Formation Control of Multiple Robots Under Packet Loss

Leila Sedghi
MSC
119222702

Thesis submitted for the degree of
Doctor of Philosophy



NATIONAL UNIVERSITY OF IRELAND, CORK

COLLEGE OF SCIENCE, ENGINEERING AND FOOD SCIENCE

SCHOOL OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY

March 2025

Supervisors: Professor Dirk Pesch
Professor Ken Brown

Contents

List of Figures . . . iii
List of Tables . . . vi
Acknowledgements . . . ix
Abstract . . . x

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Thesis Hypotheses and Goals . . . 4
1.3 Thesis Contributions . . . 5
1.4 Publications . . . 8
1.5 Thesis Outline . . . 8

2 Background and State of the Art **10**
2.1 Multi-agent Systems . . . 10
2.1.1 Centralized Architecture . . . 11
2.1.2 Distributed Architecture . . . 12
2.1.3 Hierarchical Architecture . . . 12
2.2 Formation Control . . . 13
2.2.1 Leader-follower . . . 16
2.2.2 Virtual structure . . . 16
2.2.3 Behavior-based . . . 17
2.2.4 Consensus-based . . . 17
2.3 Cooperative Transportation . . . 17
2.3.1 Autonomous Guided Vehicles . . . 18
2.3.1.1 AGV Control . . . 18
2.3.2 Unmanned Aerial Vehicles . . . 20
2.3.2.1 UAV control . . . 22
2.4 Network uncertainties . . . 23
2.4.1 Communication Delays . . . 23
2.4.2 Packet Loss . . . 24
2.5 Systematic Effects in Communication and Control . . . 25
2.5.1 Quantization Error . . . 25
2.5.2 Discretizations of Communication Intervals . . . 25
2.6 Learning-Based Approaches for Network Uncertainties . . . 26
2.7 Memory Consensus Protocol for Packet-Loss Compensation . . . 27
2.8 State of the Art . . . 29
2.8.1 Model-based predictive compensation . . . 30
2.8.2 Iterative-learning control compensation . . . 32
2.8.3 Data-driven and deep-learning compensation . . . 33

3 Formation Control of AGVs under Packet Loss **38**
3.1 Motivation . . . 38
3.2 Mathematical Modeling of AGV . . . 40
3.3 Controller Design . . . 42
3.4 Formation Control of AGVs Under Packet Loss . . . 46

3.4.1 Impact of Packet Loss on Formation Control 46
3.4.2 Long Short-Term Memory to cope with packet loss . . . 48
3.4.2.1 Architecture of LSTM prediction and control . 49
3.4.2.2 Dataset preparation and splitting 52
3.4.2.3 Application of LSTM for leader position prediction 52
3.5 Performance Evaluation 53
3.5.1 Prediction accuracy for LSTM and GRU 55
3.5.2 Simulation Results 58
3.5.2.1 Circular Path 59
3.5.2.2 Eight-Shaped Trajectory 63
3.6 Concluding remarks 68

4 Formation Control of AGVs under Packet Loss and Delay Compared with Nonlinear Controllers 70
4.1 Motivation 70
4.2 Methodology of nonlinear controller for AGVs 73
4.3 Comparison of proposed LSTM under packet loss and delay with nonlinear controllers 76
4.3.1 Simulation result 77
4.3.1.1 Impact of delay on formation control of proposed LSTM 78
4.3.1.2 Input shaping filter against delay 79
4.4 Concluding Remarks 82

5 Formation Control of Quadrotors under Packet Loss 85
5.1 Motivation 85
5.2 Mathematical Modeling of Quadrotor UAVs 87
5.3 Controller Design 89
5.3.1 Mixed H_2/H_∞ Controller 91
5.3.2 Mixed H_2/H_∞ Tracking Controller 94
5.3.3 Proposed Quadrotor Controller 95
5.4 Formation Control of Quadrotors Under Packet Loss 99
5.4.1 Impacts of packet loss on formation control 99
5.4.2 Long Short Term Memory to Cope with Packet Loss and Proposed Algorithm 100
5.5 Performance Evaluation 101
5.5.1 Prediction Accuracy for LSTM and GRU 103
5.5.2 Simulation Results 105
5.5.2.1 Spiral Path 106
5.5.2.2 Eight Shape Path 110
5.6 Concluding Remarks 115

6 Conclusion and Future Work 117
6.1 Thesis Summary and Contributions 117
6.2 Future work 120

List of Figures

2.1 A centralized control architecture 11
2.2 A distributed control architecture 12
2.3 A hierarchical control architecture 13
2.4 Examples of various formation shapes and trajectories (a) a platoon– shaped formation and a straight trajectory, (b) a hexagon-shaped formation and a circular trajectory. [Sad12] 14
2.5 AGVs formation control and cooperative transportation [AMBR17] 19
2.6 KUKA omniMove with omnidirectional wheel [KUK99] 19
2.7 Five multi-rotors, lifting and transporting a square frame payload autonomously[TLWC18] 21
2.8 Basic hierarchical control for UAV system 23

3.1 Kinematic model of robot 41
3.2 Control block diagram 44
3.3 Effect of packet loss (a) Error of follower ; (b) Zoomed view of follower error 48
3.4 LSTM architecture 50
3.5 The flowchart of the proposed approach of sending the leader position to the follower. 53
3.6 Prediction errors. (a) Error for X; (b) Error for Y; (c) Error for heading angle 56
3.7 Training and validation loss. (a) Loss for X; (b) Loss for Y; (c) Loss for heading angle 56
3.8 Closed-loop control of leaders and two followers under packet loss using LSTM prediction 59
3.9 Trajectory of the robots during formation control (packet loss—30% compared with no packet loss) (a) Leader and Followers circular trajectory ; (b) Zoomed view of Follower 1 circular trajectory, highlighting the overlap of LSTM, MCP, and no-packet-loss paths for better clarity. 60
3.10 Distance of the first follower from the leader for the circular trajectory (packet loss—30% compared with no packet loss). . 61
3.11 Distance of the second follower from the leader for the circular trajectory (packet loss—30%). 61
3.12 Trajectory of the robots during formation control (packet loss—50% compared with no packet loss) (a) Leader and Followers circular trajectory ; (b) Zoomed view of Follower 1 circular trajectory, highlighting the overlap of LSTM, MCP, and no-packet-loss paths for better clarity. 62
3.13 Distance of the first follower from the leader for the circular trajectory (packet loss—50%). 62
3.14 Distance of the second follower from the leader for the circular trajectory (packet loss—50%). 63

3.15 Trajectory of the robots during formation control (packet loss—30% compared with no packet loss) (a) Leader and Followers eight shape trajectory ; (b) zoomed view of Follower 1 eight shape trajectory, highlighting the overlap of LSTM, MCP, and no-packet-loss paths for better clarity. 65
3.16 Distance of the follower from the leader for the circular trajectory (packet loss—30%). 65
3.17 Distance of the follower from the leader for the circular trajectory (packet loss—30%). 66
3.18 Trajectory of the robots during formation control (packet loss—50% compared with no packet loss) (a) Leader and Followers eight shape trajectory ; (b) zoomed view of Follower 1 eight shape trajectory, highlighting the overlap of LSTM, MCP, and no-packet-loss paths for better clarity. 66
3.19 Distance of the follower from the leader for the circular trajectory (packet loss—50%). 67
3.20 Distance of the follower from the leader for the circular trajectory (packet loss—50%). 67

4.1 Control block diagram sliding [NDM+21]. 76
4.2 Formation control of proposed method for the circular trajectory (packet loss—20% and time-varying delay). 78
4.3 Proposed leader follower formation control 80
4.4 Proposed leader follower formation control against delay and packet loss 80
4.5 Result of [NDM+21] based on [WZ15] for the circular trajectory (packet loss—20%). 81
4.6 Result of [NDM+21] for the circular trajectory (packet loss—20%). 82

5.1 Quadrotor tracking controller structure 94
5.2 Quadrotor control scheme 96
5.3 LMI Region[SF16] 98
5.4 Effect of packet loss (a) Error of follower ; (b) zoomed view of follower error 100
5.5 Bayesian optimization results for position x 103
5.6 Prediction errors. (a) Error for x; (b) Error for y; (c) Error for z 104
5.7 Training and validation loss. (a) Loss for x; (b) Loss for y; (c) Loss for z 104
5.8 3D Trajectory of the quadrotors during formation control 106
5.9 Distance of the first follower from the leader for the spiral trajectory (packet loss—30%) (a) Distance of F1 from the leader ; (b) zoomed view distance of F1 from the leader 107
5.10 Distance of the second follower from the leader for the spiral trajectory (packet loss—30%)(a) Distance of F2 from the leader ; (b) zoomed view distance of F2 from the leader 107

5.11 Distance of the first follower from the leader for the spiral trajectory (packet loss—50%) (a) Distance of F1 from the leader ; (b) zoomed view distance of F1 from the leader 108
5.12 Distance of the second follower from the leader for the spiral trajectory (packet loss—50%)(a) Distance of F2 from the leader ; (b) zoomed view distance of F2 from the leader 108
5.13 Distance of the first follower from the leader for the spiral trajectory (packet loss—70%) (a) Distance of F1 from the leader ; (b) zoomed view distance of F1 from the leader 109
5.14 Distance of the second follower from the leader for the spiral trajectory (packet loss—70%) (a) Distance of F2 from the leader ; (b) zoomed view distance of F2 from the leader 109
5.15 3D Trajectory of the quadrotors during eight shape formation control 111
5.16 Distance of the first follower from the leader for the eight shape trajectory (packet loss—30%) 111
5.17 Distance of the second follower from the leader for the eight shape trajectory (packet loss—30%) 112
5.18 Distance of the first follower from the leader for the eight shape trajectory (packet loss—50%) 112
5.19 Distance of the second follower from the leader for the eight shape trajectory (packet loss—50%) 113
5.20 Distance of the first follower from the leader for the eight shape trajectory (packet loss—70%) 113
5.21 Distance of the second follower from the leader for the eight shape trajectory (packet loss—70%) 114

List of Tables

2.1 Summary of related work on formation control under communication uncertainties. 36

3.1 LSTM model setting. 54
3.2 GRU model setting. 54
3.3 Prediction error under packet loss. 57
3.4 RMSE of follower error for the circle path. 63
3.5 MAE of follower error for the circle path. 64
3.6 RMSE of follower error for the eight-shaped path. 67
3.7 MAE of follower error for the eight-shaped path. 68

4.1 Average tracking error norm of followers considering delay and packet loss in all methods. 82

5.1 LSTM model setting. 102
5.2 GRU model setting. 102
5.3 Prediction error under packet loss 105
5.4 RMSE of follower error for the spiral path. 110
5.5 MAE of follower error for the spiral path. 110
5.6 RMSE of follower error for the eight shape path. 114
5.7 MAE of follower error for the eight shape path. 115

I, Leila Sedghi, certify that this thesis is my own work and has not been submitted for another degree at University College Cork or elsewhere.

_____

Leila Sedghi

To my beloved family, whose unwavering support and encouragement have been my greatest strength.

Acknowledgements

This thesis would not have been possible without the support, encouragement, and guidance of many people.

First and foremost, I would like to express my deepest gratitude to my supervisor, Prof. Dirk Pesch, for his unwavering support, guidance, and invaluable insights throughout this research. His expertise, thoughtful feedback, and dedication have been fundamental in shaping my work. I am deeply appreciative of the time and effort he has devoted to helping me achieve this accomplishment. I would like to extend my sincere thanks to my co-supervisor, Prof Ken Brown, for constructive advice that is refining in my research and developing my skills.

I would like to sincerely thank Prof. John Barrett for his invaluable support at the beginning of my Ph.D. research. I am also deeply grateful to Prof. Ahmad Fakharian, my supervisor during my master's studies, for believing in me and for his tremendous support and guidance, which ultimately paved the way for my journey to Cork. Additionally, I would like to express my heartfelt gratitude to Dr. Md Noor-A-Rahim for his unwavering support and guidance throughout this thesis.

I would like to express my sincere appreciation to my company, Jaguar Land Rover, and my line managers, Sravan Kallam and Jonathan Lloyd, and my product owner, Hendrik Boettcher, for their support throughout this journey. Their flexibility, encouragement and understanding allowed me to balance my professional responsibilities with the demands of my research, making this accomplishment possible. I am equally grateful to the AI Chapter at Jaguar Land Rover for providing me with an environment that fosters innovation and exploration in artificial intelligence and broadening my understanding of real-world applications of AI, which has significantly enriched my research. I would especially like to thank my AI chapter head, Isabella Panella, whose leadership and guidance have been truly inspiring.

A special thanks goes to my husband, Masoud, whose unwavering support, patience, and encouragement have been my constant anchor throughout this journey. His belief in me has given me immense strength, and his sacrifices have allowed me to pursue this dream with confidence. I am deeply grateful to my parents, Nasser and Shamsi, and my brother, Azad, for their enduring love, understanding, and faith in me. Their constant encouragement, even from afar, has inspired me to overcome challenges. I would also like to express my heartfelt gratitude to my extended family, especially my uncle Shokrollah, my mother-in-law, Maryam, and my aunt and uncle, Jabar and Badri, for their continued support and encouragement from afar.

Finally, to everyone who has supported me along this journey, your belief in me has made this accomplishment possible. Thank you.

Abstract

This thesis addresses the challenges posed by communication packet loss in leader-follower formation control of multi-robot systems, with a focus on Automated Guided Vehicles (AGVs) and drones. It presents a comprehensive analysis of how packet loss affects linear controllers, including Linear Quadratic Regulator (LQR) for AGV position tracking and a mixed H_2/H_∞ and Proportional Derivative (PD) controller for drone attitude and position control. The results demonstrate significant positional errors under packet loss, emphasizing the necessity for predictive methods to mitigate these effects.

To tackle this problem, the thesis proposes the use of Long Short-Term Memory (LSTM) networks, which excel at retaining long-term dependencies, for predicting missing data during packet loss. A comparative analysis involving LSTM, Gated Recurrent Units (GRUs), Linear Interpolation Predictor (LIP), and the Memory Consensus Protocol (MCP) highlights LSTM's superior accuracy in trajectory prediction for AGVs and drones. The LSTM model was further integrated into closed-loop systems, where it significantly enhanced performance under feedback dynamics and external disturbances, outperforming MCP and LIP methods.

The robustness of the proposed method was validated through extensive testing under realistic scenarios, including curvy trajectories, such as circular and figure-eight paths for AGVs and spiral trajectories and figure-eight paths for drones, as well as scenarios with initial offset conditions, actuator saturation, and the presence of external disturbances like sensor noise and wind. These tests demonstrated LSTM's capability to maintain formation control accuracy in challenging environments, making it well-suited for practical applications in AGVs and drones.

This thesis also compares the LSTM-based linear control strategies with highly nonlinear controllers, such as back-stepping control and event-based sliding mode control. The comparative evaluation includes challenging conditions, such as time-varying delays, which initially caused oscillations in the system. By applying an input-shaping filter, the oscillatory behavior is mitigated, and the stability of the system is restored. The LSTM-based linear control strategy exhibits performance that is competitive with nonlinear controllers, showcasing its effectiveness as a robust alternative for handling formation control challenges in multi-robot systems. These results demonstrate that combining LSTM with

linear controllers enhances their performance under uncertainties such as packet loss, improving their robustness to a level comparable with nonlinear controllers.

Keywords Formation control, Automated Guided Vehicles (AGVs), Drones, Long Short-Term Memory (LSTM), Prediction, Packet Loss

Chapter 1

Introduction

1.1 Motivation

In recent years, the development and deployment of unmanned vehicles has grown significantly, driven by advancements in autonomy and the desire to minimize human risk in hazardous tasks. Unmanned vehicles, which can function with minimal human intervention, are now widely employed in a range of specialized sectors, such as military operations, search and rescue missions, and high-risk industrial tasks. These vehicles are classified based on their operational environment: Unmanned Aerial Vehicles (UAVs) operate in the air [SSAF+19], Unmanned Ground Vehicles (UGVs) navigate on land [MKLT95], Unmanned Surface Vehicles (USVs) travel across water surfaces [Man08], and Autonomous Underwater Vehicles (AUVs) travel under the water [ZM13].

Several practical platforms for these autonomous vehicle types have already been developed and deployed. However, one common limitation across these platforms is their typically small size and limited capacity, making them suitable only for simpler missions. Moreover, many current unmanned vehicle systems possess low levels of autonomy, with some relying heavily on remote control or operating in a semi-autonomous mode [LZ18]. To mitigate these challenges, the deployment of multiple smaller vehicles in a coordinated fleet, or multi-vehicle formation system, has emerged as an effective solution. Compared to a single large vehicle, a fleet in formation offers advantages such as greater coverage area, enhanced system robustness, and improved fault tolerance through coordinated behavior.

In the pursuit of more effective multi-vehicle formations, extensive research has

been conducted in recent decades, with formation control being one of the most actively explored areas. Formation control ensures that robots maintain a desired arrangement while performing collaborative tasks, such as transporting objects or surveying an area [CD20]. Over the past few years, significant progress has been made with successful real-world demonstrations of vehicle formation control. For example, Amazon Prime Air [JK17] has showcased successful trials of drone-based deliveries, where coordinated flight in formation plays a crucial role. In the automotive sector, truck platooning has been validated in practice, showing improvements in fuel efficiency and traffic flow [YKDvdS19]. Similarly, collaborative multi-robot systems have been employed in warehouse logistics to transport large or heavy objects [AWL+23], while using autonomous vessels is being actively explored [CHN19] to coordinate autonomous ships or boats to transport goods across water bodies. These success cases highlight the growing maturity of formation control technologies and motivate research into their robustness. However, one of the significant challenges in multi-vehicle systems is maintaining reliable communication among the agents. In real-world environments, communication links are often subject to packet loss due to interference, limited bandwidth, or hardware limitations. This disruption can degrade the performance of formation control, leading to instability or breakdown in the fleet structure.

In this thesis, we address the challenge of formation control under packet loss by focusing on two different types of unmanned vehicles: ground vehicles, specifically Automated Guided Vehicles (AGVs), and aerial vehicles, specifically UAVs. These platforms were chosen due to their growing applications in transportation, surveillance, and search-and-rescue missions, where maintaining reliable formation is critical. For example, AGVs are widely used in industrial and logistics environments, while UAVs are crucial for tasks requiring airborne operations in challenging or remote areas.

To tackle the issue of communication loss, we developed a formation control strategy that integrates deep learning techniques, specifically Long Short-Term Memory (LSTM) networks, with traditional linear controllers such as Linear Quadratic Regulator (LQR), mixed H_2/H_∞, and Proportional Derivative (PD) controllers. While complex nonlinear controllers, such as sliding mode control, can offer greater robustness against packet loss and disturbances compared to linear controllers, they are not entirely immune to communication issues. Moreover, their complexity makes them more challenging to implement and tune in practical applications. This thesis seeks to build on the existing infras-

tructure used in industry, where linear controllers like LQR and Proportional Integral Derivative (PID) are already widely implemented due to their simplicity, reliability, and ease of use [AN17, NAQ18, BB91]. In 2001, it was estimated that more than 90% of all control loops utilized PID controllers.[AN17, NAQ18]. While this percentage has since dropped to just over 50%, the PID controller remains the most widely used control method in the industry [AN17].

Linear controllers remain the backbone of industrial automation systems and robotics, favored for their well-understood design, computational efficiency, and integration into existing systems[BB91, Zha10, AN17, NAQ18, BMSP21, HG24]. These controllers are widely adopted across various industries because they are reliable, cost-effective, and require minimal computational resources. Replacing linear controllers with more complex nonlinear methods, such as sliding mode control or other adaptive techniques, would require significant changes to both hardware and software infrastructure. This not only increases the complexity of implementation but also poses challenges in terms of operator training, maintenance, and system compatibility. In contrast, this LSTM deep learning combined with linear controller does not replace the underlying linear framework; instead, we integrate LSTM networks as a lightweight prediction module. Unlike nonlinear controllers, which substitute the entire control law and demand continuous complex computation, the LSTM only supplements the existing controller with a fast prediction step. This avoids the hardware and software overhaul required by nonlinear methods while still addressing packet loss. Since LSTM inference is computationally light and can run in real time on a CPU, GPU support is only optional for offline training, making integration considerably simpler than adopting nonlinear controllers.

By integrating LSTM networks into traditional linear control frameworks such as LQR and mixed H_2/H_∞ controllers for AGV and drones, we seek to overcome the shortcomings of these linear controllers in environments with communication packet loss. LSTM, a type of recurrent neural network, is particularly suited for learning and predicting time-series data, making it an ideal candidate for handling intermittent communication in formation control of muti robots. By using LSTM, the system can predict the future states of leader robot when communication fails, thereby allowing the fleet to maintain formation and perform tasks with reduced reliance on continuous communication.

It is worth noting the difference between the methods proposed in this thesis and the dead reckoning method. While both approaches involve prediction,

their functions and objectives differ substantially. Dead reckoning (the "deduced reckoning" of sailing days) is a simple and basic mathematical procedure for finding the present location of a vessel by advancing some previous position through a known course and using the velocity information of a given length of time [DM78]. In dead reckoning, the Global Positioning System (GPS) is not available, e.g., no GPS receiver, indoor environment, etc. [TTF+20]; therefore, dead reckoning is used as a localization method that estimates the robot's position, orientation, and integrates local sensor information over time, which usually suffer from drifts [Ber16]. In this research, we do not address localization, and it is considered that each robot knows its own position accurately (e.g., the global reference is available). The focus here is instead on maintaining formation control under packet loss scenarios, which dead reckoning does not address.

In contrast to previous research, this study focuses on the use of deep learning to compensate for packet loss while robots maintain their formation. LSTM is used for AGVs in various fields such as path planning [Yan23], state estimation and sensor fusion of holonomic robots [VNGW21], data fusion of the odometry and Inertial Measurement Unit (IMU) [QHM+21] anomaly detection [ÇLDB+20] and fault detection [DZZ+21]. However, to the best of our knowledge, no research has been conducted to compensate for packet loss in the formation control of AGVs and UAVs via deep learning to enhance and improve linear controllers such as LQR and mixed H_2/H_∞ controllers.

1.2 Thesis Hypotheses and Goals

Based on the research motivation, the following research hypotheses have been investigated:

1. Can the combination of a deep learning method such as LSTM in combination with a linear control strategy, such as LQR, improve the performance of formation control for AGVs under conditions of packet loss? Can the learning algorithm compensate for the effects of packet loss in the system?
2. Can this solution be extended to other applications and controllers? For example, could combining LSTM with linear controllers like H_2/H_∞ and PD improve the performance of formation control in quadrotors, which operate in three dimensions, under packet loss?

3. Can the combination of LSTM with linear control strategies perform competitively with sophisticated nonlinear controllers, such as sliding mode control, in handling packet loss and communication delays?

Driven by the research hypotheses above, this research aims to investigate and improve the formation control performance of multiple robots, including AGVs and drones, using a decentralized leader–follower approach under conditions of communication packet loss. To achieve this, we selected linear controllers developed for single robot applications, adapted them for multi-agent formation control, and tested their performance under packet loss conditions. When the negative impact of packet loss on the linear controllers used became evident, we utilized deep learning algorithms, such as LSTM networks and GRU, to address these challenges. By integrating these learning-based algorithms, we compensate for packet loss.

The details of the research goals are illustrated below:

1. Identify and analyze the impact of packet loss on popular linear controllers used in current industry such as LQR, H_2/H_∞ and PD controllers, and understand the research gaps.
2. Develop a framework that can improve and develop current controllers using predictive capabilities of LSTM.
3. Extend the solution to different applications of formation control of AGVs and drones controlled by LQR and H_2/H_∞ controllers, respectively.
4. Compare the performance of framework against nonlinear controllers such as sliding mode controller and back stepping controller that are robust controllers against packet loss and delay used in state of the art.

1.3 Thesis Contributions

This section presents the key contributions of this thesis.

1. **Analyzing the Impact of Packet Loss on Leader-Follower Formation Control in Multi-Robot Systems**. This thesis examines the effects of packet loss on the leader-follower formation control of multiple robots using linear controllers. Two popular robotic applications AGVs and drones were chosen to explore this phenomenon in formation control settings. For AGVs, an LQR controller was implemented to maintain a specific trajectory and

ensure proper formation. Similarly, for drones, a mixed H_2/H_∞ controller combined with a PD controller was employed to manage trajectory tracking and formation maintenance through communication between the leader and followers. In both applications, leader position updates communicated to followers were subjected to packet loss modeled using a Bernoulli distribution. Using only memory-based compensation (MCP), for AGVs, packet loss caused a positional error increase of approximately 20 cm, while for drones, the error increased by around 6 cm. These results confirm that MCP alone is insufficient to maintain formation accuracy under communication disruptions, highlighting the need for predictive methods such as LSTM.

2. **Proposing LSTM for Predicting Missing Data via a specific mechanism and Compensating for Packet Loss in Formation Control**. This thesis explores four methods for predicting the leader's position in robot formation control during packet loss: LSTM, Gated Recurrent Units (GRUs), Linear Interpolation Predictor (LIP), and the MCP. LSTM is particularly advantageous due to its capacity to retain important information over time, making it well-suited for predicting missing data resulting from packet loss. Both LSTM and GRU models are trained using the leader's trajectories, with carefully chosen parameters optimized through Bayesian optimization to balance prediction accuracy, computational efficiency, and processing time. When no packet loss occurs, the network states are updated with the actual leader position. However, during packet loss, the network states are updated using LSTM predictions, as the observed leader position is unavailable. The final results show that LSTM consistently outperforms GRU, LIP, and MCP in prediction accuracy when compared to the actual data in both AGV and drone trajectories.

3. **Integrating LSTM in Closed-Loop Systems for Controlling AGVs and Drones to Analyze Formation Control Performance**. This thesis extends beyond simply comparing predicted data with real data by incorporating the chosen method, LSTM, into closed-loop systems to evaluate its impact on formation control under feedback dynamics and external influences. In this approach, LSTM is integrated into the reference generator for followers, where its predictions are utilized during packet loss of 30% and 50%. The results demonstrate that LSTM compensate packet loss and the results are very close to without loss scenario.

4. **Testing and Analyzing the Performance of the Proposed Method in Complex Scenarios**.To evaluate the effectiveness of the proposed method in realistic and challenging situations, various test conditions were designed:

   - **Complex Trajectories**: Both AGVs and drones were tasked with following complex trajectories, such as circular and figure-eight paths for AGVs and a spiral and figure-eight trajectories for drones while they are keeping their formation. Unlike straight paths, curvy trajectories with frequent direction changes present greater challenges for prediction. The proposed method demonstrated excellent performance, maintaining accuracy even under these demanding conditions.
   - **Initial Offset Conditions**: The system was tested with followers starting at an offset position, lagging behind the leaders. Throughout the test, the followers were required to adjust their speeds dynamically to maintain a fixed distance and preserve the formation along the entire trajectory.
   - **Actuator Saturation**: To reflect the constraints of real-world industrial robots, control signals in both AGVs and drones were saturated, ensuring that the robots did not exceed realistic speed limits. This test assessed the compatibility of the proposed method with practical actuator limitations.
   - **Noise and External Disturbances**: For the drone scenario, the system's robustness was evaluated in the presence of measurement noise and external disturbances. This simulated the effects of real-world factors, such as sensor noise from Inertial Measurement Units (IMUs) and environmental disturbances like wind or turbulence, which are commonly encountered in practical quadrotor operations.

   The results from these tests indicate that the proposed method is not only capable of handling complex trajectories and real-world constraints but also resilient to noise and disturbances, making it highly suitable for practical applications in AGVs and drones.

5. **Comparison Between LSTM Combined with Linear Control Strategies and Highly Nonlinear Controllers**. To assess the effectiveness of the proposed method in AGVs' formation control, a comparison was made with highly nonlinear controllers such as back-stepping control and event-based

sliding mode control. A benchmark study was established using the most comparable paper in terms of testing conditions. For fairness, the proposed method's simulation incorporated a time-varying delay, which initially caused oscillatory behavior in the system. To mitigate these oscillations, an input-shaping filter was applied, improving the system's stability. Upon evaluating the average tracking error, the LSTM-based approach demonstrated competitive performance with the nonlinear controllers, even under challenging conditions involving packet loss and delays. This underscores the robustness and adaptability of the proposed method. This study highlights the practicality of enhancing traditional linear control frameworks by integrating learning algorithms like LSTM and supplementary filters. Such an approach provides a viable alternative to developing complex nonlinear control systems, offering significant improvements in performance of current industrial linear controllers.

1.4 Publications

The research and findings of this thesis have been disseminated through the following publications, with some contributions currently under preparation for submission:

- **Published Articles:**

  Sedghi, Leila, Jobish John, Md Noor-A-Rahim, and Dirk Pesch. "Formation control of automated guided vehicles in the presence of packet loss." Sensors 22, no. 9 (2022): 3552.

  Sedghi, Leila, Zohaib Ijaz, Md Noor-A-Rahim, Kritchai Witheephanich, and Dirk Pesch. "Machine learning in event-triggered control: Recent advances and open issues." IEEE Access 10 (2022): 74671-74690.

- **Work in Progress:**

  Additional findings on formation control of quadrotors under packet loss, are under development and will soon be submitted to IEEE Access Journal.

1.5 Thesis Outline

The remainder of this dissertation is organized as follows.

Chapter 2, introduces the overview of PhD research background from multi-agent systems to formation control and cooperative transportation of AGVs and drones are discussed. After highlighting the network uncertainties impact on formation control, we reviewed the state of the arts to clarify research gap. Chapter 3 presents formation control problem for non-holonomic AGVs using LQR control in presence of packet loss and it demonstrates the vulnerability of the system to packet loss. To cope with this, LSTM framework is developed to compensate the packet loss negative impact in the AGV system. In Chapter 4, a comparative examination with nonlinear controllers is studied for AGVs scenario and explores strategies to mitigate both delay and packet loss. In Chapter 5, the LSTM framework has been applied to a new application of drones with different linear controllers of mixed H_2/H_∞ and PD to ensure the scalability of this framework to cope with packet loss. Finally, Chapter 6 concludes the thesis by summarizing the performance of the proposed method, discussing its limitations, and offering insights into future research directions.

Chapter 2

Background and State of the Art

This chapter introduces the key background concepts relevant to formation control under packet loss. It begins with an overview of multi-agent system architectures, centralized, distributed, and hierarchical, and summarizes common formation control strategies such as leader–follower, virtual structure, and behavior-based methods. The chapter then discusses cooperative transportation using Automated Guided Vehicles (AGVs) and Unmanned Aerial Vehicles (UAVs), highlighting their control frameworks and applications. Network-related challenges such as delay and packet loss are explained. Finally, the chapter reviews existing literature on formation control under unreliable communication, establishing the research gap that motivates the proposed Long Short-Term Memory (LSTM) augmented linear control approach.

2.1 Multi-agent Systems

Multi-agent systems consist of multiple interacting computing elements, referred to as agents. These agents can act autonomously, making independent decisions to achieve their design objectives [Woo09]. Additionally, they can interact with other agents in order to complete a set of goals. The agents can come in various forms including vehicles[JWBB13], robots [CW05] and drones [DYSZ14]. The field of multi-agent robotic systems emerged in the late 1980s when various researchers began working with mobile robots to perform coordinated tasks [Asa89, Ben88, AFH$^+$98]. Multi-agent control primarily includes three main architectures [YJC13, VR21]: centralized, distributed, and hierarchical. In this section, we discuss the differences among these control architectures, highlighting their advantages and disadvantages.

2.1.1 Centralized Architecture

This architecture relies on a central control agent, as depicted in Figure 2.1. The central control agent can be an external server or a designated robot within the team, such as Robot 1 controller in the Figure 2.1. To coordinate and generate control commands for the entire system, it requires the state information of all robots in the group. The primary advantage of this control structure is the global knowledge of each robot in the group, which is essential for optimal decision-making and performance. However, this architecture faces issues of robustness and reliability because the entire group of robots depends on a central controller. Additionally, it generally demands higher computational power compared to other control architectures because a single central controller is responsible for processing all the information for the entire system, handling large amounts of data, manage communications with all agents, and make real-time decisions, which can become a limitation as the number of agents increases. Consequently, the computational load on the central controller can be significantly higher than in decentralized or distributed architectures, where the workload is spread across multiple robot controllers or agents [YJC13].

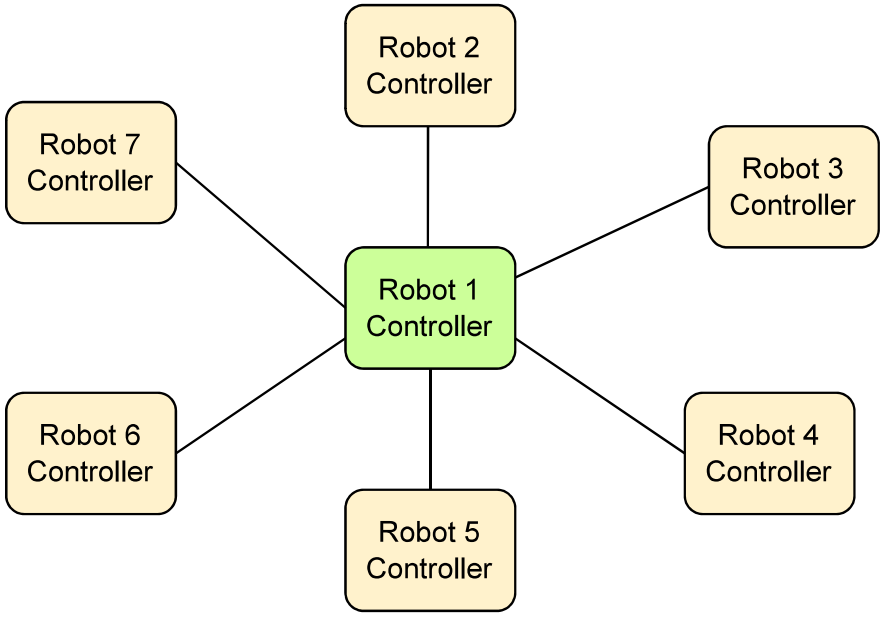


Figure 2.1: A centralized control architecture

2.1.2 Distributed Architecture

In this control architecture, each robot makes its own decisions. As shown in Figure 2.2, this approach is based on sharing information between neighboring robots. Each robot independently makes decisions based on the states of its neighbors, without relying on a single-point controller. One of the main advantages of this architecture is its reliability and its robustness, as it is more responsive to changing or unknown environments as well as agents have the ability to perform useful actions even amidst the failure of other robots [Par98]. Moreover, it is well-suited for large and scalable systems because communication is limited to interactions within the range of neighboring robots. However, this can result in suboptimal solutions in terms of stability and convergence rate [YJC13].

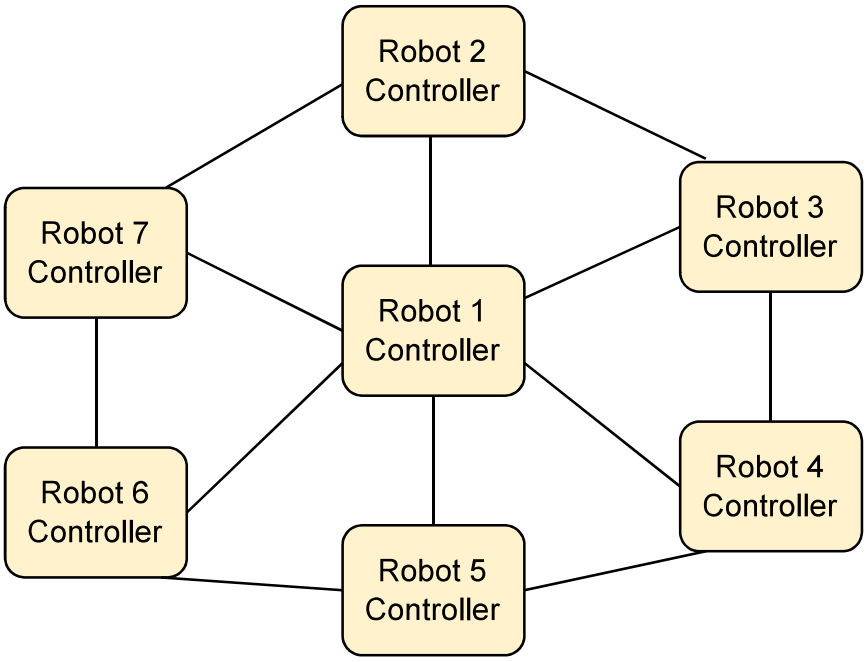


Figure 2.2: A distributed control architecture

2.1.3 Hierarchical Architecture

This architecture combines elements of both centralized and distributed control structures. There can be multiple central robots, each responsible for a local cluster of robots [CFKM95]. Figure 2.3 illustrates a hierarchical architecture

with multiple levels. At the top level, Robot 1 handles high-level decision-making and supervisory tasks, managing the subgroups or clusters of robots. Robots 2 and 3, representing Cluster 1 and Cluster 2, coordinate their respective clusters, translating high level goals from Robot 1 into specific tasks and directing them to lower levels [ACA07]. Within this hierarchy, Robots 1, 2, and 3 operate in a centralized manner. At the lowest level, Robots 4, 5, 6, 7, and 8 perform localized tasks under the guidance of the mid-level robots. These robots operate in a distributed manner, focusing on executing commands and completing tasks as instructed by the higher-level robots. In practical applications, a high-level controller might define the overall mission objectives, while lower-level controllers manage local navigation, obstacle avoidance, and task execution.

This architecture is typically implemented in environments with numerous independent tasks, each requiring an individual robot or a group of robots supervised by high level robot [NM02]. Compared to distributed architectures, hierarchical architectures are less robust but require less communication between robots. Similar to the distributed approach, this type of control is often referred to as hybrid in the literature [CFKM95].

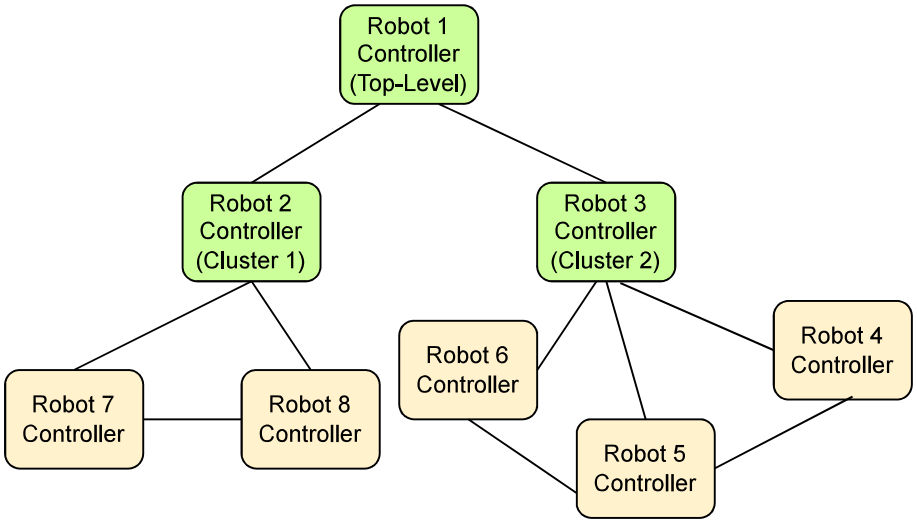


Figure 2.3: A hierarchical control architecture

2.2 Formation Control

A formation refers to a group of autonomous mobile multi-agents systems with communication capabilities that form a specific configuration (i.e., desired

positions and orientations) to collaborate and achieve a common goal as shown in Figure 2.4. Depending on the purpose of maintaining formation, we categorize and list a few applications below.

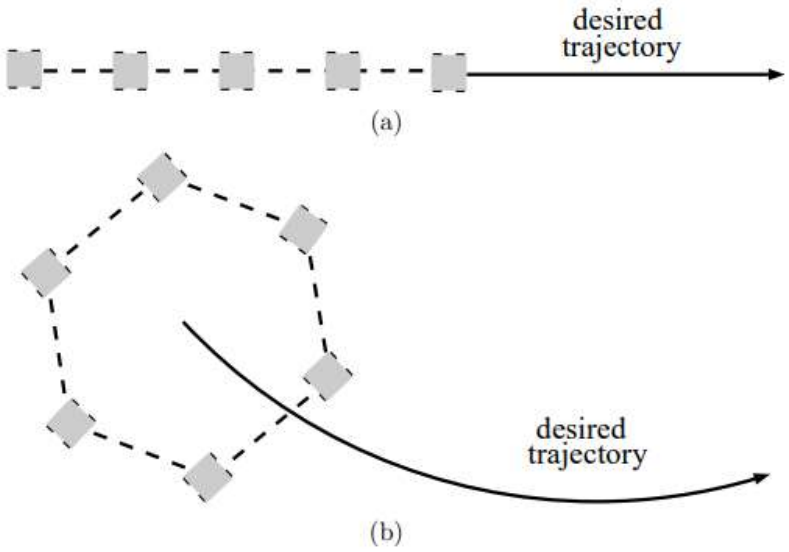


Figure 2.4: Examples of various formation shapes and trajectories (a) a platoon–shaped formation and a straight trajectory, (b) a hexagon-shaped formation and a circular trajectory. [Sad12]

- **Spatial Sensing:** The use of various sensors and mobile robots to gather data and monitor different environments, often with a focus on covering large or difficult-to-access areas. This process typically employs technologies such as UAVs, Autonomous Underwater Vehicles (AUVs), and satellites to collect information across three-dimensional spaces. The primary goals of spatial sensing include environmental monitoring, surveillance, mapping, and scientific research. It enables the detection, measurement, and analysis of physical phenomena or objects within a specified area, providing valuable data for applications like weather forecasting, resource management, disaster response, and scientific exploration. For example, drone swarms in formation are utilized for 3D scene reconstruction [MM16] and aerial filming [GNW+21]. In underwater applications [DSP16], AUVs operate in formation to take hydrographical measurements in the ocean. Formation control is also essential in space applications [LZY+18]. For instance, space-based interferometry is used to detect and study the origins of the cosmos, preferred over ground-based setups for its immunity to atmospheric disturbances and flexible reconfiguration [M+19].

- **Object Transport:** Another significant application of formation control is the cooperative transportation of objects. This involves the coordinated movement and control of multiple autonomous agents, such as drones, robots, or vessels, to collaboratively transport an object from one location to another. This type of control ensures that the agents maintain a specific formation while carrying the object, optimizing for factors such as stability, efficiency, and safety. For example, drone delivery is currently being discussed and tested, with the possibility of using drone swarms in formation becoming a promising future development [LLA+22]. Besides drones, multiple robots can collaborate to carry and transport large or heavy objects [YLL10, DN21]. Another application is robotic logistics where multiple robots working together to move heavy or large objects in warehouses or industrial settings [AWL+23]. Additionally, the potential for transportation using autonomous vessels is being actively explored [CHN19] to coordinate autonomous ships or boats to transport goods across water bodies.
- **Economic Travel:** Optimizing the movement and coordination of multiple autonomous agents, such as drones, robots, or vehicles, to minimize costs associated with energy consumption, time, and resources. This approach is essential in various applications where efficiency and cost-effectiveness are paramount. One of the advantages of traveling in formation is improved fuel efficiency. Research has shown that flying in the wake vortex of another aircraft is aerodynamically efficient, leading to reduced fuel consumption [Unt20]. Therefore, formation flying is preferred for long-haul journeys involving multiple aircraft. This concept is also applied in land transportation, such as truck platooning [YKDvdS19]. Furthermore, another application could be search and rescue team of mobile robots [QTP+20] to ensure that search and rescue operations by multiple agents are conducted in the most efficient manner, reducing time and resource expenditure.

The survey by Chen et.al. [CW05] provides a detailed study of various strategies for formation control presented in the literature. Numerous formation control strategies can be interestingly seen in the literature. Some of which are [HWZH17]: leader-follower, virtual structure, and behavior-based approach, which we will explore in the following.

2.2.1 Leader-follower

The leader-follower approach is a widely used strategy in multi-agent systems due to its simplicity of implementation. In leader-follower approach, one or more agents are assigned to be the leader(s) of the group to guide the overall movement of the formation, while the other remaining agents (followers) adjust their positions relative to the leaders. Leader-follower approaches are presented in the literature such as [LLL15, CMPT06, WWYS18]. Introducing a virtual-following robot instead of following a leader converts the formation control problem to a tracking control problem for the follower robot tracks the virtual-leader robot in [LLL15]. The leader-follower approach relies entirely on the leader, meaning the failure of the leader could impact the entire formation control. Additionally, there is typically no communication feedback from the followers to the leader, which could result in the leader moving ahead and leaving its followers behind.

2.2.2 Virtual structure

The virtual structure approach is a method in which the entire formation is treated as a single entity or "virtual structure". The desired motion is defined for the whole formation, which is, then, transformed into the desired motion of each robot for individual tracking. In fact, each agent is assigned a specific position within this virtual structure, and the agents coordinate their movements to keep these relative positions. This approach ensures that the formation behaves as a cohesive unit, similar to how a rigid body would move. Some studies, including [VdBvdWN09, CB15] proposed a virtual center defined from predefined reference trajectory for robot formation. The virtual center is used to determine the desired position for each robot at any time instance; then the robots track this position following a control command from a centralized controller. However, the virtual reference point does not have to be at the center of the formation. For example, a Virtual Vehicle (VV) is introduced for each robot in [YN08] and tried to minimize the different desired positional relations between the robot and the VV, to achieve the formation by driving all the VV to converge to a common position. In this method, each robot has its own trajectory to follow, therefore, the robots' behaviors can be simpler to describe and there is no need to inter-robot communication. As a drawback inter-robot collisions can occur in the presence of perturbations.

2.2.3 Behavior-based

Behavior-based formation control manages the movement of multiple agents, including robots, drones, or vehicles, within a group. Each agent follows local and simple rules or behaviors, and the combined effect of these behaviors results in the desired formation. This method takes inspiration from natural systems, such as bird flocking or schooling of fish, where simple local interactions create complex global patterns[BA98]. One of the main benefits of behavior-based formation control is that it is easy to implement and reacts fast to environmental change so it is highly salable and self-organizing [LC18]. As each agent follows its own set of rules and objectives, no central controller is needed. However, this approach has drawbacks, including the complexity of stability analysis and a slower convergence rate compared to leader follower method. Since behavior-based methods rely on local information and interactions between agents, this can lead to slower convergence as agents need to adjust their behavior-based on their neighbors' actions compared with leader-follower approach.

2.2.4 Consensus-based

In the consensus-based method, agents reach an agreement (consensus) on certain states like position and velocity using local interactions and communication with their neighbors [HWZH17]. In fact, consensus solves the formation problem by allowing each robot to react and compute its control velocity locally according to its neighbors state and converge to an agreed value. The consensus-based method is robust to individual agent failures, flexible in dynamic environments, and facilitates distributed decision-making. However, it requires reliable communication, may have slower response times, and can be computationally intensive.

2.3 Cooperative Transportation

The idea of cooperative transportation is inspired by daily life. When people need to move a large, heavy object, they often require assistance from others. This concept has been adapted for robotic applications, where multiple robots collaborate to transport objects. The goal of cooperative transportation is transferring the object from an initial location to a desired one via collaboration and formation control of multiple mobile robots [YO15]. Different cooperative transportation methods for multiple robots have been introduced including

leader-follower control [YWIK04], the constrain-and-move strategy [ZABW05], the chain system [YA09], cooperative towing [CFKP09], and formation control [ZMX08]. Various robots are used for cooperative transportation including various types of AGVs and UAVs. These robots are designed to work together to transport loads, optimize logistics, and operate tasks that require coordinated effort as shown in Figure 2.5.

2.3.1 Autonomous Guided Vehicles

AGVs are widely used in various industrial settings for automated material handling. Their ability to work together enhances logistics and operational efficiency. Some of the applications can be highlighted as follows[PL19, JMPH21]:

- Industrial Automation:
  - **Material Handling:** Coordinated AGVs can transport heavy or bulky materials across a factory floor.
  - **Assembly Lines:** AGVs can work together to move components to different stations along an assembly line[FDKSS21].
- Logistics and Warehousing:
  - **Order Fulfillment:** Multiple AGVs can work together to gather items from different locations within a warehouse[KKBK24].
  - **Transporting Large Items:** Coordinated efforts of several AGVs can move large pallets or containers[HZT+25].
- Search and Rescue:
  - **Disaster Response:** AGVs can work together to move debris or transport supplies in disaster-stricken areas[Dre21].
  - **Remote Operations:** Coordinated AGVs can perform complex tasks in hazardous environments where human presence is risky, such as nuclear accident sites, chemical spills, or areas affected by natural disasters.

2.3.1.1 AGV Control

The trajectory tracking problem for AGVs has attracted significant attention in recent years. A key consideration in the control of AGVs is whether the system is subject to holonomic or non-holonomic constraints.



Figure 2.5: AGVs formation control and cooperative transportation [AMBR17]

- Holonomic constraints: Constraints that depend only on the coordinates of the system and time, in fact, they are constraints that directly restrict the position of the vehicle. For example, a vehicle that can move freely in any direction on a plane (like a omnidirectional robots moving forwards, backwards, or sideways or $KUKA - omniMove^{TD}$ AGVs as depicted in Figure2.6 [KUK99]) without any restrictions other than its physical boundaries operates under holonomic constraints.



Figure 2.6: KUKA omniMove with omnidirectional wheel [KUK99]

- Non-Holonomic Constraints: Non-holonomic constraints, on the other hand, depend on both the coordinates and the velocities of the system. These constraints limit the system's ability to move in certain directions, despite the vehicle being able to reach any point in space eventually. A

common example of non-holonomic constraints is a car that can only move forward or backward and turn, but not move directly sideways (lateral motion). In this case, the vehicle's path and final position are determined by a combination of its forward/backward motion and steering.

2.3.2 Unmanned Aerial Vehicles

UAVs and multicopters (also known as multirotors as in [KFJ17, NS19]) are rotorcraft equipped with more than two rotors. They are named based on the specific number of rotors they have, such as quadcopters (4 rotors), hexacopters (6 rotors), and octocopters (8 rotors). UAVs have attracted considerable interest in research because of their diverse applications and their inherent properties, such as strongly nonlinear dynamics, under-actuated configurations[NPSL17], and high dynamics frequency while being limited by the low computational power of the on-board micro controllers, which make them ideal for showcasing various control applications [NS19]. Furthermore, UAVs or drones, are significantly used in cooperative transportation to improve efficiency and capabilities across various sectors. UAVs can operate autonomously or be remotely controlled and can work together to accomplish tasks that would be challenging for individual drones, which make them invaluable in applications ranging from logistics and agriculture to disaster response and surveillance. Some of these applications are highlighted below [SNI19, JSQ+23, GJMSBA17, MD18]:

- Aerial Delivery
  - **Package Delivery:** UAVs can collaborate to deliver packages quickly, especially in urban areas with congested roads.
  - **Medical Supplies:** They can transport urgent medical supplies, such as blood, vaccines, or medications, to remote or hard-to-reach areas.
- Construction
  - **Material Transport:** UAVs can carry and deliver lightweight construction materials to specific locations on a construction site, particularly in areas difficult to reach by ground vehicles as shown in Figure 2.7.
  - **Inspection and Maintenance:** They can work together to inspect structures and deliver maintenance tools or sensors.
- Agriculture

  - **Crop Monitoring:** UAVs can provide aerial views and data for monitoring crop health, detecting irrigation needs, and identifying pest infestations.
  - **Aerial Spraying:** Coordinated UAVs can efficiently spray pesticides or fertilizers over large fields.

- Search and Rescue
  - **Disaster Response:** UAVs can be deployed rapidly to disaster areas, providing aerial imagery, locating survivors, and delivering emergency supplies.
  - **Remote Area Assistance:** They can reach and deliver aid to individuals stranded in difficult terrains.

- Surveillance and Security
  - **Area Patrol:** UAVs can patrol large areas collaboratively, providing real-time surveillance and identifying potential security threats.
  - **Crowd Monitoring:** During large events, they can monitor crowd movements and assist in managing safety.



Figure 2.7: Five multi-rotors, lifting and transporting a square frame payload autonomously[TLWC18]

2.3.2.1 UAV control

UAVs pose unique and complex control challenges due to their significant nonlinearities, natural coupling, and underactuated dynamics. These characteristics are not merely technical hurdles but are fundamental to the dynamics of UAVs, shaping the way they are controlled and deployed. The nonlinearities in UAV dynamics arise from aerodynamic forces, torque generation, and kinematics, which are highly dependent on the UAV's current state. This complexity makes traditional linear control models insufficient for precise trajectory tracking and stability. Moreover, the natural coupling between translational and rotational dynamics means that motion in one axis inherently affects others, necessitating multi-variable control strategies that account for these interdependencies. Additionally, UAVs are underactuated systems, possessing fewer control inputs (e.g., thrust and three angular torques) than the six degrees of freedom they need to control in three-dimensional space. This underactuation restricts direct control over all motion components, requiring sophisticated algorithms, such as a predictive control, nonlinear optimization, robust control or learning-based methods, to achieve the desired behavior. UAVs are also subject to non-convex or non-holonomic constraints [NPSL17], posing substantial challenges for control. Addressing these challenges is not only crucial for ensuring stability and control but also for enabling UAVs to perform in dynamic, unpredictable environments, expanding their applicability in industrial, research, and real-world scenarios.

For example, quadrotors are inherently unstable systems, meaning they cannot maintain stable hover flight by applying a constant input to all four motors. To stabilize a quadrotor, precise control of its motor inputs at a high frequency is necessary, which varies based on its dynamic characteristics. As mentioned, quadrotors are classified as under-actuated systems since they use four control inputs to manage six degrees of freedom.

In the literature, control solutions for quadrotors are typically categorized into two design perspectives. The first design involves a single-layer control design that comprehensively considers the entire multicopter dynamics and directly computes all four control inputs simultaneously [CYWC07, CE14]. In contrast, the second design takes advantage of the naturally coupled dynamics of multicopters through a cascaded control framework [HHMS09, FLL11, NPSL17, BS07]. This approach, known as hierarchical two-layer control, is illustrated in Figure 2.8. In this approach, a high-level position controller is responsible for tracking the reference trajectory by computing the necessary thrust force and desired angles.

Meanwhile, a low-level attitude controller focuses on stabilizing the vehicle's orientation at specific set-points [HHMS09, FLL11]. One of the key advantages of hierarchical control is that each control layer independently manages the translation and rotation dynamics of the multicopter system. This separation significantly simplifies the complexity of the control designs.

The high-level position control loop typically employs a Proportional Derivative (PD) controller to regulate position and velocity. This control loop outputs a desired acceleration, which includes both the desired collective thrust and the desired attitude, with the flexibility to independently choose a heading direction. The desired attitude is subsequently managed by a low-level control loop, under the assumption that the quadrotor's attitude dynamics are significantly faster than its position and velocity dynamics. Quadrotor control strategies typically consist of a high-level PD position control loop combined with a low-level PD attitude control loop.

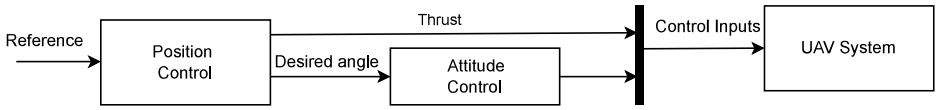


Figure 2.8: Basic hierarchical control for UAV system

2.4 Network uncertainties

Multi-agent systems rely on communication among multi-agents, such as mobile robots or drones, typically over wireless networks. Delays, packet losses, and jitter are common in such systems and can degrade the stability, performance, and reliability of formation control. This section discusses the main network-induced sources of uncertainty.

2.4.1 Communication Delays

Communication delays arise from several sources in a network stack:

- **Processing delay:** The time between packet reception and assignment to an outgoing queue [BG21].
- **Queueing delay:** The waiting time in the transmission queue before the packet starts being transmitted [BG21].

- **Transmission delay:** The time required to transmit all bits of the packet onto the link [BG21].
- **Propagation delay:** The time for the signal to propagate through the medium, proportional to the physical distance between nodes [BG21].

In practice, propagation delay is negligible for short-range wireless links, while processing and queueing delays dominate due to interference and shared communication channels. In this work, these effects are modeled collectively as a total network delay term, representing the combined latency experienced by each communication link.

2.4.2 Packet Loss

Packet loss refers to situations where transmitted data packets do not reach their intended destination. From a communication system perspective, losses can be broadly classified into two main types [BG21]:

- **Physical losses:** These occur due to errors or disruptions in the transmission medium, such as signal attenuation or interference in wireless channels. In this case, the packet is never successfully decoded at the receiver, and the data is effectively lost.
- **Buffer or queueing losses:** These occur when intermediate nodes (e.g., routers or receivers) experience buffer overflow or congestion. When the arrival rate of packets exceeds the processing or forwarding capacity, excess packets are dropped from the queue and never delivered.

Both types result in missing data at the control level, which can degrade formation stability if not properly compensated. In this work, packet loss is represented abstractly as missing information in the leader-follower communication link, regardless of the specific physical or network layer cause. As an example in our case, at a communication interval of 50 ms and a packet loss rate of 30%, a follower robot may miss leader updates for up to 150 ms, potentially leading to transient formation errors. Frequent packet loss can destabilize a control system if it heavily relies on continuous updates. In a robotic formation control system, if one robot misses a packet with critical positioning data from a leader robot, it may deviate from its intended path, disrupting the entire formation.

Communication protocols handle packet loss differently. User Datagram Protocol (UDP) often used in real-time robotic systems forgoes retransmissions to maintain

low latency, leaving losses visible to the controller. Transmission Control Protocol (TCP) mitigates packet loss through retransmission, at the cost of higher delay [KR07].

2.5 Systematic Effects in Communication and Control

While delays and packet losses are caused by network imperfections, other deterministic factors influence communication accuracy and control performance. These include quantization and discretization effects that arise from data representation and sampling choices rather than the network itself.

2.5.1 Quantization Error

Quantization refers to the process of mapping continuous signals to discrete values for transmission, which is often necessary due to limited network bandwidth, memory, or processing power. Quantization introduces small rounding errors since high-precision data (like a robot's exact position, speed, or orientation) is approximated with a limited number of bits. These errors can accumulate over time and affect the accuracy and stability of the formation. Quantization is not modeled explicitly in this work. All simulation signals in Simulink use 64-bit floating-point (double) by default, making representational error negligible compared to timing effects (e.g., 5 mm per 5 ms control step and 5 cm per 50 ms communication interval at 1 m/s).

2.5.2 Discretizations of Communication Intervals

In practical networked control systems, data exchange between agents occurs at discrete time intervals rather than continuously. This discrete sampling can introduce discretization errors, as the system states are only updated at fixed intervals, rather than reflecting real-time continuous dynamics. The mismatch between the true continuous evolution of the system and the discretely sampled updates can lead to degraded tracking performance, especially in fast-changing environments or when communication intervals are relatively large. In this work, with a control sampling time of 5 ms, a robot moving at 1 m/s travels approximately 5 mm between control updates, and with a 50 ms communication interval, it moves about 5 cm between successive packets. Since these

timing effects dominate by several orders of magnitude, quantization errors are insignificant and were therefore neglected in the analysis.

2.6 Learning-Based Approaches for Network Uncertainties

Network uncertainties, such as packet loss and delays present significant challenges in maintaining robust multi-agent system performance. Recent advancements in learning-based approaches have demonstrated their potential in addressing these challenges by leveraging data-driven predictions and control strategies. Below, we discuss various learning methods that are applicable in this context:

- **Supervised Sequence Prediction (LSTM/GRU):**

  LSTM networks, a type of recurrent neural network, are a form of supervised sequence learning, where the model is trained on labeled trajectory data to predict future states from past observations. Each input sequence corresponds to a known future target, allowing the network to learn temporal dependencies through minimization of a prediction loss such as RMSE. In multi-agent systems, LSTM networks are particularly useful for mitigating the effects of packet loss by predicting missing data or future trajectories, or reconstructing leader information for follower agents. Their architecture is designed to retain long-term dependencies, making them highly effective in handling the intermittent loss of communication in leader-follower and cooperative control setups. When it comes to time series data prediction, the results of LSTM demonstrated a significant ability to predict [AdLF17]. As we are attempting to predict the leader's trajectory, which is a time-based order sequence, this problem fits within the LSTM framework.

  The Gated Recurrent Unit (GRU) provides a computationally lighter alternative to LSTM with similar predictive behavior in many tasks.

- **Reinforcement Learning (RL):** RL offers a different paradigm by enabling agents to learn optimal policies through interactions with their environment. RL has been explored for managing network uncertainties, such as optimizing communication strategies or adapting control policies under packet loss [WNB20]. While inference can be lightweight, effective

RL typically demands more interaction data, reward shaping, and tuning than supervised forecasting—factors that may conflict with real-time and data-limited constraints common in multi-robot systems. RL remains complementary to our approach and is a candidate for future work on adaptive communication policies. While RL is a promising avenue for addressing network challenges, this thesis prioritizes LSTM for its efficiency and compatibility with linear controllers, which aligns with the requirements of resource-constrained systems.

- **Other Deep Learning Methods:** Beyond LSTM / GRU, temporal Convolutional Neural Networks (CNNs) and Transformers have been applied to spatiotemporal prediction and imputation under network imperfections, sometimes offering better long-range modeling at the cost of larger datasets and higher training complexity[CHLG17, Vas17]. Our choice of LSTM balances prediction accuracy, data/compute requirements, and straightforward integration with classical controllers.

The choice of learning method depends on the specific requirements of the system, such as computational resources, complexity of the environment, and type of uncertainty. For example, LSTM networks excel in scenarios with sequential data and high packet loss, while RL methods are ideal for adaptive communication policies. According to our hypothesis and the importance of the combination of the learning method with a linear controller while predicting time-sequenced data, LSTM networks offer robustness and flexibility, as they adapt to dynamic environments without requiring explicit modeling of all uncertainties. This makes them particularly valuable in applications involving AGVs or UAVs, where communication constraints can severely affect performance.

2.7 Memory Consensus Protocol for Packet-Loss Compensation

The Memory Consensus Protocol (MCP) is a packet-loss compensation mechanism inspired by the Zero-Order Hold (ZOH) behavior in networked multi-agent systems. When communication packets are dropped, rather than predicting the missing data using a model, each agent retains the last successfully received information from its neighbors and continues the consensus update based on that stored memory value. Thus, instead of halting motion or freezing control updates, each agent continues to compute its control input using the most re-

cently available data. This approach ensures smooth trajectory evolution even in the presence of missing communication packets[PWHB17].

Formally, for agent i receiving information from neighbor j, the effective received state $\tilde{x}_j(k)$ at discrete time step k is defined as:

$$\tilde{x}_j(k) = \begin{cases} x_j(k), & \text{if a packet is received,} \\ \tilde{x}_j(k-1), & \text{if a packet is lost.} \end{cases} \tag{2.1}$$

The control input of each agent is then updated as:

$$u_i(k) = -\sum_{j \in \mathcal{N}_i} a_{ij}\Big(x_i(k) - \tilde{x}_j(k)\Big), \tag{2.2}$$

where a_{ij} is the adjacency (communication) weight between agents i and j, and $\mathcal{N}_i$ denotes the set of neighbors of agent i.

In this formulation, each agent continues to evolve its own state $x_i(k)$ according to its local control law, while using the most recently received neighbor data $\tilde{x}_j(k)$. Therefore, MCP allows continuous motion but with temporarily outdated interaction information. This behavior introduces an implicit delay effect, since the control input depends on neighbor states from the previous successful communication instant [CHB08].

For short or isolated packet losses, this approach maintains stability and smooth motion. However, under consecutive or prolonged losses, the formation gradually drifts as agents respond to stale data. Unlike predictive methods, MCP does not attempt to estimate or reconstruct missing information—it only reuses previously received data. Hence, it provides a simple and computation-free baseline for maintaining consensus continuity but cannot fully compensate for stochastic or burst packet losses[PWHB17, PFTG14, Wak25].

In this work, MCP serves as the main baseline packet loss compensation method, against which the proposed LSTM-based predictor is evaluated. While MCP ensures motion continuity by reusing the last received neighbor states, the LSTM predictor improves performance by learning temporal dependencies and actively forecasting the missing information during packet dropouts.

2.8 State of the Art

In this section, we mainly discuss prior research that has developed algorithms and controllers to deal with packet loss in the formation control of multiple robots. As mentioned, formation control is vulnerable to network uncertainties such as channel noise, communication delay, and communication packet losses. These uncertainties can result in a variety of complex dynamic behaviours in control systems, such as divergence, oscillation, and instability [NDM+21]. Most existing studies assume ideal communication links and do not consider the effect of packet loss or delay [BYPG21, LJYX19, PYW+16, CLLH22, MCD23, GLVS20, Alr16, LWM+23].

There are some research that consider communication delay, while the packet loss is generally ignored [WTZ+18, CGP15, WLZ+21, SEMR+19, WLD+19, PGP+18, YHL+23, WDW+23].

However, packet loss in the communication network has a significant impact on the performance of any industrial process control system [LCM17], specifically the trajectory tracking performance of a robotic formation. The impact of packet loss is highlighted in [BLM+14] that considers formation control of quadrotors. The simulation results of [BLM+14] shows that the control performance in terms of tracking, collision avoidance and loss of connectivity are affected by packet loss.

The authors in [AIK21] concluded that by reducing the number of control updates and transmissions, the wireless communication channel is less congested and hence packet losses can be minimized. However, the authors did not propose any mechanisms to address the effect of packet loss, which can affect the performance of their formation control.

In the formation control of robots, holonomic and non-holonomic robots are two types of robots which are widely used in industry; as explained before, non-holonomic robots such as differential drive robots are unable to move into arbitrary directions due to a non-holonomic constraint. As a result, they require sophisticated control mechanisms, and the impact of packet loss on the formation control of non-holonomic differential drive robots is significant. Holonomic robot formation is more resilient to network uncertainties than non-holonomic robot formation [TAPJ20]. To mitigate the effects of packet loss, communication networks typically use re-transmissions to ensure that the data are received at the intended receiver. However, re-transmission will increase the

transmission delay [WLZ+21]. On the other hand, a range of control approaches such as robust control [EAM19], model predictive control [WCLY19], sliding mode control [NDM+21], and various estimators [PTT14] or filters [WG09] are widely used to address packet loss from a control perspective.

Based on the compensation strategy used, the existing research that directly addresses packet loss in formation control can be categorized into three main groups:

- Model-based predictive compensation
- Iterative-learning control compensation
- Data-driven or deep-learning compensation

2.8.1 Model-based predictive compensation

Model-based approaches explicitly incorporate packet loss within the control design, ensuring stability through analytical guarantees such as Lyapunov functions, Linear Matrix Inequality (LMIs), or robust estimation. These approaches rely on system dynamics or analytical prediction to estimate lost information.

A consensus-based tracking control strategy was studied for leader-follower formation control of multiple mobile robots under packet loss [GPP17]. A novel multiple Lyapunov functional candidate and LMIs ensure that the robots reach consensus when packet loss and communication weight (representing the rate of information flow between agents) are taken into account. Packet loss is modelled using a Bernoulli distribution and assumed to be 20% in the majority of scenarios. It is shown that the system can achieve consensus under high packet loss, but it also takes a long time to reach consensus. Their simulation results consider that one of the agents has access to the maximum amount of information, implying that their communication weight is equal to one.

A leader-follower consensus-based tracking control strategy for linearized quadcopters, considering Bernoulli-distributed communication link failures is studied in [PWHB17] . Lyapunov-based methodologies and LMIs techniques are used to design the controller. The leader (Agent 1) can only send signals to the followers (Agents 2-5), while communication between followers is undirected. Two cases are analyzed: single packet losses and consecutive packet losses. For single packet losses, the data loss rates tested were up to 40% for specific trajectory, meaning up to 40% of communication packets could be dropped before system

stability started to degrade. Effect of rate of packet loss also is considered in different typologies. In the case of consecutive packet losses, they simulated scenarios with 10% packet loss and a maximum consecutive packet loss of 5 packets. Quadrotors are going through a smooth helix trajectory. Disturbance, and other uncertainties like noise is not considered in this study.

In [NDM+21], event-triggered second-order sliding mode control is designed for consensus-based formation control of non-holonomic robots. Although sliding mode control is a robust method for counteracting packet loss and delay, the second-order sliding mode controller is a difficult controller to implement in real-world industrial scenarios. Furthermore, the main focus of this article is on event-triggered control, and the highest packet loss rate considered is 20%, modelled using the Bernoulli distribution for a circular trajectory.

Leader-follower formation control of multiple AUVs has been studied in [LLZ+22] under communication delay and packet dropout. The kernel density estimation and curve fitting methods have been used to solve the problem of communication constraints, feeding them into an input–output feedback linearization control law. Simulation and field test results show that communication constraints have a great impact on formation control and after applying their proposed method, the accuracy and stability of formation control under an unsatisfactory communication environment is improved. Packet loss is considered 30%. Although the method in this paper has shown some improvement, it seems still can not compensate the packet loss completely so it has not been tested for over 30% packet loss. It also ignores external disturbances such as ocean currents that Affect the performance.

A self–stabilizing control strategy is proposed in [YLL+23] to compensate for the negative impact of packet loss in Vehicle–to–Vehicle (V2V) wireless communication on Connected and Automated Vehicles (CAVs) and platoon stability. A self–stabilizing control strategy exploits the historical velocity information stored locally in each vehicle. The results show that the proposed self–stabilizing control strategy can significantly improve traffic stability in various packet loss scenarios. Therefore, this work provides insight into addressing the communication unreliability problems of CAVs in practice.

In [ZL18], a networked predictive controller and two algorithms are developed to cope with consecutive packet loss and communication delay for non-linear wheeled mobile robots. The authors simulated the effect of packet loss by considering a tunnel the robot travels through that is no longer detectable with

cameras and associated servers [ZL18]. A well-quantified analysis of packet loss and its effects is missing. The primary advantage of the predictive control approaches is its ability to compensate for communication delays or packet loss by estimating the system state or output, rather than merely reducing or mitigating the impact of communication constraints [CZIL23].

Decentralized outdoor formation coordination for multiple quadcopters is studied in [ALB19] by formulating the problem as a receding horizon, mixed-integer non-linear program (RH-MINLP). Each quadcopter solves this RH-MINLP to generate a time-optimal speed profile along a minimum snap spline path while maintaining its position within a desired formation. The problem includes constraints such as quadcopter kinematics, dynamics, collision avoidance, wireless communication connectivity, and geometric formations. Communication connectivity is constrained by a maximum separation distance based on minimum viable signal strength in the presence of path loss. Although the total packet loss for all quadrotors is approximately 40% to 60%, the average packet loss per quadrotor ranges between 9% and 16%. Additionally, the paths followed by each quadcopter in this study consist of a combination of linear segments, which are not highly complex trajectories, such as spirals.

These model-based methods provide theoretical robustness but are typically nonlinear, model-dependent, and computationally demanding.

2.8.2 Iterative-learning control compensation

Among the control methods used to address network uncertainties, Iterative Learning Control (ILC) design has been used to cope with packet loss in several articles [ZL21, WZG+20]. ILC is based on the class of control-theoretic learning methods, where the system improves performance by reusing tracking errors from previous iterations of a repetitive task. Unlike data-driven or deep learning approaches (e.g., LSTM networks), ILC does not require large datasets or training offline—it "learns" deterministically across repeated trials of the same motion or trajectory[BTA06].

In [XZSR16], ILC is applied to a linear system that suffers from 15% and 25% packet loss. The authors show that after 50 iterations, the system compensates for this packet loss. ILC is also used in non-linear multi-agent systems to solve the consensus problem of a leader-follower use case with packet dropouts of 10% and 40% [DSL19]. None of these ILC studies [ZL21, WZG+20, XZSR16, DSL19]

were conducted with non-holonomic AGV formation control, which is a more challenging system because of the non-holonomic constraints. Although ILC is effective approach for structured repetitive tasks, ILC-based methods are unsuitable for dynamic, non-repetitive trajectories typical in real AGV or UAV formations.

2.8.3 Data-driven and deep-learning compensation

Recent research leverages neural networks and data-driven prediction to estimate missing packets and maintain performance under packet loss.

In [VSGMP21], the authors focus on leveraging deep learning techniques, particularly LSTM and CNN models, to forecast single AGV guide errors representing the deviation from the AGV's intended path in a 5G network environment. They aim to predict AGV malfunctions by analyzing the communication between the AGV and a virtual (Programmable logic Controller) PLC utilized in Multi-Access Edge Computing (MEC). The AGV's performance is monitored in real-time by analyzing packets transmitted between the AGV and the PLC, enabling proactive adjustments to maintain trajectory adherence, even under challenging network conditions. By forecasting the mean Guide Error 10 to 15 seconds ahead, the system aims to mitigate potential malfunctions that could arise due to communication disruptions. However, their approach does not integrate existing control strategies, as the specific controller used is not mentioned. Additionally, it is unclear what trajectories are tested and how effectively their model compensates for packet loss during operation. The success rate of their predictions and the extent to which they can mitigate the effects of network disruptions remain ambiguous.

The authors of [VSGMP21] extended their work to [MVSGP24] that investigates the application of deep learning techniques for predicting guidance error variables in AGVs that are remotely controlled by a virtualized PLC. The research evaluates the performance of various deep learning architectures—specifically 1D-CNN, LSTM, and Transformer networks—within a 5G MEC infrastructure. It aims to enhance decision-making by enabling early detection of potential errors or dangerous situations, facilitating timely corrective actions to maintain the safe operation of AGVs. Through extensive experiments involving different network conditions and configurations, the study seeks to identify the most effective models for real-time deployment in an industrial setting. The results of the study indicate that Transformer models generally outperformed traditional

deep learning architectures (1D-CNN and LSTM) in predicting the guidance error of AGVs across various time windows and feature sets. Specifically, the Transformer models achieved the best performance in most scenarios, showcasing their ability to capture complex relationships in the data. However, this has been tested against delay and jitter and packet loss has not been considered. In addition, their approach does not integrate existing control strategies, as the specific controller used is not mentioned.

An LSTM predictor is combined with a linear LQR controller for AGV formation in [SJNARP22], demonstrating stable tracking under 30–50% packet loss.

A decentralized learning algorithm to restore communication connectivity during multi-agent formation control is proposed in [DKJ+23]. A trained neural network is used when the inter-agent communication is lost to generate appropriate control actions to restore connectivity. All agents use the conventional consensus protocol during the connected multi-agent dynamics, and under communication loss, only the lost agent executes the neural network predicted actions to come back to the fleet.

A novel graph convolutional flocking control strategy for nonlinear UAV systems with packet dropouts is proposed in [CKDC24], using a nonlinear control law. The deep spatiotemporal graph convolutional network extracts features from flocking data and predicts packet dropouts. With convolution kernels of size 3 and fully connected layers of 960 neurons, this method outperforms models like LSTM, ConvLSTM, and ModGNN, reducing RMS error by up to 66.70% and speeding up training by 94.63%. Flocking experiments with 10%, 20%, and 50% packet dropouts showed up to 27.37% faster convergence, maintaining robustness even at 50%. Even at high dropout rates (up to 50%), the control system showed strong robustness, achieving almost the same convergence time as under ideal conditions. Despite promising results, the study used simple trajectories like straightforward line-based paths and did not account for disturbances or noise.

These learning-based approaches show strong predictive potential but often require large training datasets. However, most of these learning-based studies focus on prediction or connectivity recovery rather than closed-loop control under packet loss. To the best of our knowledge, the present work is the first to integrate an LSTM predictor with linear controllers for leader–follower formation control of both non-holonomic AGVs and UAVs, achieving real-time packet-loss compensation with low computational cost.

Table 2.1 summarizes the major studies according to vehicle type, formation/control method, network issue considered, and mitigation technique. As evident, few studies have investigated packet-loss compensation in real time within a linear-control framework applicable to both AGV and UAV platforms.

Table 2.1: Summary of related work on formation control under communication uncertainties.

Authors	Year	Vehicle	Formation / Control	Network Issue	Technique	Limitation
Bai et al. [BYPG21]	2021	UAVs	Consensus	None	Linear control	Ideal communication
Wang et al. [WLZ+21]	2021	AGVs	Leader-follower	Delay only	Linear control	Packet loss ignored
Buonocore e t al. [BLM+14]	2014	Quadrotors	Nonlinear Null-Space-Based behavioral control	Loss (30%-40%) Bernoulli and Gilbert model	No compensation	Degraded stability and only analysis on loss, no compensation
Nandanwar et al. [NDM+21]	2021	non-holonomic AGVs	Leader-follower stochastic event-based super-twisting Sliding Mode Control (STSMC)	Loss (20%) Bernoulli model and time varying delay	Sliding-mode control	Computationally complex, nonlinear controller, model based design for uncertainties and only tested up to 20% loss
Zhang et al. [ZL18]	2018	Omni-directional wheel robots	Model-based predictive control	Delay and 10%–15% loss	MPC-based	Only one robot is considered, No formation control
Zhang et al. [ZL21]	2021	Simulated agents, no specific physical platform	Iterative Learning Control (ILC), Consensus formation	Quantization error	ILC	No packet loss and delay, no real application
Xuhui et al. [XZSR16]	2016	No specific robot or vehicle platform	LMI-based ILC	Packet loss Bernoulli	ILC	No multi agent or formation control only one plant and one controller communicating through a lossy network, slower convergence with higher loss 25%
Deng et al. [DSL19]	2019	Nonlinear multi-agent system, no specific physical platform	Leader–follower consensus (ILC)	Packet dropout (10%–40%, Bernoulli)	Distributed linear ILC with stochastic convergence compensation	Assumes perfect leader follower link and only packet loss considered between followers; repeated identical tasks, no numeric KPI
Vakaruk et al. [VSGMP21]	2021	Non-holonomic differential-drive AGV	Fault prediction (no trajectory control)	Delay, corruption, loss (40–70%)	Deep learning (CNN/LSTM) forecasting of Guide Error	Single AGV; no control integration; only prediction-based monitoring
Mozo et al. [MVSGP24]	2024	Non-holonomic AGV	Fault prediction (no trajectory control)	Delay, corruption (5G link)	Deep learning (LSTM, CNN, Transformer) forecasting of Guide Error	Prediction-only; no feedback or control integration; single AGV; no formation or motion control, no packet loss

Continued on next page

Continued from previous page

Authors	Year	Vehicle	Formation / Control	Network Issue	Technique	Limitation
Dutta et al. [DKJ+23]	2023	Multi-agent robots (formation control)	Decentralized consensus	Complete disconnection (binary link loss), not stochastic packet loss	Neural network mimicking gradient of connectivity potential	No packet loss model; single-agent reconnection, Simple straight or slightly curved motions
Yang et al. [YLL+23]	2023	CAV platoon	Self-stabilizing control	Packet loss Bernouli	Historical-velocity feedback	No adaptation mechanism, ignoring lateral dynamics, no complex trajectories (no curves or lane changes)
Li et al. [LLZ+22]	2022	AUVs	Leader-follower distributed Lyapunov-based formation controller	Delay and packet loss (30%)	Kernel density estimation and curve fitting	Not tested beyond 30% loss, Ignores external disturbances such as ocean currents, model based controller cannot predict or adapt to time-varying stochastic losses
Pan et al. [PWHB17]	2017	Quadrotors	Lyapunov–LMI distributed linear control in leader–follower consensus	Packet loss Bernoulli	Zero-order hold	Simplified linearized model; 63% on linear trajectory while degrades beyond 40% loss, Spiral trajectory test only with 10% loss and no error plotted, no disturbance
Chen et al. [CKDC24]	2024	UAVs	Flocking nonlinerar control	Packet loss (10%-50%)	Graph Convolutional Network (GCN)-based nonlinear control	Simple trajectories; no disturbance or noise modeling
Abichandani et al. [ALB19]	2019	Quad-copters	Decentralized RH-MINLP formation coordination	Total packet loss for all quadrotors (40% to 60%)	Optimization with max separation/SNR-based connectivity; round-robin scheduling	The average packet loss per quadrotor ranges between 9% and 16%, Path is combination of linear segments not complex
Sedghi et al. [SJNARP22]	2022	AGVs	Leader-follower LQR controller	Packet loss (30%-50%)	LSTM prediction, MCP, GRU	Practical implementation, no delay, no initial condition
This work	2025	AGVs / UAVs	Linear + LSTM prediction	Loss (30%-70%)	LSTM prediction	Practical implementation, generalization to unseen trajectories

Chapter 3

Formation Control of AGVs under Packet Loss

This chapter deals with the formation control problem for non-holonomic automated guided vehicles. A decentralized leader–follower approach using linear quadratic regulator control is developed in the presence of packet loss. First, a brief introduction to this problem is presented in Section 3.1. Then, Section 3.2 discusses how to model differential drive robots mathematically. An optimal Linear Quadratic Regulator (LQR) controller is designed in Section 3.3. In Section 3.4, the vulnerability of the system to packet loss is demonstrated. Then, an Long Short-Term Memory (LSTM) network is used to counteract the negative impact of packet loss. In Section 3.5, prediction accuracy of LSTM, Gated Recurrent Unit (GRU), Linear Interpolation Predictor (LIP), and Memory Consensus Protocol (MCP) are compared first. Then, simulations are used to evaluate the LSTM, LIP, and MCP performance. Finally, concluding remarks of the chapter are given in Section3.6.

3.1 Motivation

The formation control problem for multiple non-holonomic AGVs in the presence of packet loss is explained based on a decentralized leader-follower approach. The starting point of the formation control problem is presented in chapter 1 and chapter 2. Recall that the goal for formation control is to achieve a prescribed geometrical shape, using only local feedback rules. Network uncertainties, such as channel noise, communication delays, and packet loss, pose significant

challenges for maintaining stable formation control in AGVs. These network disruptions can lead to destabilizing effects in control systems, often resulting in behaviors like divergence, oscillation, or even instability [NDM+21].

As highlighted in chapter 2, to the best of our knowledge, no prior research has applied deep learning techniques to address the impact of packet loss on formation control. Motivated by this gap, we propose using LSTM networks to maintain the formation of non-holonomic differential drive AGVs, even in the face of communication failures. LSTM's strength lies in its ability to remember important information over time, making it well-suited for predicting missing data when packet loss occurs. This allows the system to compensate for lost information and maintain control.

In this chapter, we present a scenario where the leader follows a time-based trajectory, and LSTM is leveraged to predict the leader's position when communication degrades. When the actual position data of the leader is lost due to packet loss, the followers use the predicted value to continue their formation. This predictive ability is crucial for ensuring that the formation remains intact, even when wireless communication fails. LSTM ability to remember and utilize past information makes it an ideal choice for dealing with intermittent packet loss. LSTM can predict future leader positions based on historical data, allowing followers to continue the formation even when communication with the leader is compromised. Four methods are explored for predicting the leader's position: LSTM, GRUs, LIP, and the MCP. LSTM is chosen for its ability to handle long-term dependencies and predict missing data. GRUs are included for their simpler and computationally efficient architecture compared to LSTM [FZL16]. LIP method provides a simple model-free baseline that estimates missing values by linearly interpolating between the most recent valid samples, offering smoother and more accurate short-term estimates than the zero-order hold used in MCP. MCP is used as a baseline for traditional memory-based approaches that stores each neighbor's previously received data and uses it when a packet is not received by a neighbor [MW20]. MCP becomes inefficient as the number of consecutive packet losses increases [ZL18]. By comparing these methods, we aim to identify the most effective strategy for maintaining formation under packet loss. Therefore, we can address the hypothesis **Can the combination of a deep learning method such as LSTM in combination with a linear control strategy, such as LQR, improve the performance of formation control for AGVs under conditions of packet loss?**

The following are the major contributions of this chapter:

- Study of decentralized leader–follower formation control for non-holonomic AGVs using LQR and analysis of the impact of packet loss on the formation control of AGVs.
- Improving the performance of a LQR controller via machine learning, e.g., LSTM, to deal with packet losses, rather than using a highly non-linear and complicated controller such as a sliding mode controller.
- Development of a mechanism to compensate for packet loss with LSTM and the application of the mechanism to the formation control of non-holonomic differential drive robots, which are more sensitive to network uncertainties due to non-holonomic constraints.
- Comparing LSTM with GRU, LIP, and MCP for the compensation of 30 and 50 percent packet loss through simulation in MATLAB/SIMULINK.

3.2 Mathematical Modeling of AGV

As mentioned earlier, holonomic and non-holonomic robots are two types of AGVs which are widely used in industry. In non-holonomic robots like differential drive robots, the inability to move sideways is caused by the non-holonomic constraint that the wheels can roll, but can not slip [vdG20]. Another example of a non-holonomic constraint is seen in a car that can only turn its front wheels. This limitation prevents the car from moving directly sideways, as executing a parallel parking maneuver requires a more complicated set of steering maneuvers[DJ24]. In this chapter, a non-holonomic differential drive robot is considered as a model of system, which is widely used in AGV industries like Mobile Industrail Robots (MiR) and OTTO motor [Rob, htt]. A non-holonomic robot is chosen due to the model's complexity when compared to a holonomic robot. In non-holonomic robots, the number of control variables is less than the number of state variables, which complicates control. Because our objective is to derive a kinematic model, this section will discuss non-holonomic constraints.

The differential-drive robot's kinematics can be simplified using the unicycle model, where the robot is assumed to move with a linear velocity v along its heading direction θ and an angular velocity ω around its vertical axis, as shown in Figure 3.1. The robot's position is determined by coordinating the x, y which denote the Cartesian coordinates of the robot's center, and θ represents the

robot's heading angle with respect to the X and Y axes. The linear and angular velocities v and ω serve as the control inputs that drive the robot according to the following kinematic equations:

$$\begin{aligned} \dot{x} &= vcos\theta \\ \dot{y} &= vsin\theta \\ \dot{\theta} &= \omega \end{aligned} \tag{3.1}$$

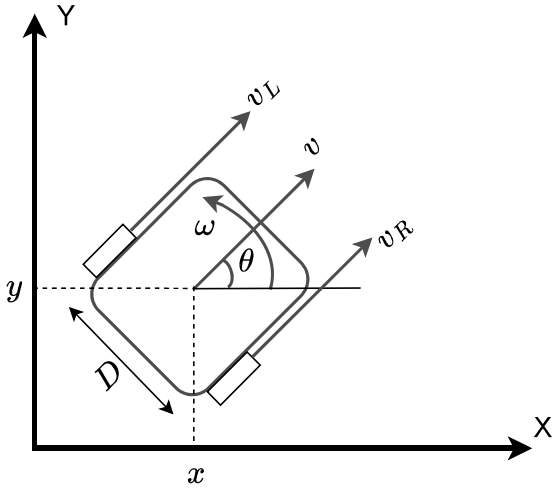


Figure 3.1: Kinematic model of robot

After obtaining v and ω from the following equations, the robot's right and left velocities can be calculated:

$$\begin{aligned} v_L &= v - \frac{\omega.D}{2} \\ v_R &= v + \frac{\omega.D}{2} \end{aligned} \tag{3.2}$$

Where the variables v_L and v_R denote the linear velocities of the left and right wheels, and D is the wheelbase, the distance between the two wheels. A positive angular velocity $\omega > 0$ corresponds to a counterclockwise (left) turn, where the left wheel moves slower and the right wheel faster than the linear speed v.

θ_{ref} shows the reference heading angle (tangent angle of each point on the path) that can be obtained from the reference trajectory (x_{ref}, y_{ref}):

$$\theta_{ref} = \arctan \frac{\dot{y_{ref}}}{\dot{x}_{ref}} + k\pi, k = 0, 1 \tag{3.3}$$

where k=0 and k=1 show forward drive and reverse drive direction, respectively.

The reference angular velocity ω_{ref} will be obtained from the time derivative of equation 3.5 and the tangential velocity v_{ref} of the robot can be obtained from equation 3.4.

$$v_{ref} = \pm\sqrt{\dot{x}_{ref}^2 + \dot{y}_{ref}^2} \tag{3.4}$$

where the sign relates to the desired drive direction (+ for forward direction and - for reverse direction)

$$\omega_{ref} = \frac{\dot{x}_{ref}.\ddot{y}_{ref} - \dot{y}_{ref}.\ddot{x}_{ref}}{(\dot{x}_{ref})^2 + (\dot{y}_{ref}^2)} \tag{3.5}$$

3.3 Controller Design

In this section, the design of the LQR tracking controller is detailed for controlling the non-holonomic robots, so that they follow a desired trajectory through a leader–follower strategy. LQR is one of the optimal control techniques, which considers the states of the system and control inputs when making optimal control decisions and computing the state feedback control gain [PTG14]. The following section is based on the design presented in [AM18] for a single robot following a reference trajectory, and as such, no novelty is claimed for this aspect of the work. The novelty of our contribution lies in extending this approach to multiple robots and analyzing the effects of packet loss. However, since this serves as the foundation for the later multi-robot extension, it is important to thoroughly review the details of the controller step by step. The reason that we chose this work that designed LQR controller is because of its simplicity and ease of implementation, while providing good accuracy, as shown in [AM18] for a single AGV. The designed LQR is a simple controller in comparison with nonlinear controllers such as a sliding mode controller, which has been proposed for AGVs in recent years [NDM+21]. Linear controllers, such

as PID and LQR, are preferred in AGV formation control due to their simplicity, ease of implementation, and proven effectiveness in handling the near-linear dynamics in current industrial AGV systems. In this work, LQR controller is responsible for generating angular velocity (ω) and linear velocity (v) for each robot's trajectory tracking. At each sampling time, the current position of the robot (x, y, and θ) is fed to the LQR controller and compared to the reference trajectory (x_{ref},y_{ref} and θ_{ref}), and the trajectory tracking error is determined [AM18]. Consider an LTI system:

$$\begin{aligned} \dot{x} &= Ax(t) + Bu(t), && t \geq 0, x(0) = 0 \\ \dot{y} &= Cx(t) + Du(t), && t \geq 0 \end{aligned} \tag{3.6}$$

where $A \in R^{n\times n}$, $B \in R^{n\times m}$, $C \in R^{p\times n}$, $D \in R^{p\times m}$ are the system matrix, control matrix, output matrix, and feed froward matrix, representing state space model, respectively. x is the state vector, u is the control vector, and y is the output vector. The goal of the LQR problem is to obtain control input to minimize the following cost function [KEJ16].

$$J(u) = \int_0^\infty [\, x^T(t)Qx(t) + u^T(t)Ru(t)]dt \tag{3.7}$$

where $Q = Q^T$ is a positive semi-definite matrix that penalizes the departure of system states from the equilibrium, and $R = R^T$ is a positive definite matrix that penalizes the control input [Des03]. The feedback control law that minimizes the value of the cost function is:

$$u = -Kx \tag{3.8}$$

where K is obtained via the following equation:

$$K = R^{-1}B^T P \tag{3.9}$$

The optimal state feedback control gain matrix K of LQR can be obtained by solving the Algebraic Riccati Equation (ARE):

$$A^T P + PA + Q - PBR^{-1}B^T P = 0 \tag{3.10}$$

Thus, to design an LQR controller, the trajectory tracking problem should be written in the form of equation 3.6. This means writing trajectory tracking problems in the form of tracking errors, which show errors with respect to a reference trajectory in x, y, and θ. These errors can be written as:

$$\begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix} = \begin{bmatrix} x_{ref} - x \\ y_{ref} - y \\ \theta_{ref} - theta \end{bmatrix} \tag{3.11}$$

where e_x, e_y, and e_θ are errors in x, y, and heading angle, respectively. These errors are in the base frame coordinate system, and to transform the error matrix to the robot coordinate, a rotation matrix has been applied to the system.

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix} \tag{3.12}$$

Finally, inputs from closed-loop system, which are ω_{cl} and v_{cl} obtained from LQR control as shown in Figure 3.2.

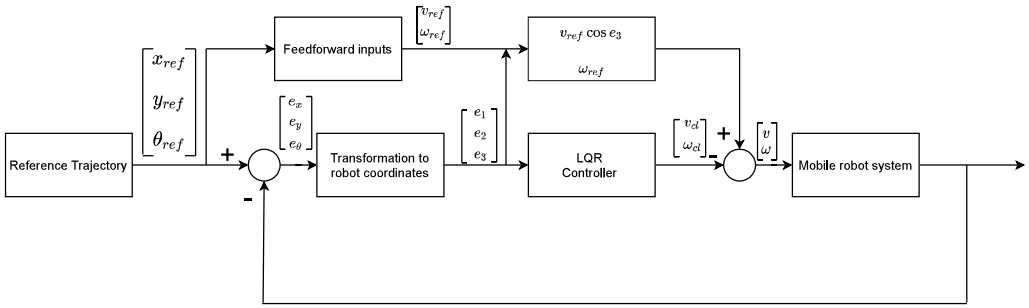


Figure 3.2: Control block diagram

Considering the kinematic model of the robot 3.1 and the robot's coordinated error 3.12, the following model is obtained.

$$\begin{bmatrix} \dot{e_1} \\ \dot{e_2} \\ \dot{e_3} \end{bmatrix} = \begin{bmatrix} \cos e_3 & 0 \\ \sin e_3 & 0 \\ 0 & 1 \end{bmatrix} . \begin{bmatrix} v_{ref} \\ \omega_{ref} \end{bmatrix} + \begin{bmatrix} -1 & e_2 \\ 0 & -e_1 \\ 0 & -1 \end{bmatrix} . \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{3.13}$$

Inputs to the equation 3.13 are:

$$\begin{aligned} v &= v_{ref} \cos e_3 - v_{cl} \\ \omega &= \omega_{ref} - \omega_{cl} \end{aligned} \tag{3.14}$$

Rewriting equation 3.13 and combining with equation 3.14 results in

$$\begin{bmatrix} \dot{e_1} \\ \dot{e_2} \\ \dot{e_3} \end{bmatrix} = \begin{bmatrix} 0 & \omega & 0 \\ -\omega & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} . \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} + \begin{bmatrix} 0 \\ \sin e_3 \\ 0 \end{bmatrix} .v_{ref} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} . \begin{bmatrix} v_{cl} \\ \omega_{cl} \end{bmatrix} \tag{3.15}$$

After linearizing 3.15 around the operating point O.P (O.P: $e_1 = e_2 = e_3 = 0, v_{cl} = \omega_{cl} = 0$), the state space model of linear system is:

$$\begin{bmatrix} \dot{e_1} \\ \dot{e_2} \\ \dot{e_3} \end{bmatrix} = \begin{bmatrix} 0 & \omega_{ref} & 0 \\ -\omega_{ref} & 0 & v_{ref} \\ 0 & 0 & 0 \end{bmatrix} . \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} . \begin{bmatrix} v_{cl} \\ \omega_{cl} \end{bmatrix} \tag{3.16}$$

Therefore, equation 3.16 is in the form of 3.6. System 3.16 is controllable if and only if its controllability matrix has a full rank. In fact, rank (B, AB, A^2B)=3 if either v_{ref} and ω_{ref} are nonzero, which is a sufficient condition only when reference inputs v_{ref} and ω_{ref} are constant. This case is only true when the trajectory is a line or circular path. However, when testing the system on more complex trajectories, such as a eight shape, the reference inputs may vary, which could lead to degraded performance. In such cases, the system might still follow the desired trajectory but with reduced accuracy, slower responses, or path deviations. The controllability of a drift-less system can be derived from Chow's theorem if the system is completely non-holonomic. As it is shown in the system 3.1, the system is completely non-holonomic as it has only one non-holonomic constraint, which is:

$$\dot{x} \sin\theta - \dot{y} \cos\theta = 0 \tag{3.17}$$

Therefore, the robot cannot move in a lateral direction to its wheels and the system is controllable [KMB05]. The controller obtained from LQR is a gain matrix K with dimensions of 2*3 because the system has three states and two

inputs. As a result, the equation is defined using the control block diagram shown in Figure 3.2.

$$\begin{bmatrix} v_{cl} \\ \omega_{cl} \end{bmatrix} = - \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \end{bmatrix} . \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} \tag{3.18}$$

To obtain feedback gain, matrices Q and R will be tuned. Q is a positive-definite/ semi-definite diagonal matrix related to state variables and R is a positive definite diagonal matrix related to input variables [Kat10]. For the robot tracking system, Q and R chosen as [AM18]:

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 10 \end{bmatrix}, R = 0.0001 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

3.4 Formation Control of AGVs Under Packet Loss

The main objective in formation control is to achieve a prescribed geometrical shape for the network of agents. In this section, the controller's performance under various packet loss conditions is evaluated. Following that, the application of an LSTM model is discussed in order to predict the leader's position in instances of packet loss, intended for utilization by the followers.

3.4.1 Impact of Packet Loss on Formation Control

As demonstrated in [AM18], this controller performs admirably well for tracking the trajectory of a single robot along a variety of paths. The following sections will evaluate the controller's performance under various packet loss conditions.

This approach aims to extend the problem to the formation control of multiple robots while accounting for packet loss. The effects of packet loss are depicted in Figure 3.3. Packet loss is modeled using a Bernoulli distribution with rates of 30% and 50%, affecting the leader's data communicated to the followers going through a circular path. Additionally, followers start with an initial offset relative to the leaders and are required to maintain a fixed distance. As shown in Figure 3.3 (a), the initial follower error is approximately 3 meters due to these initial

conditions, but it is compensated within about 10 seconds. The instantaneous position error of follower is computed as the Euclidean distance between its reference position $(x_{\text{ref}},\, y_{\text{ref}})$ and actual output position $(x_{\text{out}},\, y_{\text{out}})$, as defined in equation 3.19. As shown in Figure 3.3 (b) which is a zoomed view of Figure 3.3 (a) , even in the absence of packet loss, a small error remains due to effect of discretization error, as followers receive position updates every 50 ms (the communication interval) instead of every 5 ms (the system sampling time). In the absence of packet loss, the error remains below 5 cm after the transient response. However, when packet loss is introduced, intermittent error spikes appear due to missing leader-position updates. As shown in the zoomed view in Figure 3.3 (b), these spikes reach magnitudes of approximately 10-20 cm, indicating temporary deviations from the ideal formation spacing whenever a packet is dropped. Outside these short intervals, the mean tracking accuracy remains close to the without loss case, but the peak error clearly increases as the packet loss rate grows.

The effect of packet loss on the follower tracking error is illustrated in Figure 3.3. Packet loss is modeled as a Bernoulli process with rates of 30% and 50%, affecting the leader–follower communication during circular trajectory tracking. Both followers start with an initial offset relative to the leader and are required to converge to the fixed separation distance of approximately $1.42\,\text{m}$, indicated by the dashed line in the figure.

As shown in Figure 3.3(a), the initial error is approximately $3\,\text{m}$, but it is compensated within about $10\,\text{s}$ as the formation stabilizes. The instantaneous position error is computed as the Euclidean distance between the follower's reference position $(x_{\text{ref}},\, y_{\text{ref}})$ and its actual position $(x_{\text{out}},\, y_{\text{out}})$, as defined in (3.19). The zoomed view in Figure 3.3(b) highlights the steady-state behavior. Even without packet loss, a small residual error is observed due to the communication–sampling mismatch: position data are exchanged every $50\,\text{ms}$, whereas the system is updated every $5\,\text{ms}$. Under this nominal condition, the error remains below $5\,\text{cm}$.

When packet loss is introduced, intermittent error spikes appear whenever consecutive leader-state updates are dropped. As shown in Figure 3.3(b), these peaks reach approximately $10{-}20\,\text{cm}$ (i.e., $0.1{-}0.2\,\text{m}$ shown in figures), depending on the loss rate. Although the overall tracking accuracy as shown in Figure 3.3(a) remains close to the no-loss case, the maximum deviation increases noticeably with packet loss, revealing that peak error is the key risk factor for

formation instability.

$$E_{\text{follower}}(t) = \sqrt{\left(x_{\text{ref}}(t) - x_{\text{out}}(t)\right)^2 + \left(y_{\text{ref}}(t) - y_{\text{out}}(t)\right)^2} \tag{3.19}$$

To simulate packet loss compensation, MCP, a memory is considered in the follower robot that stores the most recent data received during packet loss. As a result, when a packet is lost, the system applies the most recently received data. The hypothesis is that the LSTM prediction model will alleviate the impact of packet loss more effectively than MCP.

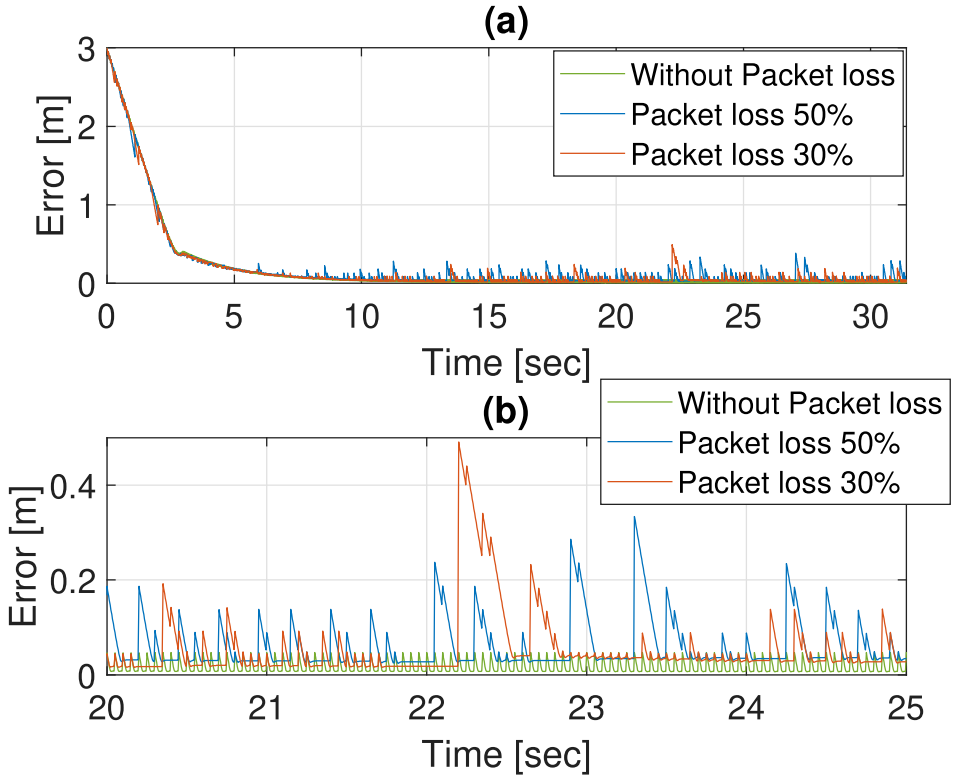


Figure 3.3: Effect of packet loss (a) Error of follower ; (b) Zoomed view of follower error

3.4.2 Long Short-Term Memory to cope with packet loss

In this section, an LSTM model is combined with the LQR controller to predict the position of the leader when packet loss occurs. The objective is to enhance the system's basic controller using LSTM rather than designing the controller from scratch. We augment existing linear (low-complexity) controllers with an LSTM

predictor to be more robust against packet loss. The LSTM exploits temporal dependencies to produce one-step-ahead state estimates, preserving tracking accuracy without replacing the baseline controller while it offers cost and time-efficient approach for industrial deployment. When it comes to time series data prediction, the results of LSTM demonstrated a significant ability to predict [AdLF17]. As we are attempting to predict the leader's trajectory, which is a time-based order sequence, this problem fits within the LSTM framework. LSTM has been addressed in numerous articles [ZZD20] in order to learn and remember long-term dependency and information. Indeed, by incorporating various gates, such as an input gate, an output gate, and a forget gate, LSTM is proposed to improve traditional recurrent neural networks (RNNs). These various gates enable LSTM to achieve a trade-off between the current and previous inputs while alleviating RNN's vanishing gradient and exploding gradient problems [ZZD20].

When it comes to time series data prediction, the results of LSTM demonstrated a significant ability to predict [AdLF17]. As we are attempting to predict the leader's trajectory, which is a time-based order sequence, this problem fits within the LSTM framework.

3.4.2.1 Architecture of LSTM prediction and control

The LSTM architecture adopted in this work follows a single-layer regression design commonly used in time-series forecasting. It consists of an input layer, a single LSTM layer, a fully connected (dense) layer, and a regression output layer. Three independent LSTM networks are trained — one for each coordinate component (x, y, and $theta$) of the leader's position. Each network acts as a sequence-to-one regression that predicts the next value of a single coordinate based on a fixed number of its past observations. Each network receives the last 10 samples of the corresponding coordinate as input, where the LSTM layer extracts temporal dependencies, the dense layer maps hidden representations, and the output layer produces the next-step prediction. This configuration provides a balance between prediction accuracy and computational efficiency suitable for real-time embedded implementation.

The input layer consists of the position of the leader, which is determined using Simulink simulation results with an existing controller. The hidden layer is in charge of remembering and storing the previous state's number of nodes. The output layer exports the leader robot's predicted position. Three of these LSTM

neural network models are used independently to predict the leader's x, y, and heading angle positions.

As illustrated in Figure 3.4, LSTM is equipped with a "gate" structure that enables it to add or remove cell state information and selectively pass information while following these steps [ZZD20]:

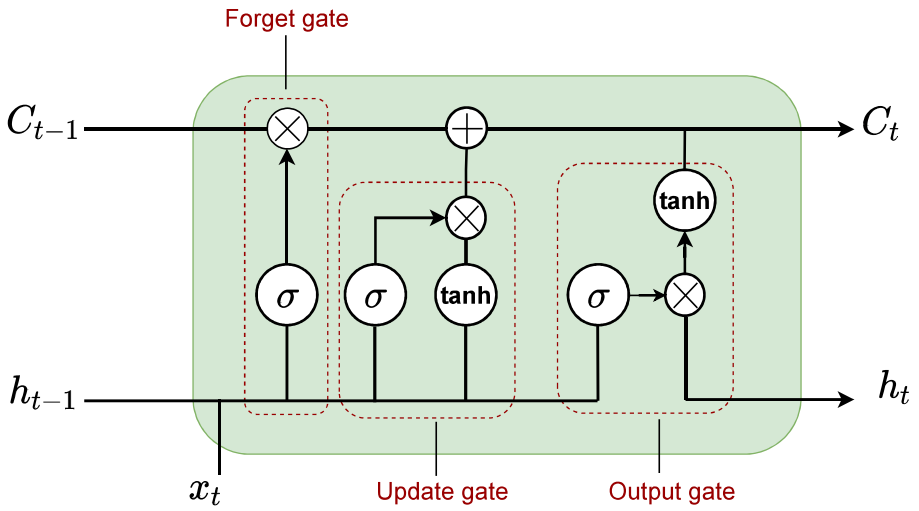


Figure 3.4: LSTM architecture

- **Forget gate f_t:** decide what information should be discarded from the previous cell state C_{t-1}.

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{3.20}$$

where f_t is the forget gate, σ is the sigmoid function, W_f is the weight matrix, b_f is the bias term, h_{t-1} is the previous hidden layer output, and x_t is the new input.

- **Updating information:** determining what information input should be stored in the cell state of the input gate, where the input information should be updated.

$$\begin{aligned} i_t &= \sigma(W_i.[h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= \tanh\left(W_C.[h_{t-1}, x_t] + b_C\right) \end{aligned} \tag{3.21}$$

where i_t is the input gate, $\tilde{C}_t$ is the candidate state of the input, σ and $\tanh$

are the sigmoid and hyperbolic tangent functions, respectively. W_i and W_C are the weight matrices, b_i and b_C are the bias terms, h_{t-1} is the previous hidden layer output, and x_t is the new input. The first equation, including σ identifies which value is to be updated, and the second equation, including $\tanh$ will generate new candidate values.

- **Updating the cell status:** updating the cell state of the present time step, which combines the new candidate memory and the long-term memory.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{3.22}$$

where C_t and C_{t-1} are the current and previous memory states, f_t is the forget gate, i_t is the input gate, $\tilde{C}_t$ is the input candidate state. State of the old cell is updated via multiplying the old state by the information that has been discarded and adding new candidate values.

- **Determining Output:** Confirming the outcome in the output gate where the output information and the cell state are used.

$$\begin{aligned} o_t &= \sigma(W_o.[h_{t-1}, x_t] + b_o) \\ h_t &= o_t * \tanh C_t \end{aligned} \tag{3.23}$$

where o_t is the output gate. W_o and b_o are the weight matrix and bias terms, respectively. h_{t-1} and h_t are the previous and current hidden layer outputs, x_t is the new input, and C_t is the current state of the memory block. The first equation, which includes σ, determines which part of the cell state will be output (o_t), and the second equation processes the cell state by $\tanh$ multiplied by the output of sigmoid layer.

Both activation functions increase the nonlinearity of the network, which are sigmoid and tanh. The sigmoid function converts values between 0 and 1, while the hyperbolic tangent function squeezes the value between –1 and 1 [SLX+20].

$$\begin{aligned} \sigma(x) &= \frac{1}{1+e^{-x}} \\ \tanh x &= \frac{e^x - e^{-x}}{e^x + e^{-x}} \end{aligned} \tag{3.24}$$

3.4.2.2 Dataset preparation and splitting

The dataset used to train the LSTM predictor was generated from leader trajectories simulated in MATLAB/Simulink using the LQR controller under perfect communication (no packet loss). Each trajectory contains position and orientation samples at a communication rate of 50 ms, covering various motion profiles including circular paths and eight-shape paths.

To ensure diversity in the training data, a total of 8 trajectories were randomly generated—four circular, four eight-shaped. For each circular and eight-shaped path, the center coordinates and radii were randomly selected within predefined workspace limits (x,y limited to -10,10 for the center and 1–6 m for the radius), while the linear speed was fixed at 1 m/s to maintain consistent motion dynamics across all samples. These trajectories were simulated using the same control parameters to reflect typical leader–follower motion under ideal communication.

All data were normalized using z-score normalization to improve convergence. To ensure fair evaluation, out of the eight trajectories, five were used for training, one for validation, and two (one circular and one eight-shaped) for testing. These test trajectories were unseen during training and used to evaluate the prediction performance of the LSTM model under simulated packet-loss conditions.

3.4.2.3 Application of LSTM for leader position prediction

As previously stated, follower robots should have access to the latest leader's position in order to maintain an accurate formation control. When packet loss occurs, followers lose access to the leader's true position. To address this, an LSTM network is trained using the leader trajectory and then its states are updated. As shown in Figure 3.5 When no packets are lost, network states are updated with the actual observed leader position received through the communication link.

In real communication systems, the receiver cannot directly determine whether a packet was lost or merely delayed—it can only detect that no new data have arrived within the expected communication interval. In this work, a packet is therefore assumed to be lost if it has not been received within 50 ms (the nominal communication period). In such cases, the system updates the LSTM using its internal state and predicted output. The predicted leader position generated by the LSTM is then used as the follower's reference until the next valid packet is received.

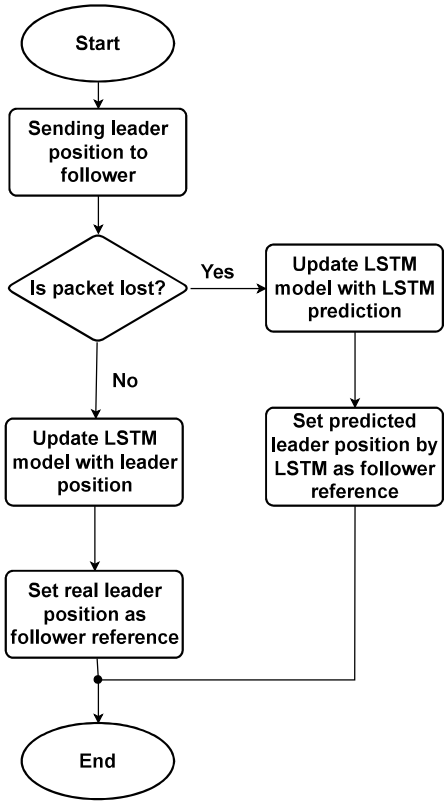


Figure 3.5: The flowchart of the proposed approach of sending the leader position to the follower.

3.5 Performance Evaluation

In this section, the performance of different prediction schemes, including LSTM and GRU, is evaluated. GRU networks share similarities with LSTM but have a simpler architecture, making them computationally lighter. Instead of the three gates used in LSTM, GRUs employ a reset gate and an update gate, both computed using fully connected layers. The reset gate controls how much of the previous state is retained, while the update gate determines how much past information is passed to the next state [CCG23]. GRUs are generally faster and easier to train than LSTMs, offering similar performance for certain tasks, though they may be less effective in handling long-term dependencies. For this reason, GRU is considered for evaluation in this study.

Furthermore, the performance of the formation control system is evaluated using different prediction methods in scenarios with varying levels of packet loss. All performance evaluations are conducted through MATLAB/SIMULINK simulations, utilizing the non-holonomic model (Equation 3.1) and the LQR

controller (Equation 3.9). It is assumed that the robots follow the commanded trajectories precisely, without accounting for skidding or disturbances. This assumption is made to isolate and analyze the specific impact of packet loss on the system's performance.

LSTM and GRU hyperparameters were selected using Bayesian optimization within the following search ranges: hidden units $\in [5, 6, 7, \ldots, 15]$, learning rate $\in [0.001, 0.005, 0.01, 0.1]$, batch size $\in \{8, 16, 32, 64, 128, 256\}$. The Adam optimizer was used in all training cases due to its stable convergence and good performance on nonlinear time-series data. The learning rate was adaptively reduced using a decay factor of 0.2 whenever the validation loss failed to improve. The number of training epochs was manually fixed at 400 for LSTM and 600 for GRU after preliminary testing to ensure convergence. LSTM and GRU hyperparameters final values used for each output component (x, y, and θ) are summarized in Table 3.1 and 3.2, respectively.

Table 3.1: LSTM model setting.

LSTM	**X**	**Y**	**Heading Angle**
Epochs	400	400	400
Hidden nodes	10	10	7
Batch size	128	128	128
Learning rate	0.005	0.01	0.1
Learning rate decay factor	0.2	0.2	0.2

Table 3.2: GRU model setting.

GRU	**X**	**Y**	**Heading Angle**
Epochs	600	600	600
Hidden nodes	15	12	8
Batch size	256	128	256
Learning rate	0.005	0.01	0.1
Learning rate decay factor	0.2	0.2	0.2

In our use case, the follower robot was expected to maintain a predefined distance from the leader. The accuracy of the proposed prediction methods was evaluated using two standard metrics: the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE). RMSE measures the square root of the average of the squared differences between the predicted and the actual values, while MAE represents the average of the absolute differences between them. RMSE and MAE are defined in Equations 3.25 and 3.26, respectively,

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i))^2} \tag{3.25}$$

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|x_i - \hat{x}_i| \tag{3.26}$$

where x_i is the observed value, $\hat{x}_i$ is the predicted value, and N is the number of data points.

3.5.1 Prediction accuracy for LSTM and GRU

As mentioned the LSTM and GRU model were trained on leader trajectories using the parameters listed in Tables 3.1 and 3.2. They were then tested with a separate set of leader trajectories, using 20% subset of the leader trajectories as the test set, while packet loss was simulated at 30% and 50%.

To provide a fairer baseline for comparison, LIP was included alongside the previously used MCP. The LIP estimates missing data points by fitting a linear trend to the nine most recent successfully received samples and extrapolating one step ahead. Unlike MCP (equivalent to a Zero-Order Hold), which simply repeats the last available value, LIP leverages short-term motion history to infer the current trajectory trend, resulting in smoother and more accurate short-term predictions under packet loss.

During each test, missing data segments caused by packet loss were reconstructed using four methods: MCP, LIP, GRU, and LSTM. The predicted values were then compared against the actual leader trajectory to evaluate accuracy. Figure 3.6 shows the prediction errors in the X, Y, and heading angle for the circular path. The LSTM and GRU models consistently yield smaller error magnitudes compared to the baseline predictors (MCP and LIP). MCP, which simply repeats the last received sample, produces the largest deviations, especially during consecutive packet losses. LIP, which estimates the missing data through linear fitting over the last nine received points, provides smoother and more accurate results than MCP, yet still performs worse than the learning-based methods. Between the two recurrent networks, LSTM demonstrates superior stability and smaller fluctuations, particularly in the heading angle, where GRU tends to accumulate error over time.

Figure 3.7 illustrates the training and validation losses of the LSTM model for

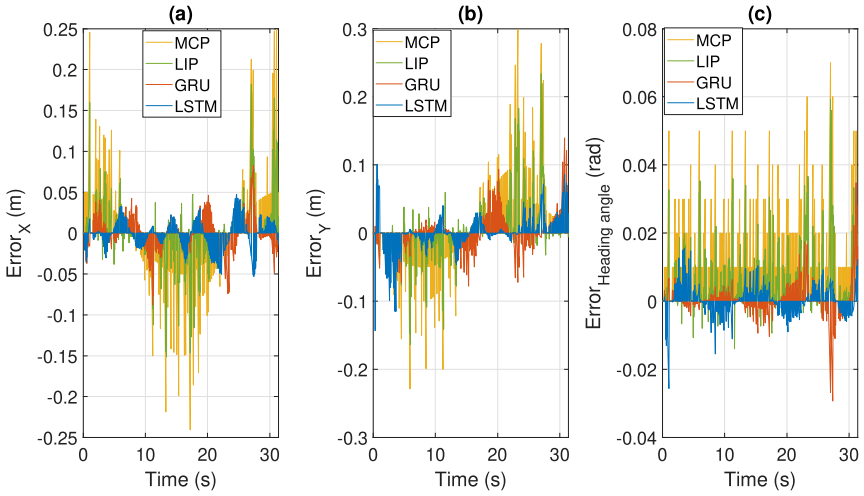


Figure 3.6: Prediction errors. (a) Error for X; (b) Error for Y; (c) Error for heading angle

the three predicted outputs X, Y, and heading angle. The loss curves show rapid convergence within the first 50 epochs, followed by stabilization without overfitting, confirming the effectiveness of the selected hyperparameters and the Adam optimizer.

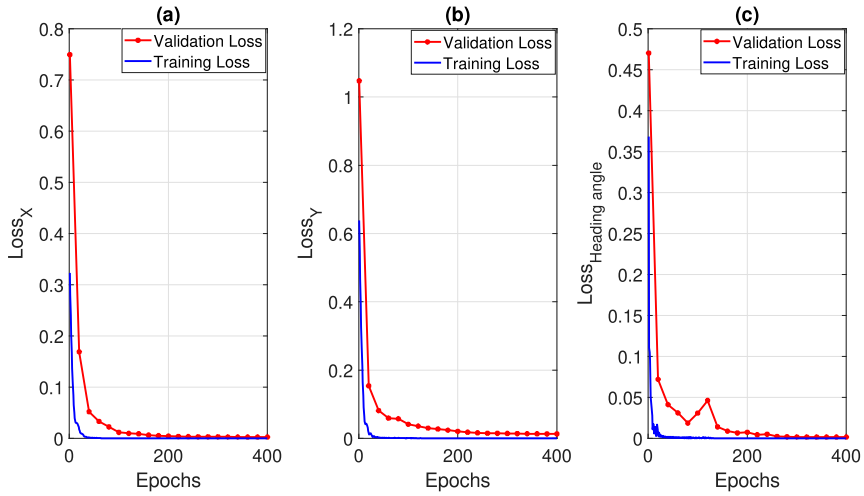


Figure 3.7: Training and validation loss. (a) Loss for X; (b) Loss for Y; (c) Loss for heading angle

Quantitative results summarized in Table 3.3 further validate these observations. Under both 30% and 50% packet loss, the LSTM model achieves the lowest RMSE and MAE values across all predicted variables, followed closely by GRU. The LIP method performs better than MCP, as it leverages multiple past samples to approximate local trajectory trends, thereby generating smoother predictions

during communication loss. In contrast, MCP exhibits the weakest performance overall since it relies on a static assumption and cannot capture dynamic changes between communication intervals.

Overall, the LSTM-based predictor demonstrates a substantial reduction in prediction error compared to both LIP and MCP across all packet loss conditions. On average, the LSTM reduces prediction errors by approximately $35{-}60\%$ relative to LIP and $60{-}80\%$ relative to MCP, for both position and heading estimates. These findings indicate that the LSTM architecture is significantly more robust to communication loss, as it effectively exploits long-term temporal dependencies in the data rather than relying on linear interpolation or static model assumptions.

Table 3.3: Prediction error under packet loss.

Metric	Loss	Method	X (m)	Y (m)	Heading (rad)
RMSE	30%	LSTM	1.01×10^{-2}	2.39×10^{-2}	2.70×10^{-3}
		GRU	1.06×10^{-2}	1.28×10^{-2}	3.50×10^{-3}
		LIP	1.94×10^{-2}	3.13×10^{-2}	7.33×10^{-3}
		MCP	3.47×10^{-2}	4.28×10^{-2}	1.10×10^{-2}
	50%	LSTM	1.49×10^{-2}	2.35×10^{-2}	3.90×10^{-3}
		GRU	1.87×10^{-2}	2.38×10^{-2}	4.80×10^{-3}
		LIP	3.44×10^{-2}	3.77×10^{-2}	1.02×10^{-2}
		MCP	5.53×10^{-2}	5.70×10^{-2}	1.59×10^{-2}
MAE	30%	LSTM	4.19×10^{-3}	6.24×10^{-3}	1.11×10^{-3}
		GRU	4.45×10^{-3}	5.46×10^{-3}	1.12×10^{-3}
		LIP	6.47×10^{-3}	7.66×10^{-3}	2.08×10^{-3}
		MCP	1.47×10^{-2}	1.60×10^{-2}	4.80×10^{-3}
	50%	LSTM	8.19×10^{-3}	1.13×10^{-2}	2.09×10^{-3}
		GRU	1.01×10^{-2}	1.23×10^{-2}	2.04×10^{-3}
		LIP	1.63×10^{-2}	1.74×10^{-2}	5.22×10^{-3}
		MCP	2.96×10^{-2}	3.03×10^{-2}	9.42×10^{-3}

In summary, four prediction methods MCP, LIP, GRU, and LSTM were evaluated to assess their effectiveness under varying levels of packet loss. The LSTM consistently achieved the highest prediction accuracy and generalization capability across all scenarios. As a result, the LSTM model was selected as the primary predictor for subsequent formation control simulations, while MCP and LIP were retained as baseline references. To avoid figure overcrowding in subsequent

simulations and because of its lower overall performance relative to LSTM, the GRU model was excluded from further analysis.

3.5.2 Simulation Results

Here, we discuss the formation control performance of three robots with one of them acting as leader. For each robot, the controller diagram depicted in Figure 3.2 was simulated using MATLAB/SIMULINK. The followers and the leader attempted to maintain their formation as they travelled along a pre-defined path. The speed of robot is considered 1 m/s. At regular communication intervals, the leader's position was communicated to all the followers via wireless broadcast communication. In the event of packet loss, LSTM predicted the leader's position for the followers. In this section, we evaluate and analyze the performance of LSTM within a closed-loop system, rather than simply comparing the predicted values to the actual values as done in Section 3.5.1. As shown in Figure 3.8, the leader's position is transmitted to the followers using the proposed LSTM architecture, highlighted with a grey background. Yellow switches determine whether to use the predicted data or real data, depending on whether packet loss has occurred. The impact of the proposed architecture is evaluated within the closed-loop system, as the predicted or real data serves as the reference for followers 1 and 2. Both the leader and followers are controlled using the LQR controller, as detailed in Section 3.3. All experimental settings are consistent with those outlined in Table 3.1.

The local control loop of each robot operates at a sampling time of 5 ms, ensuring internal stability of the individual agent. Communication among agents, however, occurs at a slower interval of 50 ms. While this does not affect the local stability guaranteed by the LQR controller, it significantly influences the formation stability, as delayed or lost information can cause temporary desynchronization between agents. Therefore, compensating for packet loss or prediction at the communication level is essential for maintaining accurate formation. When a communication packet was lost at a 50 ms interval, the corresponding predictor (LSTM, LIP, or MCP) estimated the missing leader state for that communication cycle, ensuring continuous control updates despite packet loss. Packet loss is was modelled using a Bernoulli distribution with a probability equal to 0.3 and 0.5. Packet loss has a more significant effect on formation control when the leader follows a nonlinear trajectory. Therefore, instead of using straight-line or quasi-linear trajectories or gently curved paths,

circular and eight-shaped trajectories were selected. These provide richer motion dynamics with changes in curvature and direction, making them more suitable for evaluating prediction performance and controller robustness under packet loss.

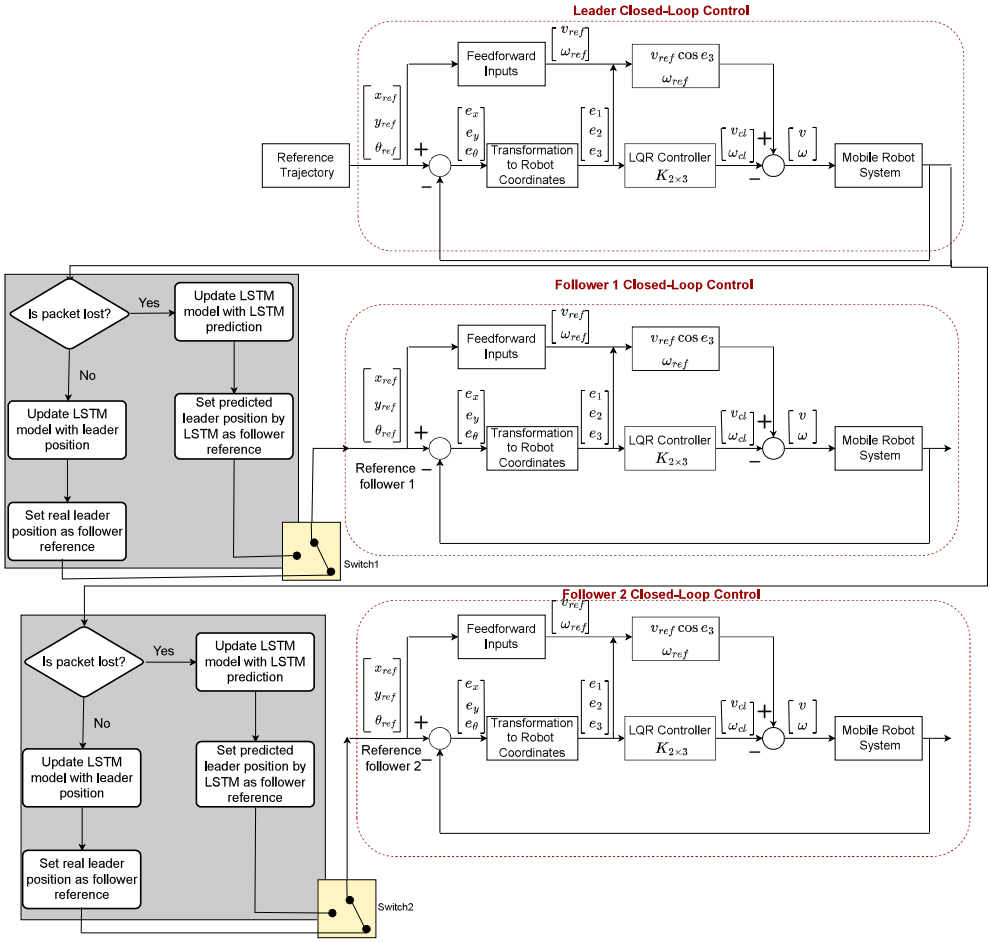


Figure 3.8: Closed-loop control of leaders and two followers under packet loss using LSTM prediction

3.5.2.1 Circular Path

In this scenario, we consider the leader and followers moving along a circular path, with packet loss set at 30%. The leader's initial position is (0, -5), while the followers start at (2, -4) and (2, -6), meaning they begin with some offset and are about 2 meters behind the leader along the x-axis. This test evaluates not only the trajectory following but also the system's performance given the initial conditions. Since the leader and followers are not synchronized from the

beginning, the followers must adjust to compensate for this and keep the fixed distance with leader. The formation distance between the leader and follower 1 is (-1, 1), and between the leader and follower 2 is (-1, -1).

Figure 3.9 (a) shows that all followers successfully maintain a circular formation despite packet loss. The zoomed-in view in Figure 3.9 (b) highlights that the LSTM predictor achieves the closest overlap with the trajectory without packet loss, while MCP and LIP show larger deviations.

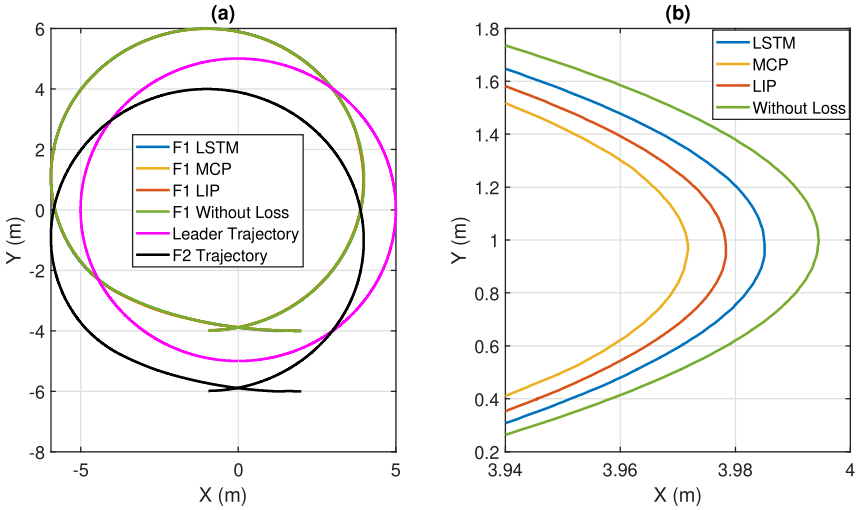


Figure 3.9: Trajectory of the robots during formation control (packet loss—30% compared with no packet loss) (a) Leader and Followers circular trajectory ; (b) Zoomed view of Follower 1 circular trajectory, highlighting the overlap of LSTM, MCP, and no-packet-loss paths for better clarity.

Figure 3.10 and Figure 3.11 illustrate the distance of first and second followers from the leader, where both followers gradually reach the desired separation after approximately 5 s and 8 s, respectively. LSTM produces smoother convergence and smaller oscillations due to its ability to capture temporal dependencies, while MCP's zero-order hold behavior leads to delay accumulation and overshoot. LIP performs slightly better than MCP by following short-term trends but is still limited under high and consecutive loss when motion becomes nonlinear.

We repeated the circular trajectory scenario with a 50% packet loss. The follower performance is illustrated in Figure 3.12. Figure 3.12 (a) shows the trajectories of the leader and followers, demonstrating that all methods maintain general formation, though noticeable deviations appear as losses increase. To illustrate this effect more clearly, a zoomed-in view in Figure Figure 3.12 (b) compares the predicted follower trajectories obtained using LSTM, LIP, and MCP against

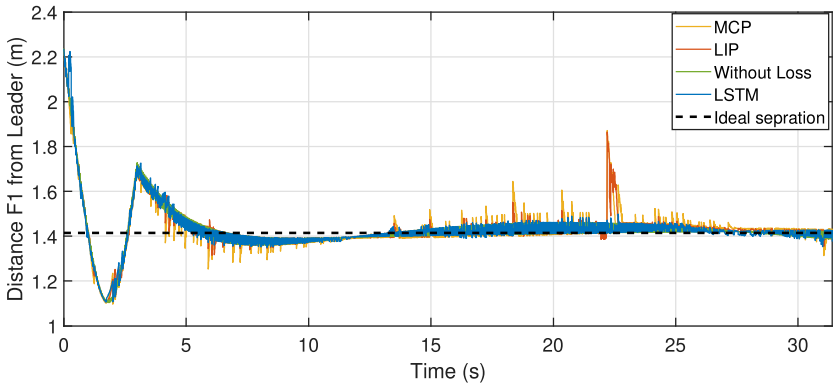


Figure 3.10: Distance of the first follower from the leader for the circular trajectory (packet loss—30% compared with no packet loss).

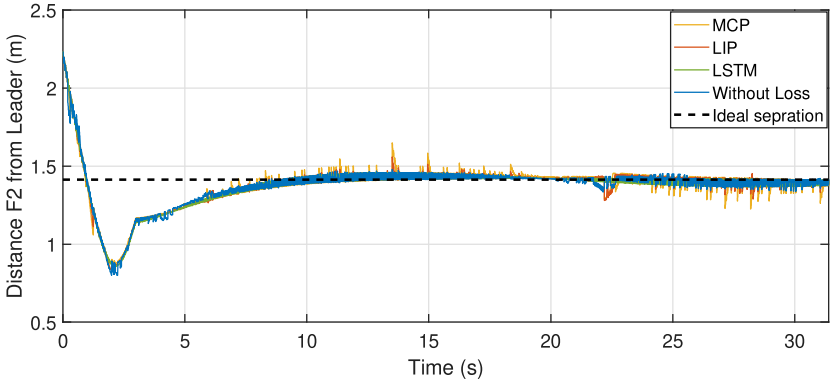


Figure 3.11: Distance of the second follower from the leader for the circular trajectory (packet loss—30%).

the ideal without loss reference. The LSTM predictor continues to provide the closest trajectory to the reference, whereas LIP exhibits minor oscillations, and MCP visibly drifts from the formation.

The corresponding distance responses between the leader and the two followers are shown in Figures 3.13 and 3.14, further confirming the superior tracking stability achieved with the LSTM-based predictor. Both followers start with an initial offset relative to the leader and are required to converge to the fixed separation distance of approximately $1.42\,\mathrm{m}$, indicated by the dashed line in the figures. After the transient phase (approximately 5 s for Follower 1 and 8 s for Follower 2), the LSTM trajectory remains tightly bounded around the ideal separation, with deviations typically about 5 cm. In contrast, both LIP and MCP exhibit larger fluctuations, with peak deviations reaching approximately 10 cm and 20 cm respectively (fluctuations go up to 1.5 and 1.6 relative to ideal

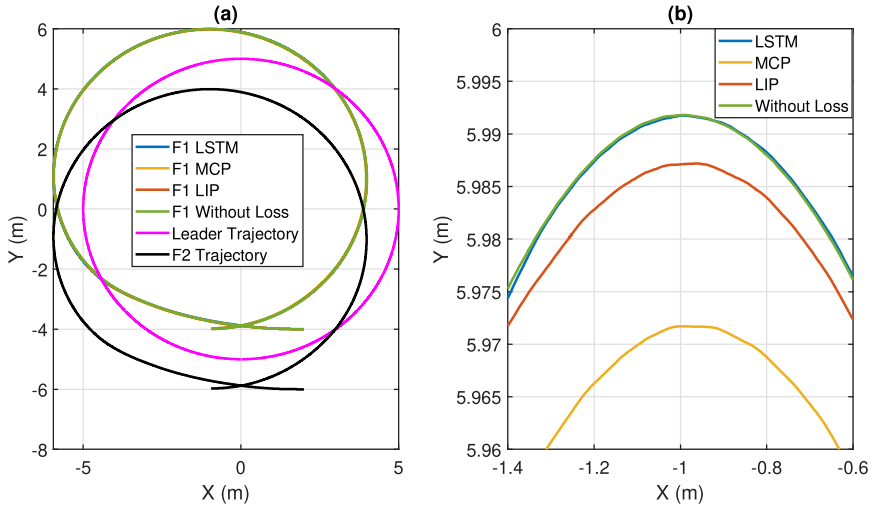


Figure 3.12: Trajectory of the robots during formation control (packet loss—50% compared with no packet loss) (a) Leader and Followers circular trajectory ; (b) Zoomed view of Follower 1 circular trajectory, highlighting the overlap of LSTM, MCP, and no-packet-loss paths for better clarity.

distance 1.42 as shown in Figures 3.13 and 3.14) during consecutive packet losses.

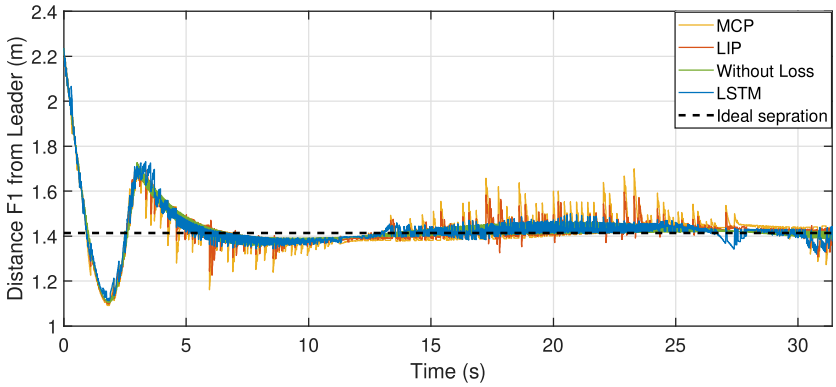


Figure 3.13: Distance of the first follower from the leader for the circular trajectory (packet loss—50%).

These transient peaks are particularly relevant in formation control, since average error values alone may conceal short but significant deviations that can destabilise inter-robot spacing. The LSTM predictor effectively suppresses both steady-state error and peak deviations, maintaining the closest adherence to the desired formation distance.

Quantitatively, Table 3.4 and Table 3.5 show RMSE and MAE results for the follower tracking error. These results show that the LSTM-based predictor

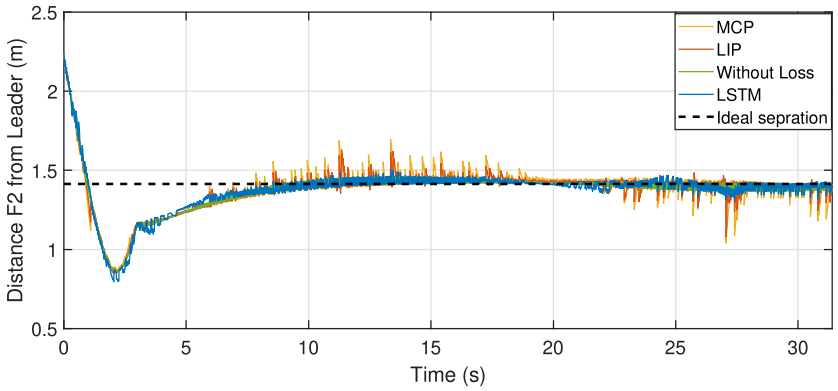


Figure 3.14: Distance of the second follower from the leader for the circular trajectory (packet loss—50%).

achieves the lowest RMSE and MAE values under both 30% and 50% packet loss. LSTM provides a consistent improvement of approximately $5{-}10\%$ in RMSE and up to 20% in MAE compared to LIP. In contrast, the MCP method exhibits noticeably larger tracking errors, with LSTM reducing RMSE by $10{-}18\%$ and MAE by $15{-}40\%$ relative to MCP, depending on the axis. These results confirm that even after closed-loop feedback, the advantage of LSTM persists, whereas MCP remains the least effective due to its inability to account for inter-sample dynamics.

Table 3.4: RMSE of follower error for the circle path.

Packet Loss	Prediction Method	RMSE-X(m)	RMSE-Y(m)	RMSE-Angle(rad)
0%	-	5.371×10^{-1}	8.62×10^{-2}	3.66×10^{-2}
30%	LSTM	5.372×10^{-1}	8.7×10^{-2}	3.73×10^{-2}
	LIP	5.375×10^{-1}	9.31×10^{-2}	3.86×10^{-2}
	MCP	5.39×10^{-1}	9.74×10^{-2}	4.16×10^{-2}
50%	LSTM	5.375×10^{-1}	8.82×10^{-2}	3.87×10^{-2}
	LIP	5.39×10^{-1}	9.43×10^{-2}	4.11×10^{-2}
	MCP	5.42×10^{-1}	1.07×10^{-1}	4.69×10^{-2}

3.5.2.2 Eight-Shaped Trajectory

Here, the formation control performance of the leader and the followers are analyzed while following an eight-shaped trajectory in different packet loss scenarios. The eight-shaped trajectory provides a more complex nonlinear motion pattern than the circular path, The leader's initial position is (0, 0), while the followers start at (-3, 2) and (-3, -2), meaning they begin with some offset

Table 3.5: MAE of follower error for the circle path.

Packet Loss	Prediction Method	MAE-X(m)	MAE-Y(m)	MAE-Angle(rad)
0%	-	1.74×10^{-1}	4.80×10^{-2}	2.30×10^{-2}
30%	LSTM	1.75×10^{-1}	4.83×10^{-2}	2.33×10^{-2}
	LIP	1.79×10^{-1}	5.28×10^{-2}	2.55×10^{-2}
	MCP	1.91×10^{-1}	6.29×10^{-2}	3.03×10^{-2}
50%	LSTM	1.77×10^{-1}	4.94×10^{-2}	2.35×10^{-2}
	LIP	1.9×10^{-1}	6.23×10^{-2}	2.98×10^{-2}
	MCP	2.10×10^{-1}	8.01×10^{-2}	3.79×10^{-2}

and are about 2 meters behind the leader along the x-axis. This test evaluates not only the eight shape trajectory following but also the system's performance given the initial conditions. Since the leader and followers are not synchronized from the beginning, the followers must adjust to compensate for this while keeping the fixed distance with leader. The formation distance between the leader and follower 1 is (-1, 2), and between the leader and follower 2 is (-1, -2).

At 30% packet loss, Figure 3.15 (a) shows that all prediction schemes preserved formation, but with noticeable deviations during the curved transitions. The zoomed section in Figure 3.15 (b) illustrates that the LSTM trajectory remains almost indistinguishable from the without loss case, while LIP performs slightly worse but still tracks the motion trend effectively. MCP, by contrast, drifts farther because it only repeats the last received state when packets are dropped.

The distance profiles in Figures 3.16 and 3.17 demonstrate that both followers start with an initial offset greater than 1 m and recover to their ideal separation within 8 s. Ideal distance is shown by dashed line and is about $2.23\,\mathrm{m}$ in figures. LSTM prediction yields the smoothest recovery and smallest steady-state oscillation. LIP shows an intermediate performance: it converges to the correct distance but exhibits minor oscillations while MCP suffers from larger deviations and slower stabilization.

With half of the packets lost (50% loss), the eight-shaped trajectory further challenges prediction accuracy. In Figure 3.18 (a), all agents still complete the formation successfully, though trajectory distortions increase. The zoomed view in Figure 3.18 (b) shows that LSTM remains closest to the without loss reference, LIP begins to lag slightly, and MCP deviates significantly.

In the distance responses (Figures 3.19, 3.20), LSTM maintains a smooth and stable convergence to the ideal separation, with minimal oscillation about 5 to 8

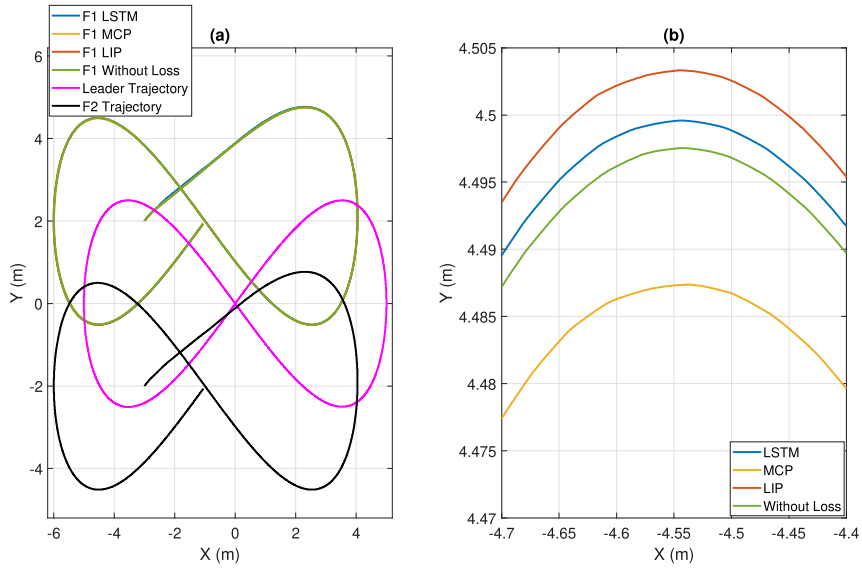


Figure 3.15: Trajectory of the robots during formation control (packet loss—30% compared with no packet loss) (a) Leader and Followers eight shape trajectory ; (b) zoomed view of Follower 1 eight shape trajectory, highlighting the overlap of LSTM, MCP, and no-packet-loss paths for better clarity.

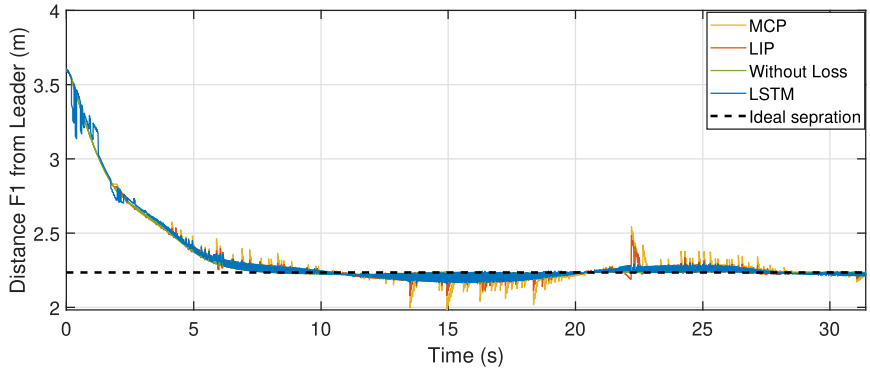


Figure 3.16: Distance of the follower from the leader for the circular trajectory (packet loss—30%).

cm. LIP continues to reduce the separation error but shows transient overshoots about 10 to 20 cm (peaks in figures goes up to 2.35 to 2.40 relative to ideal separation which is 2.23). MCP again exhibits the highest fluctuations about 20 cm to 30 cm (peaks in figures goes up to 2.45 to 2.50 relative to ideal separation which is 2.23) and slower settling.

The quantitative results of Tables 3.6 and 3.7 show that the LSTM predictor consistently achieves the lowest tracking error under both 30% and 50% packet loss. Compared to LIP, the LSTM yields improvement of approximately $5{-}10\%$

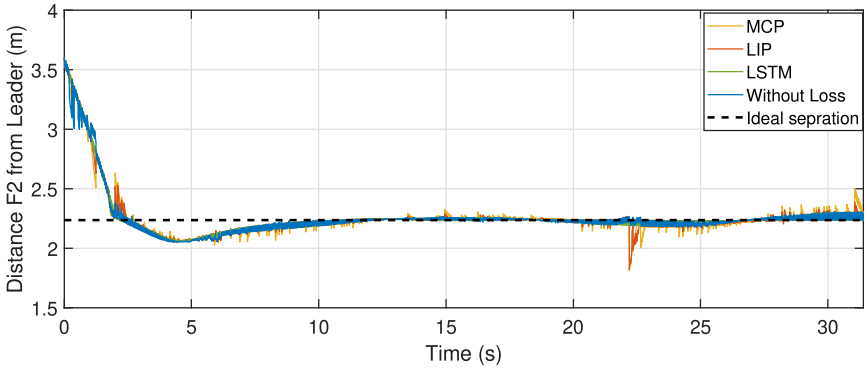


Figure 3.17: Distance of the follower from the leader for the circular trajectory (packet loss—30%).

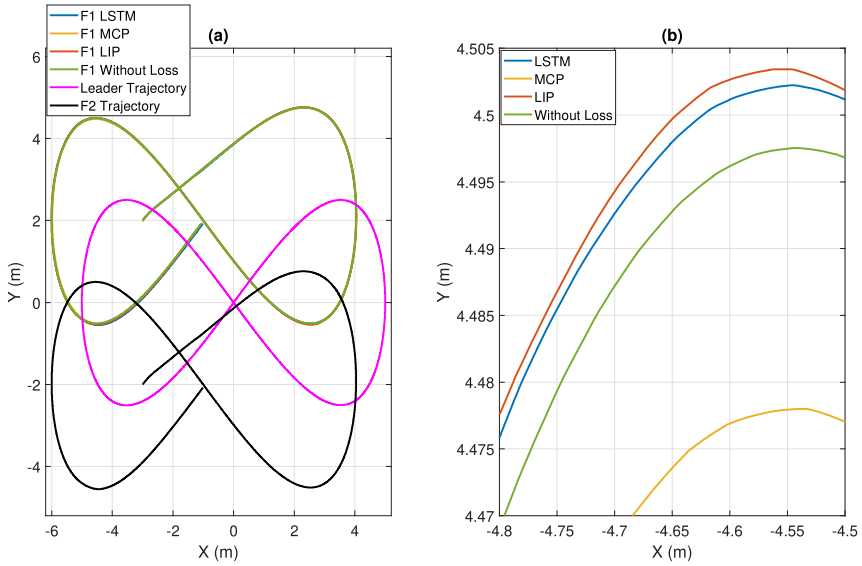


Figure 3.18: Trajectory of the robots during formation control (packet loss—50% compared with no packet loss) (a) Leader and Followers eight shape trajectory ; (b) zoomed view of Follower 1 eight shape trajectory, highlighting the overlap of LSTM, MCP, and no-packet-loss paths for better clarity.

in RMSE and up to 20% in MAE, with the largest gains observed in the lateral and angular components. The performance gap is more pronounced against MCP, where the LSTM reduces RMSE by $8{-}25\%$ and MAE by $20{-}40\%$ depending on the axis. These results confirm that the LSTM-based predictor remains the most robust scheme as packet loss increases. Although LIP performs better than MCP, it still suffers from clear tracking deviations and cannot fully compensate for packet loss.

Overall, the results across both packet-loss conditions demonstrate that LSTM

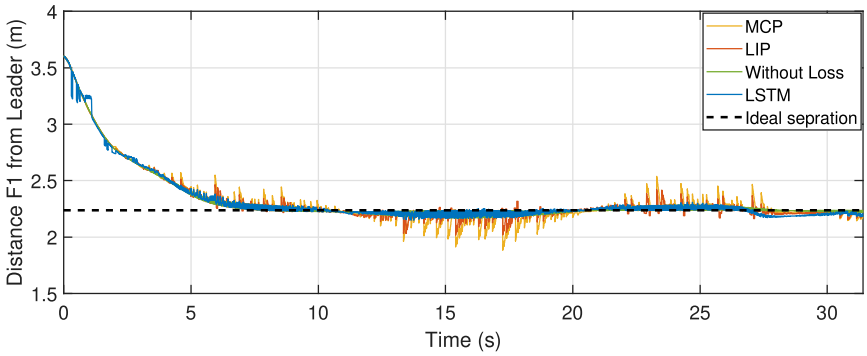


Figure 3.19: Distance of the follower from the leader for the circular trajectory (packet loss—50%).

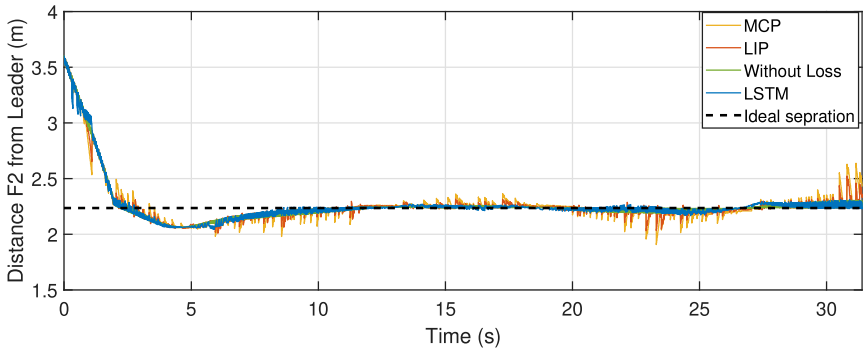


Figure 3.20: Distance of the follower from the leader for the circular trajectory (packet loss—50%).

scales robustly with trajectory complexity. While the simpler circular path mainly tested tracking smoothness, the eight-shaped path revealed LSTM's superior ability to predict nonlinear motion trends and correct for abrupt directional changes, where memory-based and interpolation-based predictors degrade more noticeably.

Table 3.6: RMSE of follower error for the eight-shaped path.

Packet Loss	**Prediction Method**	**RMSE-X (m)**	**RMSE-Y(m)**	**RMSE-Angle(rad)**
0%	-	3.60×10^{-1}	9.78×10^{-2}	5.60×10^{-2}
30%	LSTM	3.62×10^{-1}	1.004×10^{-1}	5.64×10^{-2}
	LIP	3.63×10^{-1}	1.033×10^{-1}	5.74×10^{-2}
	MCP	3.69×10^{-1}	1.13×10^{-1}	6.14×10^{-2}
50%	LSTM	3.64×10^{-1}	1.028×10^{-1}	5.79×10^{-2}
	LIP	3.71×10^{-1}	1.13×10^{-1}	6.29×10^{-2}
	MCP	3.75×10^{-1}	1.35×10^{-1}	7.18×10^{-2}

Table 3.7: MAE of follower error for the eight-shaped path.

Packet Loss	Prediction Method	MAE-X (m)	MAE-Y(m)	MAE-Angle(rad)
0%	-	1.32×10^{-1}	6.26×10^{-2}	2.71×10^{-2}
30%	LSTM	1.33×10^{-1}	6.62×10^{-2}	2.75×10^{-2}
	LIP	1.38×10^{-1}	6.89×10^{-2}	2.91×10^{-2}
	MCP	1.54×10^{-1}	8.45×10^{-2}	3.56×10^{-2}
50%	LSTM	1.35×10^{-1}	6.72×10^{-2}	2.78×10^{-2}
	LIP	1.47×10^{-1}	8.47×10^{-2}	3.45×10^{-2}
	MCP	1.76×10^{-1}	1.12×10^{-1}	4.49×10^{-2}

3.6 Concluding remarks

This chapter considers the formation control problem concerning a network of non-holonomic differential drive robots. Initially, the differential robots are modeled using unicycle model equations. Subsequently, an optimal LQR controller is developed to control a single AGV utilizing this model. Expanding upon this, the focus shifts to multiple AGVs, emphasizing the formation control aspect under conditions of packet loss. Simulations reveal that packet loss leads to an increase in inaccuracy of approximately 20 cm in AGV formation control.

To mitigate this degradation, a Memory Consensus Protocol (MCP) was first implemented, which stores the last received data during packet loss. However, MCP fails to predict motion trends under consecutive dropouts. Therefore, LSTM predictor was proposed to learn temporal dependencies and forecast missing leader states from past data. The LSTM network was trained with optimized hyperparameters to balance accuracy and computational cost. A GRU network was also evaluated for comparison, while the final system integrated the LSTM predictor with the linear LQR controller for closed-loop formation control.

Simulations are then conducted in MATLAB/SIMULINK to evaluate the performance of different prediction schemes (LSTM, LIP, and MCP). Packet loss is modeled using a Bernoulli distribution and tested under circular and eight-shaped paths. In formation control research, it is common to evaluate robustness by testing under varying packet-loss rates and trajectory profiles rather than performing repeated stochastic trials. This study follows the same approach. Future work could extend this analysis by performing more trajectories, multiple randomized trials and applying statistical tests to further quantify the variability

and significance of performance differences.

Simulation results under 30% and 50% packet loss demonstrated that the LSTM predictor significantly improved formation accuracy compared with both MCP and LIP. Across circular and eight-shaped trajectories, LSTM reduced RMSE by approximately 10–15% compared with LIP and 25–35% compared with MCP. MAE was reduced by 15–20% relative to LIP and up to 40–45% compared with MCP. Even with 50% packet loss, the LSTM maintained formation accuracy within 5% of the perfect-communication baseline. These results confirm that integrating a learning-based predictor with a conventional linear controller significantly enhances formation robustness against communication losses.

The prediction and control modules were executed in MATLAB/Simulink with a sampling time of 5 ms and a communication interval of 50 ms. The trained LSTM model required approximately 0.6 ms per prediction step, which is well below the communication period and thus suitable for real-time use, while MCP and LIP computations were negligible ($<$0.1 ms).

It is worth noting that LSTM and GRU training require considerably longer computation times, typically on the order of minutes to hours depending on data size and hyperparameter optimization. However, this process is performed offline, prior to deployment, and therefore does not affect the controller's real-time performance. Once trained, the models can be executed efficiently on standard CPUs without GPU acceleration.

In summary, this chapter validated the effectiveness of combining deep learning prediction with classical control for robust formation under communication uncertainty. This confirms our first hypothesis: "**Integrating deep learning models, such as LSTM, with traditional linear controllers can improve formation control performance in multi-robot systems facing communication challenges like packet loss.**" This hybrid LSTM–LQR approach offers a computationally efficient and scalable alternative to complex nonlinear control redesigns. Future work can include randomized multi-trial testing, statistical validation of significance, and hardware implementation to further confirm real-time performance.

Chapter 4

Formation Control of AGVs under Packet Loss and Delay Compared with Nonlinear Controllers

This chapter investigates the challenges of formation control in non-holonomic Automated Guided Vehicles (AGVs), similar to the discussion in chapter 3. It conducts a comparative examination with nonlinear controllers and explores strategies to mitigate both delay and packet loss. Initially, the motivation and background behind this comparative study are outlined in section 4.1. Following this, section 4.2 provides a brief overview of the controller developed in [NDM+21]. Then, delay is introduced into the simulations in section 4.3 where an input shaping filter is introduced to tackle delay-related issues. Simulations are conducted to evaluate the performance of both the proposed Long Short-Term Memory (LSTM) and the nonlinear controllers. Finally, concluding remarks are presented in Section 4.4.

4.1 Motivation

This chapter addresses the challenge of formation control for multiple non-holonomic AGVs in the presence of packet loss and communication delay. It builds upon the work in chapter 3, where formation control was evaluated under packet loss alone, by introducing delay as an additional network uncertainty. The study employs a decentralized leader-follower approach and investigates whether augmenting linear controllers, such as the Linear Quadratic Regulator

(LQR), with learning algorithms like LSTM can provide robust performance against these uncertainties. Furthermore, the main objective is to highlight the effectiveness of integrating simple linear controllers with learning algorithms as opposed to nonlinear controllers in order to deal with network uncertainties.

As previously mentioned, formation control is vulnerable to network uncertainties like channel noise, communication delay, and packet losses, which can result in complex dynamic behaviors such as divergence, oscillation, and instability [NDM+21]. Consequently, numerous nonlinear controllers have been developed to mitigate these uncertainties. For instance, the Sliding Mode Control (SMC) technique is widely designed as a robust control method for complex higher-order nonlinear systems under parametric uncertainties and external perturbations [GKLP21]. SMC offers stability and robustness in uncertain environments where traditional linear controllers like (Proportional Integral Derivative) PID may falter [GKLP21]. However, despite the robustness of SMC, PID controllers remain prevalent in industry due to their simplicity in design and satisfactory accuracy [NAQ18]. Notably, more than 90% of controllers used in industry are of the PID type [NAQ18, BMSP21]. To enhance the performance of PID controllers in AGVs, various hybrid PID approaches have been developed, including fuzzy PID [LLXL16], adaptive PID [ZCS+12], and optimized PID [MLS+22].

Another important linear controller to consider is the LQR algorithm, which optimizes the cost function to control dynamic systems [LWLZ14, AM18]. LQR is favored for its ability to achieve superior control performance compared to PID at a lower cost [GF21]. However, linear controllers like PID and LQR are more susceptible to network uncertainties such as channel noise, communication delay, and packet losses when extended to multi-AGV systems. Hence, the objective of this chapter is to investigate whether combining existing linear controllers with learning algorithms can effectively address uncertainties like packet loss and delay, rather than developing entirely new nonlinear controllers. This approach aligns with the industry trend of enhancing current controllers rather than creating new ones from scratch. As discussed in Chapter 3, holonomic robot formations exhibit greater resilience to network uncertainties than non-holonomic ones [TAPJ20], emphasizing the significance of validating the proposed method against packet loss and delay.

Given the prevalence of linear controllers in industry, this chapter compares the proposed LSTM-augmented LQR controller against the nonlinear consensus-super-twisting sliding mode controller developed in [NDM+21] and nonlinear

back-stepping controller developed in [WZ15]. This comparison illustrates how each method behaves under identical operating conditions, rather than serving as a strict controller-to-controller benchmark. The reason for choosing [NDM+21] is that it is the closest work we found to our own, as it uses the same application and addresses uncertainties with nonlinear controllers. However, there are key differences, such as their consensus-based approach, event-based control using a super-twisting sliding mode controller, and the consideration of time-varying delays. Despite these differences, packet loss is modeled similarly using a Bernoulli distribution, with a probability of 0.2, providing a useful point of comparison with our approach. Additionally, an input shaping filter is proposed to address delay and enhance robustness against both packet loss and delay to be able to have fair comparison in the same scenario considered in [NDM+21] .

The key contributions of this chapter include:

- Extending the analysis in Chapter 3 via investigating the combined impact of packet loss and delay on formation control, building on the packet loss analysis from Chapter 3.
- Proposing and evaluating an input shaping filter to mitigate delay-induced instability and ensure fair performance comparisons with nonlinear controllers.
- Demonstrating the feasibility and effectiveness of augmenting industry-prevalent linear controllers with LSTM, aligning with the industrial trend of enhancing existing controllers rather than replacing them with nonlinear methods.
- Benchmarking against nonlinear controllers and providing a detailed performance comparison with the nonlinear controller in [NDM+21] and [WZ15], focusing on identical scenario conditions to ensure a fair evaluation.
- Systematic validation through conducting comprehensive experiments to compare LSTM-augmented LQR with nonlinear controllers under varying levels of packet loss and delay, using metrics such as tracking error.

It should be noted that the comparisons in this chapter *are* scenario-based. We reproduce the scenario, initial conditions, desired separations, and packet-loss and delay model reported in [NDM+21, WZ15] to place our method in an identical environment. However, the full implementation details of the

second–order sliding controller are not publicly available, so we do not re-implement it. The resulting comparison is therefore *qualitative and contextual* (same scenario, different controllers), intended to indicate trends rather than a one-to-one algorithmic benchmark.

4.2 Methodology of nonlinear controller for AGVs

A novel stochastic event-based super-twisting controller, introduced in [NDM+21], aims to tackle the formation control challenges within networked multi-agent systems in the presence of uncertainties. This controller, employing event-based consensus super-twisting sliding mode techniques, is tested and validated using Pioneer P3-DX mobile robots.

The dynamics of both leader and follower robots within the multi-agent system framework are subject to network uncertainties and external disturbances, as outlined below:

$$\begin{aligned} Follower : \dot{x_i}(t) &= \rho_i^u u_i(t-\delta_i^u) + d_i(t), i = 1, ..., p \\ Leader : \dot{x_0}(t) &= \rho_0^u u_0(t-\delta_0^u) \end{aligned} \tag{4.1}$$

where $x_i(t) \in R^n$, $u_i(t) \in Rn$, and $d_i(t) \in Rn$ denote the position, control input, and bounded disturbance input, respectively, for the ith follower robot. Additionally, $x_0(t) \in R^n$ and $u_0(t) \in Rn$ represent the leader robot's position and control input, respectively. Parameters ρ_i^u and ρ_0^u represent the random packet loss coefficients for the control input of follower and leader robots, respectively. Similarly, δ_i^u and δ_0^u denote the random delays of the follower and leader robots, respectively. Likewise, ρ_i^x and ρ_0^x signify the random packet loss coefficients, while δ_i^x and δ_0^x represent the random delays of the follower and leader robot states, respectively. The discrete-time formulation of the leader–follower robot dynamics can be simplified as follows for the follower:

$$\begin{aligned} Follower : x_i(k+1) &= x_{i(\delta_i,\rho_i)}(k) + \Delta T u_i(k) + d_i(k) \\ Leader : x_0(k+1) &= x_{0(\delta_0,\rho_0}(k) + \Delta T u_0(k) \end{aligned} \tag{4.2}$$

The symbol ΔT denotes the sampling interval of the system. The states $x_{i(\delta_i,\rho_i)}$ and $x_{0(\delta_0,\rho_0)}$ are due to random delays and packet losses. A stochastic model for 4.2 is defined as follows considering expected value of states due to network uncertainties.

$$\begin{aligned} E\{x_{i,k+1}\} &= E\{x_{i,k}\} + \Delta T u_{i,k} + d_{i,k} \\ E\{x_{0,k+1}\} &= E\{x_{0,k}\} + \Delta T u_{0,k} \end{aligned} \tag{4.3}$$

where $E\{x_{i,k}\}$ represents expected value of $x_{i(\delta_i,\rho_i)}$.

$$\begin{aligned} E\{\tilde{x}_{i,k}\} &= E\{x_{i,k}\} - E\{x_{0,k}\} + \Gamma_i \\ \tilde{u}_{i,k} &= u_{i,k} - u_{0,k} \end{aligned} \tag{4.4}$$

The terms $E\{\tilde{x}_{i,k}\}$ and $\tilde{u}_{i,k}$ represent deviations in the position and control input, respectively, of follower robot i from the leader robot. Γ_i denotes the desired deviation in the position of follower i from the leader. The relative dynamics, utilizing equation 4.3, can be expressed as:

$$E\{\tilde{x}_{i,k+1}\} = E\{\tilde{x}_{i,k}\} + \Delta T \tilde{u}_{i,k} + d_{i,k} \tag{4.5}$$

The sliding-mode surface is formulated for follower agent i to facilitate the controller design as follows:

$$E\{S_{i,k}\} = E\{\tilde{x}_{i,k}\} + E\{q^{\alpha}_{i,k}\}, \quad i = 1, ..., M-1. \tag{4.6}$$

where $S_{i,k} = [s^i_{1,k}, s^i_{2,k}, ..s^i_{n,k}]^T$. With the onset of sliding phase, $Si, k = 0$. To achieve desired consensus, considered a super-twisting control law

$$\tilde{u}_{i,k} = E\{q^{\alpha}_{i,k}\} - K_1 E\{\mid S_{i,k} \mid^{\frac{1}{2}} sgn(S_{i,k})\} + E\{\varrho_{i,k}\} \tag{4.7}$$

with $E\{\varrho_{i,k+1}\} = E\{\varrho_{i,k}\} - \Delta T K_2\{sgn(E\{S_i, k\})\}$

where

$$
\begin{aligned}
E\{q_{i,k}\} = & -\frac{\gamma_i}{m_i+1}\sum_{j\in M} a_{ij}\{(E\{x_{i,k}\} - E\{x_0,k\} + \Gamma_i) \\
& -(E\{x_{j,k}\} - E\{x_{0,k}\} + \Gamma_j)\} \\
& + b_i(E\{x_{i,k}\} - E\{x_{0,k}\} + \Gamma_i) \\
= & -\frac{\gamma_i}{m_i+1}\sum_{j\in M} a_{ij}((E\{\tilde{x}_{i,k}\} - E\{\tilde{x}_j,k\}) + b_i(E\{\tilde{x}_{i,k}\} \qquad (4.8)
\end{aligned}
$$

The consensus component $q_{i,k}$ is defined based on tracking error of follower i and neighboring agents with respect to the leader. In equations 4.7 and 4.8, the parameters $\gamma_i > 0$, $\alpha \in (0,1)$, and $m_i (1 \leq m_i < M)$ denote the count of neighboring agents of i th member. The selection of gain parameters follows the constraint that they should be positive, denoted as $kij > 0 \forall i,j,$ in the absence of system perturbations. However, in the presence of perturbations, maintaining system performance solely through positive gain selections becomes unfeasible. To address this challenge, the gain parameters are chosen in following manner, i.e., $K1 = diag\{k_{11}, k_{12}, ...k_{1n}\} = 1.5\sqrt{\bar{\Omega}}$ and $K2 = diag\{k_{21}, k_{22}, ...k_{2n}\} = 1.1\bar{\Omega}$ where $\bar{\Omega}$ is a limit for uncertainties and its change. For further details regarding the assumptions and the event-triggered control component, please refer to [NDM+21].

The schematic diagram of proposed approach illustrates the control system for a follower robot i, as depicted in Figure 4.1. Sensor data from robot is transmitted to the controller through a wireless communication network prone to packet losses and delay. The consensus component $q_{i,k}^{\alpha}$ is calculated based on position of leader and follower robots, as described in eq. 4.8. The sliding surface is computed using consensus component, and deviation of leader and follower position , as outlined in eq. 4.6. The presence of a Zero-Order-Hold (ZOH) component holds the last transmitted data until next transmission occurs. ZOH is exactly similar to Memory Consensus Protocol (MCP) method explained in Chapter 3.

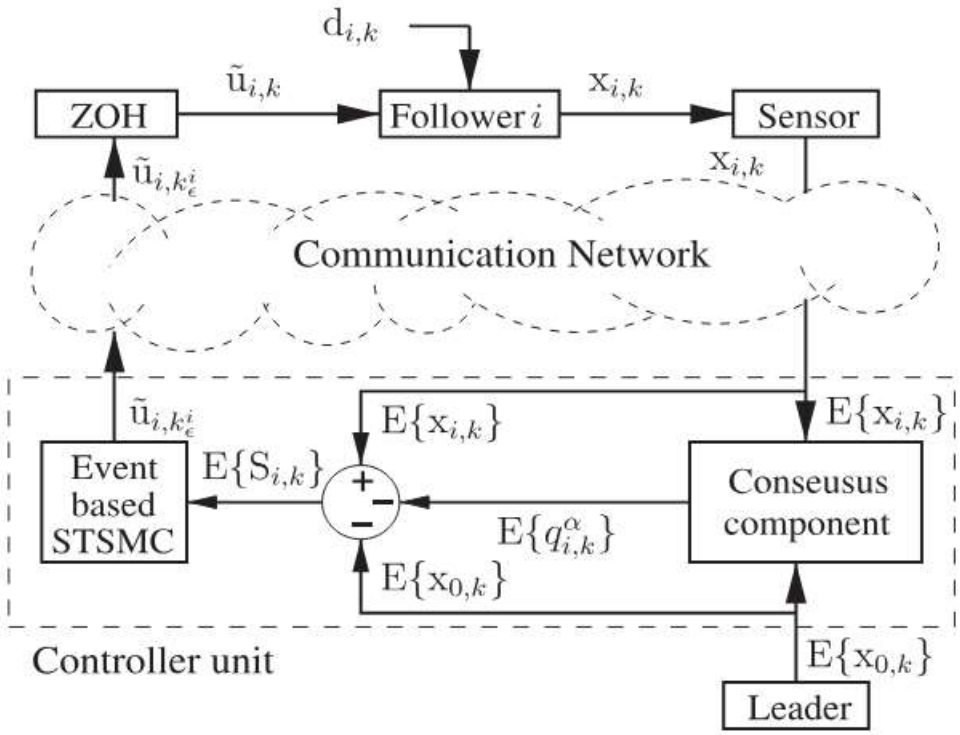


Figure 4.1: Control block diagram sliding [NDM+21].

4.3 Comparison of proposed LSTM under packet loss and delay with nonlinear controllers

The goal of this section is to compare the outcomes of the proposed LSTM method combined with LQR, as discussed in Chapter 3, with the nonlinear controllers developed in [NDM+21]. To ensure a fair and consistent comparison with nonlinear controllers, the LSTM-augmented LQR architecture introduced in Chapter 3 was adapted to match the experimental configuration of [NDM+21] and [WZ15]. The same leader–follower setup, initial conditions, and circular trajectory profile were used. Packet loss was modeled using a Bernoulli distribution with a probability of 0.2, and a time-varying delay was introduced. No architectural changes were made to the trained LSTM network. So, it was not retrained for this scenario, only reused from Chapter 3.

It is important to note that these two approaches have some fundamental differences, as outlined below:

- As mentioned, the methods developed in [NDM+21] and [WZ15] employ

super-twisting sliding control and back-stepping control, respectively, both of which are nonlinear controllers. In contrast, the approach presented in this thesis is based on linear control augmented with LSTM.

- The method in [NDM+21] applies to consensus leader-follower formation control, while this thesis addresses non-consensus leader-follower formation control. In consensus-based formation control, the aim is for all agents to collectively reach and maintain the desired formation pattern while also achieving consensus on their states or behaviors. Conversely, in non-consensus formation control, the followers simply track predefined trajectories or formations set by the leader without necessarily aiming to reach a consensus on a common state or behavior.
- The method in [NDM+21] employs event-triggered control, whereas our proposed method utilizes time-triggered control. The method in [WZ15] also utilizes time-triggered control.
- While the approach in [NDM+21] incorporates zero-order hold or MCP in its architecture, our proposed method does not use MCP.
- Furthermore, the method in [NDM+21] accounts for both delay and packet loss in the definition of the mathematical model of agents as explained in section 4.2, to enhance robustness against these factors, whereas our method aims to enhance any existing simple and linear controller by integrating LSTM to mitigate the impact of packet loss.

All simulation parameters (initial positions, desired separations, Bernoulli loss with $p = 0.2$ and delay) follow [NDM+21]. Because the internal tuning of the super–twisting controller is not fully disclosed, we compare *behavior under the same scenario*, not an exact re-implementation. Accordingly, the figures and tables should be interpreted as a fair scenario match rather than a strict controller benchmark.

4.3.1 Simulation result

In this subsection, the methodology proposed in this thesis is evaluated through simulations on a circular path with a formation exactly same as to that developed in [NDM+21]. The initial position of leader is (0,0) while the followers are initially at (-2, 0.75) and (-2, -0.75). The desired separation between the leader and followers 1 and 2 should be $\Gamma_1 = (0.75m, -0.75m)$, $\Gamma_2 = (0.75m, 0.75m)$,

respectively. The packet losses are modeled using Bernoulli distribution with a probability of 20% and the time-varying delay .

To evaluate the consistency of all approaches, a performance criterion (average tracking error norm) based on [NAQ18] is considered, as shown in Table **??**. The average tracking error norm defined as:

$$\bar{e}_i = \frac{1}{N}\sum_{k=1}^{N}\sqrt{(x_i(k) - x_{\text{ref},i}(k))^2 + (y_i(k) - y_{\text{ref},i}(k))^2} \tag{4.9}$$

where $\bar{e}_i$ represents the average tracking error norm of follower i, N is the total number of time samples, $(x_i(k), y_i(k))$ denote the actual positions of follower i at time step k, and $(x_{\text{ref},i}(k), y_{\text{ref},i}(k))$ are the desired reference positions corresponding to the leader's trajectory. The metric provides an overall measure of the tracking performance, with smaller $\bar{e}_i$ values indicating better formation maintenance and higher accuracy in following the leader's desired path. The average tracking error $\bar{e}$ is reported individually for F1 and F2 for clarity.

4.3.1.1 Impact of delay on formation control of proposed LSTM

Although the proposed method is not specially designed to handle delay, this section investigates the influence of delay on the system. Following a scenario similar to [NDM+21], the time-varying delay δ_i^x is constrained to satisfy $0 \leq \delta_i^x \leq 2s$. The outcomes of the LSTM-based approach are evaluated under conditions involving both delay and packet loss, as illustrated in Figure 4.2.

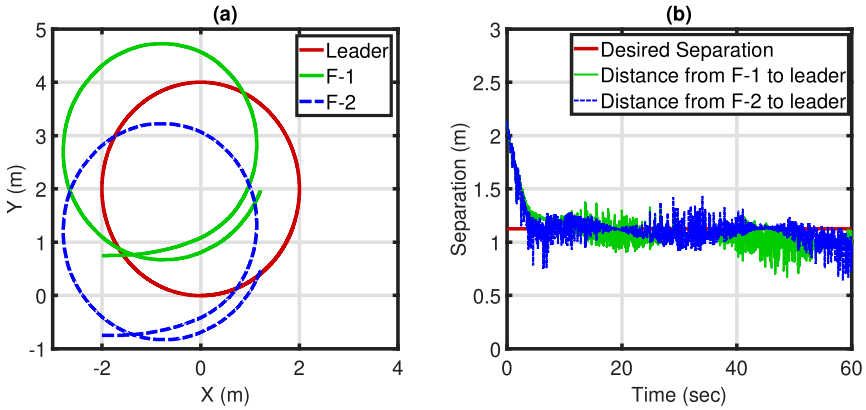


Figure 4.2: Formation control of proposed method for the circular trajectory (packet loss—20% and time-varying delay).

In particular, the presence of delay significantly affects the system, leading to

oscillations as depicted in Figure 4.2 (b). This indicates the method's susceptibility to delay, despite maintaining formation control with commendable accuracy. Although LSTM augmented with LQR is successful to maintain formation control of robots against delay and packet loss, still the oscillation caused in Figure 4.2 (b) is not desirable, therefore next section suggest method to avoid this oscillation.

4.3.1.2 Input shaping filter against delay

Digital filtering and input shaping are well-known methods for shaping reference commands to reduce vibration [SV10]. Input shaping filters have been successfully used in control systems for a long time to shape control signals in different applications including portal crane motion control, high-rise elevator motion control and robotics [MK10]. An input shaper modifies slightly the reference signals to damp the vibrations of the end-effector. This means, although the oscillations are eliminated, the tool will not move as desired since modified trajectories are commanded. Hence, an exact tracking of the desired path and a total vibrations damping seem to be two contradictory goals that cannot be simultaneously fully fulfilled. However a trade-off can be reached by minimizing the sum of the sensor deflection and the deviation between the desired and commanded signals [KLH09]. Therefore a low pass filter input shaper is used as eq. 4.10 to compensate for the oscillation occurring in Figure 4.2.

$$\mid H(jw) \mid = \frac{1}{\sqrt{(1 + (\frac{\omega}{\omega_c})^{2n})}} \tag{4.10}$$

The parameters ω_c, representing the filter cutoff frequency, and n, the filter order, are carefully chosen to balance accuracy and oscillation avoidance. A second-order filter with a cutoff frequency of 10 Hz is employed. These input shaping filters are integrated into the follower's reference, originating from the leader, which is subject to both delay and packet loss.

Consequently, the proposed method is now capable of compensating for both delay and packet loss. The block diagram of the proposed leader-follower formation control is depicted in Figure 4.3. The leader's position is transmitted to the follower through a wireless communication network susceptible to packet losses and delays. To address packet loss, LSTM prediction is utilized when such losses occur, as discussed in Chapter 3 and illustrated in Figure 3.5. For delay compensation, an input shaping filter is designed, as shown in eq. 4.10.

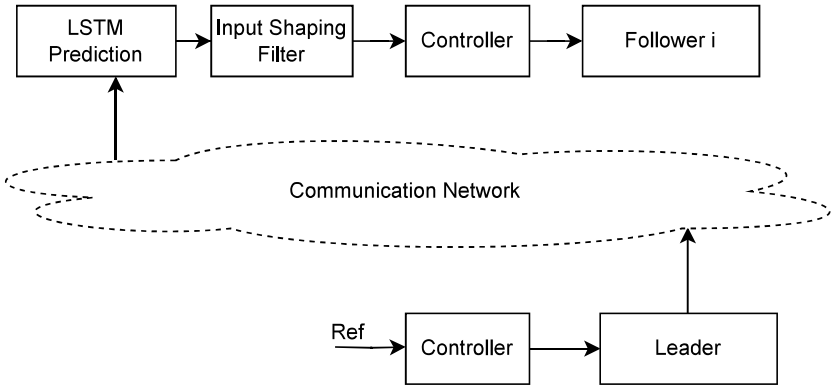


Figure 4.3: Proposed leader follower formation control

Simulation results of the proposed method using proposed input shaping filter, are presented in Figure 4.4. The formation control of the robots is successfully maintained during traversal of a circular trajectory. The oscillations observed in the separation between the leader and follower, caused by delay as shown in Figure 4.2, are effectively mitigated by the input shaping filter.

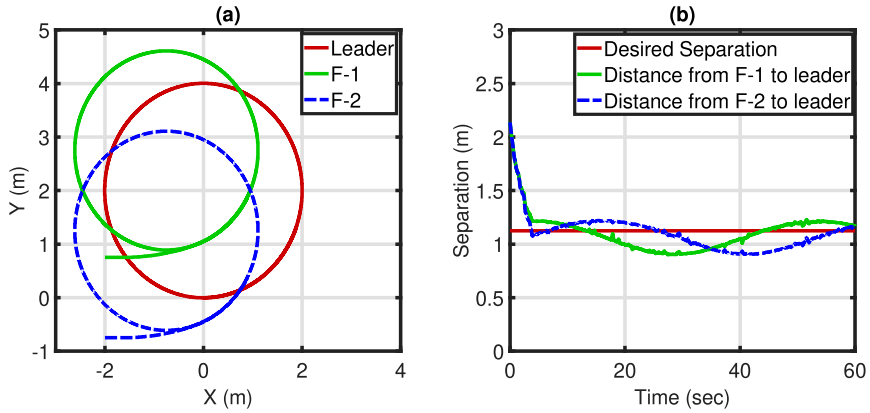


Figure 4.4: Proposed leader follower formation control against delay and packet loss

Under the replicated scenario, the proposed method achieves lower tracking error than the backstepping controller of [WZ15], as evaluated in [NDM+21]; shown in Figure 4.5. In Figure 4.5 (a), the trajectories of F1 and F2 do not complete the circular path followed by the leader. Both followers deviate outward and never return to the intended orbit. This indicates a loss of formation during the motion. The separation plot in Figure 4.5 (b) confirms this behavior: instead of converging to the desired constant distance, the follower–leader spacing oscillates and drifts away from the desired separation. In contrast, the

proposed LSTM-augmented controller preserves both the circular trajectory and the required formation under the same packet-loss and delay rate conditions as shown in Figure 4.4.

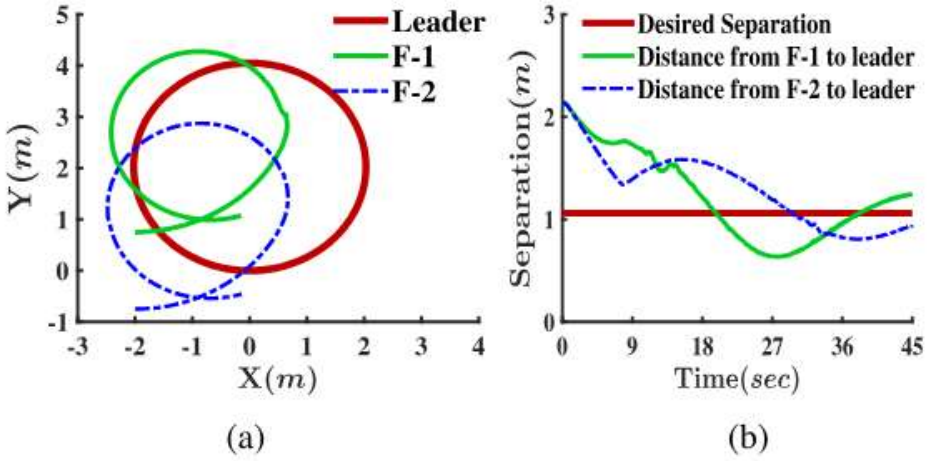


Figure 4.5: Result of [NDM+21] based on [WZ15] for the circular trajectory (packet loss—20%).

The proposed method achieves a comparable level of performance to the event-based sliding mode controller shown in 4.6 and developed in [NDM+21]. Additionally, the average tracking errors are provided in Table 4.1, allowing for comparison with the results of event-based sliding mode control [NDM+21] and back-stepping control [WZ15]. While the proposed LSTM-augmented LQR does not outperform the event-based sliding-mode controller in all cases, it remains within the same performance range without requiring a nonlinear control law. Proposed LSTM-augmented LQR also performs significantly better than the nonlinear time-triggered backstepping approach. The close performance between the proposed method and the sliding mode controller suggests that learning-based prediction can effectively compensate for communication-induced uncertainties. The input shaping filter further contributes to stability by attenuating oscillations caused by delay, resulting in smoother and more reliable formation tracking. Overall, the proposed framework achieves strong robustness against packet loss and delay with considerably lower control effort than the nonlinear controllers which locates inside control feedback loop.

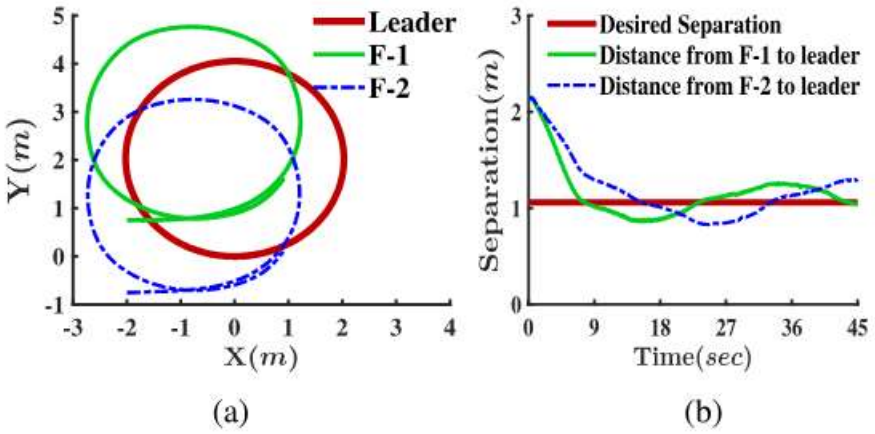


Figure 4.6: Result of [NDM+21] for the circular trajectory (packet loss—20%).

Table 4.1: Average tracking error norm of followers considering delay and packet loss in all methods.

Control Method	Avg. tracking error F1	Avg. tracking error F2
Proposed method	1.39e-2	2.19e-2
Event-based sliding-mode control [NDM+21]	1.40e-2	1.38e-2
Time-triggered backstepping controller [WZ15]	4.91e-1	4.4e-1

4.4 Concluding Remarks

This chapter addressed the formation control problem for a network of non-holonomic differential-drive robots and compared the proposed LSTM-augmented LQR controller against nonlinear approaches. A concise overview of the sliding-mode controller developed in [NDM+21] was first presented, followed by comparisons of the proposed method with both sliding-mode and backstepping controllers. However, it is important to note that the results in [NDM+21] were evaluated under both packet loss and delay conditions. An input-shaping filter was designed to mitigate the oscillations caused by delay. The comparison of average tracking error demonstrated that the proposed method achieved performance comparable to highly nonlinear controllers while maintaining a simpler and more generalizable structure. This outcome supports the central hypothesis of this research stated in 1: **Can combining LSTM with linear**

control strategies achieve performance comparable to advanced nonlinear controllers, such as sliding-mode control and backstepping controller, when dealing with packet loss and communication delays?

For clarity, all comparisons in this chapter are scenario-based, meaning that experimental conditions reported in [NDM+21, WZ15] were replicated as closely as possible with the exact scenario details, but a one-to-one controller implementation was not claimed.

An important aspect highlighted by this study concerns the computational complexity and training effort required for LSTM models compared to traditional nonlinear controllers such as backstepping or sliding mode control. While LSTM-based predictors are data-driven and adaptable to disturbances such as packet loss and delay, their training process can be computationally demanding and time-consuming, especially when optimizing hyperparameters or handling large datasets. In this work, training was performed offline, and therefore, computational effort did not impact the online control performance. For this reason, detailed benchmarking of training time was not included. However, future work could systematically evaluate the computational cost and training efficiency to provide a more quantitative comparison.

Although data-driven models such as LSTM require an offline training phase, the trained network is lightweight at runtime and does not increase the online computational load of the controller. The network was trained on multiple trajectory patterns and evaluated on unseen realizations with different radii and initial offsets, which confirms that the model learned the underlying temporal relationships rather than memorizing the data. Since training is only performed once and offline, its computational cost does not affect deployment. For this reason, a quantitative analysis of the training time was not included in this work, but may be investigated in future work.

In contrast, nonlinear controllers such as backstepping and sliding mode control are model-based and deterministic, offering systematic design once the system model is well defined. However, their performance strongly depends on accurate modeling and known parameters, which limits adaptability under unmodeled dynamics or variable network conditions. The proposed LSTM-augmented LQR takes a different approach—while the LSTM requires offline training, it operates entirely outside the real-time control loop and is only invoked to generate leader-state predictions during packet loss. The control law itself remains a standard LQR, ensuring that online computational cost is minimal

and comparable to a conventional linear controller. By contrast, nonlinear controllers execute complex model-dependent computations within each control iteration. Therefore, although LSTM-based methods require more training effort, the proposed architecture achieves a favorable balance between computational simplicity, robustness, and adaptability under packet loss and delay conditions.

In conclusion, although LSTM-based approaches demand more offline computational resources for training, they offer significant benefits in terms of adaptability and robustness in communication-impaired environments. The findings of this chapter demonstrate that integrating learning-based prediction with linear control provides a practical, scalable, and effective alternative to complex nonlinear control designs. Future work should focus on quantitatively analyzing training cost and exploring generalization limits to further validate these trade-offs.

Chapter 5

Formation Control of Quadrotors under Packet Loss

This chapter addresses the formation control problem for quadrotor Unmanned Aerial Vehicles (UAVs). A decentralized leader–follower approach based on mixed H_2/H_∞ controller and Proportional Derivative (PD) controllers are developed in presence of packet loss. Motivation and brief background is explained in section 5.1. Section 5.2 recapitulates the dynamical model of a quadrotor system. Section 5.3 discusses the control problems of a single quadrotor, employing mixed H_2/H_∞ controller and PD controllers. In section 5.4, we demonstrate the vulnerability of the system to packet loss and introduce an Long Short-Term Memory (LSTM) network to counteract its negative impact. Subsequently, In section 5.5, prediction accuracy of LSTM, Gated Recurrent Units (GRU), and Memory Consensus Protocol (MCP) are compared. Then, simulations are used to evaluate the LSTM and MCP performance. Finally, concluding remarks of the chapter is given in section5.6.

5.1 Motivation

In this chapter, LSTM is developed to control the formation of quadrotor UAVs in the presence of packet loss. As demonstrated in the chapter 3, LSTM was utilized to compensate for packet loss while robots maintained their formation, showcasing superior performance compared to alternative methods. Nonetheless, it is conceivable that these results may be confined to the Automated Guided Vehicles (AGVs) application, along with the Linear Quadratic Regulator (LQR) controller

developed therein. Herein, in this chapter, The aim is to broaden the scope of the problem concerning formation control of robots under packet loss to another application, namely quadrotors. Additionally, the LQR controller is no longer employed for trajectory tracking of quadrotor UAVs, and alternative controllers are used. The objective of this chapter is to demonstrate that our proposed solution transcends specific applications or controllers. In fact, the hypothesis of combining a simple controller with a learning method like LSTM to enhance system performance during uncertainties such as packet loss is not dependent on a particular system like AGV robots, a two-dimensional space, or a specific controller like LQR. Through this chapter, we establish our independence from application, dimensions, and controller; essentially, this method can be adapted to other applications and is scalable. The challenges in this chapter arise from the increased complexity of higher-dimensional dynamics in 3D space, where errors propagate across multiple directions (x, y, z). The additional dimension introduces state coupling (e.g., vertical movement affecting horizontal dynamics), which is absent in simpler 2D cases. Ensuring formation control in 3D also involves handling orientation and alignment challenges that are less prominent in 2D. Packet loss has a greater impact in 3D, as errors can propagate in all directions, leading to larger deviations in follower trajectories. Another key challenge is integrating LSTM predictions with various linear controllers to handle the complex dynamics of quadrotors. This requires designing systems that adapt to higher-dimensional dynamics, noise, and packet loss simultaneously. Careful integration is necessary to prevent over-reliance on inaccurate predictions, as compounded errors in 3D can destabilize the system more quickly. This involves addressing stability, aligning predictions with control inputs, and fine-tuning the controllers to ensure robust performance under packet loss.

The following are the major contributions of this chapter:

- Analysis of the impact of packet loss on the decentralized leader–follower formation control of quadrotor UAVs using mixed H_2/H_∞ controller combined with PD controllers.
- Improving the performance of a mixed H_2/H_∞ and PD controllers via LSTM to deal with packet losses,
- Demonstrating the adaptability of the LSTM prediction approach for quadrotors, alongside AGVs, highlights the effectiveness of the proposed method across diverse applications, dimensions, and control systems.

- Presenting Superior compensation capability of LSTM in scenarios with 30%, 50% and 70% packet loss, outperforming GRU, Linear Interpolation Predictor (LIP), and MCP.
- Testing performance of LSTM while ransom noise and disturbance happening in the system during spiral and eight-shape path.

5.2 Mathematical Modeling of Quadrotor UAVs

The quadrotor, a multi-rotor UAV with four rotors, has emerged as a dominant control system and a benchmark in mechatronics in recent years. Quadrotors has numerous applications such as rescue operations, aerial photography, mapping, and monitoring, further increasing its popularity. Controlling UAVs is one of the challenging problems in control systems engineering since the autonomous navigation of these vehicles demands an accurate and quick response controller. Moreover, the mathematical models of UAVs are typically nonlinear, highly sensitive to input variations, and susceptible to environmental factors, compounding the difficulty of this challenge [EF16].

A quadrotor is a type of multi-rotor helicopter equipped with two pairs of propellers. This UAV is controlled and maneuvered through variations in the rotational velocities of its propellers. Consequently, dynamic differential equations from propellers velocities to quadrotor attitude and quadrotor position are required to define the dynamic model of a quadrotor. The motion of a quadrotor aircraft involves six degrees of freedom: Two positions along the x and y; an altitude along z axes; and three orientations around the x, y, and z axes which are called 'roll' (ϕ), 'pitch' (θ), and 'yaw' (ψ). The dynamic model of a quadrotor is defined by six second-order differential equations, outlined as equations 5.1: [BS05]

$$
\begin{aligned}
m\ddot{x} &= \Big(\cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi)\Big)U_1 \\
m\ddot{y} &= \Big(\cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi)\Big)U_1 \\
m\ddot{z} &= -mg + \cos(\phi)\cos(\theta)\,U_1 \\
I_x\ddot{\phi} &= \dot{\theta}\dot{\psi}\,(I_y - I_z) + L\,U_2 \\
I_y\ddot{\theta} &= \dot{\phi}\dot{\psi}\,(I_z - I_x) + L\,U_3 \\
I_z\ddot{\psi} &= \dot{\phi}\dot{\theta}\,(I_x - I_y) + L\,U_4
\end{aligned}
\tag{5.1}
$$

The system's inputs are denoted as U1, U2, U3, U4, obtaining:

$$\begin{aligned} U_1 &= b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 &= b(-\Omega_2^2 + \Omega_4^2) \\ U_3 &= b(-\Omega_1^2 + \Omega_3^2) \\ U_4 &= d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \end{aligned} \tag{5.2}$$

where Ω_i are rotors angular velocities; I_x, I_y, and I_z are inertia around x, y, and z. m, L, g, b are quadrotor mass, lever length, gravity, trust coefficient, and drag coefficient respectively. Additional details about modelling of quatrotors can be found in [BS07, BS05, DMB06, DOMB07].

As previously emphasized, the mathematical model of a quadrotor exhibits nonlinear and unstable dynamics, rendering it highly susceptible to environmental influences. Factors such as measurement noise and external disturbances—such as wind disturbances in outdoor environments, collisions between cooperative quadrotors, and collisions with fixed objects in indoor environments—are common occurrences during flight. Therefore, controlling this robot necessitates the use of a robust controller capable of both noise suppression and disturbance rejection. To achieve this objective, a mixed H_2/H_∞ robust trajectory tracking controller is proposed for controlling the attitude and altitude of the quadrotor for the simplified model as follows:

$$\begin{aligned} m\ddot{z} &= -mg + (\cos\phi\ \cos\theta)\, U_1 \\ I_x\ddot{\phi} &= \dot{\theta}\dot{\psi}(I_y - I_z) + LU_2 \\ I_y\ddot{\theta} &= \dot{\phi}\dot{\psi}(I_z - I_x) + LU_3 \\ I_z\ddot{\psi} &= \dot{\phi}\dot{\theta}(I_x - I_y) + LU_4 \end{aligned} \tag{5.3}$$

With the system states vector defined as $[\phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}, z, \dot{z}]$, then nonlinear state space representation of the system is detailed in equation 5.4 [BS05].

Since the H_2/H_∞ controller is designed using a linear approach, a linear state space representation is required. In this thesis, the dynamic model of the system is linearized around the equilibrium point $X_{eq} = [0, 0, 0, 0, 0, 0, 0, 0]^T$ by employing the Jacobian approach. The calculated linear state space representation of this system is reported in 5.5.

$$
\begin{aligned}
\dot{X}_1 &= X_2 \\
\dot{X}_2 &= X_4 X_6 \left(\frac{I_y - I_z}{I_x} \right) + \frac{L}{I_x} U_2 \\
\dot{X}_3 &= X_4 \\
\dot{X}_4 &= X_2 X_6 \left(\frac{I_z - I_x}{I_y} \right) + \frac{L}{I_y} U_3 \\
\dot{X}_5 &= X_6 \\
\dot{X}_6 &= X_2 X_4 \left(\frac{I_x - I_y}{I_z} \right) + \frac{L}{I_z} U_4 \\
\dot{X}_7 &= X_8 \\
\dot{X}_8 &= -g + \left(\frac{\cos X_1 \, \cos X_3}{m} \right) U_1
\end{aligned}
\tag{5.4}
$$

$$
\dot{x} = Ax(t) + Bu(t), \qquad t \geq 0, x(0) = 0 \tag{5.5}
$$

where,

$$
A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}, B = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & \frac{l}{I_x} & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & \frac{l}{I_y} & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{l}{I_z} \\
0 & 0 & 0 & 0 \\
\frac{1}{m} & 0 & 0 & 0
\end{bmatrix} \tag{5.6}
$$

5.3 Controller Design

As mentioned in section 5.2, UAVs exhibit significant nonlinearities, inherent coupling, and underactuated dynamics, meaning they possess six degrees of freedom (required to fully describe the movement of a rigid body in three-dimensional space) while are controlled by only four inputs (the upward thrust and three angle torques) [Ngu19]. Furthermore, these systems are subject to various operational constraints, some of which may be non-convex or non-holonomic in nature, as highlighted by [NPSL17], being a real challenge to control.

The predominant control architecture for quadrotors consists of two control loops for position and attitude [BS07]. Quadrotor UAVs face various challenges during operation, such as environmental disturbances and obstacles, necessitating robust and efficient control systems for stabilization and improved performance. Although many control methods yield satisfactory results in ideal conditions, their effectiveness can vary. Common controllers for quadrotor UAVs include LQR, Proportional Integral Derivative (PID), H_∞, Sliding Mode Controllers (SMC), backstepping, and adaptive control [MMM22].

Our hypothesis suggests that combining linear controllers with deep learning techniques can improve the performance of UAV control systems under packet loss. In line with this, a cascaded control structure is adopted in which a mixed H_2/H_∞ controller regulates the attitude and altitude, while a PD controller is used for the outer-loop position control. PD control is widely used in quadrotor position regulation [Fäs18] because it provides a fast and well-damped response without introducing the phase lag associated with integral action.

Unlike ground vehicles, a quadrotor already behaves as an integrating system, since position is obtained by integrating the velocity and thrust generated by the inner control loops. As a result, steady-state errors are already compensated by the inner H_2/H_∞ controller, and adding an additional integral term in the outer loop is unnecessary and may even be detrimental. In practice, a full PID position controller can lead to integrator windup, oscillations, or loss of robustness during fast manoeuvres or in the presence of packet loss.

For attitude control, a more robust controller is essential due to the increased sensitivity of these dynamics to external disturbances and uncertainties. Environmental factors like wind or turbulence can quickly destabilize a quadrotor if not corrected promptly. Additionally, the coupling between roll, pitch, and yaw means that changes in one axis can affect the others, making it harder to manage with simpler control methods. Attitude control also requires fast and accurate responses, as delays in stabilizing the quadrotor can lead to instability. Furthermore, model uncertainties, sensor noise, and parametric variations are more likely to impact the attitude dynamics, necessitating a control approach that can manage these uncertainties effectively. Therefore, robust controllers like the H_∞ are used in place of simpler controllers like PD to ensure the system can handle disturbances while maintaining stability and performance[MMM22]. Altitude (z), roll (ϕ), pitch (θ), and yaw (ψ) are controlled using a mixed H_2/H_∞ controller, a robust linear static state feedback controller designed using the

Linear Matrix Inequality (LMI) approach similar to [EF16]. This controller integrates the benefits of two well-known robust control methods: the H_2 controller, which is effective against measurement noise and produces a control signal with low variation, and the H_∞ controller, which is designed to be resilient to external disturbances and model uncertainties, though it tends to result in higher variation control signals. By blending these two methods, the mixed H_2/H_∞ controller offers a robust solution that balances the need for robustness against disturbances and uncertainties while maintaining a smooth control signal.

Additionally, regional pole-placement constraints are applied to the H_2/H_∞ controller to further enhance the time-domain performance of the closed-loop system. These constraints help improve metrics like settling time, maximum overshoot, and the maximum frequency response of the system, ensuring a more desirable transient behavior [CG96]. This hybrid control design not only addresses the quadrotor's stabilization needs but also achieves a high degree of robustness to real-world uncertainties, fulfilling the hypothesis of this thesis regarding the improvement of linear controllers for UAV systems. Furthermore, regional pole-placement constraints is added to the H_2/H_∞ controller to enhance the time domain response of the close-loop system such as settling time, maximum over-shoot, and maximum frequency of the close-loop response [CG96]. This ensures a more refined transient response and overall stability of the UAV during flight operations.

5.3.1 Mixed H_2/H_∞ Controller

The following section builds upon the design proposed in [EF16] for the attitude control of a single quadrotor following a reference trajectory, and therefore, no novelty is claimed for this aspect. Our contribution lies in incorporating an additional controller, as detailed in section 5.3.3, to achieve full control of the quadrotor since [EF16] only provides partial control—and subsequently extending this approach to multiple robots while analyzing the impact of packet loss.

The robust controller design based on the LMI approach can be expressed in the standard form as shown in 5.7, where $X \in \mathbb{R}^n$ is the state vector, $U \in \mathbb{R}^m$ is the control input, and $W \in \mathbb{R}^r$ is the disturbance vector representing external disturbances effects. The matrices D_{ij} represent direct feed-through terms between the control, disturbance, and performance channels.

$$\begin{cases} \dot{X} = AX + B_1W + B_2U, \\ Z_\infty = C_1X + D_{11}W + D_{12}U, \\ Z_2 = C_2X + D_{21}W + D_{22}U. \end{cases} \tag{5.7}$$

The performance outputs are defined as:

- Z_∞: the signal constrained by the H_∞ controller to minimize the effect of external disturbances and ensure robustness.
- Z_2: the signal constrained by the H_2 controller to minimize the influence of measurement noise and ensure smoother transient behavior.

The state feedback law is defined as $U = KX$, where K is the controller gain matrix. Substituting this into equation 5.7 yields the closed-loop representation:

$$\begin{cases} \dot{X} = (A + B_2K)X + B_1W, \\ Z_\infty = (C_1 + D_{12}K)X + D_{11}W, \\ Z_2 = (C_2 + D_{22}K)X + D_{21}W. \end{cases} \tag{5.8}$$

Theorem 1 Matrix $(A + B_2K)$ in equation 5.8 will be Hurwitz (System 5.8 will be stable) and infinity-norm of the transfer function from disturbance input W to Z_∞ will be less than γ, if and only if matrix inequalities refeq:Theorem 1 have a feasible solution. [BEGFB94, CG96].

$$\begin{cases} \begin{pmatrix} (A + B_2K)P_{cl} + P_{cl}(A + B_2K)^T & B_1 & P_{cl}(C_1 + D_{12}K)^T \\ B_1^T & -I & D_{11}^T \\ (C_1 + D_{12}K)P_{cl} & D_{11} & -\gamma^2 I \end{pmatrix} & < 0 \\ P_{cl} = P_{cl}^T & > 0 \end{cases} \tag{5.9}$$

Note that in equation 5.9 P_{cl} is a symmetric matrix, I is the identity matrix with appropriate size, and γ is guaranteed H_∞ performance. Note that < 0 denotes negative definite and > 0 denotes positive definite. The proof of Theorem 1 is well-documented in the foundational literature on Linear Matrix Inequalities in System and Control Theory [BEGFB94], which provides essential insights into

the feasibility of the matrix inequalities outlined in this thesis.

Theorem 2 Matrix $(A + B_2K)$ in equation 5.8 will be Hurwitz (System 5.8 will be stable) and 2-norm of the transfer function from disturbance input W to Z_2 will be less than υ, if and only if the matrix inequalities 5.10 have a feasible solution. [BEGFB94, CG96].

$$\begin{cases} \begin{pmatrix} (A+B_2K)P_{cl} + P_{cl}(A+B_2K)^T & B_1 \\ B_1^T & -I \end{pmatrix} < 0 \\ \begin{pmatrix} Q & (C_2 + D_{22}KP_{cl}) \\ P_{cl}(C_2+D_{22}K)^T & P_{cl} \end{pmatrix} > 0 \\ Trace(Q) < \upsilon^2 \\ P_{cl} = P_{cl}^T > 0 \end{cases} \tag{5.10}$$

With guaranteed H_2 performance $\upsilon > 0$, the theoretical foundation of Theorem 2 follows standard results in LMI theory, as detailed by Boyd *et al.* [BEGFB94] and Chilali and Gahinet [CG96]. These works provide the mathematical basis for evaluating the feasibility of the matrix inequalities used to ensure the H_2 performance criterion within this framework.

Theorem 3. Consider domain D in complex numbers plane as equation 5.11 [CG96].

$$D = \{z \in \mathbb{C} : L + Mz + M^T\overline{z} < 0\} \tag{5.11}$$

Where L and M are defined in equations 5.12 and 5.13, respectively.

$$L = L^T = \{\lambda_{ij}\}_{1 \le i,j \le m} \tag{5.12}$$

$$M = \{\mu_{ij}\}_{1 \le i,j \le m} \tag{5.13}$$

Poles of the system 5.8 (eigenvalues of matrix $(A + B_2K)$ will be located on D, if and only if matrix inequalities 5.14 have a feasible solution.

$$\begin{cases} (\lambda_{ij}P_{cl} + \mu_{ij}(A + B_2K)P_{cl} + \mu_{ij}P_{cl} + P_{cl}(A + B_2K)^T\mu_{ij}^T)_{1\leq i,j\leq m} < 0 \\ P_{cl} = P_{cl}^T > 0 \end{cases} \tag{5.14}$$

Static state feedback controller K is determined by combining the H_2 and H_∞ controllers, aiming to minimize the H_2 performance υ while keeping the H_∞ performance below a specified parameter γ. This approach also fulfills regional pole-placement constraints. The minimization problem is addressed using the MATLAB LMI Toolbox [Gah94], in this thesis. It is important to note that equations 5.9, 5.10, and 5.14 are nonlinear due to the presence of the term KP_{cl}. However, this issue can be addressed by introducing a variable change $Y \triangleq KP_{cl}$, which allows for a reformulation of the system equations that may simplify the analysis and enable the application of linear control techniques [EF16]. The proof of Theorem 3 is documented in [CG96].

5.3.2 Mixed H_2/H_∞ Tracking Controller

The controller introduced in 5.3.1 represents the standard form of an H_2/H_∞ controller, functioning as a regulator controller to regulate system states to the origin. However, as the focus of this thesis is on tracking controller design, this controller is enhanced into a tracking controller using the method proposed by [Sed03]. The structure of this tracking controller is depicted in Figure 5.1.

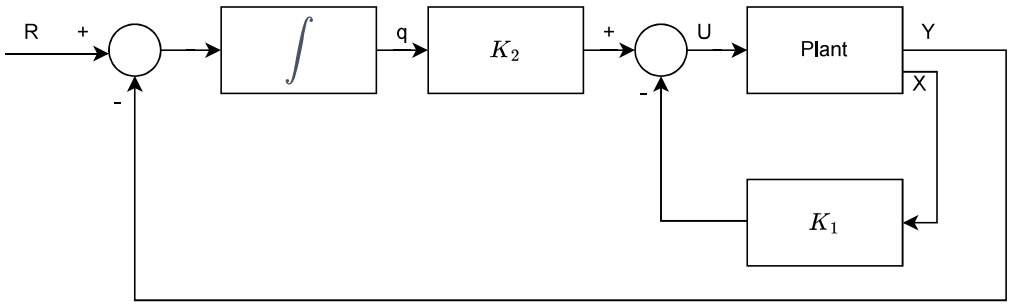


Figure 5.1: Quadrotor tracking controller structure

Theorem 4. (Developing a tracking controller using regulator controller design methods) Let's consider a dynamic system model with the following linear state-space representation:

$$\begin{cases} \dot{X} & = AX + BU \\ Y & = CX \end{cases} \tag{5.15}$$

And definition of integral state q (virtual state) as follows:

$$\dot{q} = R - Y = R - CX \tag{5.16}$$

Where, R represents the system's reference input. The combination of equations 5.15 and 5.16 yields the system 5.20, referred to as the augmented system.

$$\begin{cases} \begin{pmatrix} \dot{X} \\ \dot{q} \end{pmatrix} = \begin{pmatrix} A & 0 \\ -C & 0 \end{pmatrix} \begin{pmatrix} X \\ q \end{pmatrix} + \begin{pmatrix} B \\ 0 \end{pmatrix} U + \begin{pmatrix} 0 \\ I \end{pmatrix} R \\ \quad Y = \begin{pmatrix} C & 0 \end{pmatrix} \begin{pmatrix} X \\ q \end{pmatrix} \end{cases} \tag{5.17}$$

Lets consider a static state feedback controller as 5.18.

$$U = K_{ad} X_{ad} = \begin{pmatrix} -K_1 & K_2 \end{pmatrix} \begin{pmatrix} X \\ q \end{pmatrix} \tag{5.18}$$

If the static controller K_{ad} is designed by regulator controller design approaches, outputs of the system 5.15 tend to reference inputs. (see Figure 5.1)

Therefore, the presented mixed H_2/H_∞ regulator controller in section (5.3.1) can be improved to a tracking controller by using Theorem 4.

5.3.3 Proposed Quadrotor Controller

The proposed control scheme for the entire quadrotor system is subsequently divided into a position controller for x and y; and an altitude and attitude controller for z, 'roll' (ϕ), 'pitch' (θ), and 'yaw' (ψ), as illustrated in Figure 5.2.

Position control in diagram 5.2 are designed based on two PD controllers for x and y as per equation 5.19.

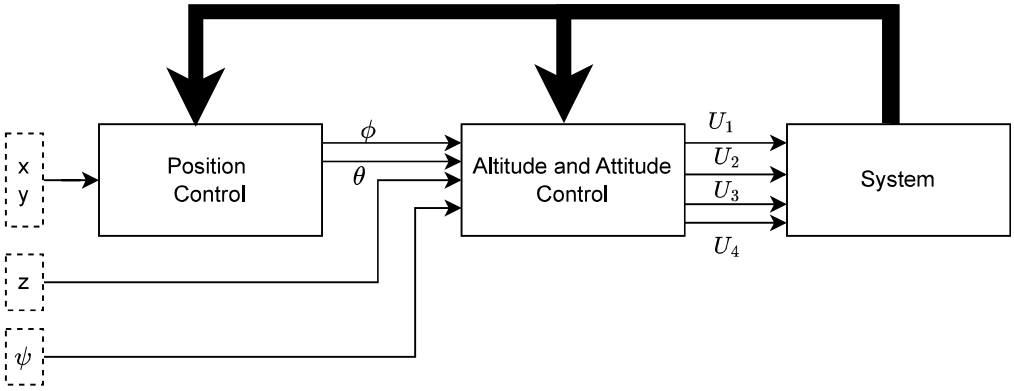


Figure 5.2: Quadrotor control scheme

$$\begin{aligned} \ddot{x} &= \frac{1}{m} U_1 \big(\cos\phi \, \sin\theta \, \cos\psi + \sin\phi \, \sin\psi \big) \\ \ddot{y} &= \frac{1}{m} U_1 \big(\cos\phi \, \sin\theta \, \sin\psi - \sin\phi \, \cos\psi \big) \end{aligned} \tag{5.19}$$

Altitude and attitude control is based on mixed H_2/H_∞ tracking controller developed in subsections 5.3.1 and 5.3.2 which is designed based on [EF16]. As previously discussed, in order to achieve a mixed H_2/H_∞ tracking controller, a mixed H_2/H_∞ regulator controller is designed for the system 5.20. However, this approach requires the system model 5.7. Thus, the system 5.20 is represented in terms of the system 5.7 as defined below.

$$\begin{aligned} \dot{X}_{ad} &= \bar{A} X_{ad} + B_1 W + B_2 U \\ Z_\infty &= C_1 X_{ad} + D_{11} W + D_{12} U \\ Z_2 &= C_2 X_{ad} + D_{22} U \\ Y &= \begin{pmatrix} C & 0 \end{pmatrix} X_{ad} \end{aligned} \tag{5.20}$$

with,

$$X_{ad} = [\phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}, z, \dot{z}, \int e_\phi, \int e_\theta, \int e_\psi, \int e_z]^T$$

$$W = \begin{pmatrix} d \\ n \end{pmatrix}$$

$$\bar{A} = \begin{pmatrix} A & 0 \\ -C & 0 \end{pmatrix}$$

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$B_1 = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}^T$$

$$B_2 = \begin{pmatrix} B \\ 0 \end{pmatrix}$$

$$Z_\infty = \begin{pmatrix} e_\phi & e_\theta & e_\psi & e_z \end{pmatrix}^T$$

$$Z_2 = \begin{pmatrix} X & U \end{pmatrix}^T$$

$$Y = \begin{pmatrix} \phi & \theta & \psi & z \end{pmatrix}^T$$

Where, A and B are parameters of equation 5.5; d is disturbance, n denotes measurement noise, and $e_{(.)}$ signifies the error signal.

The proposed region for regional pole-placement is defined as the intersection of three specific domains, each contributing to the stability and performance of the closed-loop system. First, a disk centered at the origin with a radius of 60 ensures that the poles remain relatively close to the origin. This proximity is crucial for maintaining a desirable dynamic response, as poles that are too far from the origin can lead to slower response times and potential instability. Second, inside a conic shape centered at the origin with an angle of $\theta = \frac{\pi}{8}$ defines a specific angular region where the poles can reside. This angular constraint not only limits the placement of the poles but also enhances the damping characteristics of the system, contributing to improved transient performance. Finally, the condition that the poles must lie on the left side of the line $\text{Re}(s) = -1$ guarantees that all poles possess a negative real part, a necessary condition for

the stability of the closed-loop system. By intersecting these three domains, the defined region effectively ensures that the poles are positioned in a way that balances fast response times, adequate damping, and overall system stability. This careful selection of regions facilitates robust control under varying operational conditions. This region is depicted in Figure 5.3.

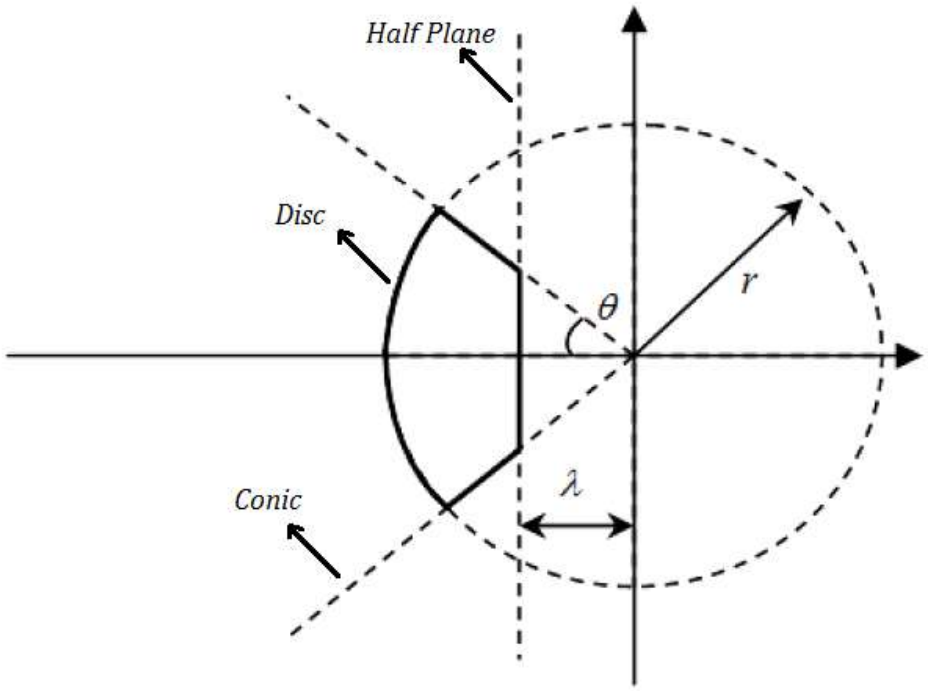


Figure 5.3: LMI Region[SF16]

The static state feedback controller denoted as K_{ad} and presented in 5.18 is determined by minimizing the H_2 performance while ensuring that the H_∞ is less than 0.25 and that the closed-loop system poles lie within the region depicted in Figure 3. The resulting controller is expressed in equation 5.21.

$$K_{ad} = \begin{pmatrix} -K_1 & K_2 \end{pmatrix} = \begin{pmatrix} -36.3 & -0.5 & -36.3 & -0.5 & -43.2 & -0.7 & -492.5 & -26.8 & 556.6 & 556.6 & 619.5 & 2765.5 \\ -49.5 & -1.8 & -8.2 & -0.1 & -9.1 & -0.1 & -28.3 & -0.7 & 380.8 & 115.6 & 124.3 & 249.6 \\ -8.2 & -0.1 & -49.5 & -1.8 & -9.1 & -0.1 & -28.3 & -0.7 & 115.6 & 380.8 & 124.3 & 249.6 \\ -13.8 & -0.2 & -13.8 & -0.2 & -87.5 & -3.3 & -50.0 & -1.3 & 192.7 & 192.7 & 657.4 & 437.7 \end{pmatrix} \tag{5.21}$$

5.4 Formation Control of Quadrotors Under Packet Loss

In this section, the formation controller performance is evaluated under various packet loss conditions. Following that, the application of an LSTM model is discussed in order to predict the leader's position in instances of packet loss, intended for utilization by the followers.

5.4.1 Impacts of packet loss on formation control

Similar to chapter 3, quadrotor control problem is extended to the formation problem of multiple quadrotor robots, while taking communication intervals and packet loss into account. The leader follower approach is considered here and the leader's position is communicated to all followers every 50ms. To simulate and compensate packet loss, a memory is considered in the follower robot that stores the most recent data received during packet loss as explained in chapter 2 as MCP. As a result, when a packet is lost, the system applies the most recently received data. There is no learning used here and we just analyze the impact of packet loss on the follower using MCP.

The instantaneous position error of follower is computed as the Euclidean distance between its actual position $(x_{\text{out}},\, y_{\text{out}},\, z_{\text{out}})$ and the desired reference position $(x_{\text{ref}},\, y_{\text{ref}},\, z_{\text{ref}})$, as defined in equation 5.22 and as shown in Figure 5.4 .

$$E_{\text{follower}}(t) = \sqrt{\Big(x_{\text{ref}}(t) - x_{\text{out}}(t)\Big)^2 + \Big(y_{\text{ref}}(t) - y_{\text{out}}(t)\Big)^2 + \Big(z_{\text{ref}}(t) - z_{\text{out}}(t)\Big)^2} \tag{5.22}$$

This metric represents the deviation of the follower from its target position in three-dimensional space and provides a direct measure of formation accuracy at each time instant.

Zoom around the worst case in Figure 5.4 (b), markers show the peak error time and value. The effects of packet loss are depicted in Figure 5.4, which illustrates how packet loss results in an increase in inaccuracy of around 6 cm when packet loss is 50% at time of 32.775 which is highlighted in Figure 5.4 (b) as a max error during formation control using MCP method.

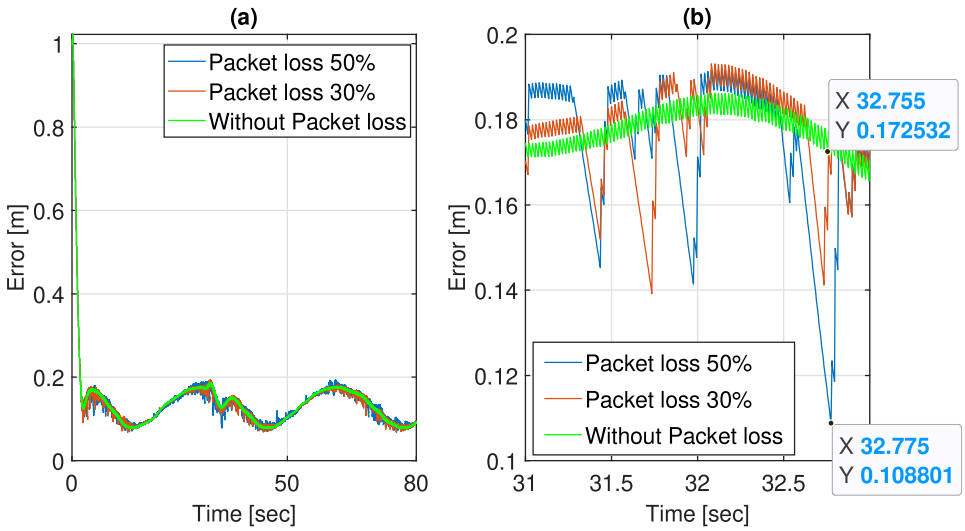


Figure 5.4: Effect of packet loss (a) Error of follower ; (b) zoomed view of follower error

5.4.2 Long Short Term Memory to Cope with Packet Loss and Proposed Algorithm

In this section, LSTM model is combined with the original control (Mixed H_2H_∞ combined with PD controllers) to predict the position of the leader when packet loss occurs, similar to chapter 3. The LSTM model can be used to compensate for network uncertainties in any existing linear or simple controller. LSTM is a type of recurrent network that reuses previously stored data and dependencies. As we are attempting to predict the leader's trajectory, which is a time-based order sequence, this problem fits within the LSTM framework.

In this thesis, follower robots maintain formation by relying on the leader's transmitted position. When packet loss occurs, the followers temporarily lose access to the leader's true position; therefore, an LSTM network—trained on the leader's past trajectory—is used to predict the missing position values. Under normal communication, the network states are updated using the actual leader data, whereas during packet loss, the most recent LSTM prediction is applied instead. The overall flow of this prediction-based compensation process is illustrated in Figure 3.5 in subsection 3.4.2.3.

5.5 Performance Evaluation

This section assesses the effectiveness of various prediction schemes, including LSTM and GRU. Additionally, the performance of the leader-follower formation control system is examined under different packet loss scenarios using these prediction methods. All evaluations are conducted through MATLAB/SIMULINK simulations.

The LSTM architecture employed for the quadrotor system is identical to that described in section 3.4.2.1 for the AGV platform. It follows a single-layer sequence-to-one regression design consisting of an input layer, an LSTM layer, a fully connected layer, and a regression output layer. Three independent LSTM networks are again trained to predict the leader's x, y, and z coordinates, respectively, using the previous ten samples of each variable.

Although the same network structure is used, the dataset is specific to the quadrotor simulations. It was generated from quadrotor leader trajectories under perfect communication conditions using the mixed H_2/H_∞ combined with PD controllers. Similar to the AGV case, the trajectories include both circular and eight-shaped paths to reflect the quadrotor's 3D motion characteristics. The training, validation, and testing splits follow the same proportions as in the AGV dataset. A total of eight trajectories were generated—four spiral and four figure-eight paths. Among these, five trajectories were used for training, one for validation, and two for testing. For each trajectory, the radius, center position, and flight height were randomly chosen within predefined bounds to introduce variability and prevent the network from overfitting to specific motion patterns. The validation set was used to fine-tune hyperparameters, while the unseen test trajectories—generated with different geometric parameters—were employed to evaluate the generalization performance of the model under new trajectory conditions and under noise and disturbances.

The parameters of the LSTM and GRU model were carefully chosen to balance prediction accuracy, computational efficiency, and processing time. The number of hidden units, batch size, and learning rate are optimized using Bayesian optimization with MATLAB's Experiment Manager tool within the following search ranges: hidden units $\in [7, 8, 9, \ldots, 30]$, learning rate $\in [0.001, 0.01, 0.1]$, batch size $\in \{8, 16, 32, 64, 128, 256\}$. The Adam optimizer was used in all training cases due to its stable convergence and good performance on nonlinear time-series data. The learning rate was adaptively reduced using a decay factor

of 0.2 whenever the validation loss failed to improve. To prevent increased computational complexity and overfitting, all models are trained for a maximum of 400 epochs for LSTM and GRU after preliminary testing to ensure convergence. As an example, the part of results provided by the Bayesian optimization process are illustrated in Figure 5.5. This test represents one of the trials that achieved the desired performance for predicting position x. Additional experiments were conducted to confirm these results. Row 10, highlighted in yellow in Figure 5.5 , represents the best-performing configuration. LSTM and GRU hyperparameters final values used for each output component (x, y, and z) are summarized in Table 5.1 and 5.2, respectively. It achieves the lowest validation RMSE and validation loss, making it the optimal choice for the experiments.

Table 5.1: LSTM model setting.

LSTM	**x**	**y**	**z**
Epox	400	400	400
Hidden nodes	21	16	29
Batch size	8	256	256
Learning rate	0.01	0.1	0.1
Learning rate decay factor	0.2	0.2	0.2

Table 5.2: GRU model setting.

LSTM	**x**	**y**	**z**
Epox	400	400	400
Hidden nodes	30	10	28
Batch size	256	8	128
Learning rate	0.1	0.1	0.1
Learning rate decay factor	0.2	0.2	0.2

In these scenarios, the follower robot was expected to maintain a predefined distance from the leader. Similar to chapter 3, the accuracy of the LSTM location prediction was assessed using the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), as defined in Equations (3.25) and (3.26). RMSE and MAE serve as a commonly employed metric for quantifying the disparity between predicted and observed values.

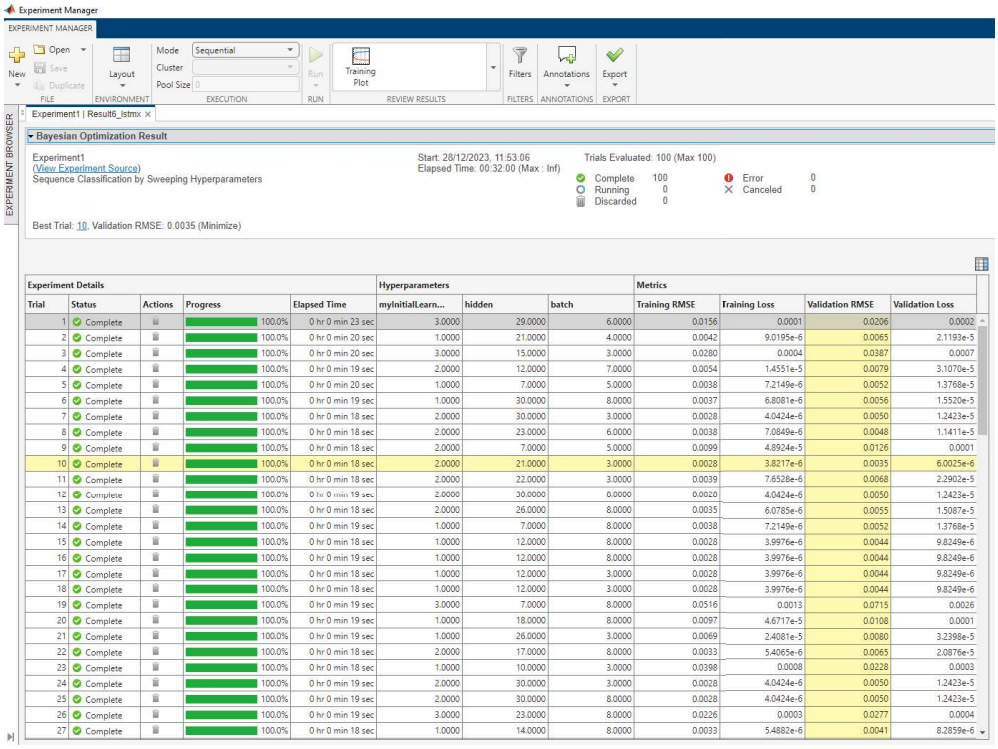


Figure 5.5: Bayesian optimization results for position x

5.5.1 Prediction Accuracy for LSTM and GRU

To assess the prediction accuracy of LSTM and GRUs, the error between the actual and predicted positions of the leader going through a spiral path is depicted in Figure 5.6 for x, y, and z. The results illustrate that LSTM exhibits superior prediction accuracy compared to GRU, particularly z direction. Additionally, Figure 5.7 presents the stable convergence of validation and training loss for LSTM that confirms the chosen LSTM configuration generalizes well to unseen trajectories without overfitting, making it suitable for real-time implementation in multi-quadrotor systems.

The RMSE and MAE between the actual and predicted values is calculated and summarized in Table 5.3. Notably, in scenarios with both 30% and 50% packet loss, LSTM consistently outperforms GRU, LIP and MCP, with MCP exhibiting the poorest performance. LIP estimates missing data points by fitting a linear trend to the nine most recent successfully received samples and extrapolating one step ahead.

As mentioned before, LSTM and GRU are both types of recurrent neural network (RNN) layers designed to handle sequential data. GRU is a simplified version

of LSTMs, use a single "update gate" which makes it more straightforward to train and run. As it is shown in Table 5.3, LSTM predictions exhibit significantly higher accuracy compared to GRU. Consequently, for subsequent simulations, only LSTM is employed to maintain clarity and avoid overcrowding the figures.

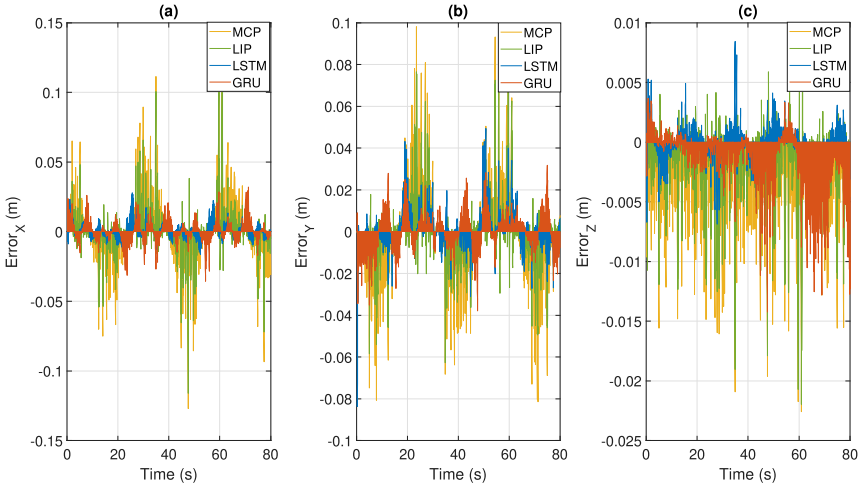


Figure 5.6: Prediction errors. (a) Error for x; (b) Error for y; (c) Error for z

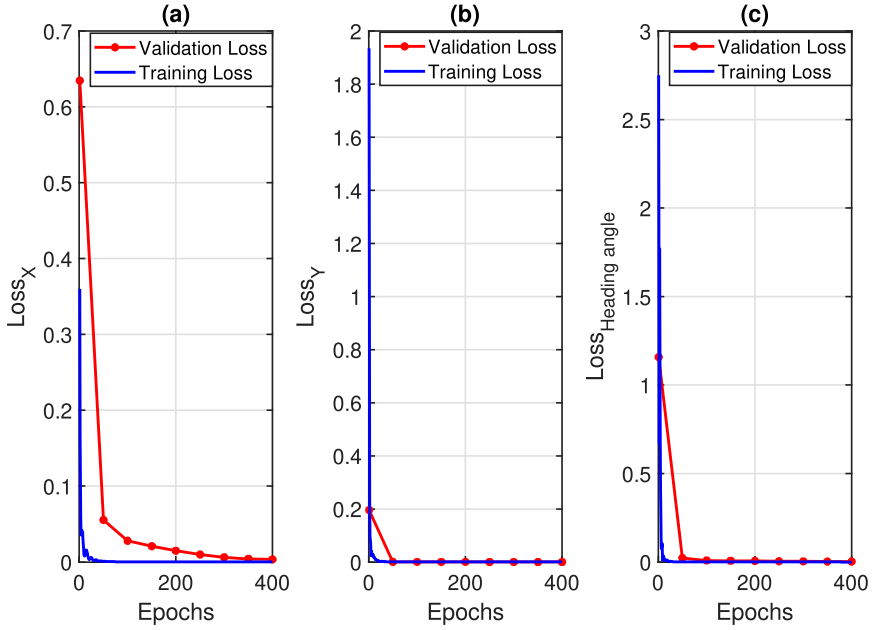


Figure 5.7: Training and validation loss. (a) Loss for x; (b) Loss for y; (c) Loss for z

The results of RMSE and MAE demonstrate that the LSTM predictor achieves approximately 70-80% reduction in prediction error compared to the MCP baseline

Table 5.3: Prediction error under packet loss

Metric	Loss	Method	X (m)	Y (m)	Z (m)
RMSE	30%	LSTM	2.76e-3	4.73e-3	6.92e-4
		GRU	3.50e-3	5.38e-3	1.20e-3
		LIP	5.25e-3	4.97e-3	1.15e-3
		MCP	1.08e-2	1.02e-2	2.30e-3
	50%	LSTM	4.80e-3	7.58e-3	1.14e-3
		GRU	6.50e-3	8.20e-3	2.38e-3
		LIP	1.44e-2	1.28e-2	3.03e-3
		MCP	2.02e-2	1.89e-2	4.44e-3
MAE	30%	LSTM	1.20e-3	1.49e-3	3.14e-4
		GRU	1.54e-3	2.27e-3	4.32e-4
		LIP	2.04e-3	1.84e-3	4.69e-4
		MCP	4.90e-3	4.48e-3	1.15e-3
	50%	LSTM	2.52e-3	3.21e-3	5.82e-4
		GRU	3.38e-3	4.15e-3	1.12e-3
		LIP	6.13e-3	5.97e-3	1.45e3
		MCP	1.04e-2	1.01e-2	2.51e-3

and around 40-50% improvement over LIP corresponding to sub-centimeter accuracy under both 30% and 50% packet loss. This improvement ensures smoother trajectory tracking, tighter inter-vehicle coordination, and more reliable payload balance during cooperative operations.

5.5.2 Simulation Results

As previously highlighted, the performance of the proposed controller was validated by simulating both the controller and the nonlinear quadrotor model 5.4. Each quadrotor's controller, as illustrated in Figure 5.2, was simulated using MATLAB/SIMULINK. Since the presented controller is designed to be robust, the simulations were conducted in the presence of external disturbance and measurement noise inputs. Therefore, the external disturbance represents factors such as wind or impact hits to the quadrotor, and 0.3 m/s^2 increases in quadrotor accelerations ($\ddot{x}$ and $\ddot{y}$) occurring at the time 30 seconds to 30.5 seconds. Additionally, the measurement noise, characterized as white noise, was incorporated into the measurements of $(\dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi})$. It is worth noting that this noise signal was selected to reflect the reality of common IMU quadrotor sensor.

After simulating each quadrotor, the performance of formation control involving three quadrotors, with one designated as the leader is discussed. The followers and the leader aimed to maintain their formation while traversing a predefined path. At regular communication intervals, the leader's position was communicated to all followers via wireless broadcast communication. In instances of packet loss, LSTM was employed to predict the leader's position for the followers. The simulations were conducted with a sampling time of 5 ms and a communication interval of 50 ms. The leader positions shared with the followers were susceptible to packet loss, which is modeled using a Bernoulli distribution with a loss rate of 30%, 50%, and 70%. The impact of packet loss on formation control is more pronounced when the trajectory followed is complex or curvy path. Therefore, leader and the followers are tested by going through spiral and eight shape path.

5.5.2.1 Spiral Path

In this test leader and followers going through a spiral trajectory as shown in Figure 5.8. The leader's initial position is (0, 0, 0), while the followers start from

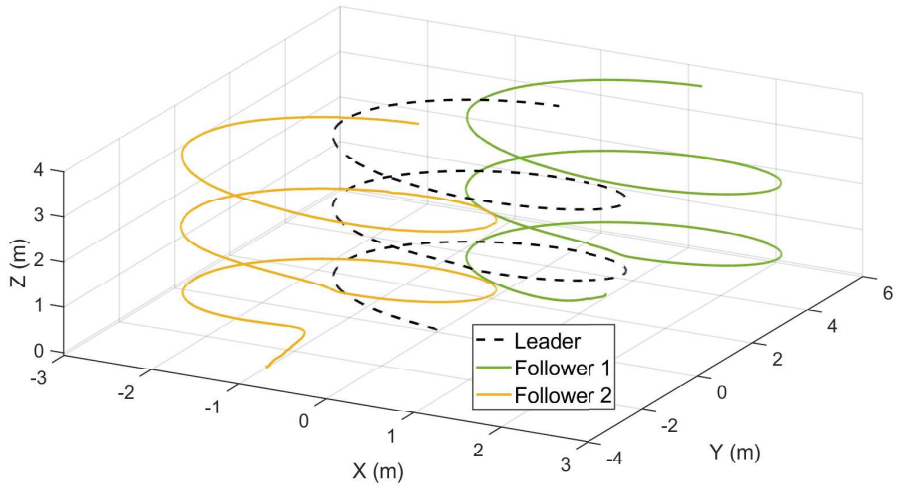


Figure 5.8: 3D Trajectory of the quadrotors during formation control

(1, 3, 0) and (-1, -3, 0), respectively. These initial positions are intentionally different from the desired formation configuration to evaluate the system's ability to converge to the intended formation. The desired relative formation distances between the leader and the followers are defined as (1, 2, 0) for Follower 1 and (-1, -2, 0) for Follower 2. This setup ensures that the followers begin with a spatial offset and must adjust their trajectories to achieve the target formation

under communication uncertainties. This configuration places both followers approximately 1 m behind their desired formation positions along the y-axis.

Figures 5.9 and 5.10 show the distance of first and second follower from the leader under 30% packet loss.

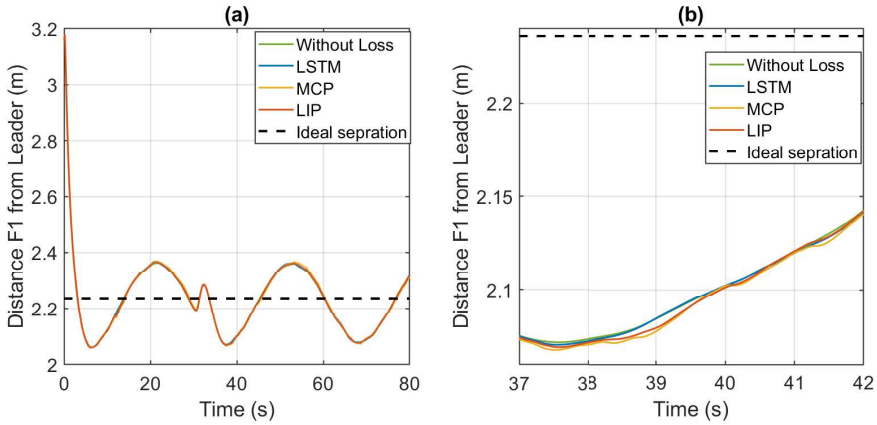


Figure 5.9: Distance of the first follower from the leader for the spiral trajectory (packet loss—30%) (a) Distance of F1 from the leader ; (b) zoomed view distance of F1 from the leader

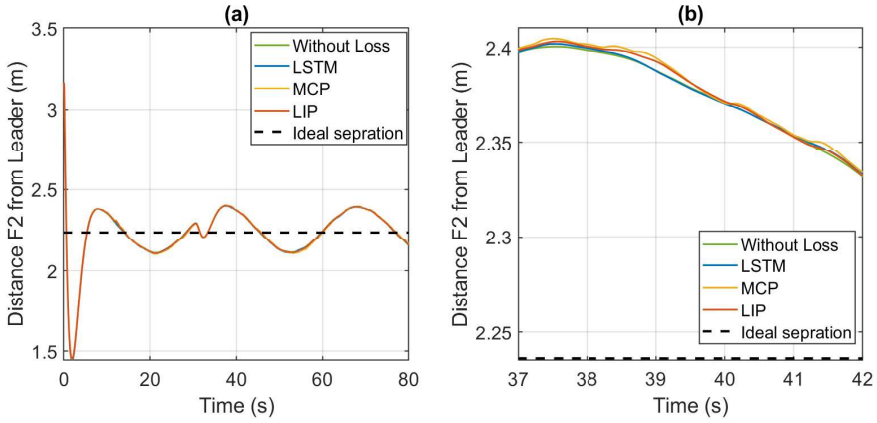


Figure 5.10: Distance of the second follower from the leader for the spiral trajectory (packet loss—30%)(a) Distance of F2 from the leader ; (b) zoomed view distance of F2 from the leader

In both cases, the initial offset is corrected within the first 4–6 seconds, and the desired formation distance is well maintained thereafter. The LSTM-based predictor closely matches the no-loss performance, whereas MCP and LIP exhibit slightly larger fluctuations which has shown in the zoomed view in Figures 5.9 (b) and 5.10 (b). Moreover, Figures 5.9, and 5.10 demonstrate the capability of the proposed robust controller to effectively track the complex spiral-like

trajectory, even in the presence of measurement noise and external disturbances. In Figures 5.9 and 5.10, the external disturbance occurring at the time 30s is promptly compensated.

The spiral trajectory scenario is repeated with a 50% packet loss. The distance for the first and second followers from the leader are depicted in Figure 5.11 and Figure 5.12, respectively.

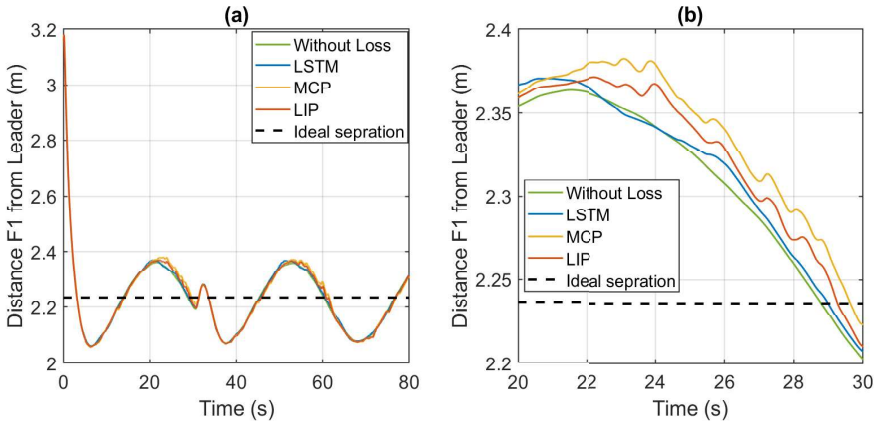


Figure 5.11: Distance of the first follower from the leader for the spiral trajectory (packet loss—50%) (a) Distance of F1 from the leader ; (b) zoomed view distance of F1 from the leader

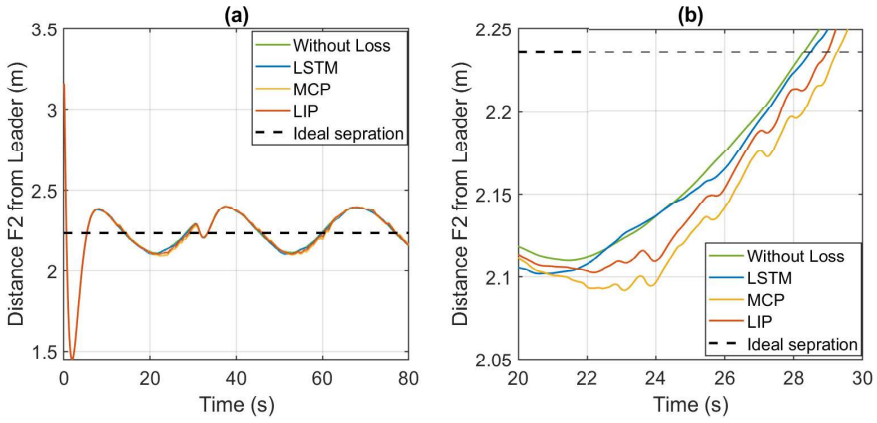


Figure 5.12: Distance of the second follower from the leader for the spiral trajectory (packet loss—50%)(a) Distance of F2 from the leader ; (b) zoomed view distance of F2 from the leader

MCP and LIP demonstrate noticeable performance deterioration, particularly during dynamic spiral segments and in response to disturbances. By contrast, LSTM maintains smooth tracking and preserves both the transient and steady-state behavior observed in the no-loss scenario.

Since LSTM has shown remarkable performance even in the presence of 30% and 50% packet loss, we further evaluate its capabilities by repeating the spiral trajectory scenario with 70% packet loss. As packet loss increases to 70%, LIP performance substantially degrades and MCP suffers further increases in deviation from the leader as shown in Figures 5.13 and 5.14.

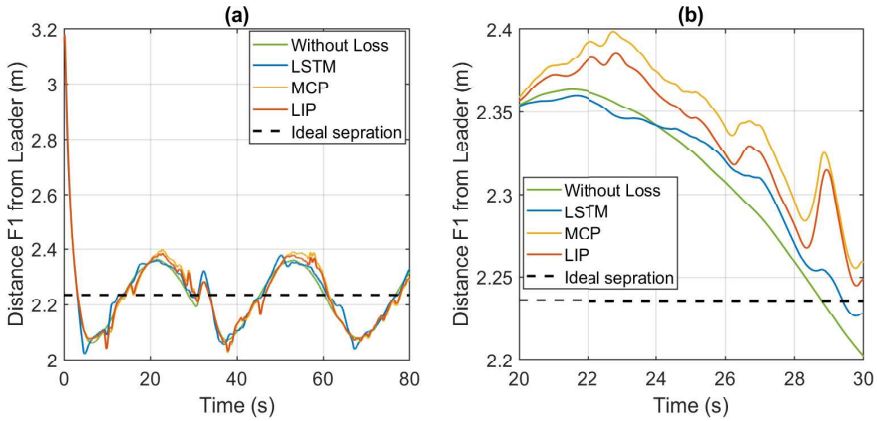


Figure 5.13: Distance of the first follower from the leader for the spiral trajectory (packet loss—70%) (a) Distance of F1 from the leader ; (b) zoomed view distance of F1 from the leader

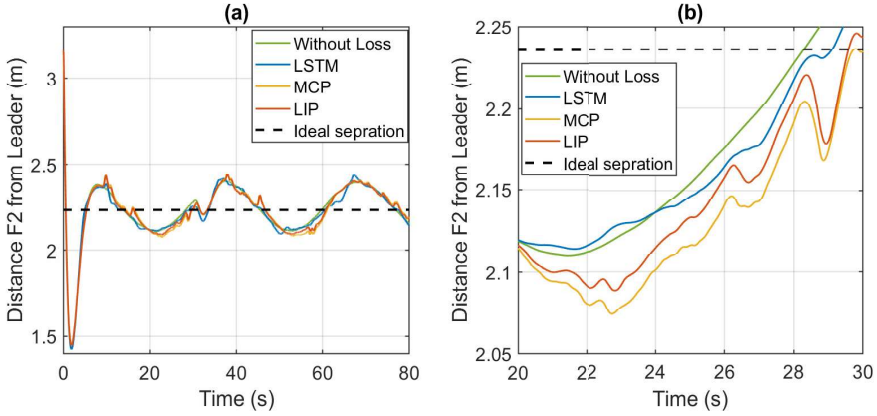


Figure 5.14: Distance of the second follower from the leader for the spiral trajectory (packet loss—70%) (a) Distance of F2 from the leader ; (b) zoomed view distance of F2 from the leader

Importantly, the LSTM predictor remains stable and close to the no-loss case, even though occasional spikes may occur in highly curved trajectory sections. This robustness can be attributed to the LSTM's capability to infer short-term temporal structure in the leader motion, effectively bridging multiple consecutive missing packets.

As Tables 5.4 and 5.5 show, MCP exhibits the poorest performance across all packet-loss levels. LIP performs better than MCP but consistently worse than LSTM. In absolute terms, the LSTM predictor keeps the RMSE and MAE close to no-loss level for 30–50% loss (differences less than 0.5 cm), and delivers a meaningful reduction at 70% loss. LSTM advantage becomes more pronounced as packet loss increases, especially when compared to MCP, where errors can rise by up to 30–40%. Even against LIP, LSTM retains a consistent margin of improvement of 8–18% across all axes and metrics base on packet loss rate.

Table 5.4: RMSE of follower error for the spiral path.

Packet Loss	Prediction Method	RMSE-x (m)	RMSE-y (m)	RMSE-z (m)
0%	-	1.05×10^{-1}	1.42×10^{-1}	1.39×10^{-2}
30%	LSTM	1.06×10^{-1}	1.421×10^{-1}	1.40×10^{-2}
	LIP	1.07×10^{-1}	1.431×10^{-1}	1.43×10^{-2}
	MCP	1.08×10^{-1}	1.433×10^{-1}	1.51×10^{-2}
50%	LSTM	1.07×10^{-1}	1.423×10^{-1}	1.41×10^{-2}
	LIP	1.08×10^{-1}	1.435×10^{-1}	1.52×10^{-2}
	MCP	1.11×10^{-1}	1.44×10^{-1}	1.65×10^{-2}
70%	LSTM	1.09×10^{-1}	1.43×10^{-1}	1.44×10^{-2}
	LIP	1.22×10^{-1}	1.44×10^{-1}	1.87×10^{-2}
	MCP	1.37×10^{-1}	1.46×10^{-1}	1.97×10^{-2}

Table 5.5: MAE of follower error for the spiral path.

Packet Loss	Prediction Method	RMSE-x (m)	RMSE-y (m)	RMSE-z (m)
0%	-	9.24×10^{-2}	9.98×10^{-2}	1.35×10^{-2}
30%	LSTM	9.26×10^{-2}	1.01×10^{-1}	1.38×10^{-2}
	LIP	9.40×10^{-2}	1.07×10^{-1}	1.43×10^{-2}
	MCP	9.56×10^{-2}	1.11×10^{-1}	1.50×10^{-2}
50%	LSTM	9.27×10^{-2}	1.02×10^{-1}	1.39×10^{-2}
	LIP	9.59×10^{-2}	1.16×10^{-1}	1.52×10^{-2}
	MCP	9.87×10^{-2}	1.22×10^{-1}	1.63×10^{-2}
70%	LSTM	9.36×10^{-2}	1.05×10^{-1}	1.4×10^{-2}
	LIP	1.10×10^{-1}	1.28×10^{-1}	1.80×10^{-2}
	MCP	1.25×10^{-1}	1.46×10^{-1}	2.03×10^{-2}

5.5.2.2 Eight Shape Path

In this test leader and followers going through an eight-shape trajectory as shown in Figure 5.15. The leader's initial position is (0, 0, 0), while the followers start

from (1, 5, 0) and (-1, -5, 0), respectively. Similar to spiral scenario, the followers use either the LSTM-based predictor, LIP, and MCP to estimate the missing leader position.

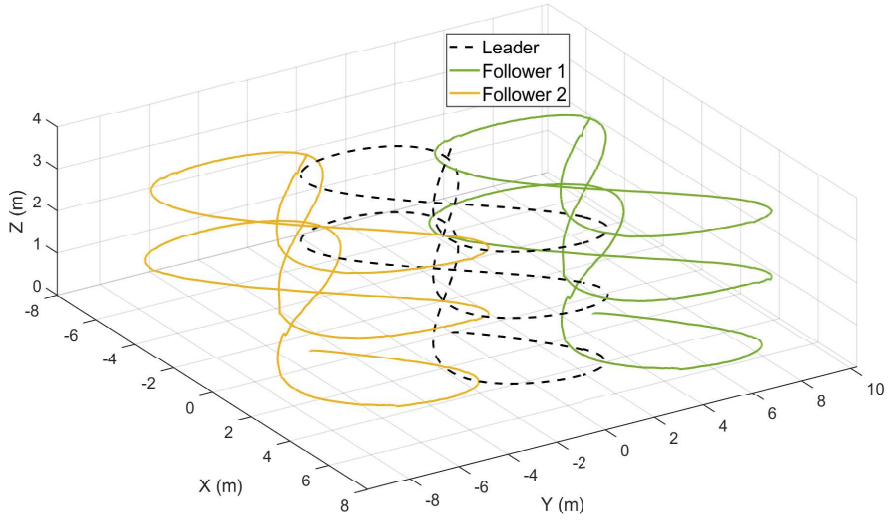


Figure 5.15: 3D Trajectory of the quadrotors during eight shape formation control

Figures 5.16 and 5.17 depict the distance of each follower from the leader under 30% packet loss.

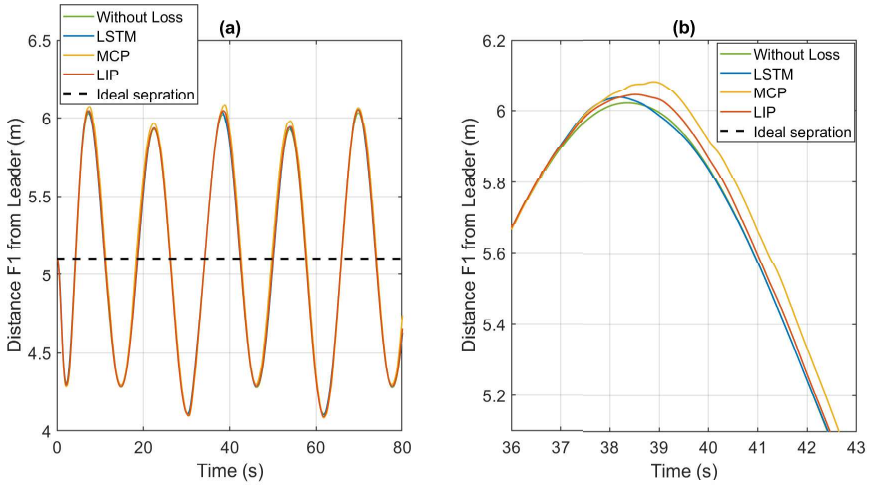


Figure 5.16: Distance of the first follower from the leader for the eight shape trajectory (packet loss—30%)

All three prediction methods produce trajectories that remain close to the no-loss case. The LSTM-based prediction is the closest to the ideal separation, whereas

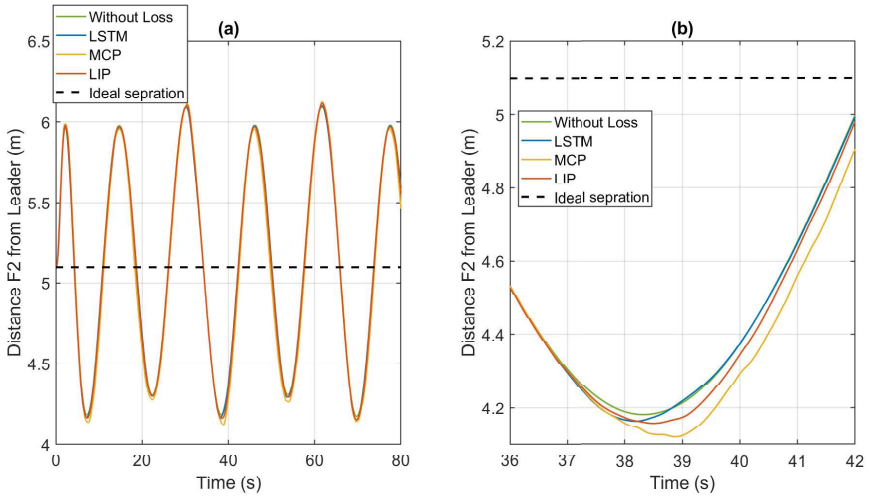


Figure 5.17: Distance of the second follower from the leader for the eight shape trajectory (packet loss—30%)

LIP and MCP introduce slightly larger oscillations. However, at this loss level the differences among the methods are small, confirming that the formation controller maintains satisfactory tracking.

When the packet loss increases to 50%, the influence of the prediction strategy becomes more evident (Figures 5.18 and 5.19).

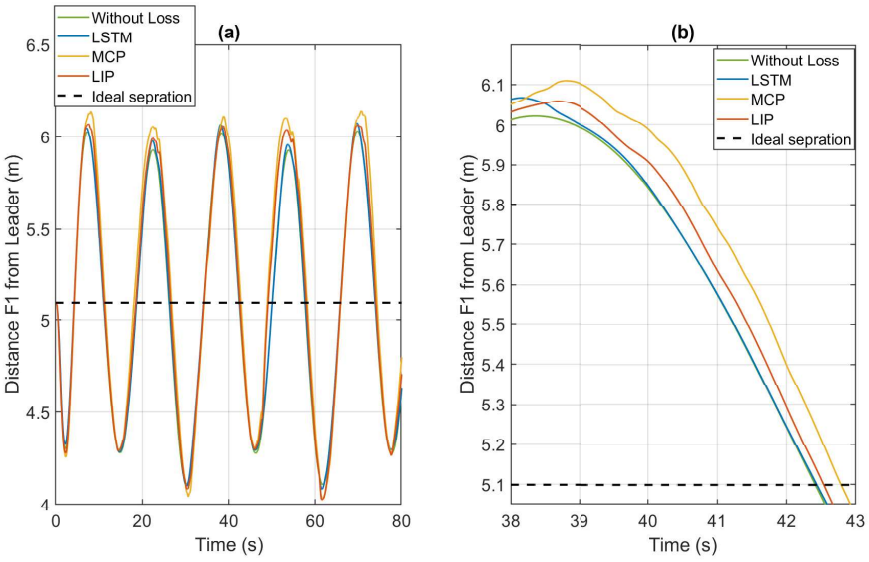


Figure 5.18: Distance of the first follower from the leader for the eight shape trajectory (packet loss—50%)

The LSTM-based predictor maintains trajectories close to the no-loss reference, whereas MCP begins to deviate more noticeably, especially near turning points of

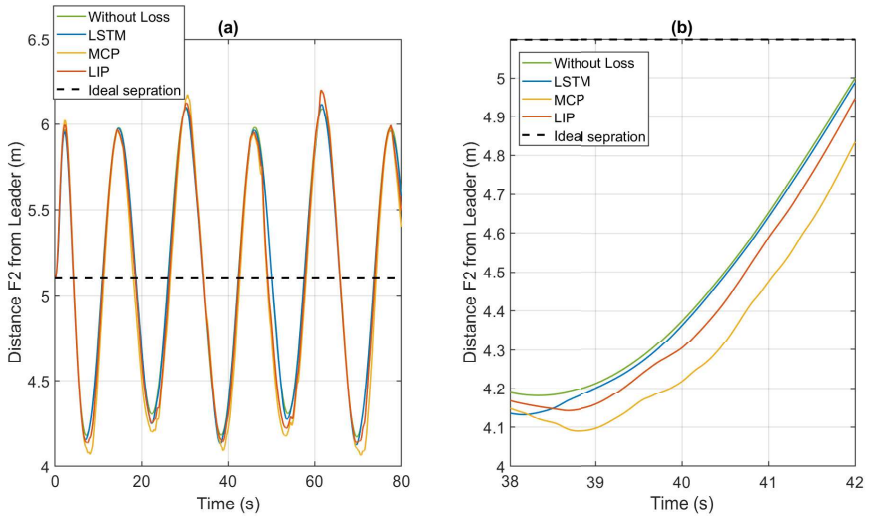


Figure 5.19: Distance of the second follower from the leader for the eight shape trajectory (packet loss—50%)

the figure-eight where state changes are faster and prediction errors accumulate.

The most challenging case occurs at 70% packet loss (Figures 5.20 and 5.21). Although all methods exhibit larger deviations than in the previous scenarios, the degradation is significantly different across predictors. LSTM maintains formation tracking with only a moderate increase in error, whereas both LIP and MCP show clear drift from the ideal separation line, particularly during periods of rapid curvature change.

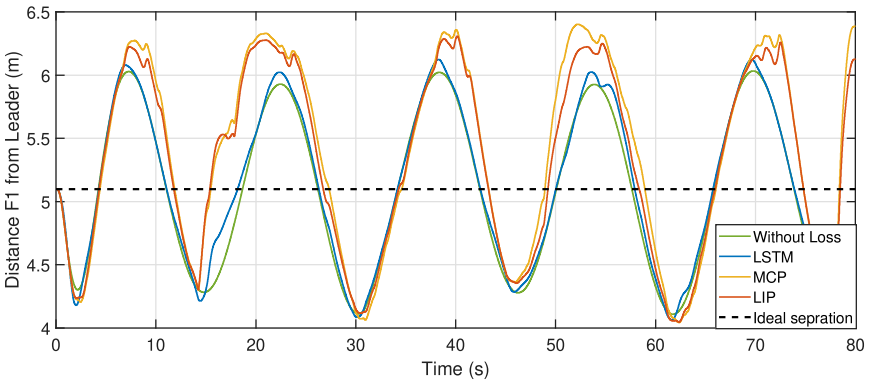


Figure 5.20: Distance of the first follower from the leader for the eight shape trajectory (packet loss—70%)

Overall, across all packet loss levels, the LSTM-based prediction consistently maintains the smallest deviation from the ideal formation, remaining close to the 0% loss baseline even under 70% packet loss. At low loss rates (30%),

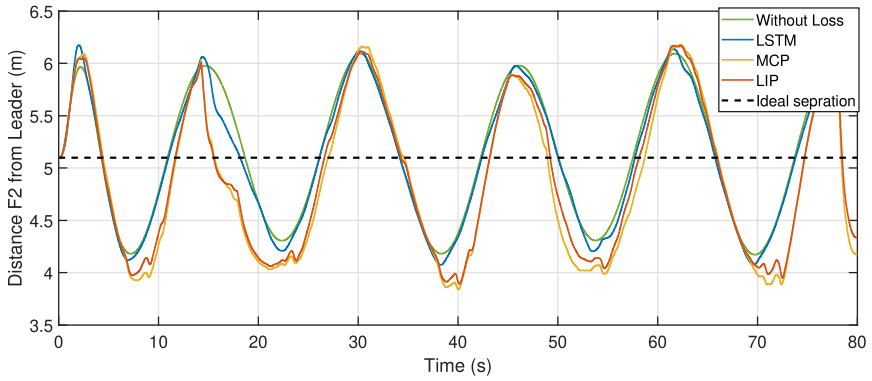


Figure 5.21: Distance of the second follower from the leader for the eight shape trajectory (packet loss—70%)

all methods perform comparably, but as packet loss increases, the gap widens: LSTM offers moderate improvement at 50% loss, and a substantial advantage at 70% loss, where both LIP and MCP exhibit clear degradation. The improvement is most pronounced along the x and y axes, where motion is more dynamic, whereas the z-axis error remains small for all methods. These results highlight that the recurrent structure of LSTM enables effective prediction of the leader's motion trend, allowing the controller to preserve formation accuracy even when more than two-thirds of the communication packets are lost. Over the eight-shape trajectory, LSTM reduces the overall tracking error by roughly 10–15% compared to LIP and 15–25% compared to MCP, with the largest gains observed at high packet-loss levels (70%).

Table 5.6: RMSE of follower error for the eight shape path.

Packet Loss	**Prediction Method**	**RMSE-x (m)**	**RMSE-y (m)**	**RMSE-z (m)**
0%	-	3.32×10^{-1}	6.38×10^{-1}	1.35×10^{-2}
30%	LSTM	3.35×10^{-1}	6.42×10^{-1}	1.38×10^{-2}
	LIP	3.53×10^{-1}	6.46×10^{-1}	1.42×10^{-2}
	MCP	3.56×10^{-1}	6.53×10^{-1}	1.51×10^{-2}
50%	LSTM	3.42×10^{-1}	6.43×10^{-1}	1.39×10^{-2}
	LIP	3.74×10^{-1}	6.64×10^{-1}	1.53×10^{-2}
	MCP	3.87×10^{-1}	6.89×10^{-1}	1.69×10^{-2}
70%	LSTM	3.50×10^{-1}	6.45×10^{-1}	1.41×10^{-2}
	LIP	4.65×10^{-1}	7.63×10^{-1}	1.95×10^{-2}
	MCP	4.98×10^{-1}	8.18×10^{-1}	2.16×10^{-2}

Table 5.7: MAE of follower error for the eight shape path.

Packet Loss	Prediction Method	MAE-x (m)	MAE-y (m)	MAE-z (m)
0%	-	2.98×10^{-1}	5.70×10^{-1}	1.36×10^{-2}
30%	LSTM	3.02×10^{-1}	5.72×10^{-1}	1.37×10^{-2}
	LIP	3.15×10^{-1}	5.80×10^{-1}	1.42×10^{-2}
	MCP	3.16×10^{-1}	5.87×10^{-1}	1.49×10^{-2}
50%	LSTM	3.06×10^{-1}	5.73×10^{-1}	1.39×10^{-2}
	LIP	3.34×10^{-1}	5.97×10^{-1}	1.50×10^{-2}
	MCP	3.44×10^{-1}	6.18×10^{-1}	1.63×10^{-2}
70%	LSTM	3.11×10^{-1}	5.78×10^{-1}	1.43×10^{-2}
	LIP	4.11×10^{-1}	6.79×10^{-1}	1.77×10^{-2}
	MCP	4.37×10^{-1}	7.23×10^{-1}	1.94×10^{-2}

5.6 Concluding Remarks

This chapter addressed the problem of formation control for a network of quadrotor UAVs operating over an unreliable wireless network, in which communication packets between agents may be dropped. The proposed solution combined traditional linear controllers with an LSTM-based predictor in order to mitigate the negative effect of packet loss on formation tracking.

The quadrotor dynamics were modelled and linearised, and a mixed H_2/H_∞ controller was used for attitude and altitude regulation, with an additional PD layer for positional control. This architecture was then extended to the multi-agent case, where position measurements from the leader were communicated to all followers at a fixed communication interval. In the presence of packet loss, basic strategies such as MCP and LIP were implemented and used as baselines.

Packet loss was found to induce tracking errors up to 6 cm for typical formation distances in the considered scenarios. To improve robustness, an LSTM network was introduced to replace missing leader position samples during communication dropouts. The LSTM and GRU were trained and tuned using Bayesian optimisation in order to balance accuracy and computational cost. The final model required only past leader positions as input, making it suitable for real-time onboard use.

Simulation results for two representative 3D leader trajectories—a spiral path and a figure-eight path—demonstrated that the proposed LSTM-based predictor consistently outperforms both MCP and LIP across packet loss levels of 30%, 50%, and 70%. While the average tracking improvements delivered by the LSTM

predictor may appear modest at low packet-loss levels (on the order of 10-15% over LIP and MCP), its contribution becomes increasingly critical as network reliability degrades. Under severe packet loss (70%), LSTM reduces formation-tracking errors by up to 28% (relative to LIP) and 35% (relative to MCP), while also suppressing peak deviations that can destabilize the entire formation. Importantly, this robustness is achieved *without increasing communication frequency* or modifying the baseline controller, demonstrating the scalability and practical utility of learning-based compensation in resource-constrained networked robotics.

These results address our hypothesis stated in 1, namely: **Can this solution be extended to other applications and controllers? For example, could combining LSTM with linear controllers like H_2/H_∞ and PD improve the performance of formation control in quadrotors, which operate in three dimensions, under packet loss?** Notably, the same predictor architecture previously applied to the AGV system (with an LQR controller) in chapter 3 was effective here when combined with an H_2/H_∞ and PD control structure for quadrotors. The independence of the predictor from the specific system and control approach highlights its versatility and potential for reuse in other cooperative robotic systems.

Although uncertainty quantification was not pursued in this work—consistent with common practice in UAV formation control—uncertainty-aware prediction, incorporating such metrics, and investigating the impact of uncertainty propagation through the control loop, are promising avenues for future research.

In summary, combining linear controllers with data-driven LSTM prediction results in a robust formation control strategy that scales gracefully under packet loss and is adaptable across platforms and control architectures. The method enhances robustness without requiring increased communication frequency, offering potential benefits in bandwidth-limited or congested wireless environments.

Chapter 6

Conclusion and Future Work

6.1 Thesis Summary and Contributions

This thesis has addressed the formation control problem for multi-agent systems, specifically focusing on non-holonomic differential drive Autonomous Ground Vehicles (AGVs) and quadrotor Unmanned Aerial Vehicles (UAVs) operating under challenging communication conditions, particularly packet loss. By combining Long Short-Term Memory (LSTM) networks with traditional control techniques, we demonstrated significant improvements in system performance and robustness.

The integration of deep learning with established control methods has shown promising results, especially in scenarios where communication disruptions can adversely affect the performance of traditional controllers. By training LSTM models to predict the positions of agents during packet loss, we demonstrated that it is possible to significantly reduce the impact of packet loss on system performance and enhancing overall system reliability. The following text summaries the key contributions of this doctoral thesis.

1. **Analyzing Impact of Packet Loss on Leader-Follower Formation Control in Multi-Robot Systems**: This thesis investigates how packet loss in wireless communication affects leader–follower formation control in multi-robot systems, focusing on AGVs and quadrotor UAVs. Both the leader and followers are controlled using their respective motion controllers; however, only the followers depend on the leader's broadcasted position to maintain formation. For AGVs, a Linear Quadratic Regulator (LQR) was used for trajectory tracking and formation maintenance, whereas for quadrotors a

mixed H_2/H_∞ controller (attitude/altitude) combined with a PD controller (position) was implemented. These controllers remain active regardless of packet loss; what is affected is the information available to the follower about the leader's position. Packet loss, modeled using a Bernoulli distribution, was applied to the communication channel transmitting the leader's position to the followers. During packet loss, the followers cannot update the formation reference and must rely on a compensation method. A baseline strategy, Memory Consensus Protocol (MCP), was first implemented, where the follower simply holds the last received leader position until new data arrives. Simulation results showed that MCP leads to a formation error increase of approximately 20 cm for AGVs and 6 cm for drones under realistic packet loss conditions. These findings confirmed that even with stable motion controllers, communication loss degrades formation accuracy, motivating the development of more advanced prediction-based solutions like LSTM.

2. **Innovative Integration of LSTM with Linear Controllers to Compensate Packet Loss for Formation Control**: In practical multi-robot and industrial systems, linear controllers such as LQR, PD, and mixed H_2/H_∞ remain the dominant choice due to their simplicity, reliability, and well-established tuning methods. Instead of replacing these controllers with more complex nonlinear or adaptive alternatives, this work introduces LSTM prediction as an additional layer that enhances communication robustness while preserving the original control architecture. The LSTM model is used during packet loss and predicts the missing leader-state information, allowing followers to maintain formation without modifying the underlying control laws. This approach keeps the system fully compatible with existing industrial implementations, while significantly improving performance under 30%, 50%, and 70% packet loss compared with memory-based techniques such as MCP and Linear Interpolation Predictor (LIP).

3. **LSTM Performance Analysis Under Packet Loss**: The LSTM-based predictor was evaluated against the MCP and the LIP for packet loss rates of 30%, 50%, and 70%. In all cases, the no-loss baseline resulted in the lowest formation tracking error, as expected. However, when packet loss occurred, the LSTM consistently provided the smallest degradation in performance compared to MCP and LIP. The results show that, although the tracking error increases with higher packet loss for all methods, the rate of deterioration is significantly lower for the LSTM predictor. At 70%

packet loss, LSTM reduces the RMSE by up to 35-45% relative to MCP and 20-30% relative to LIP, depending on the axis and trajectory type. This confirms that the LSTM model offers the most robust behaviour under communication dropouts, while still remaining below the performance of the ideal no-loss case. Unlike MCP, which holds the last received value, and LIP, which assumes linear motion between packets, the LSTM uses learned temporal patterns from the leader trajectory to provide smoother and more accurate predictions when data is unavailable. As a result, the followers maintain a formation closer to the desired separation compared to the other prediction strategies, particularly under severe packet loss conditions. The results support our two hypotheses stated in Chapter 1, one for AGVs, namely: **Can the combination of a deep learning method such as LSTM in combination with a linear control strategy, such as LQR, improve the performance of formation control for AGVs under conditions of packet loss? Can the learning algorithm compensate for the effects of packet loss in the system?** The second hypothesis focuses on drones, namely:**Can this solution be extended to other applications and controllers? For example, could combining LSTM with linear controllers like H_2/H_∞ and PD improve the performance of formation control in quadrotors, which operate in three dimensions, under packet loss?**

4. **Enhanced Performance**: Simulation results revealed that LSTM-based prediction models significantly outperformed traditional memory-based control approach namely the MCP, LIP, and Gated Recurrent Unit (GRU) networks. LSTM consistently showed better accuracy compared to MCP and LIP, particularly under packet loss scenarios in both AGVS and drones applications in complex trajectories like circular, eight shape, and spiral path. The evaluation involved initializing the followers behind the leader with specific offsets for both AGVs and quadrotors, requiring them to adjust for these initial conditions to maintain the desired formation. Additionally, the LSTM model effectively managed disturbances and noise in quadrotor operations, resulting in enhanced accuracy and responsiveness under packet loss conditions.

5. **Comparative Effectiveness**: The findings emphasized that the proposed LSTM-enhanced control approach surpassed traditional time-triggered controllers, such as backstepping, in terms of performance under packet loss. While it achieved comparable performance to event-triggered controllers

like Sliding Mode Control (SMC), our work provides insights into the conditions where LSTM can offer similar, if not better, results. Additionally, we designed an input shaping filter to cope with delay, and together with LSTM, they effectively compensated for both delay and packet loss for formation control of non-holonomic AGVs. This comprehensive analysis serves as a valuable reference for selecting appropriate control strategies in multi-agent systems, particularly for AGVs. This supports of our hypothesis stated in Chapter 1, namely: **Can combining LSTM with linear control strategies achieve performance comparable to advanced nonlinear controllers, such as sliding mode control, when dealing with packet loss and communication delays?**

In conclusion, this thesis has shown that combining LSTM with traditional control approaches can offer a powerful solution for formation control in multi-agent systems under packet loss. The hybrid approach not only improves robustness and scalability but also presents a viable alternative to existing time-triggered nonlinear controllers, demonstrating significant advancements in communication efficiency and overall system performance. This work enables further exploration of the integration of machine learning with control theory to address complex, real-world challenges.

6.2 Future work

In this section, we highlight the future directions for extending the proposed method that were not extensively explored within this thesis.

1. **Hardware Implementation**: While the current research relies heavily on simulations, the next steps should focus on the physical implementation of the LSTM-based control strategies in real-world settings. Testing in a controlled environment with actual quadrotors or AGVs will provide crucial insights into the practical performance of the system, including how it handles delays, packet loss, and unexpected disturbances. This will also help validate the simulation results and provide a basis for further refinements to the control algorithms.

2. **Exploration of Event-Based Control**: Investigate the integration of learning algorithms with event-based control strategies, moving beyond the current focus on time-triggered control. Event-based approaches could enhance efficiency and responsiveness, particularly in dynamic environ-

ments requiring timely decision-making. By adapting control actions based on events rather than fixed intervals, the system could improve its performance while minimizing communication overhead. For more information on combination of event-based control with learning approaches please refer to our survey paper [SINAR+22].

3. **Generalization to Unseen Trajectories Using Extensive Datasets and Including Statistical Tests**: In this thesis, the LSTM predictor was trained on two representative reference paths (Circular trajectories and an eight-shaped paths) and demonstrated good tracking performance under packet loss. A natural extension of this work is to investigate whether the model can be trained on a richer family of motion primitives—such as combinations of straight lines and circular arcs, which are fundamental building blocks in motion planning, suggests that complex paths can often be broken down into simpler geometric components. In particular, for non-holonomic systems such as AGVs or UAVs, many feasible trajectories can be modeled using a series of circular arcs and straight-line segments, as shown in algorithms like Dubins paths and Reeds-Shepp paths [RRK+18]. By training LSTM models extensively on a diverse set of paths composed of circles and lines, the LSTM could learn the fundamental dynamics associated with following curved and straight paths. Given the generalization capabilities of deep learning models, this approach could potentially enable LSTM networks to predict and follow arbitrary, more complex paths that are encountered in real-world applications. The hypothesis is that if the LSTM model can learn to navigate basic path components (lines and circles), it should be capable of handling more intricate and unforeseen paths by recognizing and combining these simpler motions. This approach could enable LSTM-based controllers to handle unknown or dynamic environments, improving adaptability and robustness in applications like autonomous vehicle navigation and multi-agent coordination without the need for specialized path data. The aim of this future work would not be to replace the controller, but to extend the prediction model so that it can recognize and reconstruct more complex paths by learning fundamental geometric components. It would be great to also include multiple randomized trials and applying statistical tests to further quantify the variability and significance of performance differences.

4. **Comparison with Other Learning Methods**: Comparative studies involving other learning methods, such as reinforcement learning or hybrid

approaches that integrate traditional control techniques with deep learning, could provide valuable insights into the strengths and weaknesses of various strategies. This comparison could reveal specific conditions under which certain methods excel, offering guidance for selecting the most effective control strategies for different multi-agent systems

5. **Exploration of Distributed Control**: It would be interesting to transition from a decentralized control approach, where only the leader communicates with the followers, to a distributed control approach that enables communication among followers. This could enhance the coordination among agents, allowing them to share information and adjust their actions collaboratively. Research could examine the dynamics of such systems, exploring how followers can maintain formation and react to each other's movements, thereby improving overall system robustness.

6. **Exploring Advanced Deep Learning Models for Improved Formation Control**: Future studies could delve into integrating cutting-edge deep learning algorithms, such as Transformers or attention-based models, into the formation control of multi-agent systems. These models are designed to handle sequential data and could enhance prediction accuracy by focusing on relevant past information more effectively. Research could investigate how these models perform under various conditions of packet loss and dynamic environments, potentially leading to better adaptability and resilience in real-world applications.

References

[ACA07] José Luís Azevedo, Bernardo Cunha, and Luís Almeida. Hierarchical distributed architectures for autonomous mobile robots: a case study. In *2007 IEEE Conference on Emerging Technologies and Factory Automation (EFTA 2007)*, pages 973–980. IEEE, 2007.

[AdLF17] Florent Altché and Arnaud de La Fortelle. An lstm network for highway trajectory prediction. In *2017 IEEE 20th international conference on intelligent transportation systems (ITSC)*, pages 353–359. IEEE, 2017.

[AFH+98] Rachid Alami, Sara Fleury, Matthieu Herrb, Félix Ingrand, and Frédéric Robert. Multi-robot cooperation in the martha project. *IEEE Robotics & Automation Magazine*, 5(1):36–47, 1998.

[AIK21] Sami Al Issa and Indrani Kar. Design and implementation of event-triggered adaptive controller for commercial mobile robots subject to input delays and limited communications. *Control Engineering Practice*, 114:104865, 2021.

[ALB19] Pramod Abichandani, Kyle Levin, and Donald Bucci. Decentralized formation coordination of multiple quadcopters under communication constraints. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3326–3332. IEEE, 2019.

[Alr16] Wesam M Jasim Alrawi. *Improving Leader-Follower Formation Control Performance for Quadrotors*. PhD thesis, University of Essex Colchester, UK, 2016.

[AM18] Amin Abbasi and Ata Jahangir Moshayedi. Trajectory tracking of two-wheeled mobile robots, using lqr optimal control method, based on computational model of khepera iv. *Journal of Simula-*

tion and Analysis of Novel Technologies in Mechanical Engineering, 10(3):41–50, 2018.

[AMBR17] Javier Alonso-Mora, Stuart Baker, and Daniela Rus. Multi-robot formation control and object transport in dynamic environments via constrained optimization. *The International Journal of Robotics Research*, 36(9):1000–1021, 2017.

[AN17] Ehsan Abbasi and Nader Naghavi. Offline auto-tuning of a pid controller using extended classifier system (xcs) algorithm. 2017.

[Asa89] Hajime Asama. Design of an autonomous and distributed robot system: Actress. In *Proc. of IEEE RSJ Int. Conf. Intelligent Robots and Systems*, pages 283–290, 1989.

[AWL+23] Xing An, Celimuge Wu, Yangfei Lin, Min Lin, Tsutomu Yoshinaga, and Yusheng Ji. Multi-robot systems and cooperative object transport: Communications, platforms, and challenges. *IEEE Open Journal of the Computer Society*, 4:23–36, 2023.

[BA98] Tucker Balch and Ronald C Arkin. Behavior-based formation control for multirobot teams. *IEEE transactions on robotics and automation*, 14(6):926–939, 1998.

[BB91] Stephen P Boyd and Craig H Barratt. *Linear controller design: limits of performance*, volume 78. Citeseer, 1991.

[BEGFB94] Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear matrix inequalities in system and control theory*. SIAM, 1994.

[Ben88] Gerardo Beni. The concept of cellular robot. In *Proceed. 3rd IEEE Symposium on Intelligent Control, Arlington*, pages 57–61, 1988.

[Ber16] RA Berkeveld. *Design and Implementation of Unicycle AGVs for Cooperative Control*. PhD thesis, Eindhoven University of Technology Eindhoven, The Netherlands, 2016.

[BG21] Dimitri Bertsekas and Robert Gallager. *Data networks*. Athena Scientific, 2021.

[BLM+14] Luca Rosario Buonocore, Vincenzo Lippiello, Sabato Manfredi, Fabio Ruggiero, and Bruno Siciliano. Effects of packet losses on

formation control of unmanned aerial vehicles. *IFAC Proceedings Volumes*, 47(3):1234–1240, 2014.

[BMSP21] Rakesh P Borase, DK Maghade, SY Sondkar, and SN Pawar. A review of pid control, tuning methods and applications. *International Journal of Dynamics and Control*, 9(2):818–827, 2021.

[BS05] Samir Bouabdallah and Roland Siegwart. Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. In *Proceedings of the 2005 IEEE international conference on robotics and automation*, pages 2247–2252. IEEE, 2005.

[BS07] Samir Bouabdallah and Roland Siegwart. Full control of a quadrotor. In *2007 IEEE/RSJ international conference on intelligent robots and systems*, pages 153–158. Ieee, 2007.

[BTA06] Douglas A Bristow, Marina Tharayil, and Andrew G Alleyne. A survey of iterative learning control. *IEEE control systems magazine*, 26(3):96–114, 2006.

[BYPG21] Chengchao Bai, Peng Yan, Wei Pan, and Jifeng Guo. Learning-based multi-robot formation control with obstacle avoidance. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[CB15] Lei Chen and Ma Baoli. A nonlinear formation control of wheeled mobile robots with virtual structure approach. In *2015 34th Chinese Control Conference (CCC)*, pages 1080–1085. IEEE, 2015.

[CCG23] Roberto Cahuantzi, Xinye Chen, and Stefan Güttel. A comparison of lstm and gru networks for learning symbolic sequences. In *Science and Information Conference*, pages 771–785. Springer, 2023.

[CD20] Fei Chen and Dimos V Dimarogonas. Leader–follower formation control with prescribed performance guarantees. *IEEE Transactions on Control of Network Systems*, 8(1):450–461, 2020.

[CE14] Dong Eui Chang and Yongsoon Eun. Construction of an atlas for global flatness-based parameterization and dynamic feedback linearization of quadcopter dynamics. In *53rd IEEE Conference on Decision and Control*, pages 686–691. IEEE, 2014.

[CFKM95] Y Uny Cao, Alex S Fukunaga, Andrew B Kahng, and Frank Meng. Cooperative mobile robotics: Antecedents and directions. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, volume 1, pages 226–234. IEEE, 1995.

[CFKP09] Peng Cheng, Jonathan Fink, Vijay Kumar, and Jong-Shi Pang. Cooperative towing with multiple robots. 2009.

[CG96] Mahmoud Chilali and Pascal Gahinet. H/sub/spl infin//design with pole placement constraints: an lmi approach. *IEEE Transactions on automatic control*, 41(3):358–367, 1996.

[CGP15] Rudy Cepeda-Gomez and Luis F Perico. Formation control of nonholonomic vehicles under time delayed communications. *IEEE transactions on automation science and engineering*, 12(3):819–826, 2015.

[CHB08] Chih-Chung Chen, Sandra Hirche, and Martin Buss. Sampled-data networked control systems with random time delay. *IFAC Proceedings Volumes*, 41(2):11594–11599, 2008.

[CHLG17] Zhitang Chen, Ke He, Jian Li, and Yanhui Geng. Seq2img: A sequence-to-image based approach towards ip traffic classification using convolutional neural networks. In *2017 IEEE International conference on big data (big data)*, pages 1271–1276. IEEE, 2017.

[CHN19] Linying Chen, Hans Hopman, and Rudy R Negenborn. Distributed model predictive control for cooperative floating object transport with multi-vessel systems. *Ocean Engineering*, 191:106515, 2019.

[CKDC24] Shaofeng Chen, Yu Kang, Jian Di, and Yang Cao. Graph convolutional flocking control for unmanned aerial vehicles with packet dropouts. *IEEE Transactions on Industrial Electronics*, 2024.

[ÇLDB+20] Ozan Çatal, Sam Leroux, Cedric De Boom, Tim Verbelen, and Bart Dhoedt. Anomaly detection for autonomous guided vehicles using bayesian surprise. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8148–8153. IEEE, 2020.

[CLLH22] Bor-Sen Chen, Yi-Chen Liu, Min-Yen Lee, and Chih-Lyang Hwang. Decentralized h pid team formation tracking control of large-scale quadrotor uavs under external disturbance and vortex coupling. *IEEE Access*, 10:108169–108184, 2022.

[CMPT06] Luca Consolini, Fabio Morbidi, Domenico Prattichizzo, and Mario Tosques. On the control of a leader-follower formation of nonholonomic mobile robots. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 5992–5997. IEEE, 2006.

[CW05] Yang Quan Chen and Zhongmin Wang. Formation control: a review and a new consideration. In *2005 IEEE/RSJ International conference on intelligent robots and systems*, pages 3181–3186. IEEE, 2005.

[CYWC07] Ian D Cowling, Oleg A Yakimenko, James F Whidborne, and Alastair K Cooke. A prototype of an autonomous controller for a quadrotor uav. In *2007 European Control Conference (ECC)*, pages 4001–4008. IEEE, 2007.

[CZIL23] Lei Cao, Da-Wei Zhang, Clara Mihaela Ionescu, and Guo-Ping Liu. A distributed predictive formation control strategy for cyber-physical multi-agent systems under communication constraints. *Information Sciences*, 641:119092, 2023.

[Des03] S Naidu Desineni. Optimal control systems. *Electrical engineering textbook series, CRC Press*, 2003.

[DJ24] Gregory Dudek and Michael Jenkin. *Computational principles of mobile robotics*. Cambridge university press, 2024.

[DKJ+23] Rajdeep Dutta, Harikumar Kandath, Senthilnath Jayavelu, Li Xiaoli, Suresh Sundaram, and Daniel Pack. A decentralized learning strategy to restore connectivity during multi-agent formation control. *Neurocomputing*, 520:33–45, 2023.

[DM78] Benjamin Dutton and Eldert S Maloney. Dutton's navigation & piloting. 1978.

[DMB06] Laloui Derafa, Tarek Madani, and Abdelaziz Benallegue. Dynamic modelling and experimental identification of four rotors

helicopter parameters. In *2006 IEEE international conference on industrial technology*, pages 1834–1839. IEEE, 2006.

[DN21] Hendi Wicaksono Agung Darminto and Itthisek Nilkhamhang. Hierarchical decentralized lqr control for formation-keeping of cooperative mobile robots in material transport tasks. *Engineering Journal (EJ)*, 25(12):37–50, 2021.

[DOMB07] Laloui Derafa, A Ouldali, Tarek Madani, and Abdelaziz Benallegue. Four rotors helicopter yaw and altitude stabilization. In *World Congress on Engineering*, pages 148–152. Citeseer, 2007.

[Dre21] Daniel S Drew. Multi-agent systems for search and rescue applications. *Current Robotics Reports*, 2(2):189–200, 2021.

[DSL19] Xiongfeng Deng, Xiuxia Sun, and Shuguang Liu. Iterative learning control for leader-following consensus of nonlinear multi-agent systems with packet dropout. *International Journal of Control, Automation and Systems*, 17(8):2135–2144, 2019.

[DSP16] Bikramaditya Das, Bidyadhar Subudhi, and Bibhuti Bhusan Pati. Cooperative formation control of autonomous underwater vehicles: An overview. *International Journal of Automation and computing*, 13:199–225, 2016.

[DYSZ14] Xiwang Dong, Bocheng Yu, Zongying Shi, and Yisheng Zhong. Time-varying formation control for unmanned aerial vehicles: Theories and applications. *IEEE Transactions on Control Systems Technology*, 23(1):340–348, 2014.

[DZZ+21] Xiaohu Ding, Dongdong Zhang, Liangang Zhang, Lei Zhang, Changjiang Zhang, and Bin Xu. Fault detection for automatic guided vehicles based on decision tree and lstm. In *2021 5th International Conference on System Reliability and Safety (ICSRS)*, pages 42–46. IEEE, 2021.

[EAM19] Arezou Elahi, Alireza Alfi, and Hamidreza Modares. H consensus control of discrete-time multi-agent systems under network imperfections and external disturbance. *IEEE/CAA Journal of Automatica Sinica*, 6(3):667–675, 2019.

[EF16] Masoud Emam and Ahmad Fakharian. Attitude tracking of quadrotor uav via mixed h_2/h_∞ controller: An lmi based ap-

proach. In *2016 24th Mediterranean Conference on Control and Automation (MED)*, pages 390–395. IEEE, 2016.

[Fäs18] Matthias Fässler. *Quadrotor control for accurate agile flight*. PhD thesis, University of Zurich, 2018.

[FDKSS21] Giuseppe Fragapane, Rene De Koster, Fabio Sgarbossa, and Jan Ola Strandhagen. Planning and control of autonomous mobile robots for intralogistics: Literature review and research agenda. *European journal of operational research*, 294(2):405–426, 2021.

[FLL11] Alessandro Freddi, Alexander Lanzon, and Sauro Longhi. A feedback linearization approach to fault tolerance in quadrotor vehicles. *IFAC proceedings volumes*, 44(1):5413–5418, 2011.

[FZL16] Rui Fu, Zuo Zhang, and Li Li. Using lstm and gru neural network methods for traffic flow prediction. In *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 324–328. IEEE, 2016.

[FZL+19] Shuo Feng, Yi Zhang, Shengbo Eben Li, Zhong Cao, Henry X Liu, and Li Li. String stability for vehicular platoon control: Definitions and analysis methods. *Annual Reviews in Control*, 47:81–97, 2019.

[Gah94] P Gahinet. Lmi control toolbox. *The Math Works Inc*, 1994.

[Gar19] Nuria Mateos García. Multi-agent system for anomaly detection in industry 4.0 using machine learning techniques. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 8(4):33, 2019.

[GF21] Jingming Gu and Dianjun Fang. Genetic algorithm based lqr control for agv path tracking problem. In *Journal of Physics: Conference Series*, volume 1952, page 032012. IOP Publishing, 2021.

[GJMSBA17] Higinio González-Jorge, Joaquin Martínez-Sánchez, Martín Bueno, and Pedor Arias. Unmanned aerial systems for civil applications: A review. *Drones*, 1(1):2, 2017.

[GKLP21] SJ Gambhire, D Ravi Kishore, PS Londhe, and SN Pawar. Review of sliding mode based control techniques for control system applications. *International Journal of dynamics and control*, 9(1):363–378, 2021.

[GLVS20] Jawhar Ghommam, Luis F Luque-Vega, and Maarouf Saad. Distance-based formation control for quadrotors with collision avoidance via lyapunov barrier functions. *International Journal of Aerospace Engineering*, 2020(1):2069631, 2020.

[GNW+21] Kenneth CW Goh, Raymond BC Ng, Yoke-Keong Wong, Nicholas JH Ho, and Matthew CH Chua. Aerial filming with synchronized drones using reinforcement learning. *Multimedia Tools and Applications*, 80:18125–18150, 2021.

[GPP17] Xiang Gong, Ya-Jun Pan, and Ajinkya Pawar. A novel leader following consensus approach for multi-agent systems with data loss. *International Journal of Control, Automation, and Systems*, 15(2):763, 2017.

[Gra12] Alex Graves. Long short-term memory. In *Supervised sequence labelling with recurrent neural networks*, pages 37–45. Springer, 2012.

[HG24] Tore Hägglund and José Luis Guzmán. Give us pid controllers and we can control the world. *IFAC-PapersOnLine*, 58(7):103–108, 2024.

[HHMS09] Minh-Duc Hua, Tarek Hamel, Pascal Morin, and Claude Samson. A control approach for thrust-propelled underactuated vehicles and its application to vtol drones. *IEEE Transactions on Automatic Control*, 54(8):1837–1853, 2009.

[htt] https://ottomotors.com/. Otto amr.

[HWZH17] Zhicheng Hou, Weijun Wang, Gong Zhang, and Changsoo Han. A survey on the formation control of multiple quadrotors. In *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 219–225. IEEE, 2017.

[HZL+20] Jinwen Hu, Houxin Zhang, Lu Liu, Xiaoping Zhu, Chunhui Zhao, and Quan Pan. Convergent multiagent formation control with

collision avoidance. *IEEE Transactions on Robotics*, 36(6):1805–1818, 2020.

[HZT+25] Xudong Hu, Bo Zhu, Dongkui Tan, Zhengfeng Yan, and Mingyao Yao. Cooperative transportation control of two parking automated guided vehicles. *Asian Journal of Control*, 2025.

[JK17] Sunghun Jung and Hyunsu Kim. Analysis of amazon prime air uav delivery service. *Journal of Knowledge Information Technology and Systems*, 12(2):253–266, 2017.

[JMPH21] Muhammad Atif Javed, Faiz Ul Muram, Sasikumar Punnekkat, and Hans Hansson. Safe and secure platooning of automated guided vehicles in industry 4.0. *Journal of systems architecture*, 121:102309, 2021.

[JSQ+23] Shumaila Javaid, Nasir Saeed, Zakria Qadir, Hamza Fahim, Bin He, Houbing Song, and Muhammad Bilal. Communication and control in collaborative uavs: Recent advances and future trends. *IEEE Transactions on Intelligent Transportation Systems*, 24(6):5719–5739, 2023.

[JWBB13] Qiu Jin, Guoyuan Wu, Kanok Boriboonsomsin, and Matthew Barth. Platoon-based multi-agent intersection management for connected vehicle. In *16th international ieee conference on intelligent transportation systems (itsc 2013)*, pages 1462–1467. IEEE, 2013.

[Kat10] Ogata Katsuhiko. *Modern control engineering*. 2010.

[KEJ16] Vinodh Kumar E and Jovitha Jerome. Algebraic riccati equation based q and r matrices selection algorithm for optimal lqr applied to tracking control of 3rd order magnetic levitation system. *Archives of electrical engineering*, 65(1):151–168, 2016.

[KFJ17] Kristian Klausen, Thor I Fossen, and Tor Arne Johansen. Nonlinear control with swing damping of a multirotor uav with suspended load. *Journal of Intelligent & Robotic Systems*, 88:379–394, 2017.

[KFJ18] Kristian Klausen, Thor I Fossen, and Tor Arne Johansen. Autonomous recovery of a fixed-wing uav using a net suspended by

two multirotor uavs. *Journal of Field Robotics*, 35(5):717–731, 2018.

[KKBK24] Iveta Kubasakova, Jaroslava Kubanova, Dominik Benco, and Dominika Kadlecová. Implementation of automated guided vehicles for the automation of selected processes and elimination of collisions between handling equipment and humans in the warehouse. *Sensors*, 24(3):1029, 2024.

[KLH09] Amine Kamel, Friedrich Lange, and Gerd Hirzinger. An industrial-robots suited input shaping control scheme. In *Motion and Vibration Control*, pages 177–188. Springer, 2009.

[KMB05] Gregor Klancar, Drago Matko, and Saso Blazic. Mobile robot control on a reference path. In *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.*, pages 1343–1348. IEEE, 2005.

[KR07] James F Kurose and Keith W Ross. Computer networking: A top-down approach edition. *Addision Wesley*, 12, 2007.

[KUK99] KUKA. Kuka omnimove (agvs), 1999.

[LC18] Giroung Lee and Dongkyoung Chwa. Decentralized behavior-based formation control of multiple robots considering obstacle avoidance. *Intelligent Service Robotics*, 11:127–138, 2018.

[LCM17] Yongkang Liu, Richard Candell, and Nader Moayeri. Effects of wireless packet loss in industrial process control systems. *ISA transactions*, 68:412–424, 2017.

[LJYX19] Sida Lin, Ruiming Jia, Ming Yue, and Yuan Xu. On composite leader–follower formation control for wheeled mobile robots with adaptive disturbance rejection. *Applied Artificial Intelligence*, 33(14):1306–1326, 2019.

[LLA+22] Xijing Liu, Kevin Lam, Balsam Alkouz, Babar Shahzaad, and Athman Bouguettaya. Constraint-based formation of drone swarms. In *2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, pages 73–75. IEEE, 2022.

[LLL15] Jing Luo, Cheng-Lin Liu, and Fei Liu. A leader-following formation control of multiple mobile robots with obstacle. In *2015 IEEE International Conference on Information and Automation*, pages 2153–2158. IEEE, 2015.

[LLXL16] Xinde Li, Chaomin Luo, Yefan Xu, and Pei Li. A fuzzy pid controller applied in agv control system. In *2016 International Conference on Advanced Robotics and Mechatronics (ICARM)*, pages 555–560. IEEE, 2016.

[LLZ+22] Liang Li, Yiping Li, Yuexing Zhang, Gaopeng Xu, Junbao Zeng, and Xisheng Feng. Formation control of multiple autonomous underwater vehicles under communication delay, packet discreteness and dropout. *Journal of Marine Science and Engineering*, 10(7):920, 2022.

[LWLZ14] Yang Li, Xiao Nian Wang, Si Jun Li, and Jin Zhu. Lqr based trajectory tracking control for forked agv. *Applied Mechanics and Materials*, 577:447–451, 2014.

[LWM+23] Jie Lin, Yaonan Wang, Zhiqiang Miao, Qiong Lin, Guoqiang Hu, and Rafael Fierro. Robust linear-velocity-free formation tracking of multiple quadrotors with unknown disturbances. *IEEE Transactions on Control of Network Systems*, 10(4):1757–1769, 2023.

[LZ18] Guo-Ping Liu and Shijie Zhang. A survey on formation control of small satellites. *Proceedings of the IEEE*, 106(3):440–457, 2018.

[LZY+18] Andong Liu, Wen-An Zhang, Li Yu, Huaicheng Yan, and Rongchao Zhang. Formation control of multiple mobile robots incorporating an extended state observer and distributed model predictive approach. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(11):4587–4597, 2018.

[M+19] John D Monnier et al. A realistic roadmap to formation flying space interferometry. *arXiv preprint arXiv:1907.09583*, 2019.

[Man08] Justin E Manley. Unmanned surface vehicles, 15 years of development. In *OCEANS 2008*, pages 1–4. Ieee, 2008.

[MCD23] Chi Ma, Yizhe Cao, and Dianbiao Dong. Reinforcement learning based time-varying formation control for quadrotor unmanned

aerial vehicles system with input saturation. *Applied Intelligence*, 53(23):28730–28744, 2023.

[MD18] UM Rao Mogili and BBVL Deepak. Review on application of drone systems in precision agriculture. *Procedia computer science*, 133:502–509, 2018.

[MK10] Juraj Miček and Jan Kapitulík. An input shaping filter. *IFAC Proceedings Volumes*, 43(24):259–263, 2010.

[MKLT95] Larry Matthies, Alonzo Kelly, Todd Litwin, and Greg Tharp. Obstacle detection for unmanned ground vehicles: A progress report. In *Proceedings of the Intelligent Vehicles' 95. Symposium*, pages 66–71. IEEE, 1995.

[MLS+22] Ata Jahangir Moshayedi, Jinsong Li, Nima Sina, Xi Chen, Liefa Liao, Mehdi Gheisari, and Xiaoyun Xie. Simulation and validation of optimized pid controller in agv (automated guided vehicles) model using pso and bas algorithms. *Computational Intelligence and Neuroscience*, 2022, 2022.

[MM16] Simone Milani and Alvise Memo. Impact of drone swarm formations in 3d scene reconstruction. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 2598–2602. IEEE, 2016.

[MMM22] Muhammad Maaruf, Magdi Sadek Mahmoud, and Alfian Ma'arif. A survey of control methods for quadrotor uav. *International Journal of Robotics and Control Systems*, 2(4):652–665, 2022.

[MNPZ20] Sabato Manfredi, Enrico Natalizio, Claudio Pascariello, and Nicola Roberto Zema. Stability and convergence of a message-loss-tolerant rendezvous algorithm for wireless networked robot systems. *IEEE Transactions on Control of Network Systems*, 7(3):1103–1114, 2020.

[MS16] B Madhevan and M Sreekumar. Analysis of communication delay and packet loss during localization among mobile robots. In *Intelligent Systems Technologies and Applications*, pages 3–12. Springer, 2016.

[MVSGP24] Alberto Mozo, Stanislav Vakaruk, J Enrique Sierra-García, and Antonio Pastor. Anticipatory analysis of agv trajectory in a 5g

network using machine learning. *Journal of Intelligent Manufacturing*, 35(4):1541–1569, 2024.

[MW20] Furugh Mirali and Herbert Werner. A dynamic quasi-taylor approach for distributed consensus problems with packet loss. In *2020 American Control Conference (ACC)*, pages 701–706. IEEE, 2020.

[NAQ18] Khan Nouman, Zaman Asim, and Khan Qasim. Comprehensive study on performance of pid controller and its applications. In *2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, pages 1574–1579. IEEE, 2018.

[NDM+21] Anuj Nandanwar, Narendra KUMAR Dhar, Dmitry Malyshev, Larisa Rybak, and Laxmidhar Behera. Stochastic event-based super-twisting formation control for multi-agent system under network uncertainties. *IEEE Transactions on Control of Network Systems*, 2021.

[Ngu19] Ngoc Thinh Nguyen. *Reliable hierarchical control for multicopter systems*. PhD thesis, Université Grenoble Alpes, 2019.

[NM02] Monica N Nicolescu and Maja J Matarić. A hierarchical architecture for behavior-based robots. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*, pages 227–233, 2002.

[NPSL17] Ngoc Thinh Nguyen, Ionela Prodan, Florin Stoican, and Laurent Lefèvre. Reliable nonlinear control for quadcopter trajectory tracking through differential flatness. *IFAC-PapersOnLine*, 50(1):6971–6976, 2017.

[NS19] Tiago P Nascimento and Martin Saska. Position and attitude control of multi-rotor aerial vehicles: A survey. *Annual Reviews in Control*, 48:129–146, 2019.

[Par98] Lynne E Parker. Alliance: An architecture for fault tolerant multirobot cooperation. *IEEE transactions on robotics and automation*, 14(2):220–240, 1998.

[PFTG14] Chen Peng, Min-Rui Fei, Engang Tian, and Yan-Peng Guan. On hold or drop out-of-order packets in networked control systems. *Information Sciences*, 268:436–446, 2014.

[PGP+18] Long Peng, Fei Guan, Luc Perneel, Hasan Fayyad-Kazan, and Martin Timmerman. Decentralized multi-robot formation control with communication delay and asynchronous clock. *Journal of Intelligent & Robotic Systems*, 89(3):465–484, 2018.

[PL19] Jih-Sien Peng and Yen-Chen Liu. Towards cooperative transportation of multiple mecanum-wheeled automated guided vehicles. In *Dynamic Systems and Control Conference*, volume 59155, page V002T14A004. American Society of Mechanical Engineers, 2019.

[PNK+16] Pandu Sandi Pratama, Trong Hai Nguyen, Hak Kyeong Kim, Dae Hwan Kim, and Sang Bong Kim. Positioning and obstacle avoidance of automatic guided vehicle in partially known environment. *International Journal of Control, Automation and Systems*, 14(6):1572–1581, 2016.

[PTG14] Lal Bahadur Prasad, Barjeev Tyagi, and Hari Om Gupta. Optimal control of nonlinear inverted pendulum system using pid controller and lqr: performance analysis without and with disturbance input. *International Journal of Automation and Computing*, 11(6):661–670, 2014.

[PTT14] Manh Duong Phung, Thuan Hoang Tran, and Quang Vinh Tran. Stable control of networked robot subject to communication delay, packet loss, and out-of-order delivery. *Vietnam Journal of Mechanics*, 36(3):215–233, 2014.

[PWHB17] Ya-Jun Pan, Herbert Werner, Zipeng Huang, and Marcus Bartels. Distributed cooperative control of leader–follower multi-agent systems under packet dropouts for quadcopters. *Systems & control letters*, 106:47–57, 2017.

[PYW+16] Zhaoxia Peng, Shichun Yang, Guoguang Wen, Ahmed Rahmani, and Yongguang Yu. Adaptive distributed formation control for multiple nonholonomic wheeled mobile robots. *Neurocomputing*, 173:1485–1494, 2016.

[QHM+21] Yanming Quan, Lingwei Huang, Lei Ma, Yiming He, and Ronghui Wang. Neural network-based indoor autonomously-navigated agv motion trajectory data fusion. *Automatic Control and Computer Sciences*, 55:334–345, 2021.

[QTP+20] Jorge Pena Queralta, Jussi Taipalmaa, Bilge Can Pullinen, Victor Kathan Sarker, Tuan Nguyen Gia, Hannu Tenhunen, Moncef Gabbouj, Jenni Raitoharju, and Tomi Westerlund. Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision. *Ieee Access*, 8:191617–191643, 2020.

[RAP13] Francisco C Rego, A Pedro Aguiar, and António M Pascoal. A packet loss compliant logic-based communication algorithm for cooperative path-following control. *IFAC Proceedings Volumes*, 46(33):262–267, 2013.

[RLS20] Carlos Rizzo, Angel Lagraña, and Daniel Serrano. Geomove: Detached agvs for cooperative transportation of large and heavy loads in the aeronautic industry. In *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 126–133. IEEE, 2020.

[Rob] Mobile Industrial Robots. Autonomous mobile robots (amrs).

[RRK+18] Abhijeet Ravankar, Ankit A Ravankar, Yukinori Kobayashi, Yohei Hoshino, and Chao-Chung Peng. Path smoothing techniques in robot navigation: State-of-the-art, current and future challenges. *Sensors*, 18(9):3170, 2018.

[Sad12] Anna Danuta Sadowska. *Formation control of nonholonomic mobile robots: the virtual structure approach*. PhD thesis, 2012.

[Sed03] A Khaki Sedigh. Modern control systems, 2003.

[SEMR+19] Carlos Santos, Felipe Espinosa, Miguel Martinez-Rey, David Gualda, and Cristina Losada. Self-triggered formation control of nonholonomic robots. *Sensors*, 19(12):2689, 2019.

[SF16] Leila Sedghi and Ahmad Fakharian. Robust voltage regulation in islanded microgrids: A lmi based mixed h_2/h_∞ control approach. In *2016 24th Mediterranean Conference on Control and Automation (MED)*, pages 431–436. IEEE, 2016.

[SINAR+22] Leila Sedghi, Zohaib Ijaz, Md Noor-A-Rahim, Kritchai Witheephanich, and Dirk Pesch. Machine learning in event-triggered control: Recent advances and open issues. *IEEE Access*, 10:74671–74690, 2022.

[SJNARP22] Leila Sedghi, Jobish John, Md Noor-A-Rahim, and Dirk Pesch. Formation control of automated guided vehicles in the presence of packet loss. *Sensors*, 22(9):3552, 2022.

[SLX+20] Xuanyi Song, Yuetian Liu, Liang Xue, Jun Wang, Jingzhe Zhang, Junqiang Wang, Long Jiang, and Ziyan Cheng. Time-series well performance prediction based on long short-term memory (lstm) neural network model. *Journal of Petroleum Science and Engineering*, 186:106682, 2020.

[SNI19] Behzad Shirani, Majdeddin Najafi, and Iman Izadi. Cooperative load transportation using multiple uavs. *Aerospace Science and Technology*, 84:158–169, 2019.

[SSAF+19] Hazim Shakhatreh, Ahmad H Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *Ieee Access*, 7:48572–48634, 2019.

[SV10] William Singhose and Joshua Vaughan. Reducing vibration by digital filtering and input shaping. *IEEE Transactions on Control Systems Technology*, 19(6):1410–1420, 2010.

[TAPJ20] Xiao Xuan Teh, Adnan Aijaz, Anthony Portelli, and Simon Jones. Communications-based formation control of mobile robots: Modeling, analysis and performance evaluation. In *Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 149–153, 2020.

[TLWC18] Yu Herng Tan, Shupeng Lai, Kangli Wang, and Ben M Chen. Cooperative heavy lifting using unmanned multi-agent systems. In *2018 IEEE 14th International Conference on Control and Automation (ICCA)*, pages 1119–1126. IEEE, 2018.

[TTF+20] Edoardo Topini, Alberto Topini, Matteo Franchi, Alessandro Bucci, Nicola Secciani, Alessandro Ridolfi, and Benedetto Al-

lotta. Lstm-based dead reckoning navigation for autonomous underwater vehicles. In *Global Oceans 2020: Singapore–US Gulf Coast*, pages 1–7. IEEE, 2020.

[Unt20] Simon Unterstrasser. The contrail mitigation potential of aircraft formation flight derived from high-resolution simulations. *Aerospace*, 7(12):170, 2020.

[Vas17] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[VdBvdWN09] Thijs Ha Van den Broek, Nathan van de Wouw, and Henk Nijmeijer. Formation control of unicycle mobile robots: a virtual structure approach. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 8328–8333. IEEE, 2009.

[vdG20] Xander van der Geest. Distributed formation control of nonholonomic mobile robots: Implementation and testing in robot operating system. 2020.

[VNGW21] Dinh Van Nam and Kim Gon-Woo. Deep learning based-state estimation for holonomic mobile robots using intrinsic sensors. In *2021 21st International Conference on Control, Automation and Systems (ICCAS)*, pages 12–16. IEEE, 2021.

[VR21] Janardan Kumar Verma and Virender Ranga. Multi-robot coordination analysis, taxonomy, challenges and future scope. *Journal of intelligent & robotic systems*, 102:1–36, 2021.

[VSGMP21] Stanislav Vakaruk, J Enrique Sierra-García, Alberto Mozo, and Antonio Pastor. Forecasting automated guided vehicle malfunctioning with deep learning in a 5g-based industry 4.0 scenario. *IEEE Communications Magazine*, 59(11):102–108, 2021.

[Wak25] Masashi Wakaiki. Stabilization of infinite-dimensional systems under quantization and packet loss. *IEEE Transactions on Automatic Control*, 2025.

[WCLY19] Jinhui Wu, Xu Chen, Andong Liu, and Li Yu. Predictive control for visual servoing control of cyber physical systems with packet loss. *Peer-to-Peer Networking and Applications*, 12(6):1774–1784, 2019.

[WDW+23] Jianan Wang, Xiangjun Ding, Chunyan Wang, Zongyu Zuo, and Zhengtao Ding. Affine formation control of general linear multi-agent systems with delays. *Unmanned Systems*, 11(02):123–132, 2023.

[WG09] Bao-Feng Wang and Ge Guo. Kalman filtering with partial markovian packet losses. *International Journal of Automation and Computing*, 6(4):395–400, 2009.

[WLD+19] Heng Wei, Qiang Lv, Nanxun Duo, GuoSheng Wang, and Bing Liang. Consensus algorithms based multi-robot formation control under noise and time delay conditions. *Applied Sciences*, 9(5):1004, 2019.

[WLZ+21] Liuquan Wang, Qiang Liu, Chenxin Zang, Sanying Zhu, Chaoyang Gan, and Yanqiang Liu. Formation control of dual auto guided vehicles based on compensation method in 5g networks. *Machines*, 9(12):318, 2021.

[WNB20] Qiwei Wang, Thinh Nguyen, and Bella Bose. Towards adaptive packet scheduler with deep-q reinforcement learning. In *2020 International Conference on Computing, Networking and Communications (ICNC)*, pages 118–123. IEEE, 2020.

[Woo09] Michael Wooldridge. *An introduction to multiagent systems*. John wiley & sons, 2009.

[WTZ+18] Chunyan Wang, Hilton Tnunay, Zongyu Zuo, Barry Lennox, and Zhengtao Ding. Fixed-time formation control of multirobot systems: Design and experiments. *IEEE Transactions on Industrial Electronics*, 66(8):6292–6301, 2018.

[WWYS18] Yuanzhe Wang, Danwei Wang, Shuai Yang, and Mao Shan. A practical leader–follower tracking control scheme for multiple nonholonomic mobile robots in unknown obstacle environments. *IEEE Transactions on Control Systems Technology*, 27(4):1685–1693, 2018.

[WZ15] Hui Wang and Quanxin Zhu. Finite-time stabilization of high-order stochastic nonlinear systems in strict-feedback form. *Automatica*, 54:284–291, 2015.

[WZG+20] Yuzhou Wu, Jialu Zhang, Yu Ge, Zhichao Sheng, and Yong Fang. A dropout compensation ilc method for formation tracking of heterogeneous multi-agent systems with loss of multiple communication packets. *Applied Sciences*, 10(14):4752, 2020.

[XZSR16] Bu Xuhui, Hou Zhongsheng, Jin Shangtai, and Chi Ronghu. An iterative learning control design approach for networked control systems with data dropouts. *International Journal of Robust and Nonlinear Control*, 26(1):91–109, 2016.

[YA09] Hiroaki Yamaguchi and Tamio Arai. A conversion of a cooperative transportation system with two car-like mobile robots into two-chain, single-generator chained form and its steering. *Journal of Robotics and Mechatronics*, 21(1):57, 2009.

[Yan23] Tingting Yan. Positioning of logistics and warehousing automated guided vehicle based on improved lstm network. *International Journal of System Assurance Engineering and Management*, 14(2):509–518, 2023.

[YHL+23] Ziwei Yan, Liang Han, Xiaoduo Li, Xiwang Dong, Qingdong Li, and Zhang Ren. Event-triggered formation control for time-delayed discrete-time multi-agent system applied to multi-uav formation flying. *Journal of the Franklin Institute*, 360(5):3677–3699, 2023.

[YJC13] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, 10(12):399, 2013.

[YKDvdS19] Dujuan Yang, Anique Kuijpers, Gamze Dane, and Tom van der Sande. Impacts of large-scale truck platooning on dutch highways. *Transportation research procedia*, 37:425–432, 2019.

[YLL10] Fan Yang, Shi-rong Liu, and Fei Liu. Cooperative transport strategy for formation control of multiple mobile robots. *Journal of Zhejiang University Science C*, 11(12):931–938, 2010.

[YLL+23] Yichen Yang, Zuxing Li, Yabin Li, Tianyu Cao, and Zhipeng Li. Stability enhancement for traffic flow via self–stabilizing control strategy in the presence of packet loss. *Physica A: Statistical Mechanics and its Applications*, 622:128801, 2023.

[YN08] Chika Yoshioka and Toru Namerikawa. Formation control of nonholonomic multi-vehicle systems based on virtual structure. *IFAC Proceedings Volumes*, 41(2):5149–5154, 2008.

[YO15] Alpaslan Yufka and Metin Ozkan. Formation-based control scheme for cooperative transportation by multiple mobile robots. *International Journal of Advanced Robotic Systems*, 12(9):120, 2015.

[YWIK04] Xin Yang, Keigo Watanabe, Kiyotaka Izumi, and Kazuo Kiguchi. A decentralized control system for cooperative transportation by multiple non-holonomic mobile robots. *International Journal of Control*, 77(10):949–963, 2004.

[ZABW05] Aram Zaerpoora, Majid Nili Ahmadabadia, Mohammad R Barunia, and Zhi Dong Wang. Distributed object transportation on a desired path based on constrain and move strategy. *Robotics and Autonomous Systems*, 50(2-3):115–128, 2005.

[ZCS+12] Pan Zhao, Jiajia Chen, Yan Song, Xiang Tao, Tiejuan Xu, and Tao Mei. Design of a control system for an autonomous vehicle based on adaptive-pid. *International Journal of Advanced Robotic Systems*, 9(2):44, 2012.

[Zha10] Peng Zhang. *Advanced industrial control technology*. William Andrew, 2010.

[ZL18] Tian-Yong Zhang and Guo-Ping Liu. Tracking control of wheeled mobile robots with communication delay and data loss. *Journal of Systems Science and Complexity*, 31(4):927–945, 2018.

[ZL21] Ting Zhang and Junmin Li. Iterative learning control for multi-agent systems with finite-leveled sigma-delta quantizer and input saturation. In *2021 40th Chinese Control Conference (CCC)*, pages 5637–5644. IEEE, 2021.

[ZM13] Youmin Zhang and Hasan Mehrjerdi. A survey on multiple unmanned vehicles formation control and coordination: Normal and fault situations. In *2013 International conference on unmanned aircraft systems (ICUAS)*, pages 1087–1096. IEEE, 2013.

[ZMX08] Dai Zhaohui, Wu Min, and Chen Xin. Multi-robot cooperative transportation using formation control. In *2008 27th Chinese Control Conference*, pages 346–350. IEEE, 2008.

[ZZD20] Xueqi Zhang, Meng Zhao, and Rencai Dong. Time-series prediction of environmental noise for urban iot based on long short-term memory recurrent neural network. *Applied Sciences*, 10(3):1144, 2020.