DEEP REINFORCEMENT LEARNING-BASED MOBILE ROBOT PATH PLANNING AND CONTROL SUBJECT TO MODEL UNCERTAINTY AND EXTERNAL DISTURBANCES

YEFENG YANG

PhD

The Hong Kong Polytechnic University

This programme is jointly offered by The Hong Kong Polytechnic University and Harbin Institute of Technology

2025

The Hong Kong Polytechnic University

Department of Aeronautical and Aviation Engineering

Harbin Institute of Technology

Department of Control Science and Engineering

Deep Reinforcement Learning-Based Mobile Robot Path Planning and Control Subject to Model Uncertainty and External Disturbances

Yefeng Yang

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Oct 2024

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Yefeng YANG

Abstract

Autonomous robots have attracted significant attention in recent years due to their wide-ranging applications in industry, environmental monitoring, and agriculture. However, many challenging issues must be addressed before robots can reliably execute specific tasks in complex environments. These challenges include mapping, self-localization, task allocation, planning, and control. Among these, trajectory planning and control are especially critical in design. This thesis introduces novel methodologies to address the challenges of trajectory planning and control.

1) To address the global and local planning challenges of autonomous robots, this thesis presents an enhanced Rapidly-exploring Random Tree (RRT) algorithm alongside a Deep Reinforcement Learning (DRL)-based obstacle avoidance method. Initially, a sampling-efficient RRT algorithm is proposed, which integrates the geometric properties of the environment to minimize the number of sampling nodes required in complex scenarios. Subsequently, in the local planning phase, the Network Decoupling (ND) technique is employed in the design of DRL to accelerate the training process and improve the efficacy of obstacle avoidance.

2) A DRL-based optimization framework is proposed to optimize the hyper-parameters in the traditional Recursive Fast Non-singular Terminal Sliding Mode Control(ler) (RFNTSMC). The Fast Non-singular Terminal Sliding Mode Control(ler) (FNTSMC) optimized by DRL achieves better performance in the quadrotor control problem. Thereafter, a fixed-time disturbance observer is utilized for compensating external disturbances and modeling uncertainty. The stability of the closed-loop learning-based control framework is guaranteed in a Lyapunov sense.

3) Building upon the second contribution, a distributed control framework is developed to stabilize a group of multi-quadrotors. First, a fully distributed FNTSMC and a distributed fixed-time disturbance observer are proposed to ensure the stability of the multi-quadrotor system. Subsequently, DRL techniques are employed to adaptively learn the near-optimal hyperparameters for the FNTSMCs. The stability of the framework is formally guaranteed using graph theory and Lyapunov stability analysis.

4) Building upon the second and third studies, and considering the impact of system convergence time characteristics on practical applications, we further propose an enhanced distributed predefined-time convergence control framework. This framework allows the upper bound of the convergence time for multi-agent systems to be explicitly specified. First, we derive a more generalized predefined-time stability criterion. Subsequently, we utilize this criterion to design a predefined-time stable observer. This observer simultaneously estimates external disturbances and their upper bounds, which enables the reduction of control gains, thereby mitigating chattering issues to a certain extent. Furthermore, we develop a distributed predefined-time controller to ensure that the entire quadrotor formation system achieves stability within the predefined time. Finally, extensive simulations and physical experiments are conducted to validate the effectiveness and superiority of the proposed control method.

5) Two simulation platforms are developed as auxiliary tools during the algorithm design process, and extensive physical experiments are conducted to validate the superiority and effectiveness of the proposed algorithms.

Keywords: Trajectory planning; Quadrotor control; Deep reinforcement learning; Adaptive dynamic programming; Multi-agent consensus control; Finite-time control; Fixed-time control; Predefined-time control.

Mathematical symbol announcement: The mathematical symbols in this thesis are defined independently in each chapter when they appear for the first time in this chapter since we introduce a complete system in each chapter. To fit the algorithms used in each chapter, the symbols may be different in those chapters to represent a very similar concept, please refer to the definition in its current chapter.

Publications

Published/Accepted:
Journal

1 **Yefeng Yang**, Tao Huang, Tianqi Wang, Wenyu Yang, Han Chen, Boyang Li*, and Chih-yung Wen, "Sampling efficient rapidly exploring random tree path planning and improved Actor-Critic based obstacle avoidance for autonomous robots", *Science China: Information Science*, vol. 67, no. 5, pp: 1-18, 2024.

2 **Yefeng Yang***, Tao Huang, Xinxin Wang, Chih-yung Wen, Xianlin Huang, "High-Speed Three-Dimensional Aerial Vehicle Evasion based on a Multi-stage Dueling Deep Q-Network", *Aerospace*, vol. 9, no. 11, pp. 673-689, 2022.

3 Y. Cai, **Y. Yang**, T. Huang, B. Li*, “Robust Reinforcement Learning Control Framework for a Quadrotor Unmanned Aerial Vehicle Using Critic Neural Network”, Advanced Intelligent Systems, pp. 2400427, 2025.

4 X. Wang, **Y. Yang**, Z. Zuo*, "Three-Dimensional Intercept-Angle-Constrained and Field-of-View-Constrained Guidance for Maneuvering Target", Journal of Guidance, Control, and Dynamics, pp. 1-11, 2024.

5 X. Wang*, G. Lan, **Y. Yang**, H. Lu, and X. Huang, "Terminal angle constrained time-varying sliding mode guidance law with autopilot dynamics and input saturation", Asian Journal of Control, vol. 25, no 2, pp. 1130-1144, 2023.

6 **Yefeng Yang**, Tao Huang, Tianqi Wang*, and Chih-yung Wen, "A Robust Adaptive Dynamic Programming-Based Sliding-mode Control for Quadrotors Subject to Model Uncertainty and External Disturbance". (*under review, submitted to Nonlinear Dynamics*)

7 **Yefeng Yang**, Kang Liu*, Li-Yu Lo, Tao Huang, Yanming Fu, and Chih-yung Wen, "Fixed-time Adaptive Consensus Control for Multi-Quadrotor Subject to External Disturbances Via Deep Reinforcement Learning". (*under review, submitted to Aerospace Science and Technology*)

8 **Yefeng Yang**, Kang Liu*, Tao Huang, Xinxin Wang, Yanming Fu, Chih-yung Wen, "Modified Predefined-Time Adaptive Observer-Based Fast Nonsingular Terminal Sliding Mode Control for Multi-Quadrotor Subject to External Disturbances". (*under review, submitted to IEEE Transactions on Instrumentation and Measurement*)

9 K. Liu, **Y. Yang***, W. Yang, Y. Zhang, X. Wang, C.-Y. Wen, "Adaptive Predefined-Time Disturbance Observer-Based Fast Nonsingular Sliding Mode Control for Quadrotor UAVs Under Unknown Disturbances", (under review, submitted to IEEE Transactions on Industrial Electronics)

Conference

1 **Yefeng Yang**, Xiaojun Ban*, Xianlin Huang and Chenghao Shan, "A Dueling-Double-Deep Q-Network Controller for Magnetic Levitation Ball System," 2020 39th Chinese Control Conference (CCC), Shenyang, China, pp. 1885-1890, 2020.

2 **Yefeng Yang**, Tao Huang, Tianqi Wang*, Chih-Yung Wen, "A Robust Sliding-mode Control Framework for Quadrotors Subject to Model Uncertainty and External Disturbances", 2024 American Control Conference (ACC), Toronto, Canada, pp. 3809–3814, 2024.

3 **Yefeng Yang**, Xiaojun Ban, Hongqian Lu, Tao Huang*, and Xianlin Huang, "Parameter Optimization for a Quadrotor System with External Disturbance and Uncertainty via Reinforcement Learning", 2024 43th Chinese Control Conference (CCC), Kunming, China, pp. 2480-2485, 2024.

4 Tao Huang, Hongqian Lu, **Yefeng Yang***, and Xianlin Huang, "Robust Sliding Mode Adaptive Dynamic Programming Control for Partially Unknown Second-order Nonlinear Systems", 2024 43th Chinese Control Conference (CCC), Kunming, China, 2024, pp. 2351-2356, 2024.

5 Li-Yu Lo, Yang Hu, Boyang Li; Chih-Yung Wen, **Yefeng Yang***, "An Adaptive Model Predictive Control for Unmanned Underwater Vehicles Subject to External Disturbances and Measurement Noise", 2024 14th Asian Control Conference (ASCC), Dalian, China, pp. 01-07, 2024.

6 **Y. Yang**, T. Huang*, Y. Zhang, W. Yang, L.-Y. Lo, C.-Y. Wen, "A new type of predefined time convergence criteria and its application on quadrotor systems", (Prepare to submit to 2026 Control and Decision Conference, CDC)

Acknowledgements

Completing my PhD has taken longer than expected, and I owe much to those who supported me. I am a joint PhD student at the Harbin Institute of Technology (HIT) and The Hong Kong Polytechnic University (PolyU). When I joined PolyU in September 2021 under Professor Chih-Yung Wen's group, it was already my fourth year of doctoral studies in HIT. However, due to the heavy workload from various engineering projects at HIT, I had not yet settled on a research topic when I arrived at PolyU.

First and foremost, I would like to express my heartfelt gratitude to Professor Chih-Yung Wen for allowing me to join his research group. He graciously allowed me to begin my PhD journey anew at PolyU without judgment of my previous experience. His continuous encouragement and support have been invaluable throughout my studies, particularly during the challenging three-month quarantine period in Shanghai due to COVID-19, when I made little progress in my research. His understanding during that time is something I sincerely appreciate.

I would also like to extend my sincere thanks to my two co-supervisors, Dr. Boyang Li and Dr. Tianqi Wang. Dr. Li's creative insights, patience, and assistance with experimental equipment enabled me to start my research swiftly upon joining the group. Dr. Wang's rigorous academic approach and strong mathematical expertise taught me how to write academic papers effectively.

I am grateful to my lab mates at PolyU, especially Mr. Liyu Lo and Mr. Bailun Jiang, whose assistance in hardware design saved me considerable time. I would also like to thank Mr. Wenyu Yang for his help with my physical experiments and Mr. Tao Huang, another joint PhD student from HIT and PolyU, for his companionship and support over the past six years, particularly during our time at HIT. His timely and invaluable assistance in my theoretical research was crucial to my progress. During my final six months at PolyU, I received support from two postdoctoral fellows, Dr. Yu Li and Dr. Kang Liu, whose guidance on my future research directions, writing skills, and mathematical derivations in control theory was greatly appreciated. I would also like to thank my friends from HIT, including Dr. Yue Sun, Dr. Xinxin Wang, and Dr. Jingying Li, for their guidance on my thesis, their comfort during difficult times, and their corrections when I made mistakes.

Finally, I want to express my deepest gratitude to my family for their unwavering support, love, and encouragement. I also thank all the friends in Professor Wen's lab and at HIT. Without their help,

I would have faced many more challenges.

Contents

List of Figures

List of Tables

List of Abbreviations

1) **Path planning**:

A* A Star

ACDP-RRT Adaptive Clustering-based Dynamic Programming-based RRT

ACO Ant Colony Optimization

ADP Adaptive Dynamic Programming

AI Artificial Intelligence

APF Artificial Potential Field

ApprDP Approximate Dynamic Programming

D algorithm Dijkstra Algorithm

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DWA Dynamic Window Approach

GA Genetic algorithm

JPS Jump Point Search

MA Memetic Algorithm

NRRT* Neural Rapidly-exploring Random Tree-Star

PRM Probabilistic Road Map

PSO Particle Swarm Optimization

RRT Rapidly-exploring Random Tree

RRT* Rapidly-exploring Random Tree-Star

TEB Time-Elastic Band

2) **Reinforcement Learning:**

A3C Asynchronous Advantage Actor-Critic

AC Actor-Critic

AED Adaptive Evaluation Design

CPO Conversative Policy Optimization

DDPG Deep Deterministic Policy Gradient

DL Deep Learning

DLT Deep Learning Toolbox

DP Dynamic Programming

DPPO Distributed Proximal Policy Optimization

DPPO2 Distributed Proximal Policy Optimization2

DQN Deep Q-Network

DRL Deep Reinforcement Learning

GAE Generalized Advantage Estimation

MA-DDPG Multi-agent DDPG

MA-PPO Multi-agent PPO

MDP Markov Decision Proces

ND Network Decoupling

NDP Neural Dynamic Programming

NPG Natural Policy Gradient

NN neural network

PG Policy Gradient

PID Proportional-Integral-Derivative

PPO Proximal Policy Optimization

PPO2 Proximal Policy Optimization2

RL Reinforcement Learning

SAC Soft Actor Critic

TD3 Twin Delayed Deep Deterministic Policy Gradient

TRPO Trust Region Policy Gradient

VI Value Iteration

3) **Control&Robitic:**

AFTO Appointed Fixed-time Observer

DO disturbance observer

FCU Flight Control Unit

FNSMC Fast Non-singular Sliding Mode Control(ler)

FNTSMC Fast Non-singular Terminal Sliding Mode Control(ler)

FTDO fixed-time disturbance observer

HJB Hamilton–Jacobi–Bellman

LQR Linear Quadratic Regulating

LQG Linear Quadratic Gaussian

QUAV Quadrotor Unmanned Aerial Vehicle

RFNTSMC Recursive Fast Non-singular Terminal Sliding Mode Control(ler)

ROS Robot Opeartion System

SMC Sliding Mode Control(ler)

TSMC Terminal Sliding Mode Control(ler)

UAV Unmanned Aerial Vehicle

UUB Uniformly Ultimately Bounded

PdT Predefined-time

PdTDO Predefined-time Disturbance Observer

PdTFNTSMC Predefined-time Fast Nonsingular Terminal Sliding Mode Control

4) **Other:**

DRLSP Deep Reinforcement Learning Simulation Platform

PPSP Path Planning Simulation Platform

Chapter 1

Introduction

The primary motivation for developing mobile robots is to undertake repetitive, complex, or hazardous tasks for humans. This necessity becomes even more critical in challenging environments such as ruins, landfills, or significant heights. As robotics has advanced, mobile robots have found widespread applications in industries such as manufacturing [1], agriculture [2], automation [3], scientific research [4], and defense [5]. With the rapid advancement of automation and artificial intelligence technologies, the demand for more intelligent robots has grown substantially in recent years. However, this progress has also introduced a range of unresolved theoretical and engineering challenges.

Mobile robots encompass various robot types, each serving distinct purposes. For example, underwater vehicles are employed to explore marine environments [6]; ground autonomous cars are widely used in autonomous driving [7]; multi-rotor aircraft are utilized for tasks such as terrain exploration, logistics, aerial photography, and agricultural protection [2]; fixed-wing drones are applied in reconnaissance, mapping, and remote sensing [8–10]; humanoid robots are often adopted in the service industry [11]; warehouse robots are employed for tasks like material handling, placement, and inventory management [12]; and robotic manipulators are used in industrial production and precision component manufacturing [13], among other applications. Although the structure and functionality of different robots vary greatly, the challenges that need to be addressed during task execution are remarkably similar. These challenges include task allocation, communication, localization, navigation, planning, and control.

Among the various challenges outlined, trajectory planning and control are particularly critical for mobile robot systems. Consequently, this paper provides a comprehensive study and exploration of trajectory planning and control in mobile robots. The structure of this chapter is organized as follows: Section 1.1 introduces the concept of trajectory planning. Section 1.2 reviews state-of-the-art advancements in the field of robotics control. Given the application of DRL techniques in this thesis, Section 1.4 offers a brief overview of DRL technology. Finally, Section 1.5 summarizes the

main content of the thesis.

1.1 Robotic Trajectory Planning

Path planning for mobile robots is crucial for optimizing performance in complex environments. A comprehensive path-planning framework typically consists of both global and local planners. The global planner generates paths or reference waypoints based on a worldwide map. In contrast, the regional planner is responsible for creating a smooth trajectory and avoiding previously unmapped obstacles as the robot follows the international path.

Global path planning is a fundamental problem in robotics that involves computing a collision-free trajectory for a robot to navigate from an initial position to a target destination. This process requires considering the robot's environment, which may include various obstacles and constraints. In contrast to local path planning, which addresses immediate, short-term decisions based on real-time sensor data, global path planning evaluates the entire environment to develop a comprehensive strategy for reaching the goal.

Despite advancements in global path planning, several challenges and limitations persist, especially when applying these techniques in real-world scenarios. Key issues include:

1) **High-Dimensional Spaces:** Path planning in high-dimensional spaces, such as those involving robotic manipulators or drones with complex configurations, can be computationally intensive. The number of potential paths increases exponentially with dimensions, making it challenging to find optimal solutions within a practical timeframe.
2) **Real-Time Constraints:** Applications such as autonomous driving and drone navigation often require real-time path planning. Existing algorithms frequently struggle to compute optimal paths, particularly in dynamic environments quickly.
3) **Large-Scale Environments:** Global path planning in large-scale environments, such as entire cities or extensive indoor spaces, presents significant challenges. The vast size of the climate complicates the efficient search for optimal paths.

Local planning, also known as local navigation, is a critical component of autonomous robotics that emphasizes real-time decision-making as a robot traverses its environment. Unlike global path planning, which establishes an overarching path from the start to the goal while considering the entire environment, local path planning addresses immediate, short-term decisions necessary for navigating the robot safely and efficiently within its current surroundings. Key aspects of regional planning include:

1) **Dynamic Environments:** Local planning is crucial in dynamic environments where obstacles can appear suddenly, such as pedestrians on a busy street or other robots in a warehouse. The local planner must continuously adjust the robot's path to avoid collisions.

2) **Sensor Integration:** Local planners depend heavily on real-time sensor data (e.g., LiDAR, cameras, sonar) to detect obstacles and modify the robot's trajectory. This feedback loop allows the robot to react promptly to environmental changes.
3) **Immediate Adjustments:** Local planning emphasizes the robot's immediate surroundings, making rapid decisions about subsequent movements. These decisions are typically based on a predefined window or horizon around the robot, enabling quick responses to new obstacles.

Despite decades of research, several bottlenecks remain in local planning. For example:

1) **Local Minima:** Local planning approaches based on potential fields can encounter local minima, where the robot may become "stuck" in a position without a clear path to the goal.
2) **Overreaction to Obstacles:** Local planners often exhibit strong reactions to obstacles, which can result in erratic or overly cautious behavior. For instance, a robot might execute sharp, unnecessary turns or stop frequently in response to obstacles, reducing efficiency and causing instability in its trajectory.
3) **High Computational Load:** For robots operating at high speeds or within complex environments, the computational demands of local path planning can be substantial. Balancing real-time operation with accuracy and safety remains a significant challenge.

1.2 Mobile Robot Control

Mobile robotics is a fundamental area within the field of robotics, dedicated to developing algorithms and systems that guide robot behavior with precision, reliability, and efficiency. This field integrates principles from various disciplines, including mechanical engineering [14], electrical engineering [14], computer science [15], and control theory [16], to enable robots to execute complex tasks autonomously or semi-autonomously.

Mobile robot control fundamentally involves determining the precise actions required for a robot to achieve specific objectives. These objectives may include navigating to a particular location, handling and manipulating various objects, or maintaining stability and balance. Control systems ensure the robot's actions align with its intended goals. They accomplish this by accounting for both the robot's internal states—such as joint angles or velocities—and external factors, including obstacles or environmental changes. Control systems enable robots to operate effectively and efficiently in diverse and dynamic environments by continuously monitoring and adjusting to these factors.

Among the various types of robots, quadrotors are particularly favored for their simple structure, low cost, and ease of operation. These attributes make them suitable for multiple applications, from hobbyist projects to advanced research and industrial uses. However, the nonlinear dynamics, strong coupling, and under-actuated nature of quadrotor models present significant challenges in control theory. The complexity of quadrotor dynamics makes accurate modeling difficult, necessitating so-

phisticated control algorithms to achieve stable and precise flight. As a result, quadrotor control represents a challenging and illustrative problem within robotics control. This issue is theoretically intriguing and practically significant, as it addresses real-world uncertainties and disturbances. Therefore, research in quadrotor control is crucial for advancing the field of robotics. Progress in this area can enhance the performance and reliability of quadrotors, thereby expanding their applications in areas such as surveillance, delivery, and environmental monitoring.

1.3 Multi-mobile Robot Control

In recent years, the field of multi-quadrotor control has attracted significant attention. This area of research focuses on coordinating and managing multiple quadrotor drones operating simultaneously. The growing interest in multi-quadrotor control is driven by its potential applications across various domains, including surveillance [17], search and rescue operations [18], environmental monitoring [19], and delivery services [20]. Coordinating multiple quadrotors presents unique challenges distinct from those encountered in single-quadrotor control. These challenges include ensuring collision avoidance, maintaining formation, and achieving synchronized movements, all of which demand advanced algorithms and robust communication systems. Conversely, the opportunities offered by multi-quadrotor systems include enhanced efficiency, increased coverage, and the ability to perform complex tasks that a single quadrotor cannot accomplish alone. Consequently, research in multi-quadrotor control is essential for advancing the capabilities and applications of robotic systems.

In summary, multi-quadrotor control can be categorized into several key areas: consensus control, formation control, swarm control, meeting control, synchronization control, and encirclement control.

Consensus Control: [21] The objective of consensus control is to align the states of all agents, ensuring that they eventually converge to the same state or trajectory. This approach is often employed to maintain formation or synchronize movements. For instance, in drone formation flying, all drones must maintain consistent speed and direction to achieve a stable formation.

Formation Control: [22] Formation control involves arranging agents into a predetermined geometric configuration. Common strategies include the leader-follower approach (where one agent leads and others follow), the behavior-based approach (where each agent adheres to local rules), and the virtual structure approach (where agents follow a predefined virtual structure). These methods facilitate cooperative movement within a specified formation.

Swarm Control: [23] Swarm control, also known as flocking control, emulates natural group behaviors, such as those seen in bird flocks or fish schools. Agents achieve complex collective behaviors through simple local rules, including maintaining distance, aligning directions, and avoiding collisions. This method benefits large-scale coordination tasks, such as environmental monitoring and search and rescue operations.

Meeting Control: [24] Meeting control focuses on directing multiple agents to converge at a specific location. This approach is often utilized for task allocation and resource concentration. For example, various robots may need to assemble at a designated spot to complete a task, ensuring that all robots accurately reach the intended location.

Synchronization Control: [25] Synchronization control ensures that agents remain temporally synchronized, which is crucial for tasks requiring precise timing. In multi-robot collaborative transportation tasks, all robots must start and finish simultaneously to ensure seamless execution.

Encirclement Control: [26] Encirclement control, or envelope control, involves agents surrounding a target area or object to ensure it remains within their control range. This method is commonly applied to protection and monitoring tasks. For example, drones might encircle a moving target to maintain continuous monitoring.

1.4 Deep Reinforcement Learning

This thesis employs Deep Reinforcement Learning (DRL) to develop a local planner. Recent advancements in Artificial Intelligence (AI) technology, driven by significant developments in computer science and GPU capabilities, have led to its extensive application across various fields in our daily lives. As a substantial branch of AI, Reinforcement Learning (RL) has demonstrated its potential in numerous areas, including task allocation, robotics control, navigation, and recommendation systems. Based on Markov Decision Proces (MDP) theory and Dynamic Programming (DP) theory [27–29], RL iteratively optimizes an agent's policy through interaction with the environment. Additionally, RL effectively addresses the dimensionality explosion present in DP. Consequently, RL is also referred to as Adaptive Dynamic Programming (ADP) (Approximate Dynamic Programming), Approximate Dynamic Programming (ApprDP) (Approximate Dynamic Programming), Neural Dynamic Programming (NDP) (Neuro-Dynamic Programming), and Adaptive Evaluation Design (AED) (Approximate Experience Design).

Werbos introduced the first reinforcement learning (RL) application to optimization problems in 1987 [30]. He emphasized the need for control systems to adaptively tune themselves as their complexity increases to meet evolving requirements. Werbos stated [30], "Adaptive systems, like human infants, are less agile than young monkeys but have something important to contribute as they mature." This insight laid the foundation for what is now known as ADP (Approximate Dynamic Programming) in addressing optimization problems.

In reinforcement learning control, applications of DRL can be broadly classified into two main categories: end-to-end learning and optimization of controller parameters using DRL. End-to-end DRL represents a comprehensive approach where the learning process spans the entire pipeline, from raw sensory inputs to action outputs, without requiring manual feature engineering or intermediate

processing steps. In this framework, a deep neural network (NN) is trained to directly map high-dimensional inputs, such as images or sensor data, to actions that maximize cumulative rewards. This approach harnesses the representation learning capabilities of Deep Learning (DL) to automatically extract relevant features from raw data, allowing the agent to develop complex policies and behaviors. End-to-end DRL has achieved notable success in various domains, including autonomous driving, robotic manipulation, and game playing, where it has demonstrated superhuman performance by learning directly from interactions with the environment through trial and error.

Using DRL to optimize traditional controllers involves harnessing the advanced learning capabilities of DRL to enhance the performance of conventional control systems. Traditional controllers, such as Proportional-Integral-Derivative (PID) controllers, are popular for their simplicity and effectiveness in many applications. However, tuning their parameters for optimal performance can be challenging, particularly in complex and dynamic environments. DRL can automatically adjust these parameters by learning from interactions with the environment. This method enables the controller to adapt to varying conditions, improving its robustness and efficiency. By integrating DRL with traditional control techniques, superior control performance can be achieved, making this hybrid approach highly valuable in domains such as robotics, autonomous systems, and industrial automation.

1.5 Thesis Overview

This thesis presents our contributions to robotics trajectory planning and control across four chapters, specifically from Chapter 3 to Chapter 6.

Chapter 3 introduces two simulation platforms developed to support the design and validation of our algorithms: the Path Planning Simulation Platform (PPSP) and Deep Reinforcement Learning Simulation Platform (DRLSP). The PPSP integrates several commonly used path planning algorithms as benchmarks and includes a geometric operations function library to increase map complexity during simulations. The DRLSP incorporates over 10 Deep Reinforcement Learning (DRL) algorithms and 16 physical experiments to validate the effectiveness of our proposed methods. All methods and algorithms presented in this thesis have been tested using these simulation platforms. Additionally, we have designed the function interfaces of the simulation platforms to ensure compatibility with Robot Opeartion System (ROS) for ease of experimentation.

Following the development of the simulation platforms, Chapter 4 delves into the trajectory planning problem, which is divided into two stages: the global planning process and the local planning process. For global planning, we enhanced the sampling efficiency of the traditional Rapidly-exploring Random Tree (RRT) algorithm by leveraging geometric information from the map. Specifically, we applied Density-Based Spatial Clustering of Applications with Noise (DBSCAN) to classify obstacles into clusters. Then, the sampling region of RRT was confined to a series of connected "con-

vex hull rings" and "rectangular connectors," significantly improving sampling efficiency. For local planning, DRL methods have become increasingly popular due to their adaptability and independence from explicit models. However, data obfuscation is challenging when the training data exhibits high dimensionality and similar values. To address this, we improved the structure of the critic network in DRL by applying a network decoupling technique. This technique separates the input data into two groups: sensor-related data and motion-related data. Extensive simulations and physical experiments were conducted using an autonomous ground vehicle to validate the effectiveness and superiority of the proposed planning framework.

In addition to trajectory planning, control is a critical aspect of robotics. While controlling ground robots is relatively straightforward due to their typically full actuation and limited nonlinearity under most conditions, controlling Quadrotor Unmanned Aerial Vehicle (QUAV)s presents more complex challenges. QUAVs are characterized by their nonlinear, strongly coupled, and under-actuated dynamics.

Chapter 5 addresses these challenges by focusing on QUAV control. We introduce a novel Recursive Fast Non-singular Terminal Sliding Mode Control(ler) (RFNTSMC) (Robust Finite-Time Nonlinear Sliding Mode Control) approach to stabilize the closed-loop system. A fixed-time disturbance observer (DO) estimates the equivalent disturbances affecting the system. Additionally, DRL is used to optimize hyper-parameters within the RFNTSMC, enhancing the system's robustness and adaptability. The stability of the closed-loop system, including the learning module, is rigorously guaranteed in a Lyapunov sense.

Building on the learning-based hybrid control framework discussed previously, Chapter 6 extends this framework to multi-agent systems. We introduce a fully distributed Fast Non-singular Terminal Sliding Mode Control(ler) (FNTSMC) (Finite-Time Nonlinear Sliding Mode Control) and a robust differentiator-based fixed-time DO to stabilize the closed-loop system. DRL is employed to distribute the optimization of hyper-parameters across the FNTSMCs of each quadrotor, aiming to enhance overall control performance. Given that the agents in the multi-agent system are homogeneous, we utilize DRL to train a single agent within the simulation environment, which is then generalized to the entire multi-agent group.

Based on the control method proposed in Chapter 6, in Chapter 7, a modified Predefined-time (PdT) consensus control framework is introduced for the multi-quadrotor formation control problem. A Predefined-time Disturbance Observer (PdTDO) and a Predefined-time Fast Nonsingular Terminal Sliding Mode Control (PdTFNTSMC) are proposed to estimate the external disturbances and control the multi-quadrotor system simultaneously.

The methods presented in Chapters 4 through 7 have been validated through extensive simulations and real-world experiments. Finally, Chapter 8 summarizes the key contributions of this PhD thesis and outlines potential areas for future research.

Chapter 2

Literature Review

2.1 Trajectory Planning

2.1.1 Global planning

Trajectory planning methods can be categorized into three main types: graph decomposition-based, sampling-based, and biologically inspired (Figure 2.1).

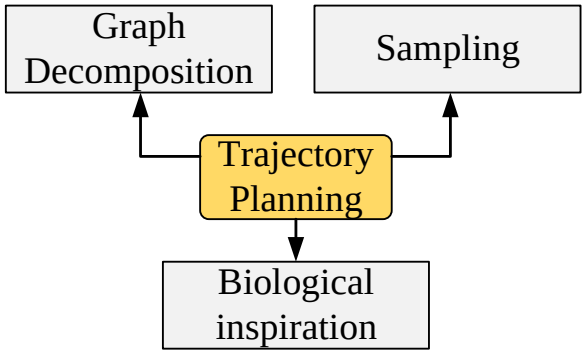


Figure 2.1: Classification of global planning algorithms.

(a) Graph Decomposition-Based Methods

Graph decomposition involves dividing the entire environment into sub-regions marked by various attributes such as obstacles, convex hulls, distances, and reachability. Trajectories are then generated using graph search techniques. These methods typically abstract the map into points, lines, and regions and assign penalties to paths between points based on constraints. Standard graph decomposition methods include Dijkstra Algorithm (D algorithm), A Star (A*) algorithm, D*-lite, and their variants.

Proposed by Dijkstra [31] in 1958, Dijkstra's algorithm is a classical global planning algorithm renowned for its optimality in solving planning problems. However, Dijkstra's algorithm requires a

search of the entire environment, regardless of the number of obstacles or the size of the environment, which can lead to excessive computation time and storage usage.

To address this limitation, A* was developed as a modification to accelerate the search process. Introduced by Dutch scientists in 1968 [32], A* uses a heuristic approach to search for paths. It calculates the cost between points based on prior environmental information and systematically searches for the optimal path during its iterative process, thus reducing search time. As illustrated in Figure 2.2, A* traverses fewer nodes compared to Dijkstra's algorithm when converging on the optimal path, as shown by the yellow grids representing the paths explored by both algorithms.

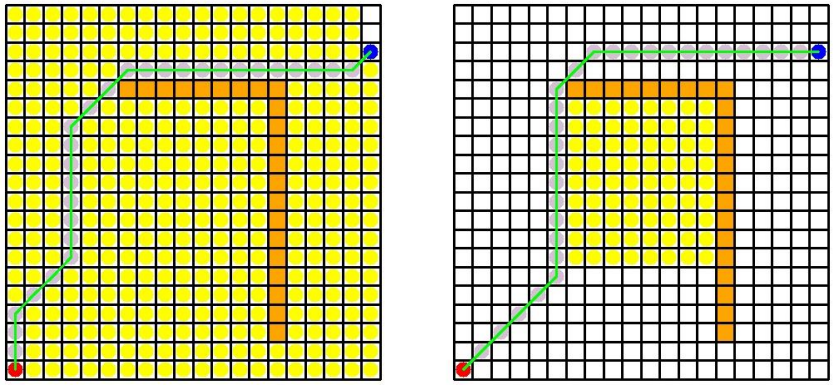


Figure 2.2: A demonstration of D algorithm (left) and A* algorithm (right).

Several improved versions of the A* algorithm have been developed to address its limitations. Dmitri Dolgov et al. [33] introduced Hybrid A*, which integrates kinematic and dynamic constraints, allowing the resulting paths to be used directly as robot control commands. Sedighi et al. [34] combined Hybrid A* with the visible graph planning method, achieving a 40% improvement in runtime for path planning in mixed discrete-continuous environments. Le et al. [35] proposed an enhanced A* algorithm that generates waypoints in narrow spaces, successfully applied to the cleaning robot hTetro. Liu et al. [36] developed a variant that balances path length and navigation safety, addressing uncertainties such as water flow and collision risks in ship trajectory planning.

(b) Sampling-based methods

In contrast to graph decomposition-based methods, sampling-based methods describe paths as sequences of waypoints or nodes, providing a feasible route from the initial point to the terminal point. The primary advantage of sampling-based methods is their ability to find accessible solutions in high-dimensional and complex maps. However, they do not guarantee the optimality of the path. Sampling-based algorithms are typically categorized into three types: Probabilistic Road Map (PRM) [37], Rapidly-exploring Random Tree (RRT) [38], and Artificial Potential Field (APF) [39].

PRM was proposed by Kavraki et al. [37]. It constructs a grid map of possible paths within a given space, distinguishing between available and occupied areas. It uses a map search method to convert

continuous space into discrete space. The fundamental idea is to generate several points within the available subspace randomly and then, along with the initial and terminal points, create a collision-free path that includes both the initial and terminal points.

RRT generates waypoints and sampling points simultaneously and is a widely used sampling-based algorithm. Several advanced methods have been introduced based on the traditional RRT. Kuffner et al. [40] proposed RRT-Connect in 2000, which searches for accessible sub-paths from both the initial and terminal positions simultaneously. The algorithm terminates when the two random trees meet, reducing the number of redundant nodes. Karaman et al. [41] introduced the Rapidly-exploring Random Tree-Star (RRT*) algorithm in 2011, demonstrating its asymptotic optimality. RRT* reconnects nodes within a specific range to replace longer path connections. Informed RRT*, proposed by Gammell et al. [42] in 2014, uses an admissible ellipsoidal heuristic to accelerate the convergence of RRT*. Recently, learning-based RRT methods have emerged. Y. Li et al. [43] proposed a neural network (NN) approximation technique to optimize RRT with kino-dynamic constraints. H. L. Chiang et al. introduced a Reinforcement Learning (RL)-RRT method, which employs a DDPG controller for obstacle avoidance and robot control, using a trained NN reachability estimator as the distance function in RRT. J. Wang et al. [44] developed Neural Rapidly-exploring Random Tree-Star (NRRT*), a neural-based RRT algorithm that predicts the probabilistic distribution of the optimal path using a NN. The dataset for this NN is generated by the A* algorithm, and simulations show that NRRT* performs competitively compared to other state-of-the-art path planning algorithms.

(c) Biological inspiration-based methods

The basic idea of bio-inspired algorithms is to abstract path-planning methods by mimicking the behavior of creatures in the natural world. Unlike graph-based and sampling-based methods, which focus on modeling the environment and solving pathfinding problems, bio-inspired methods address challenges such as local minima and complex multi-objective optimization problems. These algorithms leverage natural processes and behaviors to find solutions, often yielding robust and adaptive strategies for navigating intricate environments.

Genetic algorithm (GA), proposed by Holland [45] in 1975, simulate Darwin's theory of biological evolution to search for optimal solutions. This algorithm transforms problem-solving into a process akin to the natural evolution of chromosomes, involving operations such as crossover and mutation. GAs solve complex optimization problems by mimicking the natural selection process, which iteratively improves solutions through a simulated evolutionary process.

Memetic Algorithm (MA), introduced by Moscato [46] in 1989, heuristically simulates the mutation process supported by extensive domain knowledge. Combining population-based global search with individual-based local heuristic search, MAs provide a flexible framework for integrating various search strategies. This framework allows for forming different algorithms, such as hill climbing, simulated annealing, greedy algorithms, and taboo search, thereby enhancing the algorithm's ability

to solve complex optimization problems.

Particle Swarm Optimization (PSO), introduced by Dr. Kennedy and Eberhart [47] in 1995, addresses optimization problems through the collective behavior of particles and their interactions within a swarm. PSO is known for its simplicity and rapid convergence. However, it can suffer from premature convergence and easily fall into local optima as the problem's complexity increases. These issues are typically addressed by tuning PSO's parameters and integrating them with other methods to enhance performance and robustness.

Ant Colony Optimization (ACO), introduced by Marco Dorigo [48] in his doctoral dissertation in 1992, is inspired by the behavior of ants searching for food. Although the actions of individual ants are relatively simple, their collective behavior demonstrates remarkable intelligence. Ants deposit pheromones along their paths, which guide other ants toward food-rich areas. This pheromone-based communication enables the colony to find efficient paths, and ACO algorithms mimic this process to solve optimization problems by simulating how ants find optimal paths.

Apart from the methods mentioned earlier, many other trajectory-planning algorithms exist. Detailed overviews of multi-agent trajectory planning can be found in the works of Madridano et al. [49], Mohanan et al. [50], Zhao et al. [51], and Aggarwal et al. [52].

2.1.2 Local planning

Besides global planning, local planners focus on the robot's immediate surroundings, allowing the agent to address challenges of unforeseen environmental changes. Several well-established algorithms, such as Dynamic Window Approach (DWA) [53], APF [54], and Time-Elastic Band (TEB) [55], have been introduced as practical solutions for local path planning in robotics.

The DWA [53] emphasizes real-time obstacle avoidance by accounting for the robot's kinematic constraints and dynamic environment. It evaluates a range of possible velocities within a dynamic window, determined by the robot's current velocity and acceleration limits, and predicts trajectories to identify potential collisions. The algorithm then selects the optimal velocity that balances reaching the target and avoiding obstacles, making it particularly effective for autonomous navigation in dynamic environments such as warehouses or urban areas. However, DWA has some drawbacks, including a tendency to get trapped in local optima and an inability to guarantee a globally optimal path. Additionally, it can be computationally intensive, particularly in environments with numerators.

The APF [54] algorithm is widely used in robotics for path planning and obstacle avoidance due to its simplicity and ease of implementation. It models the robot's environment using attractive forces directed toward the goal and repulsive forces pushing away from obstacles. This approach is advantageous because it provides a straightforward mathematical model that is easy to understand and implement. However, APF has notable drawbacks, such as a tendency to get trapped in local minima, where the robot may become stuck in a suboptimal position that is not the goal. Additionally,

it can struggle with target accessibility when obstacles are near the goal, limiting its effectiveness in complex environments.

The TEB [55] algorithm is a local path-planning method used in robotics, particularly for autonomous navigation in dynamic environments. It improves upon the traditional elastic band method by incorporating time as an additional dimension, allowing for more precise trajectory optimization and dynamic obstacle avoidance. TEB is advantageous due to its flexibility in adapting constraints based on different needs and ability to produce smooth, efficient paths. However, it has some drawbacks, such as potential increases in computational complexity and the possibility of local detours in complex environments, leading to higher energy consumption.

Robotics local planning involves creating real-time, feasible paths for robots to navigate dynamic environments while avoiding obstacles. Traditional methods like DWA and APF have been widely used due to their simplicity and effectiveness. However, they often face challenges such as getting trapped in local minima or increased computational complexity. Future research increasingly focuses on Deep Reinforcement Learning (DRL)-based methods, which offer advantages like model-free planning and integrating global and local planning paradigms. DRL-based approaches can adapt to complex, unpredictable environments and enhance the robot's decision-making capabilities. Ongoing research addresses challenges related to computational efficiency and real-time adaptability in these advanced methods.

2.2 Quadrotor Control

In recent decades, Unmanned Aerial Vehicle (UAV)s have garnered significant attention within numerous academic research endeavors due to their extensive range of applications. Among them, the quadrotor is the most extensively examined and utilized type of UAV due to its straightforward mechanical structure, cost-effectiveness, and relatively unconstrained mathematical model with aerodynamics. Ensuring the stable control of a quadrotor is fundamental to achieving dependable flight performance and accomplishing intricate missions. However, quadrotor control presents significant challenges primarily due to its inherent underactuated nature, nonlinearity, and strong coupling among system components [56,57]. Additionally, tracking control becomes increasingly complex when dealing with external disturbances, uncertainties, and time-varying parameters within the quadrotor's dynamics. Thus, achieving robustness and adaptability in the quadrotor's control system is crucial for its extensive utilization.

In response to these challenges, several conventional methodologies have been proposed. Reference [58] introduced a feedback implicit PID controller based on linear matrix inequalities (LMIs) for quadrotor control under time and space constraints. Sun et al. [59] conducted an empirical comparison between two advanced control frameworks: the nonlinear model predictive controller (NMPC) [60]

and the differential flatness-based controller (DFBC) [61], in the context of high-speed flight. An adaptive fuzzy quantized controller for accurate position and attitude tracking was proposed by Zhang et al. [62]. Hou et al. [63] designed a non-singular Terminal Sliding Mode Control(ler) (TSMC) for quadrotor control and examined its performance under rotor failure. Additionally, a robust adaptive backstepping fast TSMC was introduced by Labbadi et al. [64]. Linear-based controllers have also been employed for quadrotor control, such as the finite-horizon Linear Quadratic Regulating (LQR) controller used by Cohen et al. [65], and the Linear Quadratic Gaussian (LQG) controller developed by Zioud et al. [66] for trajectory tracking. Among these methodologies, Sliding Mode Control(ler) (SMC) has garnered significant interest due to its design convenience and robustness. However, the design of the gain for the signum function in SMC critically impacts system performance. Inadequate gains can lead to sluggish system responses or controller chattering, making the design of an appropriate sliding mode approach and chattering mitigation essential concerns in the domain of SMC [64, 67, 68].

Several existing works have addressed this issue. An adaptive fractional-order SMC was proposed by Vahdanipour et al. [69] to control a quadrotor with varying loads, using an adaptive correcting coefficient to estimate load variations. Lian et al. [70] introduced a Fast Non-singular Sliding Mode Control(ler) (FNSMC) for adaptive attitude control of a quadrotor. Labbadi et al. [71] utilized a robust adaptive Fast Non-singular Terminal Sliding Mode Control(ler) (FNTSMC) to handle an uncertain quadrotor with external disturbances, also developing an SMC-based observer to estimate disturbance bounds. Mofid et al. [72] proposed an adaptive finite-time backstepping global SMC for quadrotor tracking control amidst wind disturbances and model uncertainties. Li et al. [73] investigated an Appointed Fixed-time Observer (AFTO) combined with SMC for a quadrotor UAV facing external disturbances, employing a novel disturbance observer (DO)r to estimate these disturbances within a specified settling time. Despite these advancements, two significant challenges remain unresolved. Firstly, many DOs rely on the assumption that the disturbance's derivative remains bounded, with some even assuming it to be zero, which may not always be realistic.

However, such assumptions often lead to inadequate estimation by the observer, which can negatively impact the gain of the signum function in SMC. Another significant challenge is the design of the sliding mode approach law. Determining an appropriate reaching law is complex: if set too large, it can cause oscillations in the system, while a minimal reaching law may result in sluggish response characteristics. Therefore, finding the right balance and optimizing the reaching law is crucial for achieving the desired performance in SMC.

Several methods have been proposed to address the problem of approaching law tuning. Xiong et al. [74] introduced a self-tuning SMC for an uncertain coaxial octorotor UAV. In [75], an adaptive fast TSMC with a power rate proportional reaching law was proposed for position and attitude tracking control of a quadrotor. Beyond these classical methods, the advent of Artificial Intelligence (AI) tech-

nology has generated significant interest in using Adaptive Dynamic Programming (ADP) (approximate dynamic programming) and Reinforcement Learning (RL) (reinforcement learning) as innovative approaches. ADP provides a robust framework for optimizing problems and efficiently solving the complex nonlinear Hamilton–Jacobi–Bellman (HJB) equation [76–78]. Typically, ADP involves two NNs: the actor, which generates control commands, and the critic, which evaluates the quality of the actor's output and helps optimize its performance. This ADP framework operates iteratively through interaction with the environment. Several ADP-based methodologies have been proposed for quadrotor control. Cai et al. [79] introduced a distributed ADP framework for multi-quadrotor formation control. Yi et al. [80] addressed the visual servoing feedback control problem for quadrotors using ADP. An on-policy learning framework for quadrotor tracking control was proposed by Dou et al. [81].

Nevertheless, the approaches above directly utilize ADP to estimate the optimal controller. It is important to note that ADP is inherently a data-driven algorithm, and therefore, directly learning the controller imposes significant requirements on initial system excitation. The network may converge to suboptimal control performance if the initial incentive is not sufficiently effective. On the other hand, using ADP to optimize the hyper-parameters within a classical controller offers a preferable alternative. This method maintains the stability of the closed-loop system throughout the learning process, especially during the initial stages. By restructuring the closed-loop system to parameterize the hyper-parameters that require optimization, it becomes possible to train the actor-network within ADP to achieve an approximate optimal solution while ensuring system stability.

2.3 Multi-quadrotor Control

Multi-agent control encompasses various sub-branches, as detailed in Section 1.3. Among these fields, multi-agent consensus control is particularly significant in swarm robotics. Specifically, multi-agent formation control is a specialized case of multi-agent consensus control. In this context, the reference trajectory for the i-th agent, $traj_{d,i}$, is uniquely defined by the group's center, $traj_d$, and an offset $bias_d$.

Multi-agent consensus control consists of two categories: leaderless consensus control and leader-follower control. Nuño et al. [82] propose a leaderless consensus formation control for a group of two-wheel differential ground robots with output feedback. An adaptive leaderless consensus control for strict-feedback nonlinear systems with unknown model parameters and control directions is investigated in [83]. In [84], the authors consider directed topologies and introduce a leaderless consensus control law using an event-trigger mechanism. Mei et al. [85] investigate a multi-agent leaderless consensus control protocol for uncertain multi-agent systems and directed graphs. Wand et al. [86] introduce a novel control framework to address leaderless consensus in heterogeneous Euler–Lagrange

systems with unknown disturbances. However, leaderless control protocols are generally not applicable to multi-robot systems, such as multi-quadrotor systems, because a physical or virtual leader usually generates a reference signal for the multi-agent system.

As for leader-follow consensus control, many scholars have introduced solid results. In [87], Zhang et al. propose an innovative event-trigger-based leader-follower consensus controller for linear multi-agent systems. A closed-loop state estimator is introduced to decrease the triggering time, and the consensus control protocol has been successfully applied to vehicle platoons. A distributed adaptive leader-follower control protocol for a class of strict-feedback nonlinear systems is proposed in [88]. Considering convergence time, a finite-time leader-follower consensus controller based on the back-stepping theory that compensates for mismatched disturbances is introduced in [89]. Fan et al. [90] propose an event-trigger and sliding mode control (SMC)-based finite-time consensus control framework. A distributed state estimator is used to estimate the tracking errors of each agent, given that the entire topological graph is not fully connected and there are unknown disturbances. Among the controllers mentioned, the SMC is superior due to its robustness, fast response, and simple design. In contrast, other controllers are difficult to adjust in complex environments, are susceptible to disturbances, require high accuracy of the system model, and are computationally complex. Therefore, the sliding mode controller performs better in handling uncertainties and external disturbances.

However, tuning the gains in sliding mode control (SMC) and SMC-related frameworks is challenging. If the gains are set too high, there will be noticeable chattering or overshoot; if the gains are too low, the state response may be sluggish. Therefore, designing an appropriate law to adaptively tune the hyper-parameters in SMC is a critical problem. Deep reinforcement learning (DRL) is an ideal approach for complex function approximation and is naturally suitable for SMC parameter optimization. Additionally, DRL has already been utilized in conjunction with traditional controllers. In [91], approximate dynamic programming (ADP), a DRL technique from the control theory perspective, is integrated with conventional SMC to stabilize multi-agent systems. However, this approach decouples the ADP learning process from the design of the sliding mode surface, effectively using pure ADP control without SMC. Mousavi et al. [92] propose a multi-agent consensus control framework that combines fuzzy control, SMC, and ADP, utilizing fuzzy-based ADP to optimize the sliding mode surface of the system. Similar to the previous approach, the final controllers are predominantly ADP-based rather than SMC-based. Therefore, using DRL to optimize the parameters of SMC directly is of significant research importance.

2.4 Deep Reinforcement Learning

In recent years, artificial intelligence (AI) technology has found extensive applications in almost all areas of life due to significant advancements in computer science and GPU technology. As

a crucial branch of AI, reinforcement learning (RL) theory and technology have demonstrated their potential across various fields, such as task allocation problems, robotics control, navigation, and recommendation systems. Based on Markov Decision Process (MDP) theory and dynamic programming (DP) theory, RL iteratively optimizes the agent's policy through interactions with the environment. Moreover, RL effectively addresses the problem of dimensional explosion inherent in DP. Therefore, RL is also referred to as Approximate Dynamic Programming (ADP), Neuro-Dynamic Programming (NDP), and Adaptive Engineering Design (AED) [93–95].

The first application of using RL to tackle optimization problems was proposed in 1987 by Werbos [96]. He suggested, ' It is necessary for scholars to consider how to make control systems tune themselves adaptively as their complexity increases to meet people's requirements.' Werbos emphasized that 'Adaptive systems, like human infants, are less agile than young monkeys but have something important to contribute as they mature.' This idea forms the foundation of ADP for solving optimal control problems.

Reinforcement Learning (RL) requires an effective function approximation unit. In the early development of RL, various methods, such as tabular approaches, fuzzy logic, and Gaussian processes, were used as approximators. Nowadays, Deep Learning (DL) techniques are widely employed in RL methods, giving rise to the theory of Deep Reinforcement Learning (DRL). In 2015, researchers from DeepMind at Google [97] groundbreaking introduced the first DRL algorithm, Deep Q-Network (DQN). This was the first instance where an agent trained by DRL outperformed human players in 23 Atari computer games. DQN uses video streams captured from the screen as input, with a NN consisting of convolutional layers for feature extraction and fully connected layers for optimal action learning.

Following this, several improved versions of DQN were proposed. H. V. Hasselt et al. [98] introduced Double-DQN, which mitigates the overestimation of the value function. It achieves this by decoupling action selection and evaluation using two sets of parameters: the online network for selecting actions and the target network for evaluating them. This approach reduces overestimation, leading to more stable and reliable learning, and has demonstrated improved performance in reinforcement learning tasks.

Similarly, Z. Wang et al. [99] proposed Dueling-DQN to address the overestimation problem during the training process. This method introduces a novel network architecture that separates the estimation of the state value function from the advantage function, representing each action's relative value in a given state. By doing so, Dueling-DQN allows the model to learn which states are valuable without understanding each action's effect on every state. This leads to more efficient learning and better performance, particularly in environments with many similarly valued actions.

Considering that DQN-based algorithms are limited to discrete action spaces, researchers from DeepMind at Google [100] developed another Deep Reinforcement Learning (DRL) algorithm named

Deep Deterministic Policy Gradient (DDPG) to address continuous action space problems in DRL. DDPG is a state-of-the-art algorithm in reinforcement learning specifically designed for environments with continuous action spaces. It combines the strengths of deterministic policy gradients with DL techniques to learn optimal policies. DDPG operates off-policy, allowing it to learn from data generated by different policies, and uses methods such as Experience Replay and slowly updating target networks to ensure stable training. This makes it particularly effective for tasks requiring precise control, such as robotic manipulation and autonomous driving.

Subsequently, several advanced Deep Reinforcement Learning (DRL) algorithms were proposed. V. Mnih et al. [101] introduced Asynchronous Advantage Actor-Critic (A3C) in 2016. A3C enhances the traditional actor-critic method by running multiple instances of the agent in parallel, each interacting with its copy of the environment. This asynchronous approach helps stabilize training and improves efficiency by decorating the data. A3C is known for its simplicity, robustness, and ability to achieve high performance on various standard reinforcement learning tasks, making it suitable for continuous and discrete action spaces.

J. Schulman et al. [102] proposed Proximal Policy Optimization (PPO) in 2017. PPO balances simplicity, stability, and sample efficiency, making it a popular choice for training agents in complex environments. PPO improves upon previous methods by using a clipping mechanism to limit the size of policy updates, ensuring that the new policy does not deviate excessively from the old one. This approach helps maintain stable and reliable learning, making PPO effective for various tasks, from robotic control to playing video games.

An improved version of Deep Deterministic Policy Gradient (DDPG) named Twin Delayed Deep Deterministic policy gradient (Twin Delayed Deep Deterministic Policy Gradient (TD3)) was proposed in 2018 by S. Fujimoto et al. [103]. TD3 enhances performance and stability in continuous control tasks by incorporating three key techniques: Clipped Double Q-Learning, which mitigates overestimation bias by learning two Q-functions and using the smaller value; Delayed Policy Updates, which updates the policy less frequently than the Q-functions to improve learning stability; and Target Policy Smoothing, which adds noise to target actions to prevent the policy from exploiting errors in the Q-function. These innovations make TD3 a robust choice for complex environments requiring precise and stable control.

More detailed reviews of Deep Reinforcement Learning (DRL) algorithms can be found in [104–106].

Chapter 3

Simulation Platform Establishment

3.1 Research Background and Motivation

Simulation technology is a vital branch in robotics trajectory planning and control. Simulation enables the safe and efficient testing of algorithms and strategies in a virtual environment before real-world implementation. These techniques allow for the precise modeling of robot dynamics, environmental interactions, and potential obstacles, ensuring that the planned trajectories are feasible and optimized for performance. Using simulations, engineers can identify and rectify issues such as collisions, inefficiencies, and unexpected behaviors without risking damage to physical robots or their surroundings. This accelerates the development process and enhances the reliability and safety of robotic systems in complex and dynamic environments.

Simulation techniques are also vital in robotics control, providing a risk-free environment to test and refine control algorithms before deploying them on physical robots. These techniques allow engineers to model and predict the behavior of robots under various conditions, ensuring that control strategies are robust and effective. By simulating different scenarios, including potential failures and environmental interactions, developers can optimize control systems for performance and safety without the high costs and risks associated with real-world testing. This accelerates the development process, enhances the reliability of robotic systems, and ultimately leads to more efficient and safer robotic operations.

There are some open-sourced, well-established simulation platforms or semi-physical simulation platforms. However, these platforms are all designed for more general purposes, which may not be suitable for our research. Therefore, to better serve our research and physical experimental equipment, it is still necessary to establish our simulation platforms.

3.2 Path Planning Simulation Platform

3.2.1 Platform Establishment

The overall structure of the Path Planning Simulation Platform (PPSP) is shown in Figure 3.1. The source code of the PPSP has been open-sourced to Github.

https://github.com/Yang-Yefeng/PathPlanningAlgorithms

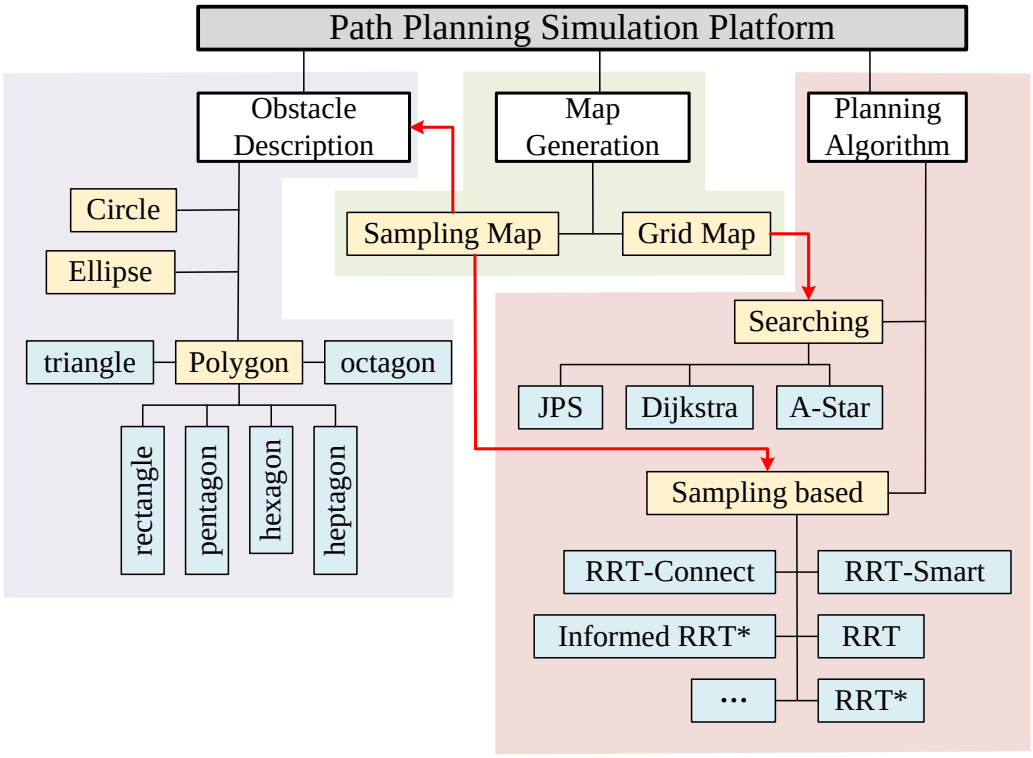


Figure 3.1: The structure of the PPSP.

It is demonstrated that the main body of the PPSP can be divided into three modules, namely,

(a) obstacle description module

(b) stochastic map generation module

(c) planning algorithm module

(a) obstacle description module

The obstacle description module is an essential map generation and planning algorithm component. For planning algorithms such as Dijkstra, A-Star, and Jump Point Search (JPS), only grid maps are required, where obstacles can be simply defined using "0" for free grids and "1" for occupied grids. Therefore, unique characteristics for obstacles in grid maps are not designed. However, obstacles in sampling-based maps (or continuous maps) require more detailed design.

In sampling maps, obstacles can be categorized into polygons and ellipses. For simplicity, poly-

gons are limited to having up to 8 edges, meaning only triangles, rectangles, pentagons, hexagons, heptagons, and octagons are defined. Additionally, circles are considered exceptional cases of ellipses.

To facilitate definition, coding, and description, we represent polygons using their circumscribed circles and angular offsets rather than the coordinates of their vertices. This approach is preferred because manually inputting the coordinates of all vertices, especially for polygons with a more significant number of edges, is cumbersome and tedious. Instead, the coordinates of the vertices can be automatically computed and recorded in *.txt* or *.csv* files, simplifying the design process. Figure 3.2 provides a basic description of the polygon-shaped obstacle design.

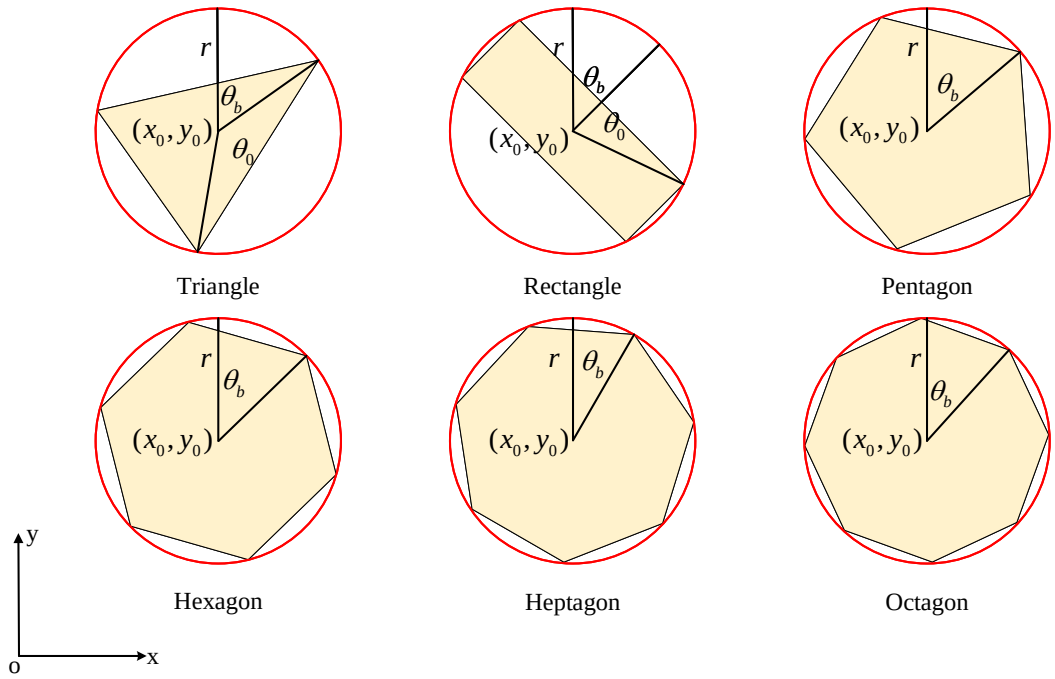


Figure 3.2: The basic description of the polygon design.

For simplicity, only triangles and rectangles are designed to be non-equilateral. In Figure 3.2, polygons with 5 to 8 edges can be uniquely defined by their center (x_0, y_0), radius r, and angular bias θ_b. Additionally, an extra parameter θ_0 is required to design triangles and rectangles.

Similarly, the parameterized design of circles and ellipses is shown in Figure 3.3.

An ellipse can be uniquely defined by the center (x_0, y_0), short-axis a, long-axis b, and angular bias θ_b. Especially, θ_b is neglected when $a = b$ causes the ellipse to degenerate into a circle.

The obstacles defined in a specific map are stored in a Python class. The coordinates of the vertices are recorded for polygons, while for circles and ellipses, the center, long-axis, short-axis, and angular bias are recorded.

(b) stochastic map generation module

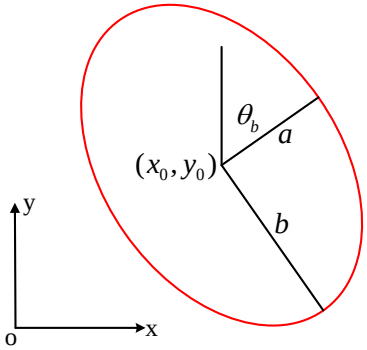


Figure 3.3: The basic description of the circle and ellipse design.

The stochastic map generation module is responsible for generating maps. The maps can be either generated randomly or specifically. For random map generation, Algorithms 1 and 2 are respectively adopted for grid maps and sampling maps.

Algorithm 1 Random grid map generation

Input (1) Map size (x_m, y_m), Number of obstacle grids N_o

Output Map with obstacles

Map initialization: $map()$

Obstacle initialization: $obs()$

for $i := 1$ **to** N **do**

while (x_i, y_i) is an obstacle **do**

$x_i =$ randomly choose in $0, 1, \cdots, x_m - 1$

$y_i =$ randomly choose in $0, 1, \cdots, y_m - 1$

Define new obstacle: $obs_i = (x_i, y_i)$

Add the new obstacle to the obstacle class: $obs.add(obs_i)$

Add all obstacles into the map: $map.add(obs)$

return $map()$

Algorithm 2 Random sampling map generation

Input (1) Map size (x_m, y_m), Number of obstacle grids N_o

Output Map with obstacles

Map and obstacle initialization: $map()$, $obs()$

for $i := 1$ **to** N **do**

Randomly select a new obstacle in $\{Polygon, Ellipse\}$.

if The new obstacle is an Ellipse **then**

Randomly determine (x_i, y_i), a, b, and θ_b.

Define new obstacle: $obs_i = \{(x_i, y_i), a, b, \theta_b\}$.

else

Randomly determines the number of edges from 3 to 8.

Random determine (x_i, y_i), r, θ_b, and θ_0 (triangles and rectangles).

Define new obstacle: $obs_i = \{(x_i, y_i), r, \theta_b, \theta_0\}$ or $obs_i = \{(x_i, y_i), r, \theta_b\}$.

Add the new obstacle to the obstacle class: $obs.add(obs_i)$.

Add all obstacles into the map: $map.add(obs)$

return $map()$

Users can also specially design the configuration of obstacles by manually defining the coordinates, the number, the type, and the parameters of each obstacle. The following figures demonstrate the maps: Figure 3.4 for sampling maps and Figure 3.5 for grid maps.

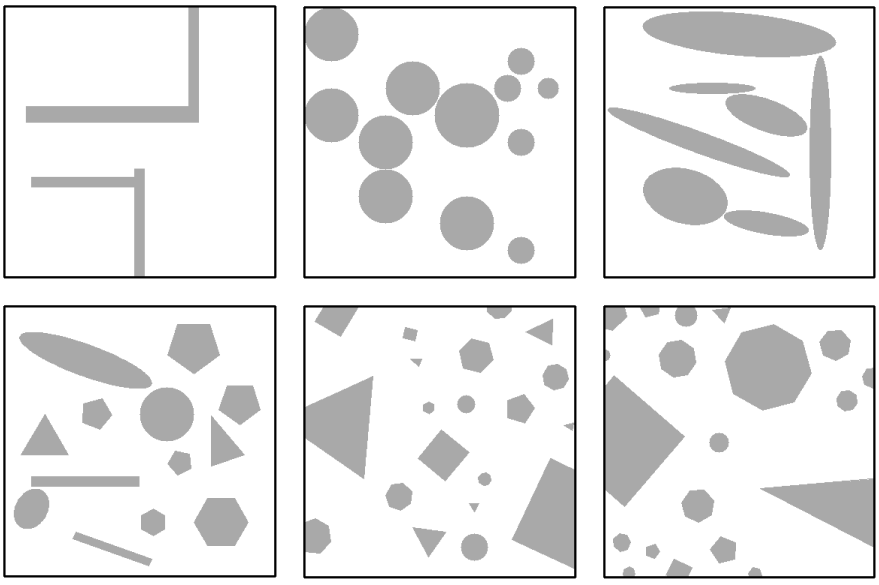


Figure 3.4: Some demonstrations of sampling maps.

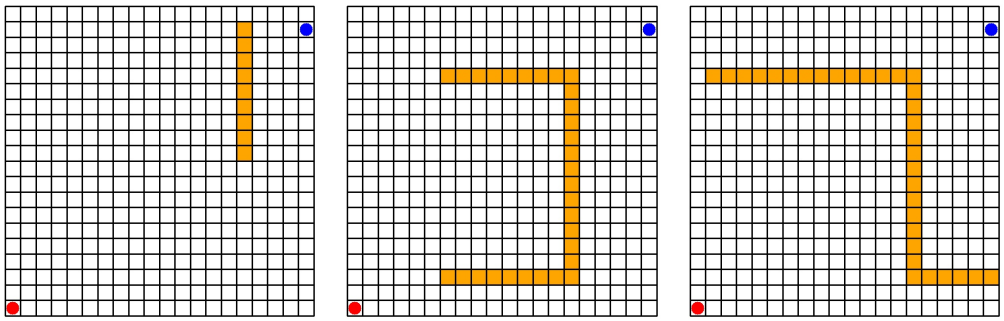


Figure 3.5: Some demonstrations of grid maps.

We also establish a complete geometric computation function library, which is designed as an auxiliary tool for map generation and path planning algorithms. Table 3.1 lists some of those functions and their functionalities.

Table 3.1: Some geometric operation functions.

	Function	**Functionality**
1	point_is_in_ellipse	Determine whether a point is inside an ellipse
2	line_is_in_ellipse	Determine whether a line segment is inside an ellipse
3	line_is_in_many_polys2	Determine whether a line segment is inside multiple polygons
4	dis_point_2_line_segment	Calculate the distance of point to two line segments
5	uniform_sample_on_ellipse	Uniformly sample on an ellipse
6	cal_two_pt_set_dis	Calculate the distance of two point sets
7	get_convex_hull_area	Generate the convex hull of an area
8	get_minAeraEllipse	Calculate the circumscribed ellipse of an area
9	get_convex_hull_triangle	Spilt a convex hull into some triangles
10	point_is_in_poly	Determine whether a point is inside a polygon
11	line_is_in_poly	Determine whether a line segment is inside a polygon
12	is_two_points_same_side_line	Determine whether two points lie on the same side of a line segment
13	dis_point_2_line	Calculate the distance between a point and a line
14	uniform_sample_in_ellipse	Uniformly sample in an ellipse
15	find_convex_hull	Find a convex hull of a point set
16	convex1_is_in_convex2	Determine whether convex hull 2 contains convex hull 1
17	uniform_sample_in_triangle	Uniformly sample in an triangle
18	uniform_sample_between_two_hull	Uniformly sample in two convex hulls

3.2.2 Some Demonstrations

This subsection illustrates some demonstrations of the planning results.

(1) Dijkstra Algorithm

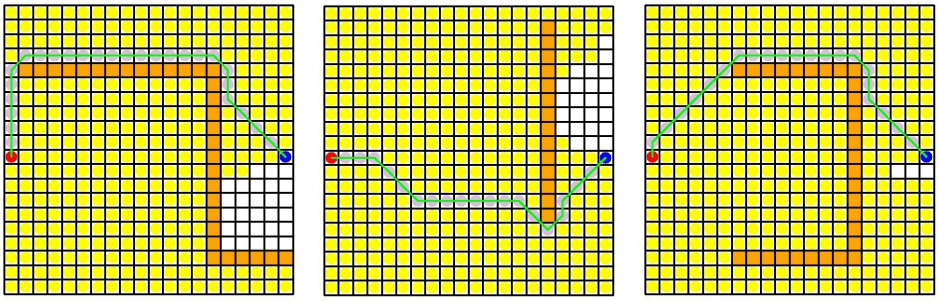


Figure 3.6: Some planning results of Dijkstra algorithm.

(2) A-Star Algorithm

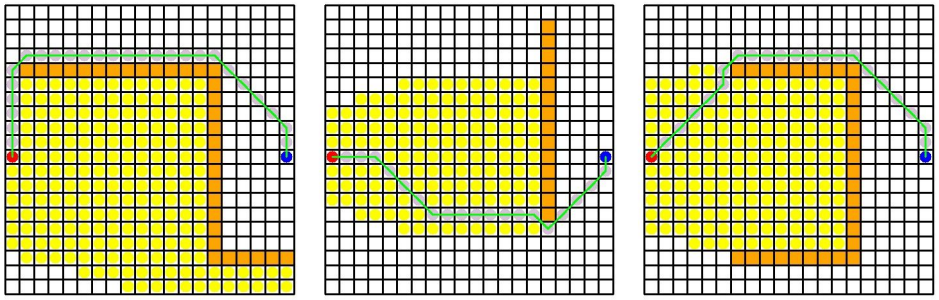


Figure 3.7: Some planning results of A-Star algorithm.

(3) JPS Algorithm

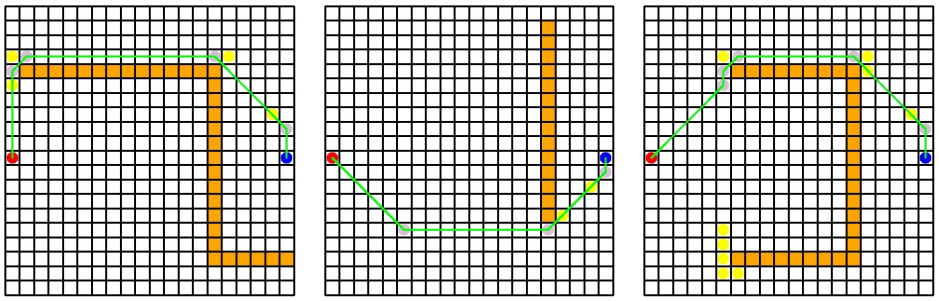


Figure 3.8: Some planning results of JPS algorithm.

(4) RRT Algorithm

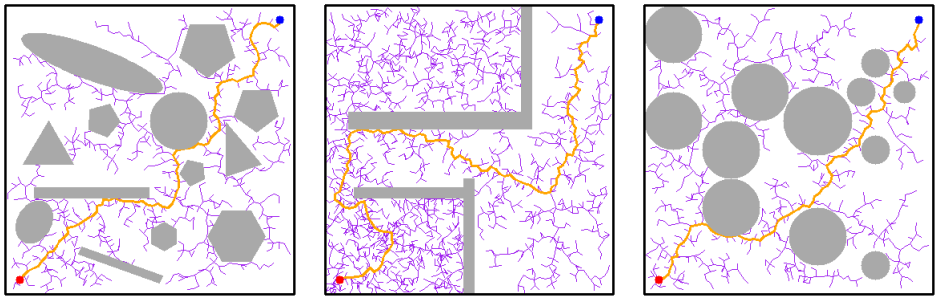


Figure 3.9: Some planning results of RRT algorithm.

(5) RRT-Connect Algorithm

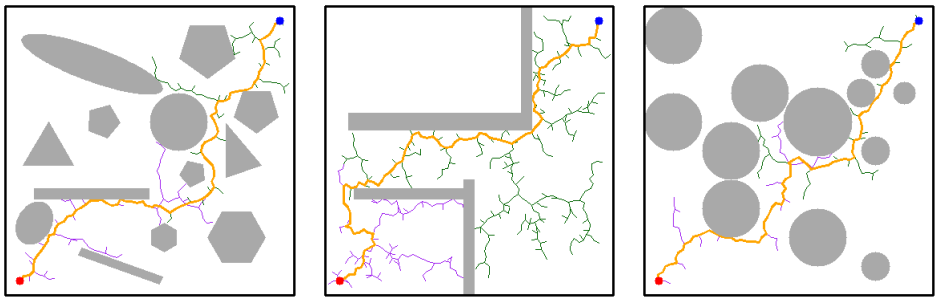


Figure 3.10: Some planning results of RRT-Connect algorithm.

(6) RRT-Connect-Smart Algorithm

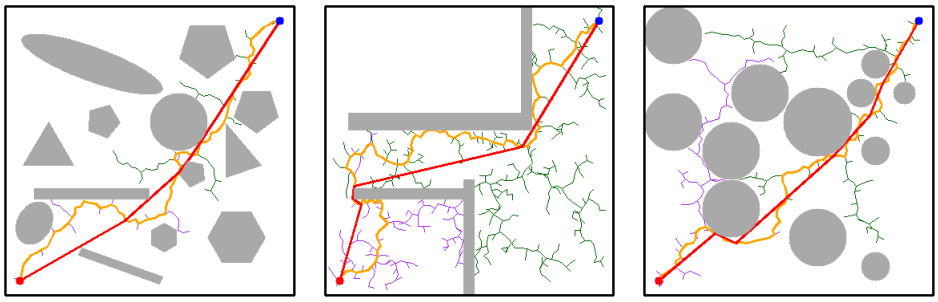


Figure 3.11: Some planning results of RRT-Connect-Smart algorithm.

(7) RRT-Smart Algorithm

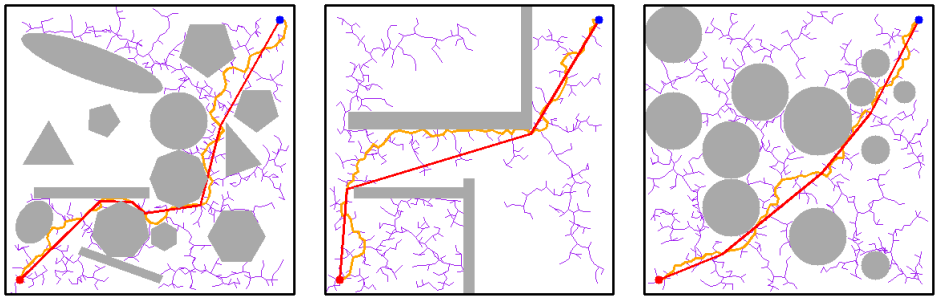


Figure 3.12: Some planning results of RRT-Smart algorithm.

(8) RRT-Star Algorithm

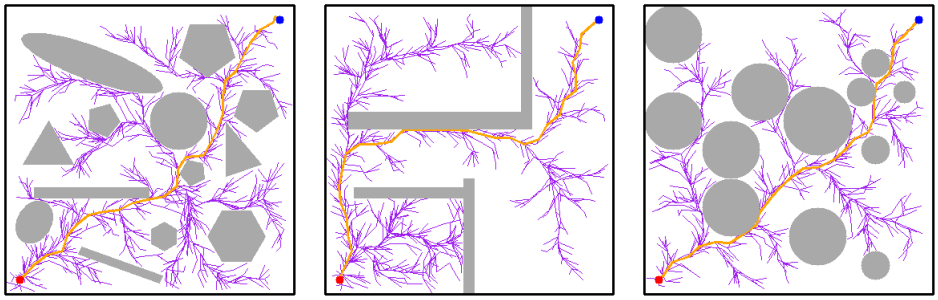


Figure 3.13: Some planning results of RRT-Star algorithm.

(9) RRT-Star-Smart Algorithm

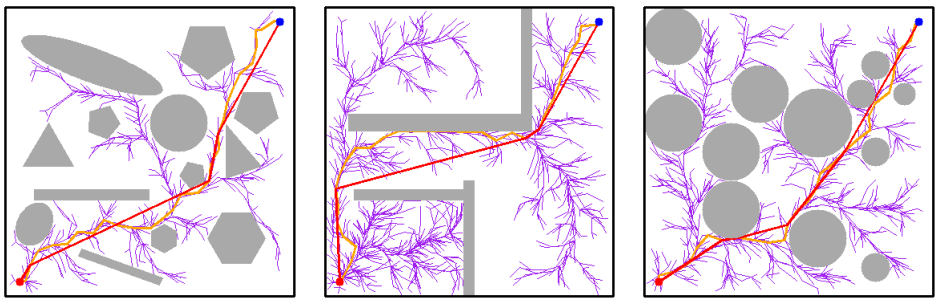


Figure 3.14: Some planning results of RRT-Star-Smart algorithm.

3.3 Deep Reinforcement Learning Simulation Platform

3.3.1 Platform Establishment

The structure of the Deep Reinforcement Learning Simulation Platform (DRLSP) is shown in Figure 3.15. The source code of the DRLSP has been open-sourced at Github:

https://github.com/HKPolyU-UAV/ReinforcementLearningPlatform

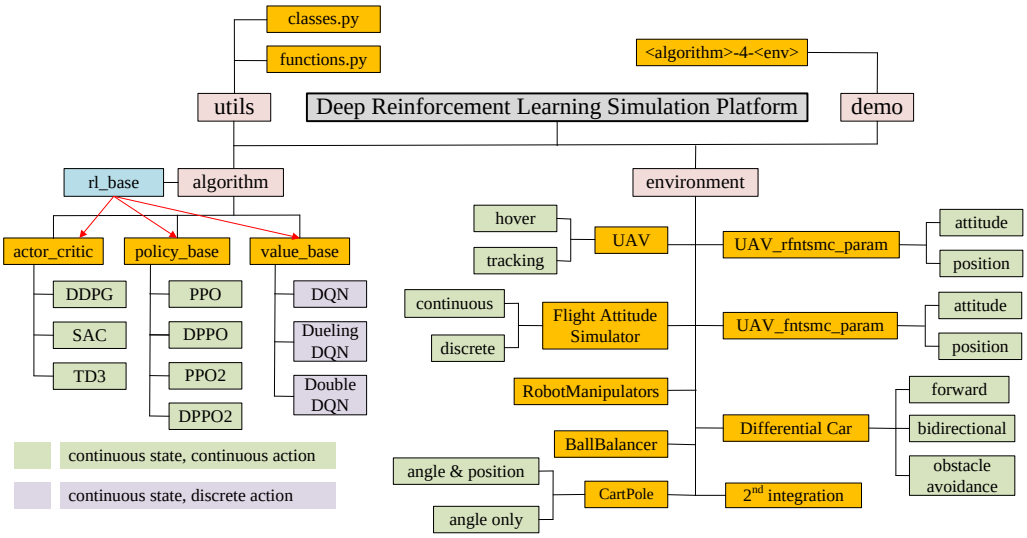


Figure 3.15: Structure of DRLSP.

As shown in Figure 3.15, the entire simulation platform can be divided into four parts, say, (a.) utilization module, (b.) algorithm module, (c.) environment module, and (d.) demonstration module.

(a) utilization module

The utilization module includes some commonly used classes and functions in this DRLSP. For example, Table 3.2 lists some classes and their functionality in the "classes.py" of the utilization module.

Additionally, the "functions.py" integrates some commonly used functions.

(b) Algorithm Module

The algorithm module is a core part of the entire simulation platform. Currently, this platform includes 10 algorithms (shown in Figure 3.15), namely, DQN, Double-DQN, Dueling-DQN, DDPG, TD3, Soft Actor Critic (SAC), PPO, Proximal Policy Optimization2 (PPO2), Distributed Proximal Policy Optimization (DPPO), and Distributed Proximal Policy Optimization2 (DPPO2).

Among the aforementioned DRL algorithms, DQN, Double-DQN, and Dueling-DQN are value-based DRL algorithms. DDPG, TD3, and SAC are actor-critic-based algorithms, while the remaining four are policy-based algorithms.

(c) environment module

Table 3.2: Some classes and their functionality in "classes.py".

	Class	Functionality
1	Actor	The Actor net in DRL
2	Critic	The Critic net in DRL
3	TD3Critic	The Critic net utilized in TD3
4	ReplayBuffer	Replay buffer used in value-based methods
5	RolloutBuffer	Buffer used in policy-based methods
6	PPOActorCritic	The AC framework used in PPO
7	PPOActor_Gaussian	The Actor with gaussian noise used in PPO
8	Normalization	State normalization
9	RewardScaling	Scale the immediate reward
10	SharedAdam	The optimizer used for multi-process learning methods

The environment module is another essential part of the entire simulation platform. The authenticity of the environment setup largely determines the applicability of the controllers learned by DRL algorithms in the real world, which is why we determined to establish the DRLSP by ourselves.

Several well-established open-sourced DRL simulation platforms have been found on GitHub, for example, TianShou designed by Tsinghua University and Lizhi-sjtu designed by Shanghai Jiao Tong University, et al. However, most platforms primarily aim for RL algorithms rather than RL for control. Most environments related to control systems in those platforms are designed to be overly simplistic, making it impossible for them to reasonably approximate real-world physical processes. That is the motivation for us to establish our simulation platform.

Inspired by some popular RL libraries, such as OpenAI Gym, the environment defined in our platform consists of a mathematical model, `rl_base`, graphical visualization, state update, episode termination criteria, reward function, and reset. Figure 3.16 provides the structure of an environment.

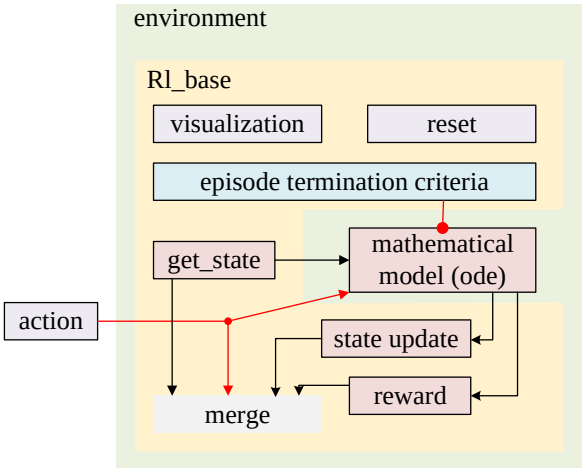


Figure 3.16: The structure of an environment.

All environments are designed with a unified interface to facilitate invocation, debugging, and data storage. Table 3.3 lists all environments in this platform.

Table 3.3: Environments in DRLSP

Environment	Description
UAV (hover, tracking)	UAV control
Flight Attitude Simulator (continuous, discrete)	Flight simulator control
Robot Manipulators	Robot Manipulators control
Ball Balancer	An 1-dimensional ball balancer on a beam
CartPole (angle only)	A cart pole with free position
CartPole	A cart pole with both angle and position
UAV_rfntsmc_param (position, attitude)	Parameter optimization in RFNTSMC of a UAV
UAV_fntsmc_param (position, attitude)	Parameter optimization in FNTSMC of a UAV
Differential Car (forward)	A differential car only move forward
Differential Car (bi-directional)	A differential car move forward and backward
Differential Car Obstacle Avoidance	Obstacle avoidance of a differential car
2^{nd} integration	Second-order integration control

(d) demonstration module This module includes some demonstrations, and all demos follow the naming rule "<DRL algorithm name>-4-<environment name>.py". For example, "PPO-4-BallBalancer1D.py" means using PPO algorithm to learn a controller for a one-dimensional ball balance system. More detailed descriptions are given in section 3.3.2.

3.3.2 Some Demonstrations

This section conducts some well-trained demonstrations. The recorded gifs can be found in the README.md file of our GitHub repository. (https://github.com/HKPolyU-UAV/ReinforcementLearningPlatform) We only use some screenshots to represent the corresponding training environments.

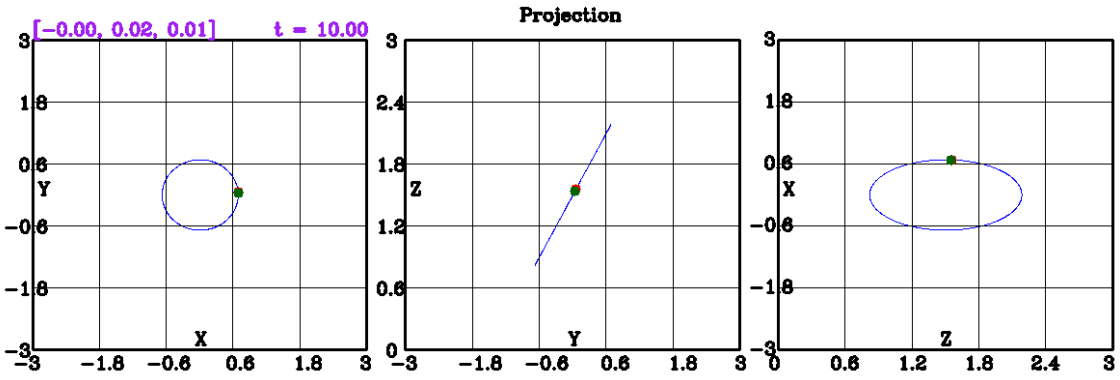


Figure 3.17: Environments of UAV position control-related systems.

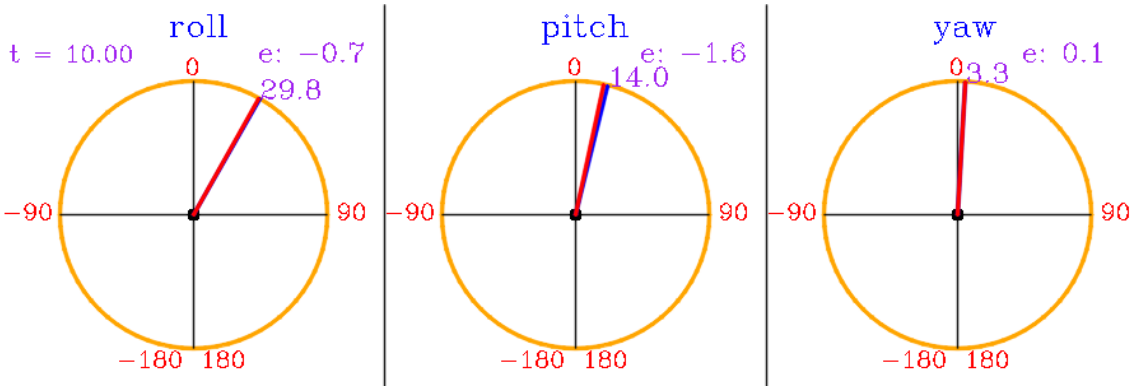


Figure 3.18: Environment of UAV attitude control.

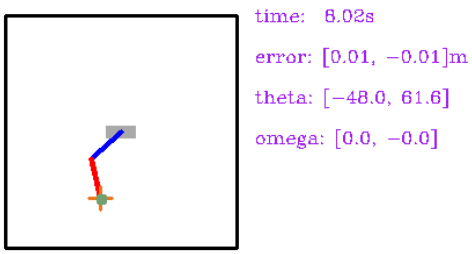


Figure 3.19: Environment of two-link robot manipulator.

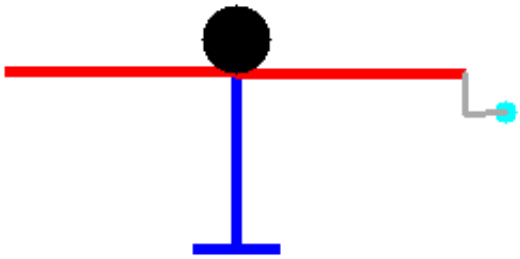


Figure 3.20: Environment of one-dimensional ball balance system.

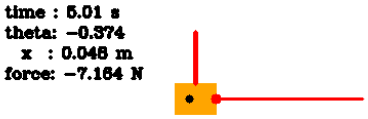


Figure 3.21: Environment of CartPole control-related systems.

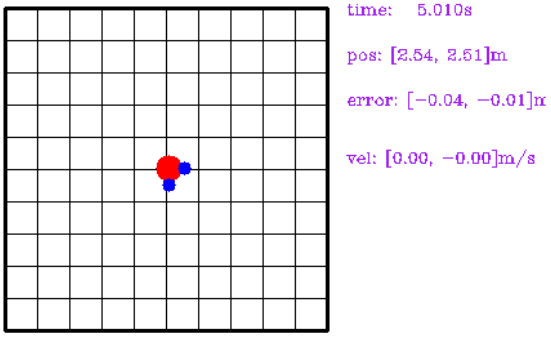


Figure 3.22: Environment of second-order integration system.

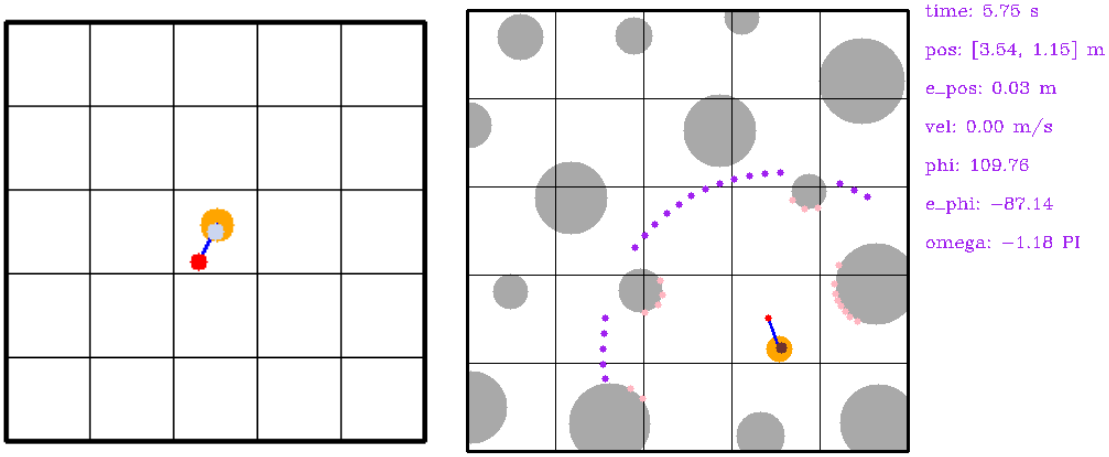


Figure 3.23: Environments of UGV-related systems.

3.4 Conclusion

This section gives a detailed introduction to the two simulation platforms related to our research: the path planning simulation platform and the deep reinforcement learning simulation platform. Simulation projects of the work studied in later sections are all extracted from the two simulation platforms.

Chapter 4

Sampling Efficient Global Path Planning and Obstacle Avoidance

4.1 Research Background

Although widely investigated, robotics trajectory planning remains a hot topic worthy of further study. For global planning, RRT is a basic and efficient algorithm. However, many state-of-the-art methods primarily focus on enhancing the growth of random trees, leading to limited improvements in sampling efficiency. Additionally, intelligently selecting sampling regions presents a promising approach to significantly boost sampling efficiency. In light of these challenges, this chapter introduces a novel method named Adaptive Clustering-based Dynamic Programming-based RRT (ACDP-RRT) to efficiently address the global path planning problem.

For local planning, APF, DWA, and TEB have been proposed to tackle obstacle avoidance. However, in complex environments, precisely defining these boundaries can be challenging, and any imprecision in their specification may result in excessively conservative control strategies. Consequently, many DRL-based methods have been developed. Despite advancements in enhancing robot adaptability, inherent limitations in DRL can negatively impact the effectiveness of local planners. One significant issue is data obfuscation, where different dimensions of structured data input into the neural network have distinct physical meanings but similar values. Data obfuscation highlights the limited feature extraction capability of NNs when dealing with complex data compositions. Therefore, optimizing the structure of NNs in DRL-based algorithms is crucial to mitigate the effects of data obfuscation.

Considering the aforementioned challenges in robot trajectory planning, the main contributions of this chapter are as follows.

1) A novel ACDP-RRT algorithm for robot global path planning is proposed. Compared with several existing algorithms, the ACDP-RRT leverages the geometric characteristics of obstacles

to precisely define the sampling region and guide the growth direction of the tree structure.

2) A novel Network Decoupling (ND)-Actor-Critic (AC) algorithm is introduced for robot local planning. In a pioneering approach, the utilization of ND technology effectively segregates sensor data and robot motion data within the NN, leading to improved data quality. The integration of ND technology enhances the rate of obstacle avoidance, thereby improving the overall performance of the ND-AC local planner.
3) All proposed methods are integrated into a complete autonomous navigation system for ground vehicles. Comprehensive simulations and experiments are conducted to verify the robustness and effectiveness of the proposed methods.

4.2 Sampling Efficient Global Path Planner Design

In this section, we propose the ACDP-RRT global planning algorithm, which includes six steps:

1) Obstacle adaptive clustering
2) Convex hull and key point generation
3) Distance matrix and topological map generation
4) Path search in the topological map
5) Sampling region determination
6) Random tree growth

Then, the feasibility analysis of the ACDP-RRT is performed. Figure 4.1 illustrates the procedures of the ACDP-RRT algorithm.

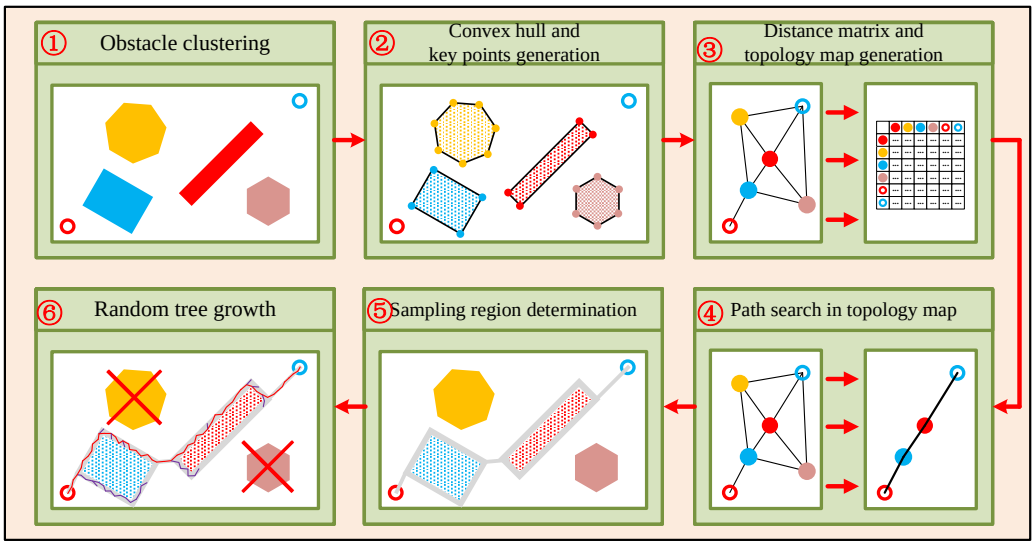


Figure 4.1: Schematic representation of ACDP-RRT.

The proposed ACDP-RRT represents an extension of the classical RRT algorithm. The original RRT algorithm employs a uniform sampling strategy across the entire map, which tends to be suitable

for relatively simple maps without significant complexities. In such scenarios, the uniform sampling approach does not present significant challenges or obstacles to efficient path planning. However, the search efficiency of the RRT algorithm significantly diminishes in the presence of highly complex maps. This can be attributed to the propensity of the random tree's expansion trajectory to encounter obstacles and the inherent lack of foresight in the random tree's growth direction. In contrast to the RRT, ACDP-RRT incorporates the geometric information of obstacles in the map into its path planning strategy, providing directional guidance for the growth of the random tree, thereby improving the sampling efficiency compared to the RRT algorithm.

The ACDP-RRT algorithm extracts the geometric information of the obstacles in the map. The obstacles are grouped into clusters, each bounded by a convex hull. A topological map with a distance matrix is then created to represent the map. The path from the initial position to the final position in the topological map with minimum cost is computed using DP [107]. Finally, the complete sampling region for the RRT is determined.

4.2.1 Algorithm design

1) Obstacle adaptive clustering

Obstacles in the environment are abstracted as convex polygons. To classify obstacles on the map, we use the DBSCAN [108] algorithm. Obstacle clustering is a vital module in the pre-processing of path planning. Although a plethora of clustering algorithms, such as k-means [109] and k-means++ [110], can be effectively employed to cluster obstacles, k-means-based methods require the number of clusters and the initial clustering center for each cluster. In contrast, the DBSCAN algorithm is a density-based clustering method that does not require any prior information about the number of obstacle clusters.

The DBSCAN algorithm classifies data into different clusters and identifies noise using two hyper-parameters: ε and N_0. Here, ε represents the search radius of each data point, while N_0 denotes the minimum number of data points required to form a dense region. Algorithm 3 illustrates the pseudo-code of the adaptive clustering process.

2) Generation of convex hulls and key points

Each cluster is bounded by a circumscribed convex hull. Some vertices of the convex hull are defined as the key points of the cluster. **Definition 4.1** defines the key points.

Definition 4.1 (Key points) *The free space of the map is denoted by $\mathscr{X}_{free}$. For a convex hull $\mathscr{H}$ and the corresponding vertex set $\mathscr{V}_H$, the key point set $\mathscr{K}$ is defined as follows:*

$$\mathscr{K} = \left\{ v | v \in (\mathscr{V}_H \cap \mathscr{X}_{free}) \right\} \tag{4.1}$$

Algorithm 3 Adaptive clustering [108]

Input ε, N_0, obstacle space $\mathscr{X}_{obs} = \cup_{i=1}^{N} \mathscr{X}_{obs}^{i}$ ▷ $\mathscr{X}_{obs}^{i}$ is the ith obstacle
Output Number of clusters: nc, all clusters: C

```
Initialization: wa = {}, nc = 0, C = {}, C' = {}
while X_obs is not empty do
    Pop the first element from X_obs and push it to wa
    while wa is not empty do
        Pop w, the first element of wa
        Push w to C'
        Find all obstacles belong to the same cluster as w by using function find_C(X_obs, w)
        The result of find_C(X_obs, w) is stored in new
        for _new ∈ new do
            Push _new to wa
    Push C' to C
    Empty C'
    nc+ = 1
return nc, C
```

3) Distance matrix and topological map generation

A topological map is a way of modeling a map by simplifying it to only contain essential information. It typically consists of nodes and edges, where each node represents a significant feature or region, and edges denote connections between these features or regions. In this chapter, each obstacle cluster is abstracted as a node, and the interconnections between clusters are modeled as edges.

A distance matrix M is employed to record the distances between different clusters, as well as the robot's origin and destination positions on the map. The topological map is a graphical representation of M. Specifically, M is a square matrix of dimension $(N+2)\times(N+2)$, where N denotes the number of clusters.

Definition 4.2 defines the distance between cluster c_i and cluster c_j.

Definition 4.2 (Distance between cluster c_i and cluster c_j) *We denote $\mathscr{K}_i$ as the key point set of the ith obstacle cluster c_i, $\mathscr{K}_j$ as the key point set of the jth obstacle cluster c_j, and $\mathscr{X}_{obs}$ as the obstacle space. For all $p \in \mathscr{K}_i$, $q \in \mathscr{K}_j$, the distance between c_i and c_j is:*

$$d_{i,j} = d_{j,i} = \begin{cases} \min\limits_{p,q} \|p-q\|_2 & \text{line segment } pq \text{ does not pass through } \mathscr{X}_{obs} \\ -1 & \text{otherwise} \end{cases} \tag{4.2}$$

Algorithm 4 shows the pseudo-code of distance matrix generation.

Algorithm 4 Distance matrix generation

Input Key point sets of all clusters: $\mathscr{K} = \cup_{i=1}^{n} \mathscr{K}_i$

Input $\mathscr{K}_{n+1} = \{\mathscr{S}\}, \mathscr{K}_{n+2} = \{\mathscr{T}\}$ ▷ $\mathscr{S}$ is starting position, and $\mathscr{T}$ is target position

Output Distance matrix M

for $i := 1$ **to** $n+2$ **do** ▷ Traversal of matrix rows
  $M_{i,i} = -1$
  for $j := i+1$ **to** $n+2$ **do** ▷ Traversal of matrix columns
    for $\forall p \in \mathscr{K}_i, \forall q \in \mathscr{K}_j$ **do**
      $M_{i,j} = d_{i,j}, M_{j,i} = d_{j,i}$
return M

4) Topological path search in topological map

DP [107] is utilized to find the topological path with minimum cost in the topological map. Unlike Dijkstra's algorithm [111], which is a single-source shortest path algorithm, DP solves an optimization problem in a graph iteratively by identifying optimal substructures and overlapping subproblems. Consequently, DP can be more efficient in terms of memory and computational resources for large maps [107].

The topological path consists of alternating nodes and edges. Specifically, in the topological path, a node (n) represents either an obstacle cluster, the starting position, or the target position, while an edge (e) represents the connection between two clusters. The cost between the i-th and j-th clusters is given by $M_{i,j}$ (or $M_{j,i}$).

It is important to note that the path found in the topological map represents an abstract expression of the sampling region rather than the final global path. Specifically, the topological path only determines the neighborhoods of obstacle clusters and the connectors between the corresponding clusters. In **Step *5)***, the topological path is utilized to restore the sampling regions in the map accordingly. This means that the actual sampling regions are derived from the vicinity of the obstacle clusters and the connections between them as indicated by the topological path.

5) Determination of sampling region

The complete sampling region is constructed by combining the convex hulls and connectors identified after the path is computed in ***Step 4)***. A convex hull is a polygon that encapsulates an obstacle cluster, while a connector (represented as an edge in the topological path) is a slender rectangle that links two adjacent convex hulls.

If a convex hull does not enclose any other obstacle cluster areas, it is simplified to a convex hull ring. This simplification helps in reducing computational complexity and improving the clarity of the sampling region.

To further optimize the sampling region, we adjust the endpoints of the connectors. This adjustment is performed without altering the path computed in ***Step 4)***. The goal of moving the connectors is to

minimize the distance between adjacent connectors, thereby refining the efficiency of the sampling process. **Algorithm 5** provides the pseudo-code for the connector movement principle.

Algorithm 5 Connector movement

Input (1) Two adjacent connectors (edges) e_i and e_{i+1}

Input (2) The obstacle cluster (node) between e_i and e_{i+1}: n_i

Output Updated e_1 and e_2

Denote the vertices set of n_i as $v = \cup_{j=1}^{N} v_j$
Keep the first endpoint of e_i and the second endpoint of e_{i+1} fixed.
Set $dis = \infty$.
for $id1 := 1$ **to** N **do**
    for $id2 := 1$ **to** N **do**
        Set v_{id1} as the second endpoint of e_i, v_{id2} as the first endpoint of e_{i+1}
        if Neither e_i nor e_{i+1} overlap the obstacle region **then**
            if $\|v_{id1} - v_{id2}\|_2 < dis$ **then**
                $dis = \|v_{id1} - v_{id2}\|_2$
                Update the second endpoint of e_i to v_{id1}
                Update the first endpoint of e_{i+1} to v_{id2}
return e_i, e_{i+1}

6) Random tree growth

Unlike conventional RRT-based methods, the sampling region of ACDP-RRT is limited to the region generated in ***Step 5)*** rather than the entire free space of the map. In ACDP-RRT, the tree begins growing only within the first sub-region. As the random tree expands and its nodes appear in the second sub-region, the growth in the first sub-region is halted, and growth is initiated in the second sub-region. This process of transitioning from one sub-region to the next continues until the tree reaches the final sub-region.

The global planning algorithm concludes when the target position is included within the random tree. The growth principle of the random tree within each sub-region follows the same approach as traditional RRT methods, maintaining consistency in the expansion strategy.

4.2.2 Probabilistic completeness proof

Probabilistic completeness refers to the property wherein the likelihood of discovering a feasible solution if one exists, tends to converge to unity as the number of sampling episodes increases [112].

Let $\mathscr{X}$ be the space of the environment. The free space in $\mathscr{X}$ is denoted as $\mathscr{F}$. The obstacles in $\mathscr{X}$ are denoted as $\mathscr{X}_{obs}$. The start state of the robot is denoted as x_s. The target state of the robot is denoted as x_t. $\mathscr{B}_r(x)$ denotes the ball of radius r and centered at x. For simplicity, we assume that

there exists $d_0 > 0$, and a target region $\mathscr{X}_t \in \mathscr{F}$, such that $\mathscr{X}_t = \mathscr{B}_{d_0}(x_t)$. Then, the pseudo-code of traditional RRT (**Algorithm 1** in [113]) can be given by

Algorithm 6 RRT

Input x_s, x_t, $\mathscr{X}_t$, d, K, τ

Output The random tree τ

```
for i = 1 to K do
    x_rand ← RANDOM_SAM()
    x_near ← NEIGHBOR(x_rand, τ)
    x_new ← NEW_NODE(x_rand, x_near, d)
    if COLLISION_FREE(x_near, x_new) then
        τ.add_vertex(x_new)
        τ.add_edge(x_near, x_new)
```

In **Algorithm 6**, τ is the random tree, K is the number of iterations, and d is the distance $\|x_{near} - x_{new}\|$. Noting that x_{rand} is a vertex that uniformly sampled in $\mathscr{X}$, $x_{near} \in \tau$ is the vertex which is nearest to x_{rand}, and x_{new} is on the line segment between x_{near} and x_{new}. In addition, the function $COLLISION\_FREE()$ is utilized to check if the path from x_{near} to x_{new} is collision-free. If so, x_{new} is added as a vertex of τ, and the line segment between x_{near} and x_{new} is added as an edge of τ. Otherwise, the candidate vertex x_{new} is discarded.

We now introduce a preliminary lemma and theorem to support the probabilistic completeness proof. Suppose that a branch in the random tree corresponds to a collision-free path τ from the initial point x_s to the destination x_t. Let the length of this path be L, and let the number of vertices on the path be $p > \frac{5L}{d}$, where $d < \delta_0$ and δ_0 is the minimum distance between the random tree and the obstacles. With these assumptions, we can now state the following lemma and theorem.

Lemma 4.1 (Lemma 1 in [113]) *By the above analysis, we denote the vertexes on the path as $x_0 = x_s$, x_1, $\cdots$, and $x_p = x_t$. We assume that the length between two consecutive points is less than $d/5$, namely, $\|x_i - x_{i+1}\| \leq d/5$ for $0 \leq i < p$. Suppose that RRT has reached the ith vertex x_i. Then, there must be another vertex x_i' such that $x_i' \in \mathscr{B}_d(x_i)$ (otherwise, x_i cannot exist). If a new random node x_{rand} is in the near region of x_{i+1}, namely, $x_{rand} \in \mathscr{B}_d(x_{i+1})$. Then, the line segment between x_{rand} and x_{near} in τ lies in free space $\mathscr{F}$.*

Further, note the fact that $\|x_{rand} - x_{near}\| \leq \|x_{rand} - x_i'\| \leq \|x_i' - x_i\| + \|x_i - x_{i+1}\| + \|x_{i+1} - r_{rand}\| \leq 3 \cdot \frac{d}{5} < d$, which means that $x_{rand} = x_{new}$. Then, a theorem can be given as follows.

Theorem 4.1 (Theorem 1 in [113]) *Based on Lemma 4.1, the probability that RRT fails to reach $\mathscr{X}_t$ after k iterations is at most $\frac{p}{p-1}k^p e^{-p_0 k}$, namely,*

$$Pr[X_k < p] \leq \frac{p}{p-1}k^p e^{-p_0 k},$$

where Pr is the probability that RRT fails to reach the goal after k iterations, p is the number of vertexes on the path, p_0 is the probability that x_{rand} falls into the ball $\mathscr{B}_d(x_i)$.

Theorem 4.1 shows that the probability decays exponentially with k. For the ACDP-RRT algorithm, the comprehensive sampling region is divided into multiple concatenated sub-regions, which are interspersed with connectors and convex hulls (or convex hull rings). The following observations can be made:

a). The principle underlying the proposed ACDP-RRT implies that two adjacent sampling areas are bound to have overlapping segments.

b). Based on the principle of ACDP-RRT, the presence of obstacles within certain sub-sampling areas does not compromise the connectivity between adjacent areas.

Therefore, based on the analysis, it can be inferred that all sampling sub-regions are interconnected. Given that the random tree generation process in ACDP-RRT follows the same principles as traditional RRT, we can conclude that ACDP-RRT maintains the probabilistic completeness of RRT. Hence, ACDP-RRT is proven to be probabilistically complete.

4.3 DRL-Based Obstacle Avoidance Method Design

In this section, we present the ND-AC local planning algorithm, which represents an advancement over the classical AC-based method. Both AC and ND-AC share a common learning framework. However, traditional AC primarily focuses on the agent's learning process within the environment, often neglecting the importance of data pre-processing. As a result, when structured data contains excessive information, the quality of neural network learning can deteriorate in traditional AC. The introduction of a network decoupling pre-processing module in ND-AC significantly enhances training performance compared to traditional AC methods. By decoupling the network's input processing, ND-AC can handle different aspects of the input data independently. This approach improves the network's ability to extract meaningful information from the input, leading to enhanced training performance and, ultimately, superior overall performance compared to traditional AC algorithms.

4.3.1 AC framework

The basic local planning learning framework utilized in this study is AC [114], which integrates Policy Gradient (PG) [115] and Value Iteration (VI) [116] methods. The proposed AC allows the

benefits of both PG and VI to coexist within DRL. The AC framework simultaneously updates the policy network (actor) and the value network (critic). In the context of the local planner, the actor serves as the robot's controller, responsible for obstacle avoidance. The critic, in this case, is an NN-based cost function that evaluates and quantifies the performance of the actor. Figure 4.2 illustrates the relationship between the AC framework and the local planner.

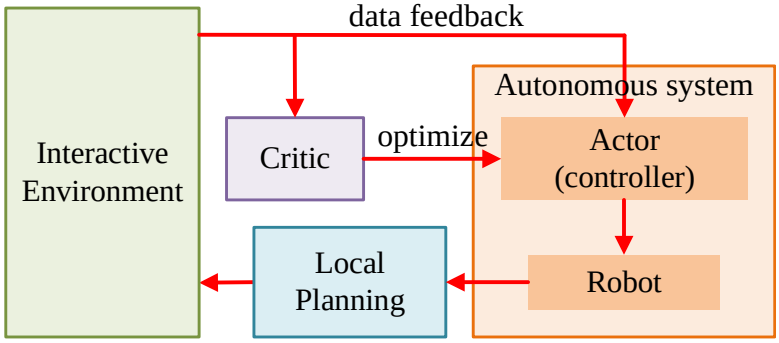


Figure 4.2: The relationship between AC framework and the local planner.

The objective of the actor is to maximize the expectation of cumulative reward:

$$\begin{aligned}\eta(\theta) &= \mathbb{E}_{\tau\sim\pi_\theta}[G_t|S=s,A=a] \\ &= \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)] \\ &= \sum_{s\in S} d^{\pi_\theta}(s) \sum_{a\in A} \pi_\theta(a|s) Q_{\pi_\theta}(s,a;\theta),\end{aligned} \tag{4.3}$$

where $\eta(\theta)$ is the objective function, π_θ is the policy parameterized by θ, s is the state, a is the action, τ is a trajectory $(s_1,a_1,s_2,a_2,\cdots,s_n,a_n,\cdots)$ induced by π_θ, $R(\tau)$ is the reward function, $d^{\pi_\theta}(s)$ is the distribution of the state space with the current policy π_θ, $Q_{\pi_\theta}(s,a;\theta)$ is the state-action value function of the tuple $\langle s,a\rangle$, and θ is the parametric vector of the actor-network in the AC framework.

According to policy gradient theory (**Theorem 1** in [115]), the gradient of $J(\theta)$ with respect to θ is given by:

$$\begin{aligned}\nabla_\theta\eta(\theta) &= \nabla_\theta \sum_{s\in S} d^{\pi_\theta}(s) \sum_{a\in A} Q_{\pi_\theta}(s,a)\pi_\theta(a|s) \\ &= \sum_{s\in S} d^{\pi_\theta}(s) \sum_{a\in A} Q_{\pi_\theta}(s,a)\nabla_\theta\pi_\theta(a|s) \\ &= \sum_{s\in S} d^{\pi_\theta}(s) \sum_{a\in A} \pi_\theta(a|s) Q_{\pi_\theta}(s,a) \frac{\nabla_\theta\pi_\theta(a|s)}{\pi_\theta(a|s)}.\end{aligned} \tag{4.4}$$

It follows that:

$$\nabla_\theta\eta(\theta) = \mathbb{E}_{\pi_\theta}[Q_{\pi_\theta}(s,a)\nabla_\theta \ln\pi_\theta(a|s)]. \tag{4.5}$$

The actor net is introduced to approximate policy $\pi_\theta(a|s)$. A critic net is unitized to approximate the state-action value function $Q_\pi(s,a)$. The objective of the critic net is to maximize the state-action value function $Q_\pi(s,a)$ by minimizing the temporal difference (TD) error. The formula for the TD error δ_t is given by:

$$\delta_t = r_{t+1} + \gamma Q_{\pi_\theta}(s_{t+1}, a_{t+1}) - Q_{\pi_\theta}(s,a), \tag{4.6}$$

where r_{t+1} is the immediate reward of step $t+1$ and δ_t is the loss function of the critic network whose gradient can be computed automatically by the Deep Learning Toolbox (DLT). Algorithm 7 provides the pseudo-code of the one-step training of AC, where τ is the soft update rate, and the $learn()$ is automatically realized by DLT.

Algorithm 7 One-step learning of AC

Input (1)Target actor-network: $\pi(s;\theta)$, evaluation actor-network: $\pi'(s;\theta')$
Input (2)Target critic network: $q_\pi(s,a;\omega)$, evaluation critic network: $q'_\pi(s,a;\omega')$
Output Updated $\pi(s;\theta)$, $\pi'(s;\theta')$, $q_\pi(s,a;\omega)$, and $q'_\pi(s,a;\omega')$

Get action a_{t+1} at time step $t+1$: $a_{t+1} \leftarrow \pi(s_{t+1};\theta_t)$
Get values Q_{t+1} and Q_t at time step $t+1$ and t: $Q_{t+1} \leftarrow q_\pi(s_{t+1}, a_{t+1};\omega_t)$, $Q_t \leftarrow q'_\pi(s_t, a_t;\omega'_t)$
Get reward r_{t+1} at time step $t+1$ from the environment
critic net training:
    $loss = \frac{1}{2}(r_{t+1} + \gamma Q_{t+1} - Q_t)^2$
    Compute ω'_{t+1} ▷ Computed automatically by DLT
actor net training:
    $loss = -q'_\pi(s_t, \pi'(s_t;\theta'_t);\omega'_t)$
    Compute θ'_{t+1} ▷ Computed automatically by DLT
parameter update
    $\omega_{t+1} = \tau\omega_t + (1-\tau)\omega'_{t+1}$
    $\theta_{t+1} = \tau\theta_t + (1-\tau)\theta'_{t+1}$
return $\pi(s;\theta)$, $\pi'(s;\theta')$, $q_\pi(s,a;\omega)$, and $q'_\pi(s,a;\omega')$

4.3.2 Network decoupling technology

Theoretically, neural networks (NNs) with more than two layers possess the capability to approximate functions of arbitrary complexity. However, their practical performance may be unreliable due to limitations in data quality and the potential for the optimizer to get trapped in local minima. To address these challenges, Network Decoupling (ND) techniques are employed to enhance NN learning performance. By decoupling the network's input processing, ND helps mitigate the adverse effects of insufficient data quality and local minima, thereby improving the overall performance and reliability of NN-based function approximation.

Figure 4.3 provides a comparative demonstration of the original actor network and the decoupled

actor network. The input data to the neural network (NN), which represents the state space of the autonomous robot, consists of two parts: the motion part (s_1 in Figure 4.3) and the sensor part (s_2 in Figure 4.3). The layers l_1, l_2, and l_3 are fully connected hidden layers with the `ReLU` activation function, while l_4 is the output layer with the `tanh` activation function. The number in each bracket following $l.$ denotes the number of neurons in the corresponding layer. The green circular node indicates the concatenation of the outputs from l_{13} and l_{23}.

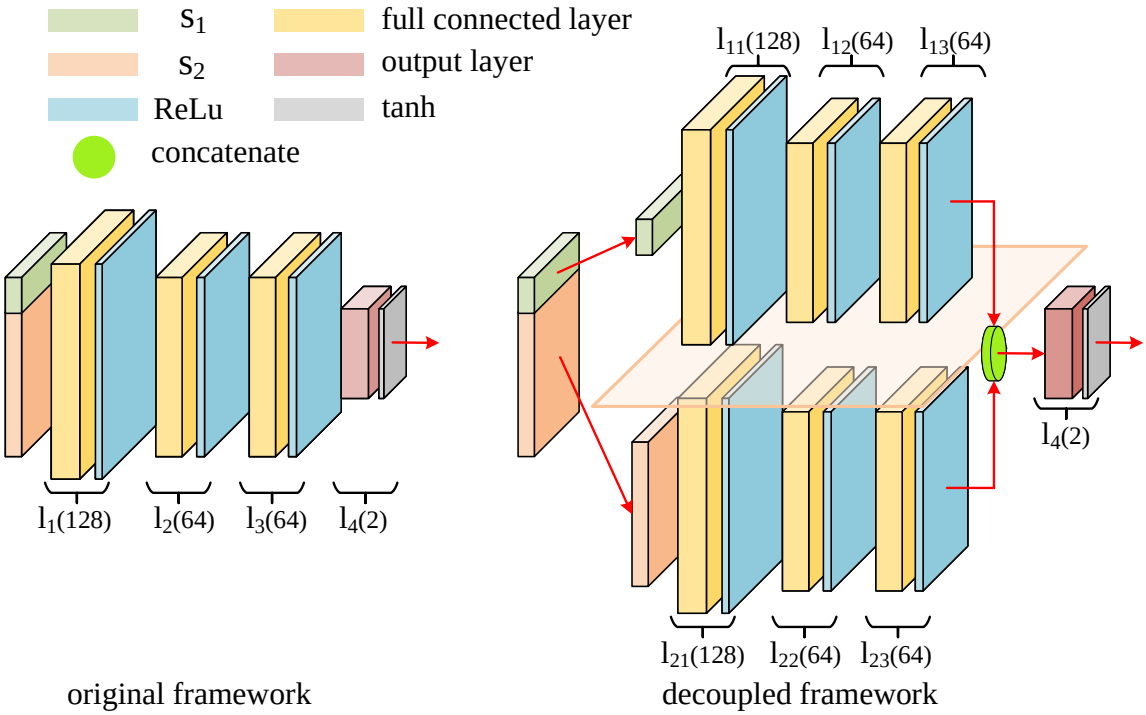


Figure 4.3: Comparison of the actor networks in the original learning framework and the decoupled learning framework.

The physical meanings of these two parts are different. The sensor-related data represents the spatial distribution of obstacles around the robot, while the motion-related data encompasses information about the robot's position, orientation, and velocity. The implementation of Network Decoupling (ND) technology facilitates the independent processing of these two types of data. By doing so, the two distinct sub-networks can independently extract their unique features without interference from each other. The concatenation of the outputs from these sub-networks occurs prior to the final hidden layer. Ultimately, the output of the actor-network is influenced by both the motion and LiDAR components.

4.4 Simulation and Experiments

This section contains comparative simulations and real-world experiments to demonstrate the robustness and effectiveness of the proposed global and local planner. Figure 4.4 shows the flowchart of the systemic path planning framework for a real autonomous robot system.

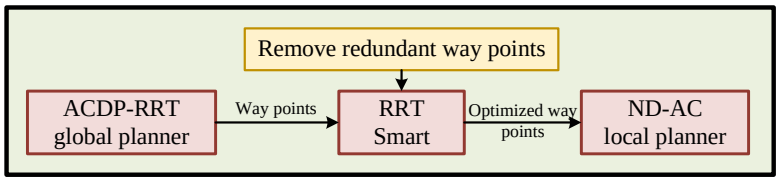
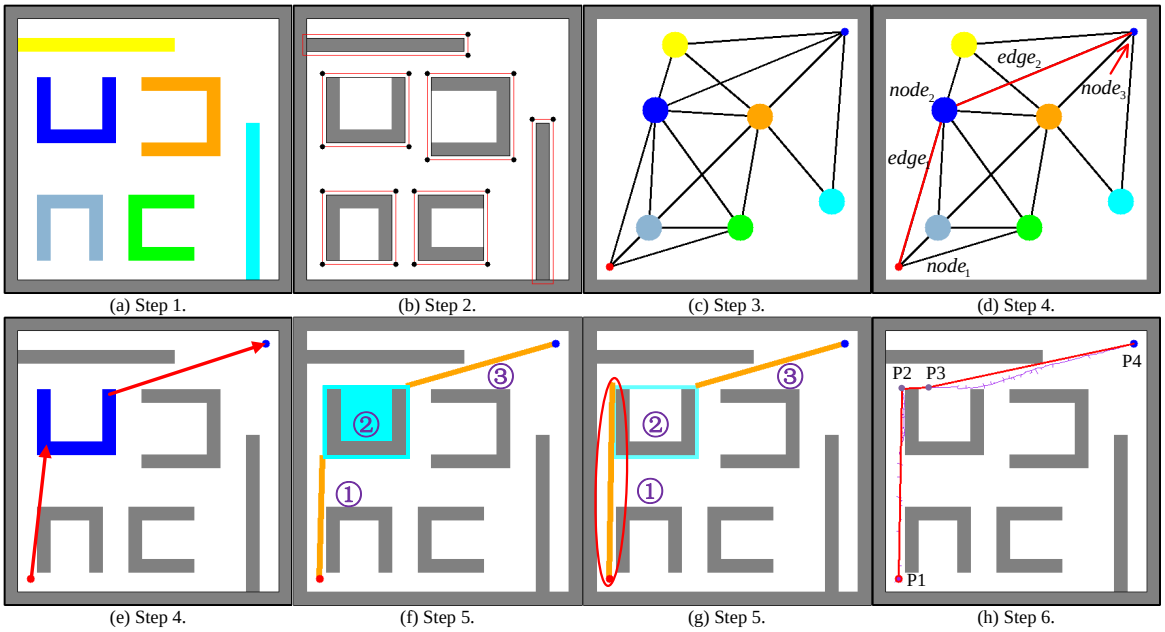


Figure 4.4: Data flow diagram of the systemic path planning framework.

As shown in Figure 4.4, the complete path planning task is divided into two parts: global planning and local planning. The RRT-Smart technique is utilized as a bridge between the global planner and the local planner. Initially, the ACDP-RRT global planner generates a global path for the autonomous robot, represented by a sequence of nodes. However, this path often contains an excessive number of redundant nodes, which increases the burden on the local planner. To address this issue, the RRT-Smart methodology is applied to effectively eliminate redundant nodes and optimize the path. RRT-Smart connects the furthest node on the path to the current node if the node is directly visible. The detailed optimization process of RRT-Smart is described in **Algorithm 2** in [117]. After optimization with RRT-Smart, the number of nodes in the global path is significantly reduced. The remaining nodes in the optimized path are considered targets for the local planner. The robot then sequentially advances towards each of these nodes until it reaches the vicinity of the final node.

4.4.1 Simulation results of the global planner

Figure 4.5 demonstrates the detailed process of global planning. In Figure 4.5(a), six clusters are generated and obstacles belonging to different clusters are marked with different colors. After that, six convex halls (the red rectangles in Figure 4.5(b)) and the corresponding key points (the black solid points in Figure 4.5(b)) are generated. Additionally, all convex hulls are extended outward by half the size of the robot to ensure that the robot can pass through the gaps between obstacle clusters. Then, as shown in Figure 4.5(c), the topological map abstracted from Figure 4.5(b) is developed. In Figure 4.5(d) and (e), the topological path (the red lines) is generated in the topological map. Based on the result in **Step 4**, the sampling region is determined in **Step 5**. The sampling region shown in Figure 4.5(f) is the combination of one convex hull (the cyan region in Figure 4.5(f)) and two connectors (the orange rectangles in Figure 4.5(f)). The simplified sampling region is shown in Figure 4.5(g), which is, the convex hull is simplified as a convex hull ring. Finally, based on **Step 5**, the global path containing four-way points (P1 to P4 in Figure 4.5(h)) is generated.



(a) Step 1. (b) Step 2. (c) Step 3. (d) Step 4.

(e) Step 4. (f) Step 5. (g) Step 5. (h) Step 6.

Figure 4.5: Global planning process of the ACDP-RRT.

Figure 4.6 shows the planning results in different scenarios using ACDP-RRT. We compare our proposed ACDP-RRT algorithm with two RRT-based planning algorithms: RRT and RRT*. For each office scenario depicted in Figure 4.6, we select any two of the four corners of the room as a $(\text{start}, \text{target})$ tuple. The dimensions of the office scenarios are $11\,\text{m} \times 11\,\text{m}$. The four corners are located at $(1\,\text{m}, 1\,\text{m})$, $(1\,\text{m}, 10\,\text{m})$, $(10\,\text{m}, 1\,\text{m})$, and $(10\,\text{m}, 10\,\text{m})$. There are $C_6^2 = 12$ $(\text{start}, \text{target})$ tuples. Each tuple is tested 50 times in each scenario. We evaluate the time and memory efficiency of each algorithm based on the time consumption and the number of nodes in the random tree when generating paths.

Figure 4.7 shows the efficiency of each algorithm across all ten scenarios. The box plots indicate the standard deviation, while the line within each box represents the mean value of the 600 simulation results. The top sub-figure of each scenario illustrates time efficiency, and the bottom sub-figure depicts memory efficiency. The ten figures clearly demonstrate that the performance of the proposed ACDP-RRT algorithm significantly surpasses that of the conventional RRT and RRT*.

The time efficiency can be assessed by comparing the top sub-figures in Figure 4.7. Although RRT* is an enhanced version of RRT, its primary improvements focus on path length and the proof of asymptotic optimality. However, path length is not a criterion in this study because RRT-Smart [117] is applied after path generation, making the length of the global path a non-critical factor. Generally, the time efficiency of RRT* is lower than that of RRT due to the need for tree re-connecting within the neighborhood of new nodes. In contrast, the time efficiency of ACDP-RRT is significantly better than that of the other two algorithms in all scenarios. ACDP-RRT searches the map more direc-

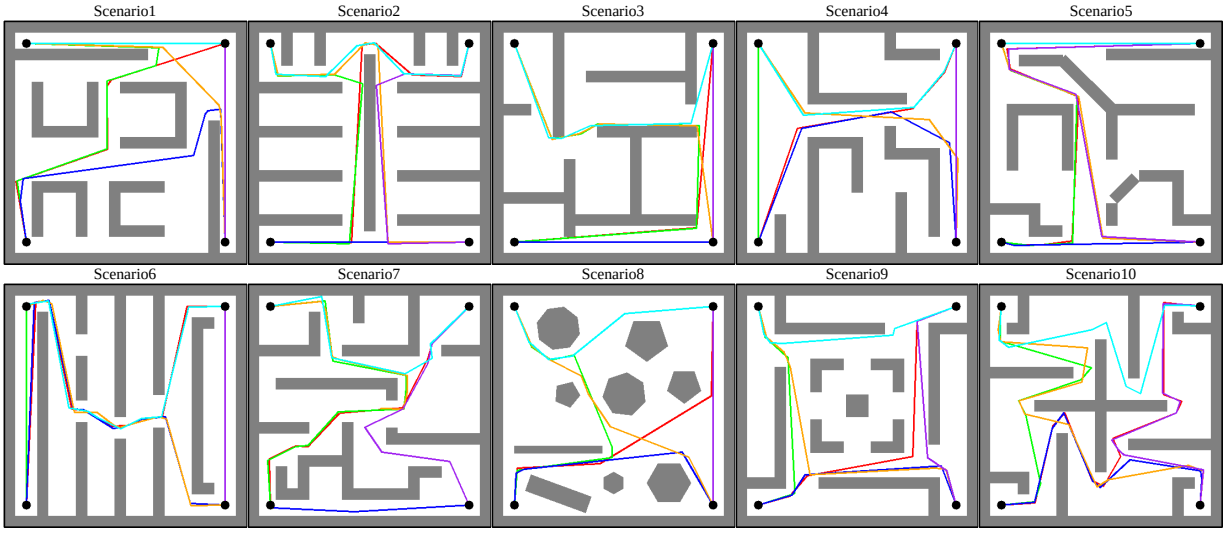


Figure 4.6: Planning results of ACDP-RRT. Different paths are indicated by different colours.

tionally and avoids collision detection when the tree grows within connectors, thus saving additional computational resources.

The memory efficiency is illustrated in the bottom sub-figures of each scenario in **Figure 4.7**. The figures indicate that the number of nodes generated by RRT and RRT* for a global path is similar. In contrast, ACDP-RRT requires only $\frac{1}{5}$ to $\frac{1}{2}$ as many nodes to generate a global path. The improvement in time efficiency is less pronounced than the improvement in memory efficiency because ACDP-RRT involves some pre-processing before it begins searching the map.

4.4.2 Simulation results of the local planner

In this subsection, we evaluate the performance of the proposed ND-AC local planner using a two-wheel differential mobile robot as the test platform. This choice allows for a thorough validation of the ND-AC local planner's effectiveness and applicability in real-world scenarios. The two-wheel differential ground vehicle is selected for its nonholonomic characteristics, which means that its steering capabilities are restricted to adjusting the speed difference between its two wheels, unlike an omnidirectional robot such as a four-wheeled mecanum wheel robot. The nonholonomic nature of the platform makes the local planner's design more challenging, highlighting the advanced capabilities of the ND-AC-based local planner.

***1)* Dynamic Model for Two-Wheeled Differential Robot**

A dynamic model of the two-wheeled robot is required for local planning. As in [118], the dynamic model is given by:

$$\dot{x} = \frac{r}{2}(\omega_L + \omega_R)\cos\varphi, \quad \dot{y} = \frac{r}{2}(\omega_L + \omega_R)\sin\varphi, \quad \dot{\varphi} = \frac{r}{r_b}(\omega_L - \omega_R), \ , \tag{4.7}$$

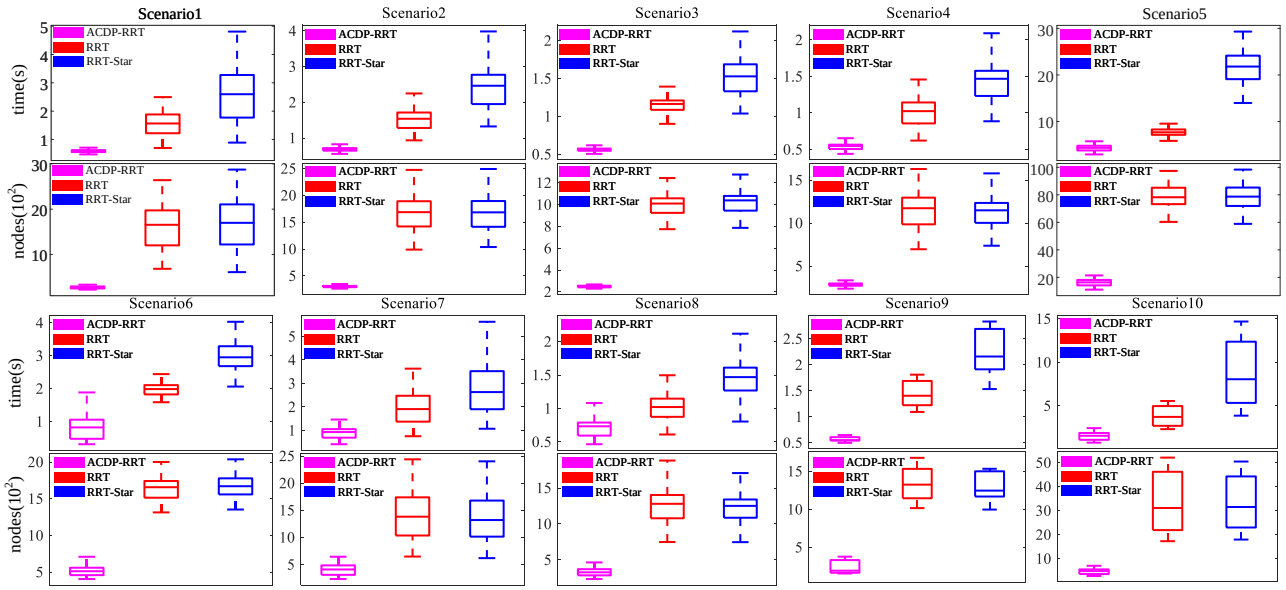


Figure 4.7: Evaluation of the time and memory efficiency for each algorithm from scenario (1) to (10).

where $[x,y]$ is the location, φ is the yaw angle, r is the radius of the wheel, r_b is the radius of the robot base, and $[\omega_L,\omega_R]$ represents the rotation speeds of the left and right wheels, respectively. In the training environment, we limit the maximum linear velocity to V_{max} and the maximum wheel speed to ω_{max} to match the motion capability of the robot. The action space is given by:

$$a = [\omega_L, \omega_R]. \tag{4.8}$$

The state variable is a 45-dimensional vector given by:

$$s = [\tilde{e}_x, \tilde{e}_y, \tilde{x}, \tilde{y}, \varphi, \dot{x}, \dot{y}, \dot{\varphi}] + Lidar(), \tag{4.9}$$

where $\tilde{e}_x = k(t_x - x)/X$, $\tilde{e}_y = k(t_y - y)/Y$, $\tilde{x} = kx/X$, and $\tilde{y} = ky/Y$ are normalized to improve the generalization ability of the learned policy, $[t_x,t_y]$ is the location of the target, X and Y are the dimensions of the map, $[x,y]$ is the location of the robot, k is a static gain, $Lidar()$ is a 37-dimensional vector down-sampled from the raw data of the LiDAR in intervals $[0°,90°]$ and $[270°,360°]$ with a resolution of 5°.

***2)* Learning framework setup**

The training environment for the proposed DRL method is implemented on a computer equipped with an i7-11700 CPU and NVIDIA-RTX 2060 GPU. The software platforms used are Ubuntu 20.04, ROS Noetic, and PyTorch 1.8.1. Figure 4.8 shows an illustration of the learning environment.

The reward function consists of position reward r_1, orientation reward r_2, and sparse reward r_3.

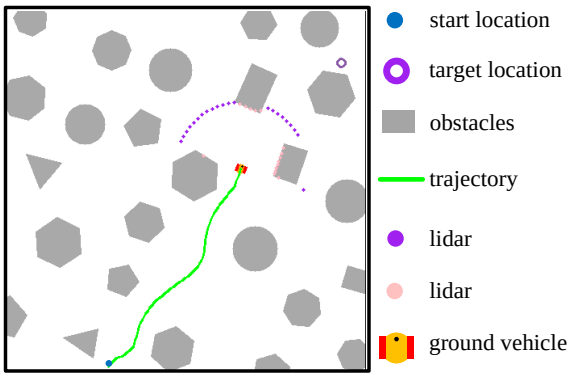


Figure 4.8: An illustration of the learning environment. The purple LiDAR detection points mean that no obstacles are detected at the corresponding angle. The pink LiDAR detection points indicate the obstacles are detected at the corresponding angle.

r_1 is a positive value ($=5$) if the position error decreases; otherwise, it is negative ($=-5$). r_2 is a positive value ($=2$) if the angular error decreases; otherwise, r_2 is a negative value ($=-2$). r_3 is set to be positive ($=10$) if the vehicle reaches the destination successfully; otherwise, r_3 is set to zero. The complete formation of the reward function is the sum of the following three parts.

$$r = r_1 + r_2 + r_3. \tag{4.10}$$

The three parts of the immediate reward function r are given by

$$\begin{cases} r_1 = \begin{cases} 5 & \text{positional error is smaller} \\ -5 & \text{otherwise} \end{cases} \\ r_2 = \begin{cases} 2 & \text{angular error is smaller} \\ -2 & \text{otherwise} \end{cases} \\ r_3 = \begin{cases} 0 & \text{collision} \\ 10 & \text{otherwise} \end{cases} \end{cases} \tag{4.11}$$

Table 4.1: Hyper-parameters of the learning framework.

Parameter	Value	Parameter	Value
γ	0.99	N	60000
up_C	0.01	up_A	0.01
$noise$	0.25	$noise\_clip$	0.5
lr_c	1e-3	lr_a	1e-4
$batch$	512	n_d	5

In each episode, the agent explores different maps generated by a stochastic map generator to

enhance the generalization of the learned policy. The obstacles in the maps are modeled as convex polygons or circles. Table 4.1 lists several hyperparameters of the proposed learning framework. The discount factor is denoted by γ, up_C represents the soft update rate of the critic network, and up_A is the soft update rate of the actor-network. The parameters *noise* and $noise\_clip$ denote the variance and maximum value of the Gaussian distributed noise, respectively. The learning rates of the critic and actor networks are represented by lr_c and lr_a, respectively. The capacity of the replay buffer is denoted by N, $batch$ refers to the number of samples the neural network processes each time, and n_d indicates the number of steps by which the TD3 actor network delays the update.

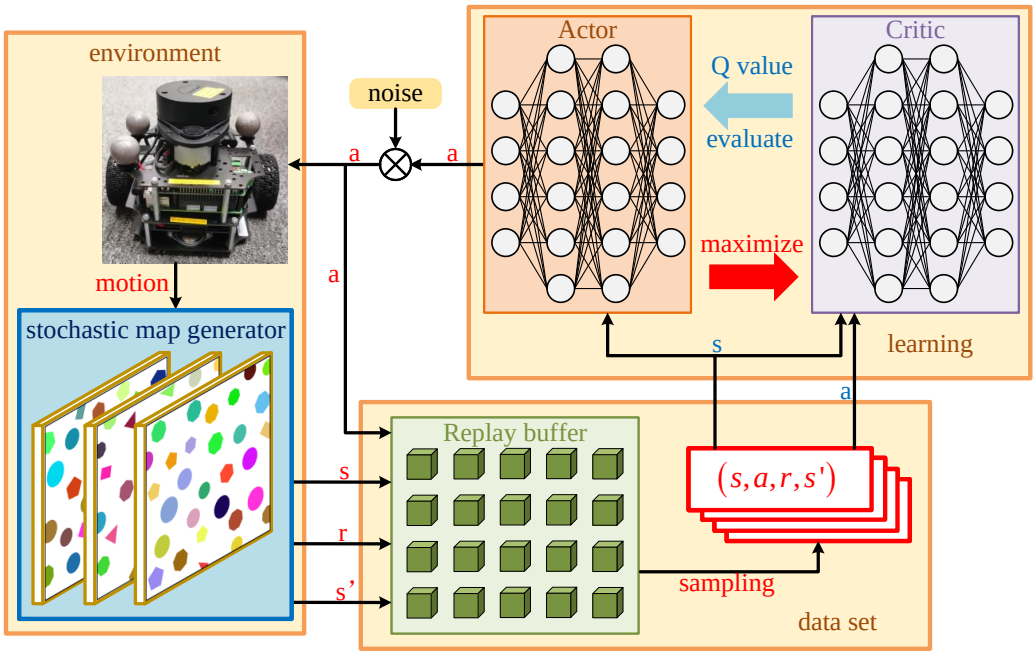


Figure 4.9: The complete DRL-based local planner learning framework.

Figure 4.9 illustrates the complete learning process of the proposed DRL method. The learning framework is composed of three interconnected modules: an environment module, a learning module, and a dataset module. The environment module receives the agent's actions from the learning module and updates the states of the mobile robot accordingly. It then sends a tuple (s,a,r,s') to the dataset. Meanwhile, the learning module updates both the actor and critic networks by sampling batches of tuples from the dataset at each time step. This creates a recursive process among the three modules.

***3)* Policy learning and simulation**

We assess the performance of the proposed ND technology by comparing the success rate of the mobile robot in reaching its destination on various maps. To evaluate the effectiveness of ND technology, we implement two state-of-the-art AC-based DRL methods: TD3 [119] and DDPG [120]. Figure 4.10 illustrates the success rate and immediate reward throughout the training process.

Figure 4.10 shows that integrating the network decoupling mechanism into the training framework

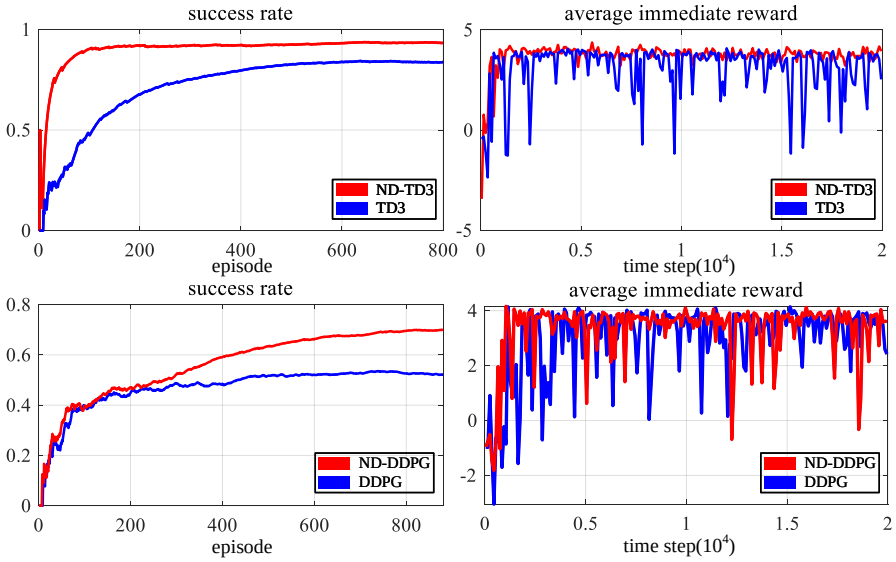


Figure 4.10: Training processes of two AC-based DRL algorithms: TD3 and DDPG. The two on the left are the success rates of the robot in obstacle avoidance. The two on the right are the immediate rewards.

results in a higher success rate. The lower success rate of DDPG compared to TD3 can be attributed to TD3's use of a double network, actor-network delayed update, and target-policy smoothing regularization, which collectively mitigate overestimation of the critic network and improve policy smoothness. The figure for immediate rewards further supports the effectiveness of the ND mechanism, as the rewards for planning algorithms incorporating ND are consistently higher than those for the original algorithms. The fluctuations in immediate rewards are due to the addition of Gaussian-distributed noise to both the hidden layers and output of the actor-network, which ensures thorough exploration of the state space. Consequently, we choose TD3, which demonstrates a higher success rate, as the foundational method for the proposed ND-AC local planner in real-world experiments.

In Figure 4.11, we list four examples of successful obstacle avoidance to give a clear graphical representation of local planning.

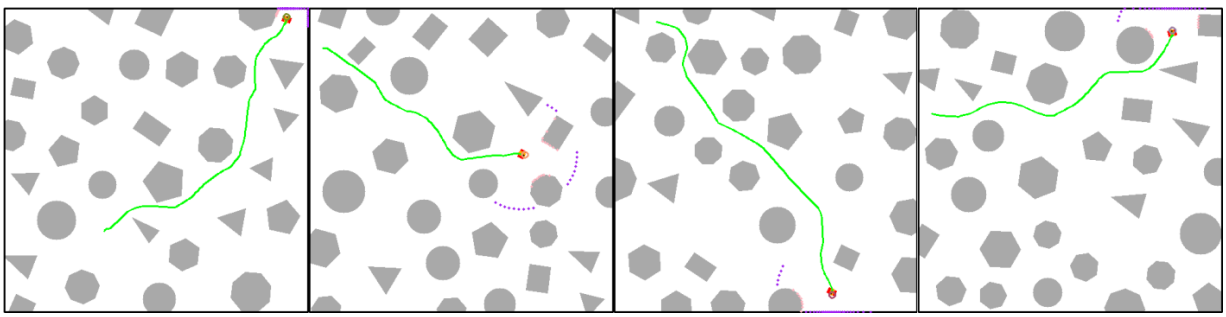


Figure 4.11: A graphical demonstration of local planning.

In addition, extensive Monte Carlo experiments are conducted to validate the conclusions of the proposed ND technology and to demonstrate the capabilities of the well-trained ND-AC local planner. We perform 1000 Monte Carlo experiments on maps containing between 1 and 30 obstacles to evaluate the performance of both AC and ND-AC local planners. The starting positions, target positions, shapes, sizes, and locations of the obstacles for each experiment are stochastically initialized to ensure the robustness of the results. Figure 4.12 illustrates the success rates of the AC and ND-AC local planners across maps with varying numbers of obstacles. Notably, the success rates for the ND-AC and original AC local planners in obstacle-free environments are 0.999 and 0.987, respectively.

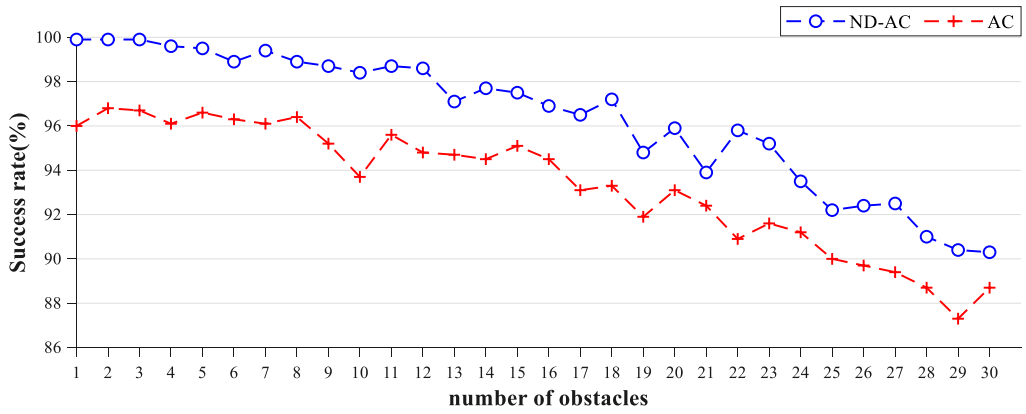


Figure 4.12: Comparative experiments on the success rates of AC local planner and ND-AC local planner in maps with a different number of obstacles.

Figure 4.12 clearly demonstrates that the performance of the ND-AC local planner surpasses that of the original AC local planner. The success rate of the ND-AC local planner remains above 90 % even when 30 obstacles are present on the map. The effectiveness and robustness of the proposed ND-AC local planner are further validated by 30,000 episodes of Monte Carlo simulation experiments.

4.4.3 Simulation and experiments with the integrated planner

A complete path planning framework integrates both global and local planning. In this section, we present simulation and experimental results that combine the proposed global and local planning algorithms. The NanoRobot, as shown in Figure 4.13, is used to test the trained local planner. The NanoRobot is equipped with a Raspberry Pi 4B processor (CPU frequency: 1.5 GHz) running Ubuntu 20.04, ROS Noetic, and PyTorch 1.8.1. Its sensors include a rplidar-SUPER single-wire LiDAR mounted on top of the robot and two encoders connected to the motors. Markers on the robot's body are used for positioning within the VICON motion capture environment. The maximum linear and angular velocities are set to $V_{max} = 0.7\,\mathrm{m/s}$ and $\omega_{max} = 10\,\mathrm{rad/s}$, respectively. The LiDAR detection range is 8 m. In the learning environment, we reduce the LiDAR distance value to the range

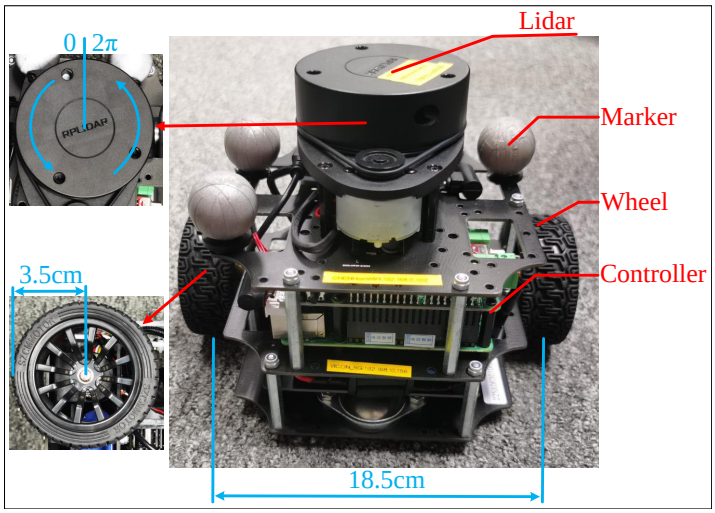


Figure 4.13: NanoRobot platform equipped with a single wire LiDAR.

$[0.15\,\mathrm{m}, 2\,\mathrm{m}]$, with $2\,\mathrm{m}$ being sufficient for obstacle avoidance in both simulation and experiments. Additionally, the minimum detection range is set to $0.15\,\mathrm{m}$ to align with the hardware properties of the LiDAR.

1) Simulation results

Four corners $(1\,\mathrm{m}, 1\,\mathrm{m})$, $(1\,\mathrm{m}, 10\,\mathrm{m})$, $(10\,\mathrm{m}, 10\,\mathrm{m})$, and $(10\,\mathrm{m}, 1\,\mathrm{m})$ are set as the start and target positions, respectively. Ten different groups of simulations are performed for each map. The complete path planning simulation results for different scenarios are shown in Figure 4.14.

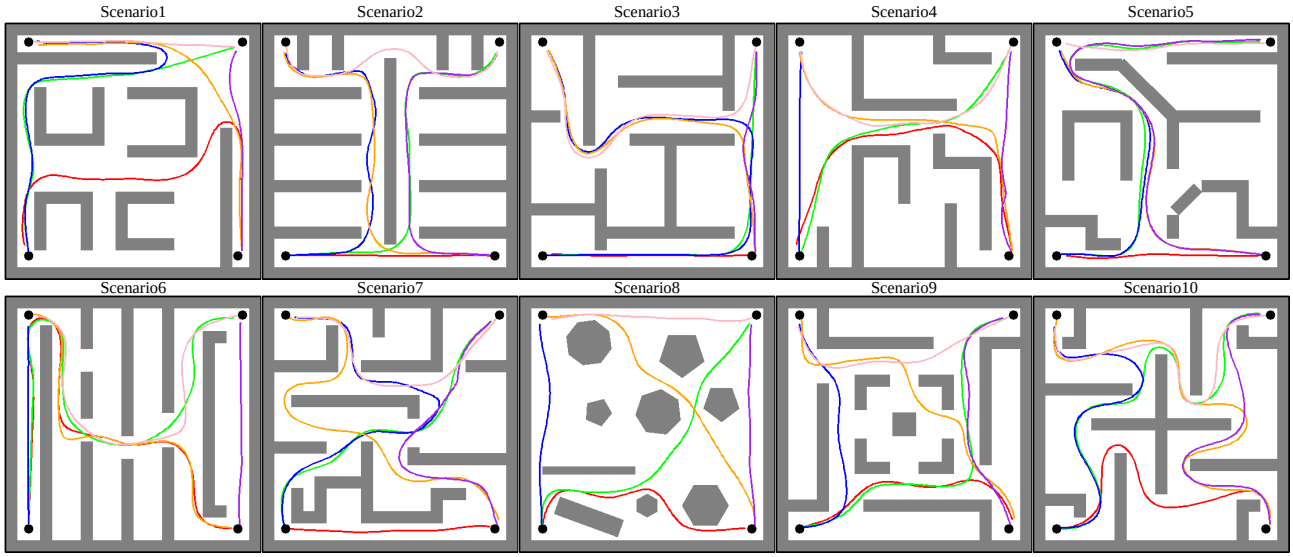


Figure 4.14: Demonstrations of complete path planning simulation experiments. The trajectories between different starting and ending points are highlighted in different colors.

2) Real-world experiment results

Finally, experiments are conducted in real-world scenarios to further validate the performance of the proposed methods. The scale of both the physical and simulation scenarios is $1:2.5$. We use six office scenarios similar to those created in the simulation. Figure 4.15 presents snapshots of the complete path planning results. The first and second rows display the official maps for the simulation and the natural world, respectively. The last row shows the actual trajectories of the mobile robot. The results indicate that the robot follows the waypoints smoothly and successfully avoids obstacles. These demonstrations underscore the effectiveness and robustness of the ACDP-RRT global planner and the ND-AC local planner. Additionally, the runtime of the local planner on the onboard Raspberry Pi is less than 2 ms. The ND-AC local planner, as a compact end-to-end solution, involves only minimal computations during onboard operation, representing a significant advantage over classical motion planning methods. Thus, the DRL-based local planner ensures effective navigation and safety by effectively addressing environmental changes.

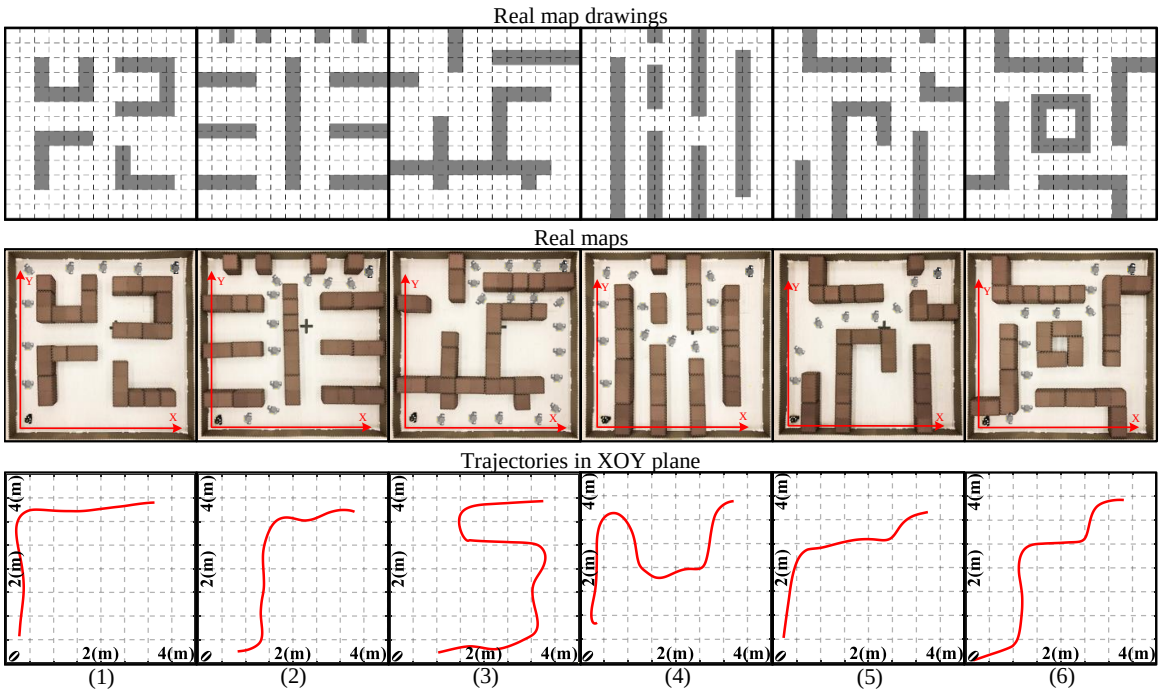


Figure 4.15: Physical experiments of the corresponding six office building scenarios. The dimension of the grids shown in the maps is $0.3m \times 0.3m$. The perimeter is made of plastic plates with a thickness of $1cm$. Opaque rough tapes are attached to the surfaces of the plastic boards to ensure the quality of the LiDAR data. The initial position is the bottom left corner and the target position is the top right corner.

4.5 Conclusion

This chapter presents an innovative planning framework designed to address trajectory planning problems in autonomous robots. For global planning, an ACDP-RRT global planner is introduced, which efficiently generates global waypoints for local planning. Compared to conventional global planning algorithms, the ACDP-RRT approach enhances sampling efficiency and reduces memory costs by intelligently determining the sampling region within the environment. For local planning, an ND-AC-based local planner is employed, which iteratively learns an approximately optimal policy that integrates both the robot's controller and planner. The integration of ND technology not only expedites the training process of the AC framework but also improves the robot's success rate in obstacle avoidance. Real-world experiments validate the robustness and effectiveness of the entire planning framework. In future work, we plan to pursue two main research directions: firstly, investigating path planning methodologies tailored to scenarios with unknown or partially known maps, and secondly, exploring the implementation and adaptation of the proposed path planning algorithms within a multi-agent system context.

Chapter 5

Approximate Optimal Recursive Sliding Mode Control for Quadrotors and Adaptive Parameter Optimization via Deep Reinforcement Learning

5.1 Research Background

In recent decades, UAVs have garnered significant attention in numerous academic research endeavors due to their extensive range of applications, including surveillance [121], forest fire detection [122], and disaster rescue operations [123]. The quadrotor is the most extensively examined and utilized among the various types of UAVs. This is primarily due to its straightforward mechanical structure, cost-effectiveness, and relatively unconstrained mathematical model with aerodynamics.

Ensuring stable control of a quadrotor is fundamental to achieving reliable flight performance and accomplishing complex missions. However, controlling quadrotors presents significant challenges due to their inherent under-actuated nature, nonlinearity, and strong coupling among the system's components [124, 125]. Furthermore, tracking control becomes increasingly complex when dealing with external disturbances, uncertainties, and time-varying parameters within the quadrotor's dynamics. Consequently, achieving robustness and adaptability in the quadrotor's control system is crucial for widespread utilization.

Among the methodologies discussed in Chapter 2, SMC has garnered considerable interest due to its design convenience and robustness. However, the design of the gain for the signum function in SMC significantly impacts system performance. Inappropriate gains can lead to sluggish system response or controller chattering, making the design of a practical sliding mode approach and the mitigation of chattering critical concerns in the field of sliding mode control [126–128].

Nevertheless, two aspects are still present: specific challenges and unresolved issues. First and foremost, a notable issue arises concerning the disturbance observers discussed in the papers above. These observers typically rely on the assumption that the derivative of the disturbance remains bounded and, in some cases, even assume the derivative of the disturbance to be zero. Such assumptions often lead to inadequate estimation by the observer, which impacts the gain of the signum function in SMC.

Secondly, another significant challenge lies in the design of the sliding mode approach law. Finding an appropriate-reaching law proves to be a non-trivial task. If the reaching law is set too high, it can lead to oscillations within the system. Conversely, the system may exhibit sluggish response characteristics if the reaching law is too low. Striking a balance and determining an optimal reaching law thus becomes crucial for achieving the desired system performance in SMC.

ADP (or DRL) is an excellent mathematical tool for function application, making it well-suited for optimizing the hyperparameters of traditional SMCs. Given the challenges prevalent in this field, this chapter makes a fourfold contribution, which can be summarized as follows:

1) An approximate optimal RFNTSMC method is proposed for quadrotors. First, In contrast to methodologies introduced in [129], [130], we improve by removing the requirement of the disturbances being slowly time varient. The proposed FTDOs can accurately estimate the external disturbances in both rotational and translational subsystems, and the estimation errors converge to a set near zero. Second, using the DRL theory, we can learn the approximate optimal control gains of the RFNTSMC by minimizing the discounted cumulative cost function. This approach not only alleviates the burden of manual parameter tuning but also ensures the optimality of the parameters.
2) Dissimilar to methods investigated in [121, 131, 132], the method proposed in this study does not weaken the stability of the system. The learning frameworks introduced in [121, 131, 132] proved to be UUB due to the introduction of neural networks. In contrast, the learning method proposed in this chapter only optimizes the hyper-parameters of the RFNTSMCs rather than directly generating neural network-based controllers. Therefore, the stability of the closed-loop system with a network remains consistent with that of the system without a deep network.
3) Unlike works proposed in [133–135], we have developed a faster, more efficient, and stable training method that isolates the rotational and translational loops during the training process. The implementation proposed in this study simplifies the nonlinear mapping that the neural network needs to approximate, thereby reducing training time and enhancing training stability.

Different sliding mode surfaces are adopted compared to work proposed in [136]. Additionally, the FTDO proposed in this study incorporates two exponential terms designed to enhance the convergence speed for both large and small observation errors.

Notations: In this study, $\mathrm{diag}(\cdot)$ denotes the diagonal matrix. $\circ$ is the Hadamard product operator. $C_{(\cdot)}$, $S_{(\cdot)}$, and $T_{(\cdot)}$ respectively denote $\sin(\cdot)$, $\cos(\cdot)$, and $\tan(\cdot)$ functions. $|x|$ is the absolute value

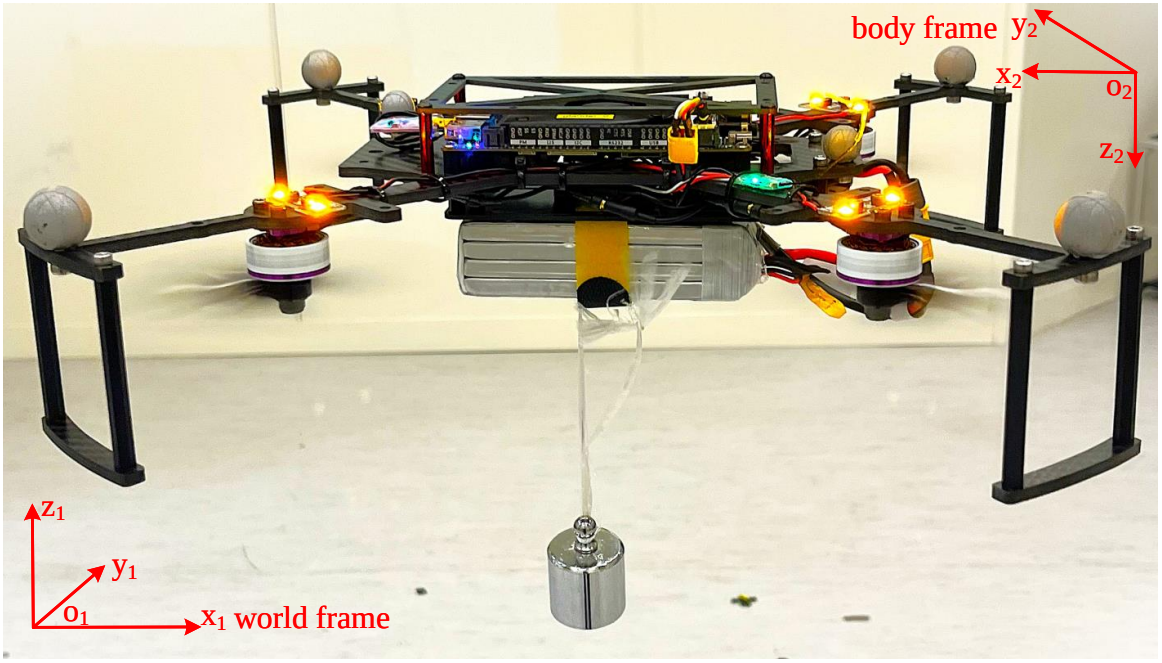


Figure 5.1: Physical configuration of a quadrotor.

function by column vector x elements. $\mathrm{sgn}(\cdot)$ is the signum function. x^a is the power function by elements of column vector x. $\lfloor x \rceil^\alpha = |x|^\alpha \circ \mathrm{sgn}(x)$.

5.2 Problem Formulation and Preliminaries

5.2.1 System Modeling

The physical configuration of a quadrotor is shown in Fig. 5.1. Referring to literature [137], the dynamic model of the quadrotor can be yielded as

$$\begin{aligned} \ddot{\eta} &= \frac{u_f}{m} A(\rho) - \frac{k_t}{m}\dot{\eta} - \mathbf{g} + \frac{d}{m}, \\ \dot{\omega} &= J_0^{-1}\left[-k_r\omega - \omega \times (J\omega) + \Delta + \tau\right], \\ \dot{\rho} &= W(\rho)\omega, \end{aligned} \tag{5.1}$$

where $\eta = [x, y, z]^\top$ is the position of the quadrotor in world-fixed frame, m is the mass, k_t is the translational drag coefficient, u_f is the thrust; $\omega = [\phi, \theta, \psi]^\top$ is the attitude in world-fixed frame, $J_0 = \mathrm{diag}(J_x, J_y, J_z)$ is the inertial tensor matrix, k_r is the rotational drag coefficient, τ is the control torque, and $\mathbf{g} = [0, 0, g]^\top$. Rotation vector $A \triangleq A(\rho)$ and transformation matrix $W \triangleq W(\rho)$ are respectively

defined by

$$A = \begin{bmatrix} C_\varphi C_\psi S_\theta + S_\varphi S_\psi \\ C_\varphi S_\psi S_\theta - S_\varphi C_\psi \\ C_\theta C_\varphi \end{bmatrix}, W = \begin{bmatrix} 1 & S_\varphi T_\theta & C_\varphi T_\theta \\ 0 & C_\varphi & -S_\varphi \\ 0 & S_\varphi / C_\theta & C_\varphi / C_\theta \end{bmatrix},$$

and d and Δ are disturbances acting on translational and rotational loops, respectively.

Similar to [137], we decouple the mathematical model into a rotational and translational subsystem. ***1) Rotational Subsystem*** We denote the reference attitude command as $\rho_d = [\varphi_d, \theta_d, \psi_d]^\top$ and define the tracking error as $e_\rho = \rho - \rho_d$. Using Eq. (5.1), we have

$$\begin{aligned} \dot{e}_\rho &= W\omega - \dot{\rho}_d, \\ \ddot{e}_\rho &= \dot{W}\omega + W\dot{\omega} - \ddot{\rho}_d, \end{aligned} \tag{5.2}$$

where

$$\dot{W} = \begin{bmatrix} 0 & \dot{\varphi} T_\theta C_\varphi + \frac{\dot{\theta} S_\varphi}{C_\theta^2} & -\dot{\varphi} S_\varphi T_\theta + \frac{\dot{\theta} C_\varphi}{C_\theta^2} \\ 0 & -\dot{\varphi} S_\varphi & -\dot{\varphi} C_\varphi \\ 0 & \frac{\dot{\varphi} C_\varphi C_\theta + \dot{\theta} S_\varphi S_\theta}{C_\theta^2} & \frac{-\dot{\varphi} S_\varphi C_\theta + \dot{\theta} C_\varphi S_\theta}{C_\theta^2} \end{bmatrix}.$$

Further, define the equivalent disturbance of inner-loop subsystem as $\Delta_\rho = J_0^{-1}\Delta - \ddot{\rho}_d$. Substituting Δ_ρ into (5.2) gives

$$\ddot{e}_\rho = \dot{W}\omega + W f_\rho + W J_0^{-1}\tau + \Delta_\rho, \tag{5.3}$$

where $f_\rho = -J_0^{-1}[k_r\omega + \omega \times (J\omega)]$. Eq. (5.3) can be given as

$$\ddot{e}_\rho = A_\rho + B_\rho \tau + \Delta_\rho, \tag{5.4}$$

where $A_\rho = \dot{W}\omega + W f_\rho \in \mathbb{R}^{3\times 1}$ and $B_\rho = W J_0^{-1} \in \mathbb{R}^{3\times 3}$.

Remark 5.1 *Note that the second-order derivative of ρ_d is known for pure attitude control. However, for quadrotor position control, the expected attitude commands are generated by the translational subsystem. Therefore, $\ddot{\rho}_d$ is covered into Δ_ρ and regarded as part of the unknown disturbances.*

2) Translational Subsystem The translational subsystem generates the quadrotor's thrust and expected pitch and roll angles. However, a so-called "virtual control input" is required due to the underactuated characteristic of the quadrotor. Specifically, the error dynamic of the translational subsystem is given by

$$\ddot{e}_\eta = -\frac{k_t}{m}\dot{\eta} + u_\eta - \ddot{\eta}_d + \Delta_\eta, \tag{5.5}$$

where $u_\eta = [a_x, a_y, a_z]^\top$ is the virtual expected linear acceleration of the quadrotor that is generated by the FNTSMC controller introduced in a later section, $e_\eta = \eta - \eta_d$ is the position tracking error, and

$$\Delta_\eta = \frac{u_f}{m}\begin{bmatrix} C_\phi C_\psi S_\theta + S_\phi S_\psi \\ C_\phi S_\psi S_\theta - S_\phi C_\psi \\ C_\theta C_\varphi \end{bmatrix} + \frac{d}{m} - \mathbf{g} - u_\eta.$$

Furthermore, the translational and rotational subsystems are connected by the following nonlinear mapping:

$$\begin{aligned} u_f &= m\sqrt{a_x^2 + a_y^2 + (a_z + g)^2}, \\ \varphi_d &= \arcsin\frac{m\left(a_x S_\psi - a_y C_\psi\right)}{u_f}, \\ \theta_d &= \arctan\frac{a_x C_\psi + a_y S_\psi}{a_z + g}. \end{aligned} \tag{5.6}$$

We make the following assumption to our system model:

Assumption 5.1 *[138] The disturbances Δ_ρ and Δ_η and their derivatives are all bounded by some unknown positive constants, namely, $||\dot{\Delta}_\rho|| \leq \overline{\Delta}_\rho$ and $||\dot{\Delta}_\eta|| \leq \overline{\Delta}_\eta$ with $\overline{\Delta}_\rho, \overline{\Delta}_\eta > 0$. The yaw angle is bounded as $-\pi \leq \psi \leq \pi$. To avoid singularities, the roll and pitch angles are bounded as $-\frac{\pi}{2} < \varphi < \frac{\pi}{2}$ and $-\frac{\pi}{2} < \theta < \frac{\pi}{2}$, respectively.*

Remark 5.2 *In real-world experimental scenarios, the attitude angle ranges are automatically limited by the quadrotor's Flight Control Unit (FCU) to satisfy the conditions required in Assumption 5.1.*

Based on the analysis and summation aforementioned, the problem we consider is described as follows:

Control Objective: Given quadrotor system (5.1) and a set of reference trajectories $\{x_d, y_d, z_d, \psi_d\}$, where $\{x_d, y_d, z_d\}$ is the reference position and ψ_d is the reference yaw angle, design a type of ADP-based robust sliding-mode control law such that

$$\lim_{t\to T_\eta}(x(t) - x_d(t)) = 0, \ \lim_{t\to T_\eta}(y(t) - y_d(t)) = 0, \tag{5.7a}$$

$$\lim_{t\to T_\eta}(z(t) - z_d(t)) = 0, \ \lim_{t\to T_\rho}(\psi(t) - \psi_d(t)) = 0, \tag{5.7b}$$

with T_η and T_ρ being the settling time.

5.2.2 Preliminaries

In this section, we introduce several definitions and lemmas that form the theoretical foundation of our approach.

Definition 5.1 *[139] For system $\dot{x} = f(x), f(0) = 0, x \in \mathbb{R}^n$ is the system state. The system globally and uniformly converges to a neighborhood Ω of the origin in a finite time if there exists a locally bounded function $T_0 : \mathbb{R}^n \to \mathbb{R}_+ \cup \{0\}$ such that $x \in \Omega$ when $t \geq T_0(x_0)$. The system is considered globally uniformly finite-time convergent to Ω if function T_0 is globally bounded by some positive number T_m. T_0 is called the settling-time function.*

Definition 5.2 *[139] Let $r = [r_1, r_2, \cdots, r_n], r_i > 0$ be a generalised weight vector. The dilation associated with the generalised weight r is $\Lambda_r(\lambda) = \mathrm{diag}(\lambda^{r_1}, \lambda^{r_2}, \cdots, \lambda^{r_n})$ for $\lambda > 0$. A function field $f : \mathbb{R}^n \to \mathbb{R}^n$ (or a function $f : \mathbb{R}^n \to \mathbb{R}$) is r-homogeneous with a degree of d if and only if $\forall x \in \mathbb{R}^n$ and $\forall \lambda > 0$, $\lambda^{-d}\Lambda_r^{-1}(\lambda) f(\Lambda_r(\lambda)x) = f(x)$ holds if f is a function field or $\lambda^{-d} f(\Lambda_r(\lambda)x) = f(x)$ holds if f is a function.*

Definition 5.3 *[139] A function field $f : \mathbb{R}^n \to \mathbb{R}^n$ (or function $f : \mathbb{R}^n \to \mathbb{R}$) is said to be homogeneous in the p-limit $(p = 0, \infty)$ with an associated tuple $(\overline{r}, \overline{d}, \overline{f})$ if*

$$\lim_{\lambda \to p} \sup_{x \in \mathbb{R}^n \setminus \{0\}} \left\| \lambda^{-\overline{d}} \Lambda_{\overline{r}}^{-1}(\lambda) f(\Lambda_{\overline{r}}(\lambda)x) - \overline{f}(x) \right\| = 0,$$

where $\overline{r}$ is a generalised weight vector, $\overline{d} \in \mathbb{R}$, and $\overline{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a function field or

$$\lim_{\lambda \to p} \sup_{x \in \mathbb{R}^n \setminus \{0\}} \left\| \lambda^{-\overline{d}} f(\Lambda_{\overline{r}}(\lambda)x) - \overline{f}(x) \right\| = 0$$

if f and $\overline{f} : \mathbb{R}^n \to \mathbb{R}$ are functions.

Definition 5.4 *[139] A function field $f : \mathbb{R}^n \to \mathbb{R}^n$ is said to be homogeneous in the bi-limit if it is homogeneous in the 0-limit and ∞-limit simultaneously.*

Lemma 5.1 *[140] For system $\dot{x} = f(x), f(0) = 0, x \in \mathbb{R}^n$. The system is practically fixed-time stable if there exists a Lyapunov function V for the system, such that*

$$\dot{V} \leq -\left(a_1 V^{a_2} + a_3 V^{a_4}\right)^k + \delta, \tag{5.8}$$

where $a_1, a_3 > 0$, $a_2 k < 1 < a_4 k$, $\delta > 0$. Specifically, the settling time is given by

$$T_m \leq \frac{1}{a_1^k \vartheta^k (1 - a_2 k)} + \frac{1}{a_3^k \vartheta^k (a_4 k - 1)},$$

and the residual set is given by

$$\left\{x : V \leq \min\left\{a_1^{-\frac{1}{a_2}}\left(\frac{\delta}{1-\vartheta^k}\right)^{\frac{1}{a_2 k}}, a_3^{-\frac{1}{a_4}}\left(\frac{\delta}{1-\vartheta^k}\right)^{\frac{1}{a_4 k}}\right\}\right\}$$

with $t \to T_m$*, where* $\vartheta \in (0,1)$

Lemma 5.2 *[140] For system* $\dot{x} = f(x), f(0) = 0, x \in \mathbb{R}^n$*. If there exists a Lyapunov function* $V(x)$ *such that* $\dot{V}(x) \leq -aV^b(x), 0 < b < 1, a > 0$*, then the origin of the system is finite-time stable. The adjusting time of* $V(x)$ *is* $T_a \leq \frac{V^{1-b}(0)}{a(1-b)}$*.*

Lemma 5.3 *[141] For* $\forall x_i \in \mathbb{R}$*,* $i = 1, 2, \cdots, n$*,* $0 < p \leq 1$*, there is*

$$\left(\sum_{i=1}^{n} |x_i|\right)^p \leq \sum_{i=1}^{n} |x_i|^p \leq n^{1-p}\left(\sum_{i=1}^{n} |x_i|\right)^p.$$

Lemma 5.4 *For scalar* $a, b \in \mathbb{R}_+$ *and a function* $\varkappa(a,b) = \frac{a(1+b)}{1+a}$*. Then, there exists* $\kappa_0 > 0$ *such that* $\varkappa(a,b) = \kappa_0 a + b$ *if* $a > b$ *and* $\varkappa(a,b) = a + \kappa_0 b$ *if* $a < b$ *hold.*

Proof 5.1 *Solving* $\varkappa(a,b) = \kappa_0 a + b$ *yields* $\kappa_0 = \frac{a-b}{a+a^2}$*, which requires the condition* $a > b$*. Similarly, solving* $\varkappa(a,b) = a + \kappa_0 b$ *yields* $\kappa_0 = \frac{a(b-a)}{ab+b}$*, which requires the condition* $a < b$*. This completes the proof.*

5.3 System Design

5.3.1 Rotational subsystem stability

To accurately estimate external disturbances, an FTDO is designed as

$$\begin{aligned}
\dot{z}_{\rho 1} &= \zeta_\rho m_{\rho 1}\left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\alpha_1} + (1-\zeta_\rho) n_{\rho 1}\left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\beta_1} + z_{\rho 2}, \\
\dot{z}_{\rho 2} &= \zeta_\rho m_{\rho 2}\left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\alpha_2} + (1-\zeta_\rho) n_{\rho 2}\left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\beta_2} + z_{\rho 3} + A_\rho + B_\rho \tau, \\
\dot{z}_{\rho 3} &= \zeta_\rho m_{\rho 3}\left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\alpha_3} + (1-\zeta_\rho) n_{\rho 3}\left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\beta_3},
\end{aligned} \tag{5.9}$$

where $z_{\rho 1}$, $z_{\rho 2}$, and $z_{\rho 3}$ are the estimates of e_ρ, $\dot{e}_\rho$, and Δ_ρ, respectively, $\tilde{e}_{\rho 1} = e_\rho - z_{\rho 1}$ is the estimation error of e_ρ; α_1, α_2, α_3, β_1, β_2, β_3 are all positive constants; ζ_ρ is a switching parameter; $k_0 > 0$ is a scale factor; $m_{\rho i}$ and $n_{\rho i}$, $i = 1, 2, 3$ are some positive constants that need to be designed.

To guarantee the stability of the FTDO, we let $\alpha_1 = \frac{3}{4}$, $\alpha_2 = \frac{2}{4}$, $\alpha_3 = \frac{1}{4}$, $\beta_1 = \frac{5}{4}$, $\beta_2 = \frac{6}{4}$, $\beta_3 = \frac{7}{4}$,

$$\zeta_\rho = \begin{cases} 0 & \text{if } ||\tilde{e}_{\rho 1}||_2 > e_\rho^*, \\ 1 & \text{if } ||\tilde{e}_{\rho 1}||_2 \leq e_\rho^*, \end{cases}$$

with e_ρ^* being the threshold of the estimation error of e_ρ, and for $i = 1,2,3$, $m_{\eta,i}$, $n_{\eta,i}$ are such that matrices

$$\Gamma_{\rho 1} = \begin{bmatrix} -m_{\rho 1} & 1 & 0 \\ -m_{\rho 2} & 0 & 1 \\ -m_{\rho 3} & 0 & 0 \end{bmatrix}, \Gamma_{\rho 2} = \begin{bmatrix} -n_{\rho 1} & 1 & 0 \\ -n_{\rho 2} & 0 & 1 \\ -n_{\rho 3} & 0 & 0 \end{bmatrix}$$

are Hurwitz.

Using (5.9), it is easy to verify that the estimation errors $\tilde{e}_{\rho 1} = e_\rho - z_{\rho 1}$, $\tilde{e}_{\rho 2} = \dot{e}_\rho - z_{\rho 2}$, and $\tilde{e}_{\rho 3} = \Delta_\rho - z_{\rho 3}$ are generated by

$$\begin{aligned} \dot{\tilde{e}}_{\rho 1} &= -\zeta_\rho m_{\rho 1} \left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\alpha_1} - (1-\zeta_\rho) n_{\rho 1} \left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\beta_1} + \tilde{e}_{\rho 2}, \\ \dot{\tilde{e}}_{\rho 2} &= -\zeta_\rho m_{\rho 2} \left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\alpha_2} - (1-\zeta_\rho) n_{\rho 2} \left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\beta_2} + \tilde{e}_{\rho 3}, \\ \dot{\tilde{e}}_{\rho 3} &= -\zeta_\rho m_{\rho 3} \left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\alpha_3} - (1-\zeta_\rho) n_{\rho 3} \left\lfloor \frac{\tilde{e}_{\rho 1}}{k_0} \right\rceil^{\beta_3} + \dot{\Delta}_\rho. \end{aligned} \tag{5.10}$$

Further, for $i = 1,2,3$, we define

$$\xi_i = \frac{\tilde{e}_{\rho i}}{k_0}, \Upsilon_{mi} = m_i' \lfloor \xi_1 \rceil^{\alpha_i}, \Upsilon_{ni} = n_i' \lfloor \xi_1 \rceil^{\beta_i}.$$

Then, system (5.10) can be transformed into

$$\begin{aligned} \dot{\xi}_1 &= -\zeta_\rho \Upsilon_{m1} - (1-\zeta_\rho)\Upsilon_{n1} + \xi_2, \\ \dot{\xi}_2 &= -\zeta_\rho \Upsilon_{m2} - (1-\zeta_\rho)\Upsilon_{n2} + \xi_3, \\ \dot{\xi}_3 &= -\zeta_\rho \Upsilon_{m3} - (1-\zeta_\rho)\Upsilon_{n3} + \dot{\Delta}_\rho, \end{aligned} \tag{5.11}$$

where $m_i' = m_{\rho i}/k_0$ and $n_i' = n_{\rho i}/k_0$ with $i = 1,2,3$.

The following theorem summarizes the effectiveness of our FTDO:

Theorem 5.1 *Under Assumption 5.1, for the FTDO (5.9), the estimation error $\tilde{\Delta}_\rho = \Delta_\rho - z_{\rho 3}$ converges to a neighborhood of the origin $\mathscr{N}_\rho$ within a fixed time $\mathscr{T}_{\rho 1}$, and the boundary of $\mathscr{N}_\rho$ can be denoted as $N_\rho = \max\{||x||_2 : x \in \mathscr{N}_\rho\}$.*

Proof 5.2 *The proof process consists of two steps. First, we show the existence of a Lyapunov function*

that can make system (5.11) globally asymptotically stable when $\dot{\Delta}_\rho = 0$*. Then, based on step* 1*, we further prove that the entire error system is still fixed-time convergent to a neighborhood of the origin when* $\dot{\Delta}_\rho \neq 0$*.*

Step 1*: To begin with, assuming* $\dot{\Delta}_\rho = 0$*, we construct three vector fields*

$$f(\xi) = \begin{bmatrix} -\zeta_\rho \Upsilon_{m1} - (1-\zeta_\rho)\Upsilon_{n1} + \xi_2 \\ -\zeta_\rho \Upsilon_{m2} - (1-\zeta_\rho)\Upsilon_{n2} + \xi_3 \\ -\zeta_\rho \Upsilon_{m3} - (1-\zeta_\rho)\Upsilon_{n3} \end{bmatrix}, \overline{f}(\xi) = \begin{bmatrix} -\Upsilon_{m1} + \xi_2 \\ -\Upsilon_{m2} + \xi_3 \\ -\Upsilon_{m3} \end{bmatrix}, \underline{f}(\xi) = \begin{bmatrix} -\Upsilon_{n1} + \xi_2 \\ -\Upsilon_{n2} + \xi_3 \\ -\Upsilon_{n3} \end{bmatrix}.$$

Then, Eq. (5.11) can be simplified to

$$\dot{\xi} = \begin{cases} \overline{f}(\xi) & ||\tilde{e}_\rho||_2 > e_\rho^* \\ \underline{f}(\xi) & ||\tilde{e}_\rho||_2 \leq e_\rho^* \end{cases}. \tag{5.12}$$

By Theorem 10 in [142], system (5.12) is globally asymptotically stable for both $||\tilde{e}_\rho||_2 > e_\rho^*$ *and* $||\tilde{e}_\rho||_2 \leq e_\rho^*$ *without considering* $\dot{\Delta}_\rho$*. This completes the proof of Step 1.*

Step 2: We define $\overline{r} = [1, \alpha, 2\alpha - 1]^\top$ *and* $\overline{d} = \alpha - 1$ *with* $\alpha = \frac{3}{4}$*. Then,* $\forall \lambda > 0$*, we have*

$$\begin{aligned} &\lambda^{-\overline{d}} \Lambda_{\overline{r}}^{-1}(\lambda) \overline{f}(\Lambda_{\overline{r}}(\lambda)\xi) \\ =&\lambda^{-\overline{d}} \cdot \mathrm{diag}(\lambda^{-1}, \lambda^{-\alpha}, \lambda^{-2\alpha+1}) \overline{f}(\Lambda_{\overline{r}}(\lambda)\xi) \\ =&\mathrm{diag}(\lambda^{-\alpha}, \lambda^{1-2\alpha}, \lambda^{2-3\alpha}) \overline{f}([\lambda\xi_1, \lambda^{\alpha}\xi_2, \lambda^{2\alpha-1}\xi_3]^\top) \\ =&\begin{bmatrix} -m_1' \lfloor \xi_1 \rceil^{\alpha} + \xi_2 \\ -m_2' \lfloor \xi_1 \rceil^{2\alpha-1} + \xi_3 \\ -m_3' \lfloor \xi_1 \rceil^{3\alpha-2} \end{bmatrix} = \overline{f}(\xi). \end{aligned} \tag{5.13}$$

According to Definitions 5.2 and 5.3, we know that $\overline{f}$ *is* $\overline{r}$*-homogeneous with a degree of* $\overline{d}$*. Further, using the result in (5.13) yields*

$$\lim_{\lambda \to 0} \sup_{\xi \in \mathbb{R}^3 \setminus \{0\}} ||\lambda^{-\overline{d}} \Lambda_{\overline{r}}^{-1}(\lambda) f(\Lambda_{\overline{r}}(\lambda)\xi) - \overline{f}(\xi)|| = 0, \tag{5.14}$$

which indicates that $f(\xi)$ *is homogeneous in the* 0*-limit with tuple* $(\overline{r}, \overline{d}, \overline{f})$*. By following very similar derivations, we can also conclude that* f *is homogeneous in the* ∞*-limit with tuple* $(\underline{r}, \underline{d}, \underline{f})$*,* $\underline{r} = [1, \beta, 2\beta - 1]^\top$*,* $\underline{d} = \beta - 1$*, and* $\beta = \frac{5}{4}$*.*

By Definition 5.4, one has f *is in the bi-limit of tuples* $(\overline{r}, \overline{d}, \overline{f})$ *and* $(\underline{r}, \underline{d}, \underline{f})$*. Based on step 1, we define two positive constants* $m_0 \geq \max\{1, \alpha, 2\alpha - 1\}$ *and* $m_\infty \geq \max\{1, \beta, 2\beta - 1\}$*. Then, inspired by Theorem 2.20 of [143] and Corollary 2.21 of [143], Lemma 5.5 holds.*

Lemma 5.5 *There exists a Lyapunov function $V(\xi)$ for system $\dot{\xi} = f(\xi)$ disturbed by $|\dot{\Delta}_\rho|$ such that*

$$\dot{V}(\xi) \leq -c_v \varkappa(V^{\frac{m_0+d_0}{m_0}}, V^{\frac{m_\infty+d_\infty}{m_\infty}}) + c_\Delta \varkappa(|\dot{\Delta}_\rho|^{\frac{m_0+d_0}{m_0}}, |\dot{\Delta}_\rho|^{\frac{m_\infty+d_\infty}{m_\infty}}) \tag{5.15}$$

with c_v and c_Δ being some positive constants.

Combine Lemma 5.4 and Lemma 5.5 with $\delta = c_\Delta \varkappa\,(|\dot{\Delta}_\rho|^{\frac{m_0+d_0}{m_0}}, |\dot{\Delta}_\rho|^{\frac{m_\infty+d_\infty}{m_\infty}})$, one has

$$\dot{V} \leq \begin{cases} -c_v \kappa_0 V^{\frac{4}{3}} + c_v V^{\frac{7}{6}} + \delta, & 0 \leq V \leq 1, \\ -c_v V^{\frac{4}{3}} + c_v \kappa_0 V^{\frac{7}{6}} + \delta, & V > 1. \end{cases} \tag{5.16}$$

It is easy to verify that each case in (5.16) satisfies the form shown in Lemma 5.1. Therefore, the observer is practically fixed-time stable; the estimation error converges to a neighborhood of the origin in a fixed time. The settling time can be given by:

$$T_m \leq \max\left(\frac{4}{c_v \kappa_0 \vartheta} + \frac{7}{c_v \vartheta}, \frac{4}{c_v \vartheta} + \frac{7}{c_v \kappa_0 \vartheta}\right), \vartheta \in (0,1]. \tag{5.17}$$

The residual set is given by

$$\Omega = \{x : V(x) \leq \min\{\Omega_1, \Omega_2\}\}, \tag{5.18}$$

where

$$\begin{aligned} \Omega_1 &= \min\left\{(c_v \kappa_0)^{-\frac{4}{3}} \left(\frac{\delta}{1-\vartheta}\right)^{\frac{4}{3}}, c_v^{-\frac{6}{7}} \left(\frac{\delta}{1-\vartheta}\right)^{\frac{6}{7}}\right\}, \\ \Omega_2 &= \min\left\{c_v^{-\frac{4}{3}} \left(\frac{\delta}{1-\vartheta}\right)^{\frac{4}{3}}, (c_v \kappa_0)^{-\frac{6}{7}} \left(\frac{\delta}{1-\vartheta}\right)^{\frac{6}{7}}\right\}, \\ \delta &= c_\Delta \varkappa\left(|\dot{\Delta}_\rho|^{\frac{m_0+d_0}{m_0}}, |\dot{\Delta}_\rho|^{\frac{m_\infty+d_\infty}{m_\infty}}\right). \end{aligned}$$

The proof is completed.

Next, we design an integral-type sliding mode surface σ_ρ as

$$\begin{aligned} s_\rho &= \dot{e}_\rho + k_{\rho 1} e_\rho + \gamma_\rho \lfloor e_\rho \rceil^{\alpha_\rho}, \\ \sigma_\rho &= s_\rho + \int_0^t \lambda_\rho \lfloor s_\rho \rceil^{\beta_\rho} dt, \end{aligned} \tag{5.19}$$

where $k_{\rho 1} > 0$, $\gamma_\rho > 0$, $\lambda_\rho > 0$, $\alpha_\rho > 1$, and $0 < \beta_\rho < 1$ are positive scalars.

Remark 5.3 *As we will see in the proof of Theorem 5.2, $s_\rho = 0$ can already guarantee the convergence of e_ρ. However, to enable s_ρ to reach the sliding surface more smoothly, we further introduce another sliding surface $\sigma_\rho = 0$.*

Assuming there are no external disturbances, the equivalent control law is given by

$$\tau_{eq} = -B_\rho^{-1}\left(A_\rho + k_{\rho 1}\dot{e}_\rho + \gamma_\rho \alpha_\rho \left\lfloor e_\rho \right\rceil^{\alpha_\rho - 1} \circ \dot{e}_\rho + \lambda_\rho s_{\rho 1} + \dot{k}_{\rho 1} e_\rho + \dot{\gamma}_\rho \left\lfloor e_\rho \right\rceil^{\alpha_\rho}\right). \tag{5.20}$$

Thereafter, considering the existence of the external disturbances, a compensation term is further required to maintain σ_ρ on the sliding mode surface, which is designed as

$$\tau_{sw} = -B_\rho^{-1}[z_{\rho 3} + k_{\rho 2}\,\mathrm{sgn}(\sigma_\rho)], \tag{5.21}$$

where $k_{\rho 2} > 0$. The complete control law is thus given by

$$\tau = \tau_{eq} + \tau_{sw}. \tag{5.22}$$

Remark 5.4 *Note that parameters $k_{\rho 1}$, $k_{\rho 2}$, γ_ρ, and λ_ρ designed in Eqs. (5.20) and (5.21) are optimized by the ADP theory proposed in later sections. Therefore, $\dot{k}_{\rho 1}$ and $\dot{\gamma}_\rho$ cannot be ignored in the stability analysis.*

Then, we can establish the following theorem.

Theorem 5.2 *Under Assumption 5.1, the closed loop system of the rotational subsystem of the quadrotor (5.4) under the FTDO (5.9) and the FNTSMC (5.22) is finite-time stable if the control gain satisfies $k_{\rho 2} > N_\rho$.*

Proof 5.3 *To begin with, we show the finite-time convergence of σ_ρ. Select the Lyapunov function candidate as $V_{\rho 1} = \frac{1}{2}\sigma_\rho^\top \sigma_\rho$. Let $\kappa_\rho = k_{\rho 2} - N_\rho$. Differentiating $V_{\rho 1}$ gives*

$$\begin{aligned}
\dot{V}_{\rho 1} &= \sigma_\rho^\top \left(\ddot{e}_\rho + k_{\rho 1}\dot{e}_\rho + \gamma_\rho \left|e_\rho\right|^{\alpha_\rho - 1} \circ \dot{e}_\rho + \lambda_\rho s_{\rho 1} + \dot{k}_{\rho 1} d_\rho + \dot{\gamma}_\rho \left\lfloor e_\rho \right\rceil^{\alpha_\rho}\right) \\
&= \sigma_\rho^\top \left[\tilde{\Delta}_\rho - k_{\rho 2}\,\mathrm{sgn}(\sigma_\rho)\right] \\
&\leq -\kappa_\rho \left\|\sigma_\rho\right\|_2 \\
&= -\sqrt{2}\kappa_\rho V_{\rho 1}^{\frac{1}{2}} \leq 0.
\end{aligned} \tag{5.23}$$

By Lemma 5.2, σ_ρ converges to zero in a finite-time $\mathscr{T}_{\rho 2}$, which is bounded by $\mathscr{T}_{\rho 2} \leq \frac{\sqrt{2V_\rho(0)}}{\kappa_\rho}$.

Then, we show that the error $e_\rho(t)$ converges to the origin in a finite time on the sliding surface σ_ρ.

In fact, on the sliding surface σ_ρ, one has

$$s_\rho = -\int_0^t \lambda_\rho \left\lfloor s_\rho \right\rceil^{\beta_\rho} dt,\ \dot{s}_\rho = -\lambda_\rho \left\lfloor s_\rho \right\rceil^{\beta_\rho}. \tag{5.24}$$

Choose a Lyapunov function candidate for s_ρ as $V_{\rho 2} = \frac{1}{2} s_\rho^\top s_\rho$. By following the same derivation process as in (5.23), one has

$$\dot{V}_{\rho 2} = -\lambda_\rho \left\| s_\rho^{\beta_\rho + 1} \right\|_1 \leq -2^{\frac{\beta_\rho + 1}{2}} \lambda_\rho V_{\rho 2}^{\frac{\beta_\rho + 1}{2}}. \tag{5.25}$$

Invoking Lemma 5.2 concludes that s_ρ converges to zero within a finite-time $\mathscr{T}_{\rho 3}$, which is bounded as

$$\mathscr{T}_{\rho 3} \leq \frac{V_{\rho 2}(0)^{\frac{1-\beta_\rho}{2}}}{\lambda_\rho 2^{\frac{\beta_\rho + 1}{2}} \frac{1-\beta_\rho}{2}} = \frac{\left\| s_\rho \right\|^{1-\beta_\rho}}{\lambda_\rho \left(1 - \beta_\rho\right)}. \tag{5.26}$$

Thereafter, $s_\rho = 0$ implies $\dot{e}_\rho = -k_{\rho 1} e_\rho - \gamma_\rho \left\lfloor e_\rho \right\rceil^{\alpha_\rho}$. Now, consider a Lyapunov function candidate $V_{\rho 3} = \frac{1}{2} e_\rho^\top e_\rho$. Differentiating $V_{\rho 3}$ and using Lemma 5.3 yield

$$\begin{aligned}
\dot{V}_{\rho 3} &= e_\rho^\top \left(-k_{\rho 1} e_\rho - \gamma_\rho \cdot \left\lfloor e_\rho \right\rceil^{\alpha_\rho} \right) \\
&= -k_{\rho 1} \left\| e_\rho \right\|_2^2 - \gamma_\rho \left\| e_\rho^{1+\alpha_\rho} \right\|_1 \\
&\leq -k_{\rho 1} \left\| e_\rho \right\|_2^2 - \gamma_\rho \left\| e_\rho \right\|_2^{1+\alpha_\rho} \\
&= -\sqrt{2}(k_{\rho 1} \left\| e_\rho \right\|_2 + \gamma_\rho \left\| e_\rho \right\|_2^{\alpha_\rho}) V_{\rho 3}^{\frac{1}{2}} \\
&= -k_{\rho 0} V_{\rho 3}^{\frac{1}{2}},
\end{aligned} \tag{5.27}$$

where $k_{\rho 0} = \sqrt{2}(k_{\rho 1} \left\| e_\rho \right\|_2 + \gamma_\rho \left\| e_\rho \right\|_2^{\alpha_\rho}) > 0$.

By Lemma 5.2, e_ρ also converges to zero in a finite-time $\mathscr{T}_{\rho 4}$, and the settling time is bounded by $\mathscr{T}_{\rho 4} \leq \frac{\sqrt{2V_\rho(0)}}{k_{\rho 0}}$. As a result, the convergence time of the closed-loop rotational subsystem is bounded by

$$\mathscr{T}_\rho \leq \mathscr{T}_{\rho 1} + \mathscr{T}_{\rho 2} + \mathscr{T}_{\rho 3} + \mathscr{T}_{\rho 4}. \tag{5.28}$$

The proof is completed.

5.3.2 Translational subsystem stability

Similarly, an FTDO for disturbance compensation of the translational subsystem can be designed as follows:

$$\begin{aligned}
\dot{z}_{\eta 1} &= \zeta_\eta m_{\eta 1} \left\lceil \frac{\tilde{e}_{\eta 1}}{k_0} \right\rfloor^{\alpha_1} + (1-\zeta_\eta) n_{\eta 1} \left\lceil \frac{\tilde{e}_{\eta 1}}{k_0} \right\rfloor^{\beta_1} + z_{\eta 2}, \\
\dot{z}_{\eta 2} &= \zeta_\eta m_{\eta 2} \left\lceil \frac{\tilde{e}_{\eta 1}}{k_0} \right\rfloor^{\alpha_2} + (1-\zeta_\eta) n_{\eta 2} \left\lceil \frac{\tilde{e}_{\eta 1}}{k_0} \right\rfloor^{\beta_2} + z_{\eta 3} - \frac{k_t}{m}\dot{\eta} + u_\eta - \ddot{\eta}_d, \\
\dot{z}_{\eta 3} &= \zeta_\eta m_{\eta 3} \left\lceil \frac{\tilde{e}_{\eta 1}}{k_0} \right\rfloor^{\alpha_3} + (1-\zeta_\eta) n_{\eta 3} \left\lceil \frac{\tilde{e}_{\eta 1}}{k_0} \right\rfloor^{\beta_3},
\end{aligned} \tag{5.29}$$

where $z_{\eta 1}$, $z_{\eta 2}$, and $z_{\eta 3}$ are the estimates of e_η, $\dot{e}_\eta$, and Δ_η, respectively; $\tilde{e}_{\eta 1} = e_\eta - z_{\eta 1}$ is the estimation error of e_η;

$$\zeta_\eta = \begin{cases} 0 & ||\tilde{e}_{\eta 1}||_2 > e_\eta^*, \\ 1 & ||\tilde{e}_{\eta 1}||_2 \leq e_\eta^*, \end{cases}$$

with e_η^* being the threshold of $\tilde{e}_\eta$. $m_{\eta,i}$, $n_{\eta,i}$, $i = 1,2,3$ are such that matrices

$$M_{\eta 1} \begin{bmatrix} -m_{\eta 1} & 1 & 0 \\ -m_{\eta 2} & 0 & 1 \\ -m_{\eta 3} & 0 & 0 \end{bmatrix}, M_{\eta 2} = \begin{bmatrix} -n_{\eta 1} & 1 & 0 \\ -n_{\eta 2} & 0 & 1 \\ -n_{\eta 3} & 0 & 0 \end{bmatrix}$$

are Hurwitz.

Similar to the result presented in Theorem 5.1, observer (5.29) is fixed-time stable and the estimation error $\tilde{\Delta}_\eta = \Delta_\eta - z_{\eta 3}$ converges to a neighborhood of the origin $\mathscr{N}_\eta$, which can be bounded as $N_\eta = \max\{||x||_2 : x \in \mathscr{N}_\eta\}$.

Also, the integral-type sliding mode surfaces for the translational subsystem are given by

$$\begin{aligned}
s_\eta &= \dot{e}_\eta + k_{\eta 1} e_\eta + \gamma_\eta \left\lfloor e_\eta \right\rceil^{\alpha_\eta}, \\
\sigma_\eta &= s_\eta + \int_0^t \lambda_\eta \left\lfloor s_\eta \right\rceil^{\beta_\eta} dt,
\end{aligned} \tag{5.30}$$

where $k_{\eta 1} > 0$, $\gamma_\eta > 0$, $\lambda_\eta > 0$, $\alpha_\eta > 1$ and $0 < \beta_\eta < 1$ are positive constants. Assuming there are no external disturbances, an equivalent control law is given by

$$u_{eq} = \frac{k_t}{m}\dot{\eta} + \ddot{\eta}_d - k_{\eta 1}\dot{e}_\eta - \gamma_\eta \left| e_\eta \right|^{\alpha_\eta - 1} \circ \dot{e}_\eta - \lambda_\eta s_{\eta 1} - \dot{k}_{\eta 1} e_\eta - \dot{\gamma}_\eta \left\lfloor e_\eta \right\rceil^{\alpha_\eta}. \tag{5.31}$$

To maintain σ_η on the sliding mode surface when there exist disturbances acting on the system, a

switching control law is further required, which takes the following form:

$$u_{sw} = -z_{\eta 3} - k_{\eta 2}\,\mathrm{sgn}(\sigma_\eta), \tag{5.32}$$

where $k_{\eta 2} > 0$ is a positive constant. Then, the complete control law is

$$u_\eta = u_{eq} + u_{sw}. \tag{5.33}$$

We then have the following result. The proof follows a similar procedure as Theorems 5.1 and 5.2 and is therefore omitted.

Theorem 5.3 *Under Assumption 5.1, the closed loop system of the the translational subsystem (5.5) under the FTDO (5.29) and the FNTSMC (5.33) is finite-time stable if the control gain satisfies* $k_{\eta 2} > N_\eta$.

Remark 5.5 *Although the proposed control framework can guarantee the stability of the quadrotor system, hyperparameter tuning is also a vital problem for controller design [144]. The following section will discuss how the DRL technique is utilized as the hyper-parameter optimizer for the sliding mode surfaces and controllers to achieve better performance and enhance the system's robustness.*

5.4 Deep Reinforcement Learning for Parameter Optimization

Although the controllers designed in the previous sections ensure finite-time stability, there remains potential for enhancing their performance through optimizing the hyper-parameters within these controllers. DRL is a practical approach to parameter optimization. In this section, we will adopt this technology further to optimize the design parameters of our proposed control strategy. We will show that our DRL algorithm ensures the uniform ultimate boundedness of the system trajectory during the learning process.

5.4.1 System re-organization

Using Lemma 5.2, we know that the rotational subsystem controller is parameterized by $k_{\rho 1}$, $k_{\rho 2}$, γ_ρ, λ_ρ, α_ρ, and β_ρ. For simplicity and ease of programming, we fix $\alpha_\rho = 2.5$ and $\beta_\rho = 0.99$ (so are in simulations and physical experiments), and ADP is utilized to learn $k_{\rho 1}$, $k_{\rho 2}$, γ_ρ, and λ_ρ. Similarly, in Lemma 5.3, $\alpha_\eta = 2.5$ and $\beta_\eta = 0.99$ (so are in simulations and physical experiments) are fixed, and ADP is utilized to learn $k_{\eta 1}$, $k_{\eta 2}$, γ_η, and λ_η in translational subsystem. Without loss of generality, the control inputs of the two controllers can be written as

$$\begin{aligned} \tau &= \tau\left(\Theta_\rho\right), \quad & \Theta_\rho &= \left[k_{\rho 1}, k_{\rho 2}, \gamma_\rho, \lambda_\rho\right]^\top \\ u_\eta &= u_\eta\left(\Theta_\eta\right), & \Theta_\eta &= \left[k_{\eta 1}, k_{\eta 2}, \gamma_\eta, \lambda_\eta\right]^\top. \end{aligned} \tag{5.34}$$

Then, decoupling Θ_ρ and Θ_η yields

$$\begin{aligned}\tau &= B_\rho^{-1}[G_\rho\Theta_\rho - (A_\rho + z_{\rho 3}) - \dot{k}_{\rho 1}e_\rho - \dot{\gamma}_\rho \lfloor e_\rho \rceil^{\alpha_\rho}],\\ u_\eta &= G_\eta\Theta_\eta + \frac{k_t}{m}\dot{\eta} + \ddot{\eta}_d - z_{\eta 3} - \dot{k}_{\eta 1}e_\eta - \dot{\gamma}_\eta \lfloor e_\eta \rceil^{\alpha_\eta},\end{aligned} \tag{5.35}$$

where $G_\rho = [-\dot{e}_\rho, -\operatorname{sgn}(\sigma_\rho), -|e_\rho|^{\alpha_\rho - 1} \circ \dot{e}_\rho, -s_{\rho 1}] \in \mathbb{R}^{3\times 4}$ and $G_\eta = [-\dot{e}_\eta, -\operatorname{sgn}(\sigma_\eta), -|e_\eta|^{\alpha_\eta - 1} \circ \dot{e}_\eta, -s_{\eta 1}] \in \mathbb{R}^{3\times 4}$.

Note that the ADP training framework does not consider external disturbances since the FTDOs compensate them. Therefore, the analysis given in the following does not cover Δ_ρ and Δ_η related terms. Hence, substitute Eq. (5.35) into Eqs. (5.4-5.5), we have

$$\begin{aligned}\ddot{\bar{e}}_\rho &= -z_{\rho 3} - \dot{k}_{\rho 1}e_\rho - \dot{\gamma}_\rho \lfloor e_\rho \rceil^{\alpha_\rho} + G_\rho\Theta_\rho,\\ \ddot{\bar{e}}_\eta &= -z_{\eta 3} - \dot{k}_{\eta 1}e_\eta - \dot{\gamma}_\eta \lfloor e_\eta \rceil^{\alpha_\eta} + G_\eta\Theta_\eta.\end{aligned} \tag{5.36}$$

Define an augmented state vector $\chi = [e_\rho, \dot{\bar{e}}_\rho, e_\eta, \dot{\bar{e}}_\eta]^\top \in \mathbb{R}^{12}$. Then, we can verify that

$$\dot{\chi} = \mathscr{F} + \mathscr{G}\Theta, \tag{5.37}$$

where $\mathscr{F} = [(W\omega - \dot{\rho}_d)^\top, (-z_{\rho 3} - \dot{k}_{\rho 1}e_\rho - \dot{\gamma}_\rho \lfloor e_\rho \rceil^{\alpha_\rho})^\top, -\dot{\eta}_d^\top, (-z_{\eta 3} - \dot{k}_{\eta 1}e_\eta - \dot{\gamma}_\eta \lfloor e_\eta \rceil^{\alpha_\eta})^\top]^\top \in \mathbb{R}^{12\times 1}$ is the augmented system matrix, $\mathscr{G} = [\mathbf{0}, G_\rho, \mathbf{0}, \mathbf{0}; \mathbf{0}, \mathbf{0}, G_\eta, \mathbf{0}]^\top \in \mathbb{R}^{12\times 8}$ is the augmented control matrix, and $\Theta = [\Theta_\rho^\top, \Theta_\eta^\top]^\top \in \mathbb{R}^{8\times 1}$ is the augmented hyper-paremeters to be optimized.

5.4.2 HJB Equation and NN approximation

Define $\mathscr{G}\Theta = u$. Let the discounted cumulative cost function of augmented system Eq. (5.37) be

$$J(t) = \int_t^\infty e^{-\gamma(s-t)} \left[Q_{\chi(s)} + u(s)^\top R u(s) \right] ds, \tag{5.38}$$

where $Q_\chi = \chi^\top Q \chi$, and $Q \in \mathbb{R}^{12\times 12}$, $R \in \mathbb{R}^{6\times 6}$ are symmetric positive definite diagonal matrices, and $0 < \gamma < 1$ is a discount factor. Using the Leibniz's rule to J yields

$$\dot{J}(t) = -Q_\chi - u(t)^\top R u(t) + \gamma J(t). \tag{5.39}$$

Denote $J \triangleq J(t)$ and $u \triangleq u(t)$. The HJB equation is

$$H \equiv Q_\chi + u^\top R u + J_\chi(\mathscr{F} + u) - \gamma J = 0. \tag{5.40}$$

Letting $\partial H/\partial u = 0$ yields $u = -\frac{1}{2}R^{-1}J_\chi$, $\Theta = \mathscr{G}^\dagger u$, where $\mathscr{G}^\dagger = \left(\mathscr{G}^\top \mathscr{G}\right)^{-1}\mathscr{G}^\top$ is the pseudo-inverse matrix of $\mathscr{G}$.

Based on the Weierstrass high-order approximation theorem [145], the index function J and its partial derivative $J_\chi = \partial J/\partial \chi$ can be respectively approximated by

$$J = W_1^\top \Phi + \varepsilon,\ J_\chi = \Phi_\chi^\top W_1 + \varepsilon_\chi, \tag{5.41}$$

where ε and $\varepsilon_\chi = \partial \varepsilon_\chi/\partial \chi$ are approximation errors, $W_1 \in \mathbb{R}^N$ is the weight, whose dimension N is the number of neurons, $\Phi \in \mathbb{R}^{N\times 1}$ is the activation function, and $\Phi_\chi = \partial \Phi/\partial \chi \in \mathbb{R}^{N\times 12}$ is the Jacobi matrix of Φ to χ. Substituting u into (5.40) yields

$$Q_\chi + J_\chi^\top \mathscr{F} - \frac{1}{4}J_\chi R^{-1}J_\chi^\top - \gamma J = 0. \tag{5.42}$$

Then, substituting (5.41) into (5.42) gives

$$W_1^\top \Phi_\chi \mathscr{F} - \frac{1}{4}W_1^\top \Phi_\chi R^{-1}\Phi_\chi W_1 + Q_\chi - \gamma W_1^\top \Phi = \varepsilon_H, \tag{5.43}$$

where $\varepsilon_H = \frac{1}{4}\varepsilon_\chi^\top R^{-1}\varepsilon_\chi - \varepsilon_\chi^\top \mathscr{F} + \frac{1}{2}W_1^\top \Phi_\chi R^{-1}\varepsilon_\chi$ is the residual error of HJB equation. Assuming $\|\mathscr{F}\| \le k_\chi \|\chi\|$, $\|\varepsilon\| \le \varepsilon_m$, $\left\|\varepsilon_\chi\right\| \le \varepsilon_{m\chi}$, $\|\Phi\| \le \Phi_m$, $\left\|\Phi_\chi\right\| \le \Phi_{m\chi}$ for some position constants k_χ, ε_m, $\varepsilon_{m\chi}$, Φ_m, and $\Phi_{m\chi}$. We then have the following result:

Theorem 5.4 *Under Assumption 5.1, consider system (5.37). Let u be any admissible bounded controller, let $\hat{J} = \hat{W}_1^\top \Phi, \hat{J}_\chi = \Phi_\chi^\top \hat{W}_1$ be respectively the approximations of J and its partial derivative J_χ. Then, the trajectory of the entire closed-loop system during the learning process is Uniformly Ultimately Bounded (UUB) under the approximated control law*

$$u_1 = -\frac{1}{2}R^{-1}\Phi_\chi^\top \hat{W}_2, \tag{5.44}$$

with $\hat{W}_1$ and $\hat{W}_2$ generated by

$$\begin{aligned}
\dot{\hat{W}}_1 &= -\alpha_1 \frac{\sigma_2}{\left(\sigma_2^\top \sigma_2 + 1\right)^2}(\sigma^\top \hat{W}_1 + Q_\chi + u_1^\top R u_1), \\
\dot{\hat{W}}_2 &= -\alpha_2(F_2\hat{W}_2 - F_1\bar{\sigma}^\top \hat{W}_1 - \frac{1}{4}\bar{D}_1\hat{W}_2 M^\top \hat{W}_1),
\end{aligned} \tag{5.45}$$

where $\sigma_2 = \Phi_\chi(\mathscr{F} + u), \bar{\sigma}_2 = \sigma_2/\left(\sigma_2^\top \sigma_2 + 1\right)$, and $\bar{D}_1 = \Phi_\chi R^{-1}\Phi_\chi^\top, M = \sigma_2/\left(\sigma_2^\top \sigma_2 + 1\right)^2$.

Proof 5.4 *Take the Lyapunov function as*

$$L = J + \overbrace{\frac{1}{2}\tilde{W}_1^\top \alpha_1^{-1}\tilde{W}_1}^{L_1} + \overbrace{\frac{1}{2}\tilde{W}_2^\top \alpha_2^{-1}\tilde{W}_2}^{L_2}, \tag{5.46}$$

where $\tilde{W}_1 = W_1 - \hat{W}_1$ *and* $\tilde{W}_2 = W_1 - \hat{W}_2$ *is the estimation error. Differentiating L yields*

$$\dot{L} = \dot{J} + \overbrace{\tilde{W}_1^\top \alpha_1^{-1}\dot{\tilde{W}}_1}^{\dot{L}_1} + \overbrace{\tilde{W}_2^\top \alpha_2^{-1}\dot{\tilde{W}}_2}^{\dot{L}_2}. \tag{5.47}$$

For the first term,

$$\begin{aligned} \dot{J} &= (\Phi_\chi^\top W_1 + \varepsilon_\chi)^\top (\mathscr{F} + u) \\ &= (\Phi_\chi^\top W_1 + \varepsilon_\chi)^\top (\mathscr{F} - \frac{1}{2}R^{-1}\Phi_\chi^\top \hat{W}_2) \\ &= W_1^\top \Phi_\chi \mathscr{F} - \frac{1}{2}W_1^\top \bar{D}_1 W_1 + \frac{1}{2}W_1^\top \bar{D}_1 \tilde{W}_2 + \varepsilon_1, \end{aligned} \tag{5.48}$$

where $\varepsilon_1 = \varepsilon_\chi^\top (\mathscr{F} - \frac{1}{2}R^{-1}\Phi_\chi^\top \hat{W}_2) \in \mathbb{R}$. *Define* $\sigma_1 = \Phi_\chi (\mathscr{F} + u) \in \mathbb{R}^N$, *and for ideal NN controller approximation, we have* $u = \frac{1}{2}R^{-1}\Phi_\chi W_1$. *Then, (5.48) can be further put forward as*

$$\begin{aligned} \dot{J} &= W_1^\top \sigma_1 + \frac{1}{2}W_1^\top \bar{D}_1 W_1 - \frac{1}{2}W_1^\top \bar{D}_1 \tilde{W}_2 - W_1^\top \Phi_1 u + \varepsilon_1 \\ &= W_1^\top \sigma_1 + \frac{1}{2}W_1^\top \bar{D}_1 \tilde{W}_2 + \varepsilon_1. \end{aligned} \tag{5.49}$$

Recall (5.43) that the HJB equation can be derived as

$$\begin{aligned} \varepsilon_H &= W_1^\top \Phi_\chi \mathscr{F} - \frac{1}{4}W_1^\top \Phi_\chi R^{-1}\Phi_\chi W_1 + Q_\chi - \gamma W_1^\top \Phi \\ &= W_1^\top \sigma_1 + \frac{1}{4}W_1^\top \bar{D}_1 W_1 - \gamma W_1^\top \Phi + Q_\chi. \end{aligned} \tag{5.50}$$

Substituting (5.50) into $\dot{J}$ *and doing some manipulations yields*

$$\dot{J} = \dot{\bar{J}} + \frac{1}{2}W_1^\top \bar{D}_1 \tilde{W}_2 + \varepsilon_1, \tag{5.51}$$

where $\dot{\bar{J}} = \varepsilon_H - Q_\chi - \frac{1}{4}W_1^\top \bar{D}_1 W_1 + \gamma W_1^\top \Phi$. *For the second term, there is*

$$\begin{aligned} \dot{L}_1 &= \tilde{W}_1^\top M(\sigma_2^\top \hat{W}_1 + Q_\chi + u_1^\top R u_1) \\ &= \tilde{W}_1^\top M(\sigma_2^\top \hat{W}_1 - \sigma_1^\top W_1 + \gamma W_1^\top \Phi + \frac{1}{4}\hat{W}_2^\top \bar{D}_1 \hat{W}_2 - \frac{1}{4}W_1^\top \bar{D}_1 W_1 + \varepsilon_H) \end{aligned} \tag{5.52}$$

Note that

$$\begin{aligned}\sigma_2^\top \hat{W}_1 - \sigma_1^\top W_1 &= \left[\Phi_\chi\left(\mathscr{F}+u\right)\right]^\top \hat{W}_1 - \left[\Phi_\chi\left(\mathscr{F}+u\right)\right]^\top W_1 \\ &= -\tilde{W}_1\Phi_\chi\mathscr{F} - \frac{1}{2}\hat{W}_2^\top \bar{D}_1\hat{W}_1 + \frac{1}{2}W_1^\top \bar{D}_1 W_1\end{aligned} \tag{5.53}$$

Substituting (5.50) and (5.53) back to (5.52), using the fact $W_1 = \hat{W}_1 + \tilde{W}_1 = \hat{W}_2 + \tilde{W}_2$*, and doing some manipulations yield*

$$\begin{aligned}\dot{L}_1 &= \tilde{W}_1^\top M\left(\frac{1}{2}\hat{W}_2^\top \bar{D}_1\tilde{W}_1 + \frac{1}{4}\tilde{W}_2^\top \bar{D}_1\tilde{W}_2 - \tilde{W}_1^\top\Phi_\chi\mathscr{F} + \gamma W_1^\top\Phi + \varepsilon_H\right) \\ &= \tilde{W}_1^\top M\left[\frac{1}{4}\tilde{W}_2^\top \bar{D}_1\tilde{W}_2 + \gamma W_1^\top\Phi + \varepsilon_H + \tilde{W}_1^\top\left(\frac{1}{2}\bar{D}_1\hat{W}_2 - \Phi_\chi\mathscr{F}\right)\right] \\ &= \tilde{W}_1^\top M\left(\frac{1}{4}\tilde{W}_2^\top \bar{D}_1\tilde{W}_2 + \gamma W_1^\top\Phi + \varepsilon_H - \sigma_2^\top\tilde{W}_1\right) \\ &= \dot{\tilde{L}}_1 + \frac{1}{4}\tilde{W}_1^\top M\tilde{W}_2^\top \bar{D}_1\tilde{W}_2,\end{aligned} \tag{5.54}$$

where $\dot{\tilde{L}}_1 = \tilde{W}_1^\top M\left(-\sigma_2^\top\tilde{W}_1 + \varepsilon_H + \gamma W_1^\top\Phi\right)$*. Finally, adding* $\dot{J}$*,* $\dot{L}_1$*, and* $\dot{L}_2$ *yields*

$$\dot{L} = \dot{J} + \dot{\tilde{L}}_1 + \varepsilon_1 + \frac{1}{2}W_1^\top \bar{D}_1\tilde{W}_2 + \frac{1}{4}\tilde{W}_1^\top M\tilde{W}_2^\top \bar{D}_1\tilde{W}_2 + \tilde{W}_2^\top\alpha_2^{-1}\dot{\hat{W}}_2. \tag{5.55}$$

In (5.55), there is

$$\begin{aligned}&\frac{1}{4}\tilde{W}_1^\top M\tilde{W}_2^\top \bar{D}_1\tilde{W}_2 \\ =&\frac{1}{4}\tilde{W}_2^\top \tilde{W}_2 M\bar{D}_1\tilde{W}_1 \\ =&\frac{1}{4}\tilde{W}_2^\top W_1 M\bar{D}_1\tilde{W}_1 - \frac{1}{4}\tilde{W}_2^\top \hat{W}_2 M\bar{D}_1\tilde{W}_1 \\ =&\frac{1}{4}\tilde{W}_2^\top W_1 M\bar{D}_1\tilde{W}_1 + \frac{1}{4}\tilde{W}_2^\top \bar{D}_1\hat{W}_2 M^\top\hat{W}_1 - \frac{1}{4}\tilde{W}_2^\top \bar{D}_1 W_1 M^\top W_1 + \frac{1}{4}\tilde{W}_2^\top \bar{D}_1\tilde{W}_2 M^\top W_1\end{aligned} \tag{5.56}$$

and

$$\begin{aligned}&\tilde{W}_2\alpha_2^{-1}\dot{\tilde{W}}_2 \\ =&-\tilde{W}_2^\top\alpha_2^{-1}\dot{\hat{W}}_2 \\ =&\tilde{W}_2^\top\left(F_2\hat{W}_2 - F_1\bar{\sigma}_2^\top\hat{W}_1 - \frac{1}{4}\bar{D}_1\hat{W}_2 M^\top\hat{W}_1\right) \\ =&\tilde{W}_2^\top F_2\hat{W}_2 - \tilde{W}_2 F_1\bar{\sigma}_2^\top\hat{W}_1 - \frac{1}{4}\tilde{W}_2^\top \bar{D}_1\hat{W}_2 M^\top\hat{W}_1.\end{aligned} \tag{5.57}$$

Substituting (5.56) and (5.57) into (5.55) yields

$$\begin{aligned}\dot{L} =& -Q_\chi - \frac{1}{4}W_1^\top \bar{D}_1 W_1 + \gamma W_1^\top \Phi + \varepsilon_H + \varepsilon_1 + \tilde{W}_1^\top \bar{\sigma}_2 \left(-\bar{\sigma}_1^\top \tilde{W}_1 + \varepsilon_H/m_s + \gamma W_1^\top \Phi\right) \\ &+ \frac{1}{2}W_1^\top \bar{D}_1 \tilde{W}_2 + \frac{1}{4}\tilde{W}_2^\top \bar{D}_1 W_1 M^\top \tilde{W}_1 - \frac{1}{4}\tilde{W}_2^\top \bar{D}_1 W_1 M^\top W - 1 + \frac{1}{4}\tilde{W}_2^\top \bar{D}_1 \tilde{W}_2 M^\top W_1 \\ &+ \tilde{W}_2^\top F_2 W_1 - \tilde{W}_2^\top F_2 \tilde{W}_2 - \tilde{W}_2 F_1 \bar{\sigma}_2^\top W_1 + \tilde{W}_2^\top F_1 \bar{\sigma}_2^\top \tilde{W}_1,\end{aligned} \tag{5.58}$$

where $m_s = \sigma_2^\top \sigma_2 + 1 \in \mathbb{R}$.

Based on the assumptions above, there is

$$\|\varepsilon_1\| \leq \varepsilon_\chi k_\chi \|\chi\| + \frac{1}{2}\varepsilon_\chi \Phi_\chi \sigma_{min}(R)\left(\|W_1\| + \left\|\tilde{W}_2\right\|\right). \tag{5.59}$$

Obviously, there exists a positive constant ε_0 *such that* $\|\varepsilon_H\| \leq \varepsilon_0$ *and a positive constant* q_0 *such that* $Q_\chi > \chi^\top q_0 \chi$. *For clarity, we define*

$$\Xi = \left[\chi^\top, \tilde{W}_1^\top \bar{\sigma}_2, \tilde{W}_2^\top\right]^\top = \left[\Xi_1^\top, \Xi_2^\top, \Xi_3^\top\right]^\top,$$

then $\dot{L}$ *can be further derived as*

$$\begin{aligned}\dot{L} \leq& -\chi^\top q_0 \chi + \frac{1}{4}\|W_1\|^2 \|\bar{D}_1\| + \varepsilon_0 + \gamma W_1 \Phi + \varepsilon_\chi k_\chi \|\chi\| + \frac{1}{2}\varepsilon_\chi \Phi_\chi \sigma_{min}(R)\left(\|W_1\| + \left\|\tilde{W}_2\right\|\right) \\ &- \Xi_2^\top \Xi_2 + \Xi_2^\top \left(\frac{\varepsilon_H}{m_s} + \gamma W_1^\top \Phi\right) + \frac{1}{2}W_1^\top \bar{D}_1 \Xi_3 + \frac{1}{4m_s}\Xi_3^\top \bar{D}_1 W_1 \Xi_2 - \frac{1}{4m_s}\Xi_3^\top \bar{D}_1 W_1 \bar{\sigma}_2^\top W_1 \\ &+ \frac{\|\bar{\sigma}_2\| \|\bar{D}_1\| \|W_1\|}{4m_s}\Xi_3^\top \Xi_3 + \Xi_3^\top F_2 W_1 - \Xi_3^\top \Xi_3 - \Xi_3^\top F_1 \bar{\sigma}_2^\top W_1 + \Xi_3^\top F_1 \Xi_2 \\ =& \frac{1}{4}\|W_1\|^2 \|\bar{D}_1\| + \varepsilon_0 + \gamma \|W_1\| \Phi_m + \frac{1}{2}\varepsilon_{m\chi}\Phi_{m\chi} \|W\|_1 \sigma_{min}(R) - \Xi^\top \Gamma_1 \Xi + \Xi^\top \Gamma_2,\end{aligned} \tag{5.60}$$

where

$$\Gamma_1 = \begin{bmatrix} q_0 I_{12} & 0 & \\ 0 & I & -\frac{1}{8m_s}\bar{D}_1 W_1 - \frac{1}{2}F_1 \\ 0 & -\frac{1}{8m_s}\bar{D}_1 W_1 - \frac{1}{2}F_1 & F_2 + \frac{\|\bar{\sigma}_2\| \|\bar{D}_1\| \|W_1\|}{4m_s} \end{bmatrix}$$

and

$$\Gamma_2 = \begin{bmatrix} \varepsilon_{m\chi} k_f \\ \frac{\varepsilon_0}{m_s} + \gamma W_1^\top \Phi \\ \begin{cases} \frac{1}{2}\varepsilon_{m\chi}\Phi_{m\chi}\sigma_{min}(R) + \frac{1}{2}\bar{D}_1 W_1^\top \\ + \left(F_2 - F_1 \bar{\sigma}_2^\top - \frac{1}{4m_s}\bar{D}_1 W_1 \bar{\sigma}_2^\top\right) W_1 \end{cases} \end{bmatrix}.$$

Define

$$\Gamma_3 = \frac{1}{4}\|W_1\|^2\|\bar{D}_1\| + \varepsilon_0 + \gamma\|W_1\|\Phi_m + \frac{1}{2}\varepsilon_{m\chi}\Phi_{m\chi}\|W\|_1\sigma_{min}(R) \tag{5.61}$$

Therefore, $\dot{L} \leq 0$ *if*

$$\|\Xi\| > \frac{-\|\Gamma_2\| + \sqrt{\|\Gamma_2\|^2 + 4\|\Gamma_1\|\|\Gamma\|_3}}{2\|\Gamma_2\|}. \tag{5.62}$$

This completes the proof.

Remark 5.6 *The introduction of Theorem 5.4 guarantees the UUB stability during the NN training process. However, many learning-based frameworks can only handle control problems with a fixed pre-defined reference trajectory (see, [121], [139], et al.). However, the DRL proposed in this study employs only a single-layer network without activation functions to approximate complex functions, which inherently limits its network capacity and expressive power. Additionally, given that the proposed method in this study follows an offline learning-online deployment control framework, it is crucial to bridge the gap between theoretical simulations and practical physical experiments. To address this, in contrast to prior methodologies, we adopt two key strategies: 1) replacing the traditional single-layer network with a more expressive multi-layer deep network and 2) employing a set of randomly generated reference trajectories to train the NNs, thereby enhancing the robustness of the NNs and significantly narrowing the gap between simulation and experimentation. These enhancements aim to improve the robustness and adaptability of the proposed control framework.*

5.4.3 NN Training

The DRL algorithm used in this study is selected as Proximal Policy Optimization (PPO) [146]. PPO is a policy iteration-based reinforcement learning algorithm that integrates concepts from Conservative Policy Optimization (CPO) [147], Policy Gradient (PG) [148], and Natural Policy Gradient (NPG) [149]. It ensures near-monotonic policy improvement during the learning process. PPO serves as an engineering approximation to Trust Region Policy Optimization (TRPO) [150], as TRPO involves computationally intensive calculations of the Hessian matrix. Consequently, PPO is widely favored in robotics control applications due to its balance of performance and computational efficiency. The pseudocode code for PPO is illustrated in Algorithm **??**. The detailed computational principle of PPO can be found in some highly-stared Github repositories (*e.g., see* [1]). Table 5.1 lists some related parameters of PPO.

[1] Our self-developed DRL simulation platform: https://github.com/HKPolyU-UAV/ReinforcementLearningPlatform

Table 5.1: Some related parameters of the PPO optimizer.

Symbol	Value	Symbol	Value	Symbol	Value	Symbol	Value
T_m	20	dt	0.01	std_0	0.35	std_{min}	0.1
std_d	0.05	std_{dN}	500	γ	0.95	K_{ep}	25
b_s	$T_m/dt*2$	a_{lr}	10^{-4}	c_{lr}	10^{-3}	c_{en}	0.01
λ	0.95	c_{min}	0.8	c_{max}	1.2		

Fig. 5.2 demonstrates the structures of PPO's actor and critic networks in both rotational and translational subsystems' network training. In Fig. 5.2, the actor network is a multi-input-multi-output network. The input of the actor network is the concatenate of the tracking error and the first-order derivative of the tracking error. The output of the actor network is the optimized hyper-parameter of the RFNTSMC proposed in Theorems 5.2 and 5.3. However, the critic network is a multi-input-single-output NN. The input of the critic network is the concatenate of the input and output of the actor-network. The output is a scalar value that quantifies the quality of the input state-action tuple generated from the actor-network.

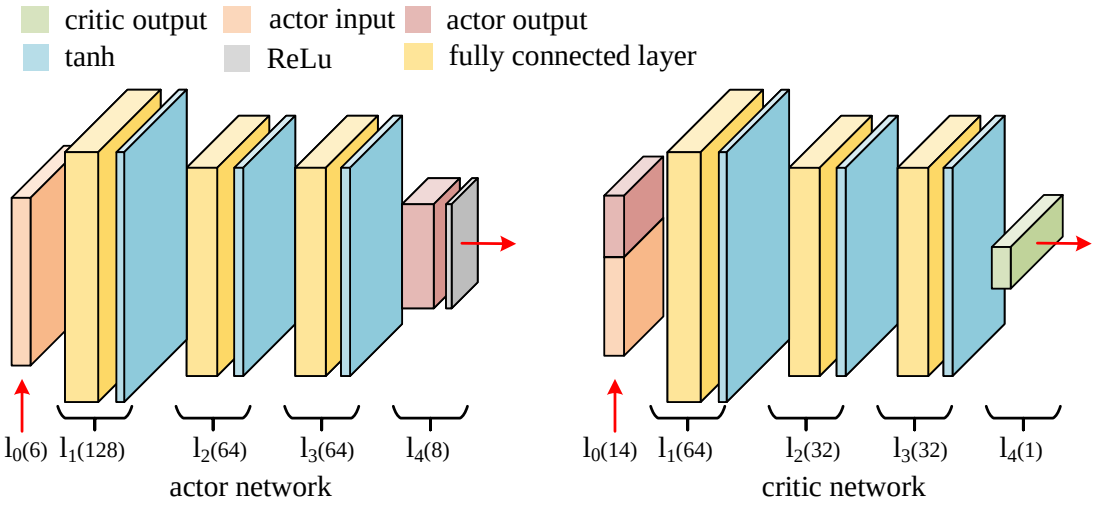


Figure 5.2: The structures of the NNs in PPO.

Additionally, the activation function of the last layer in the actor network must be selected as 'ReLU' rather than 'tanh' because the hyper-parameters of the RFNTSMCs must be positive. The output dimension of the actor network is '8' rather than '4' because we separate the parameter optimization of x, y, and z directions in real-world application scenarios to enhance the flexibility of the parameter optimizer. We extend k_{p1} as $[k_{p1,x}, k_{p1,y}, k_{p1,z}]$ and k_{p2} as $[k_{p2,x}, k_{p2,y}, k_{p2,z}]$, $p = \rho, \eta$ to respectively represent the parameters in x, y, and z directions.

Rotational susbsystem training:

The augmented state x_ρ and some parameters of RFNTSMC Θ_ρ can be respectively given by

$$\begin{aligned} \chi_\rho &= \left[e_\rho^\top, \dot{e}_\rho^\top\right]^\top, \\ \Theta_\rho &= \left[k_{\rho 1,x}, k_{\rho 1,y}, k_{\rho 1,z}, k_{\rho 2,x}, k_{\rho 2,y}, k_{\rho 2,z}, \gamma_\rho, \lambda_\rho\right]^\top. \end{aligned} \tag{5.63}$$

The reward function is defined as

$$J_\rho(t) = \int_{t=0}^{\infty} e^{-\gamma(s-t)} \left[\chi_\rho^\top R_{\chi,\rho} \chi_\rho + \tau^\top R_\tau \tau\right] ds, \tag{5.64}$$

where $R_{\chi,\rho} = \text{diag}(R_\rho, R_{\dot{\rho}})$, $R_\rho = \mathbf{I}_3$, $R_{\dot{\rho}} = 0.01\mathbf{I}_3$, and $R_\tau = 0.01\mathbf{I}_3$, and $\gamma = 0.99$ is the discount factor. The random reference attitude angle set is defined as

$$\rho_d = A_{r\rho} \sin\left(\omega_{r\rho} t + \phi_{r\rho}\right) + A_{\rho 0}, \tag{5.65}$$

with $0 \leq ||A_{r\rho}||_\infty < \pi/2$, $2\pi/3 \leq ||\omega_{r\rho}||_\infty \leq 4\pi/3$, $0 \leq ||\phi_{r\rho}||_\infty \leq \pi/2$, and $||A_{\rho 0}||_\infty = 0$.

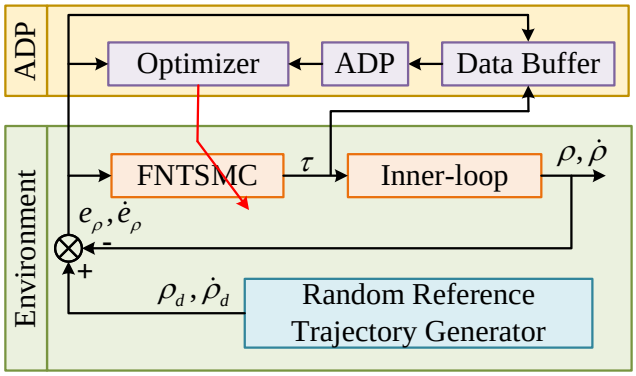


Figure 5.3: Logic block diagram of the rotational subsystem training process.

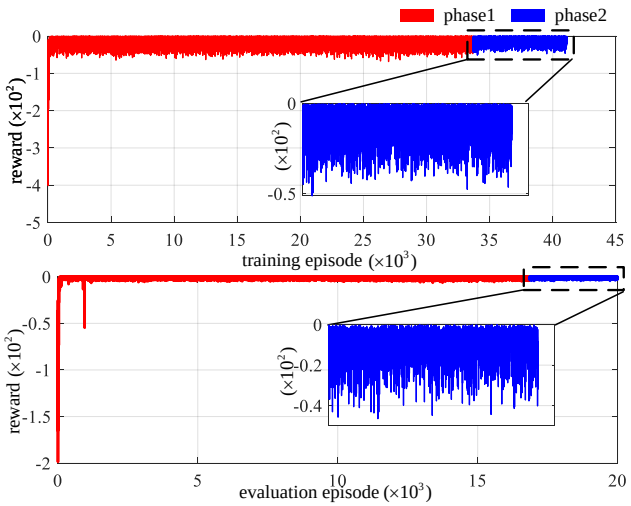


Figure 5.4: Cost for rotational learning subsystem in the training process (upper) and evaluation process (lower).

Fig. 5.3 demonstrates the diagram of the rotational subsystem training process. Fig. 5.4 records the rotational subsystem's cost function during the training and evaluation processes. Fig. 5.4 illustrates that the NN optimizer is well-established after around 1000 training episodes. We evaluate the performance of the DRL-optimized system every two training episodes. Additionally, to further improve the performance of the NN optimizer, a multi-phase network training technique [151] is utilized.

Translational Subsystem Training

Similar to the rotational subsystem, the augmented state x_η and hyper-parameter Θ_η are respectively given by

$$\begin{aligned} \chi_\eta &= \left[e_\eta^\top, \dot{e}_\eta^\top\right]^\top, \\ \Theta_\eta &= \left[k_{\eta 1,x}, k_{\eta 1,y}, k_{\eta 1,z}, k_{\eta 2,x}, k_{\eta 2,y}, k_{\eta 2,z}, \gamma_\eta, \lambda_\eta\right]^\top, \end{aligned} \tag{5.66}$$

where $k_{\eta(1,2),p}, p = x,y,z$ represents the components of $k_{\eta(1,2)}$ in the x, y, and y directions, respectively. The reward function is defined as

$$J_\eta(t) = \int_{t=0}^{\infty} e^{-\gamma(s-t)} \left[\chi_\eta^\top R_{\chi,\eta} \chi_\eta + u_\eta^\top R_\eta u_\eta\right] ds, \tag{5.67}$$

where $R_{\chi,\eta} = \text{diag}(R_\eta, R_{\dot{\eta}})$, $R_\eta = \mathbf{I_3}$, $R_{\dot{\eta}} = 0.05\mathbf{I_3}$, $R_u = 0.01\mathbf{I_3}$, and $\gamma = 0.99$.

The random reference trajectory for the translational loop training is defined as

$$\eta_d = A_{r\eta} \sin\left(\omega_{r\eta} t + \phi_{r\eta}\right) + A_{\eta 0}, \tag{5.68}$$

with $||A_{r\eta}||_\infty \leq 1.5$, $\pi/10 \leq ||\omega_{r\eta}||_\infty \leq 2\pi/5$, $||\phi_{r\eta}||_\infty \leq \pi/2$, and $||A_{\eta 0}||_\infty = 0$.

The logic block diagram for translational subsystem training is given in Fig. 5.5, and Fig. 5.6 represents the training and the evaluation results of the translational subsystem. Similarly, we utilized a three-phase training approach to enhance the robustness and generalization capacity of the NN optimizer. The lower subfigure in Fig. 5.6 initially shows a brief fluctuation process. This fluctuation is due to the unconvergence of the NN weights. The training process tends to be stable in the second and third phases.

After training the NNs, we test the comparative performance of different control frameworks (with or without DRL, with or without FTDO), which is recorded in Fig. 5.7. We fix $\phi_{r\eta} = \mathbf{0}_3$ and $A_{\eta 0} = \mathbf{0}_3$, uniformly sampling $A_{r\eta} \in [0, 1.5]$ and $T_{r\eta} \in [0.2\pi, 0.5\pi]$. A comparison of the "green-blue" (or "orange-red") reward surfaces shows that the performance of the proposed SMC (with or without a fixed-time observer) optimized by DRL surpasses that of without DRL. The superiority of the proposed fixed-time observer is evident from the comparison of the "orange-green" (or "red-blue") reward surfaces. Additionally, the traditional RFNTSMC without DRL or a disturbance observer exhibits the least favorable performance among the four frameworks.

5.5 Simulation

This section demonstrates some comparative simulations that can reveal the superiority and effectiveness of our proposed control framework. Different from other training frameworks (see [131], [134], [152]),

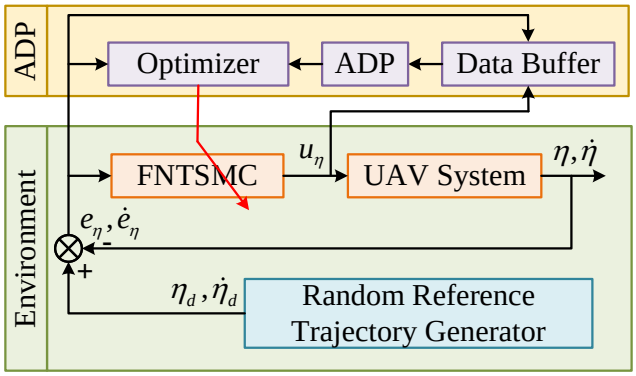


Figure 5.5: Logic block diagram of the translational subsystem training process.

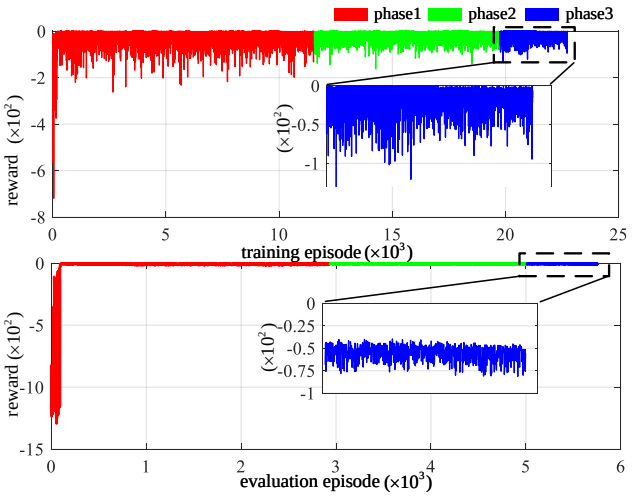


Figure 5.6: Cost for translational learning subsystem in the training and evaluation process.

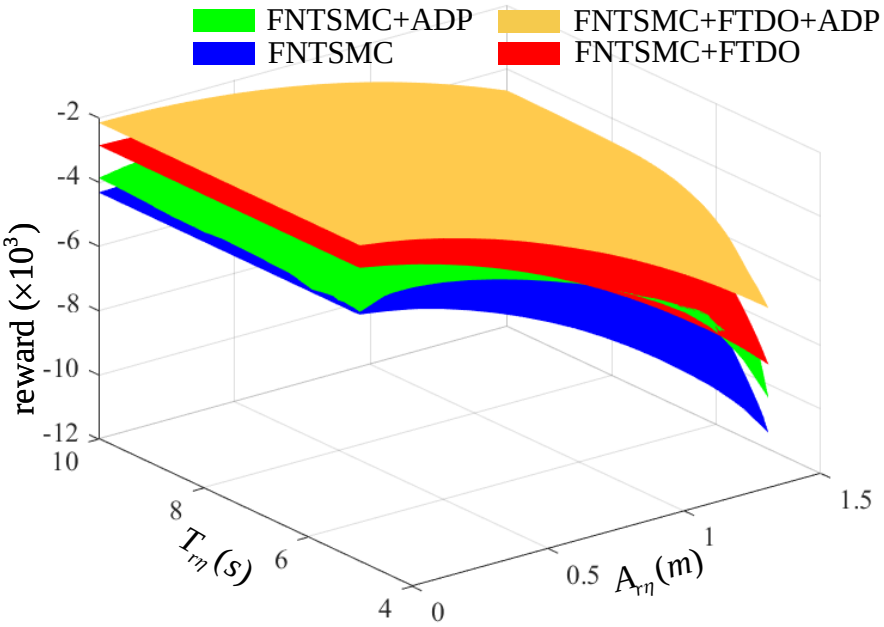


Figure 5.7: The cost surface under different control frameworks.

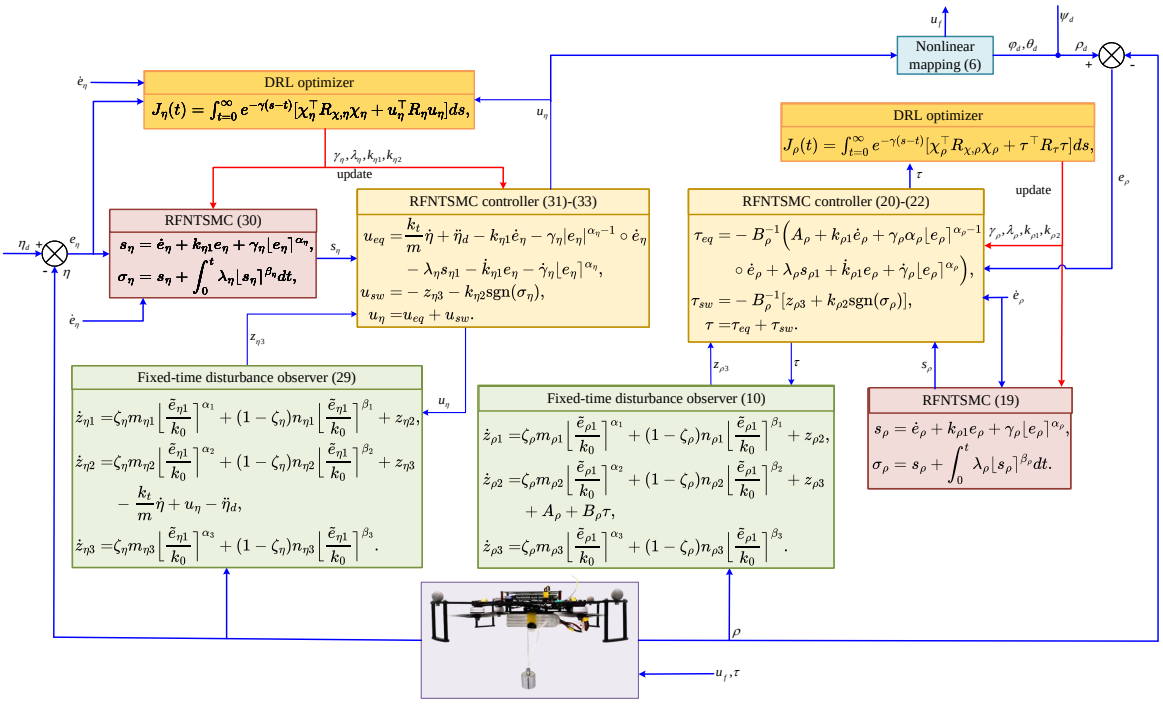


Figure 5.8: The diagram of the quadrotor control system.

we train the RL-based optimizers separately to enhance the robustness and efficiency. Specifically, in the mathematical model, $m = 0.850$, $J_0 = 10^{-3} \times \text{diag}[4.113, 4.113, 8.255]$, $k_\eta = 0.001$, and $k_r = 0.001$. The disturbance signal is composed of a combination of sine, cosine, constant, ramp, and sine functions induced by other sine functions. To demonstrate the system's entire control diagram, the proposed method's control flow and the signal pathways of some important variables are illustrated in Fig. 5.8.

5.5.1 Simulation Group 1: Fixed-point control

The initial location of the quadrotor is set to be $(0,0,0)$m, and the target location is $(5,5,3)$m. Fig. 5.9 records the comparative state response of five different control frameworks, and Fig. 5.10 illustrates the output of the observer.

Fig. 5.9 clearly illustrates the performance differences among various control methods in fixed set-point regulation control. First, methods incorporating disturbance observers (such as 'RFNTSMC + FTDO' and 'RFNTSMC + FTDO + DRL') demonstrate significantly better control performance than their counterparts without observers. This highlights the effectiveness of the proposed disturbance observer. Second, although some control methods do not utilize observers, Fig. 5.10 reveals that the external disturbances during the first 10 seconds are inherently minor. Thus, by comparing Backstepping with sliding mode-based control methods without observers (such as 'FNTSMC' and 'RFNTSMC + DRL'), it is evident that sliding mode-based methods inherently exhibit superior dis-

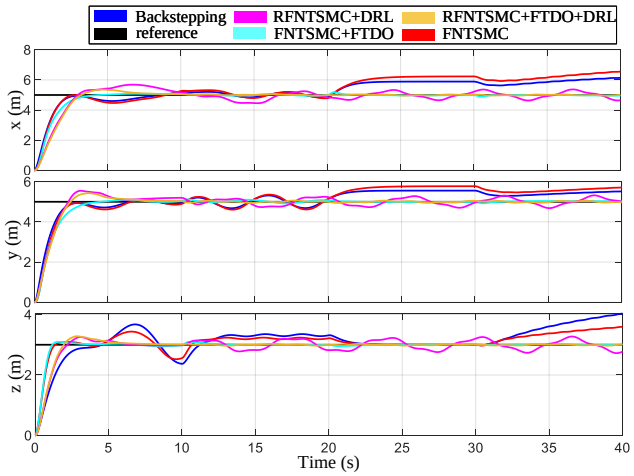


Figure 5.9: State response of the quadrotor in the fixed-point simulation.

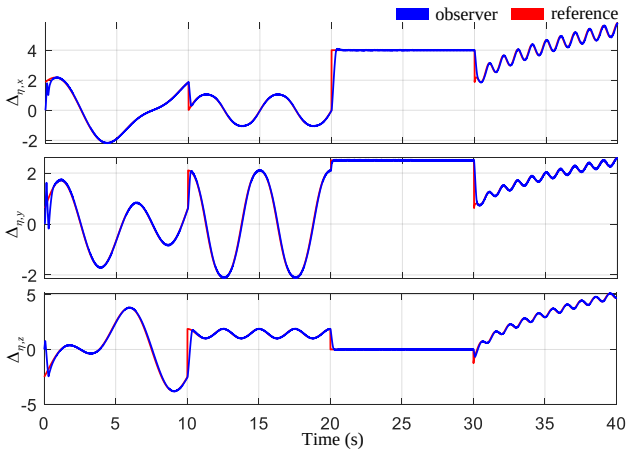


Figure 5.10: OOutputof the observer in the fixed-point simulation.

turbance rejection due to their strong robustness. Finally, comparing 'RFNTSMC + FTDO + DRL' with 'RFNTSMC + FTDO' shows that 'RFNTSMC' with DRL-based adaptive parameter tuning outperforms 'RFNTSMC' with fixed gains. These analyses confirm the effectiveness and superiority of the proposed control framework. Correspondingly, Fig. 5.10 presents the output of the observer. The observer's performance aligns precisely with the theoretical claims outlined in Theorem 5.1. Specifically, it can be observed that the proposed observer can track complex disturbance signals within a very short time frame, thereby providing the controller with adequate and accurate model compensation.

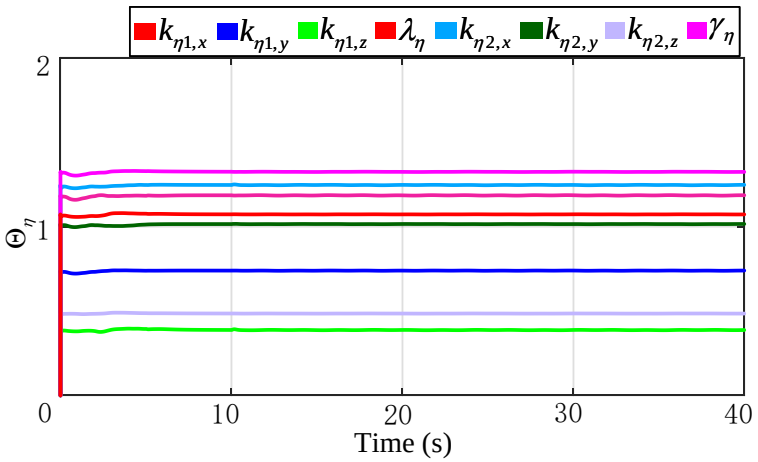


Figure 5.11: hyper-parameters tuned by DRL in the fixed-point simulation.

Fig. 5.11 demonstrates the corresponding time-variant hyper-parameters of the proposed SMC tuned by DRL. As illustrated in Fig. 5.11, all control parameters are initially set to zero. When

the controller is activated, the parameters adaptively adjust in real time based on the tracking error and stabilize after approximately 4 seconds. This behavior aligns perfectly with the response curve of 'RFNTSMC + FTDO + DRL' (represented by the orange line in Fig. 5.9). Notably, after 4 seconds, the controller gains remain almost constant. This is because the control error determines the controller gains; as the error approaches zero, the gains converge to a steady state.

5.5.2 Simulation Group 2: Trajectory tracking control

After verifying the algorithm with a fixed-point simulation, we further test our control framework under a more complicated working environment. The reference trajectory is set to be time-variant, which is defined by

$$\eta_d = [5\cos(0.2\pi t), 5\sin(0.2\pi t), 3\sin(0.2\pi t)]^\top. \tag{5.69}$$

Similar to simulation group 1, Figs. 5.12 and 5.13 respectively record the comparative state response and the output of the observer.

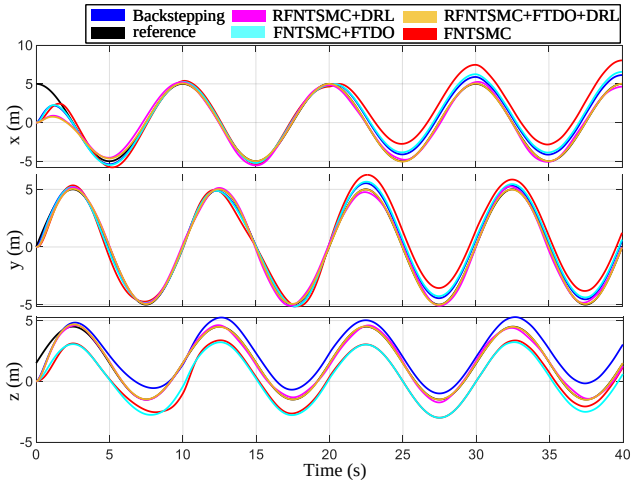


Figure 5.12: State response of the quadrotor in the tracking simulation.

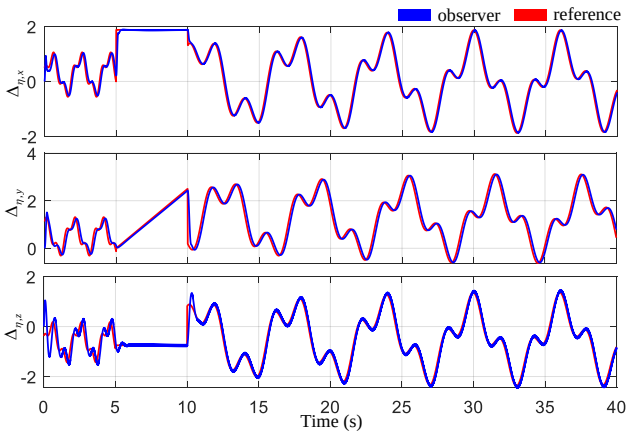


Figure 5.13: OOutputof the observer in the tracking simulation.

The patterns revealed in Figs. 5.12 and 5.13 are fundamentally consistent with those in Figs. 5.9 and 5.10. Compared to the orange curve ('RFNTSMC + FTDO + DRL'), the responses under other control methods exhibit shortcomings such as insufficient response, severe overshoot, and slow convergence, further demonstrating the superiority of the proposed method. Similarly, 5.13 records the output of the observer. Unlike in simulation group 1, we introduced a pure ramp signal and a sinusoidal signal induced by a sine function into the disturbance signal. Despite this, Fig. 5.13 clearly shows that the observer can rapidly and accurately track the estimated signals.

The control gains tuned by DRL are represented in Fig. 5.14. The curves depicted in Fig. 5.14

are nearly identical to those in Fig. 5.11, as the control errors ultimately approach zero, resulting in similar outputs from the neural network. Although minor differences exist in the initial few seconds of adjustment, the tuning process concludes rapidly. Consequently, the control gain adjustment curves for the two sets of simulations are fundamentally the same.

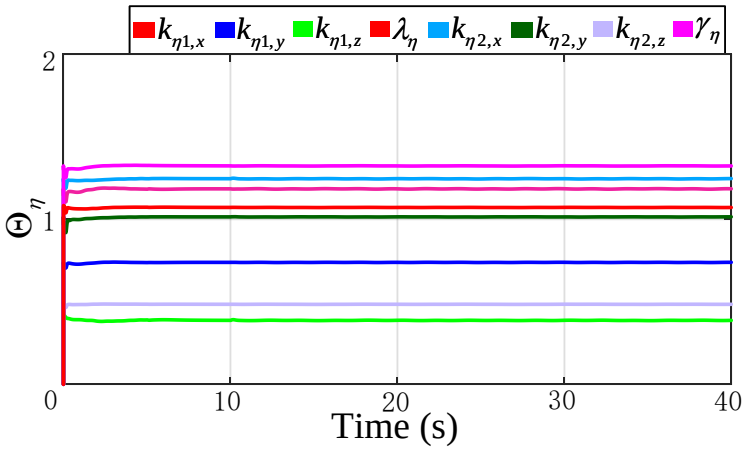


Figure 5.14: hyper-parameters tuned by DRL in the tracking simulation.

5.6 Real World Experiments

Based on the simulations in the previous section, this section presents physical experiments further to demonstrate the performance of our proposed control framework.

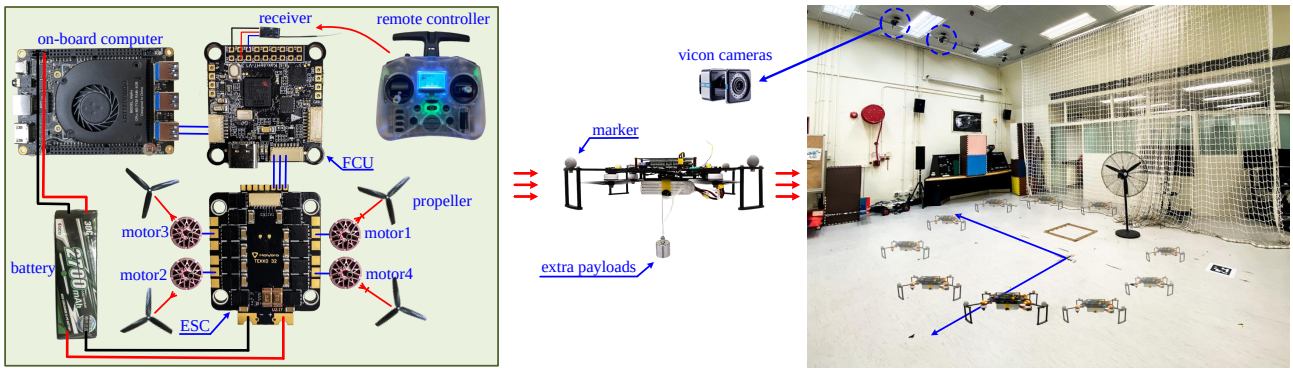


Figure 5.15: The quadrotor configuration and physical experiment environment.

Fig. 5.15 shows the quadrotor configuration in our physical experiments. The quadrotor's mass (excluding the battery) is $m = 0.850$kg. The external payload weighs (200g). The flight control unit (FCU) of the quadrotor is Holybro Kakute H7 V1.3, which integrates PX4 firewire, and the onboard computer is LattePandaAlpha 864s running Ubuntu 20.04 and ROS Noetic. The information transmission between FCU and the onboard computer is connected with a USB to TTL module and achieved

via MAVROS. FCU is responsible for attitude stabilization, and the onboard computer controls the translational subsystem. In the experimental setup, fans are positioned along the x and y axes to serve as the second disturbance source for the quadrotor, except for the variable payloads. The entire physical experiment was conducted in a VICON indoor positioning system, where the quadrotor's position was tracked using 'markers' on the body.

5.6.1 Experiment Group 1: Fixed-point control

The initial position is set at around $(-3m, 2m, 0.4m)$ and the target location is $(0m, 0m, 1m)$. Several different controllers are compared in this experiment platform. The performance is recorded in Fig. 5.16, and Fig. 5.17 is the corresponding output of the proposed fixed-time observer.

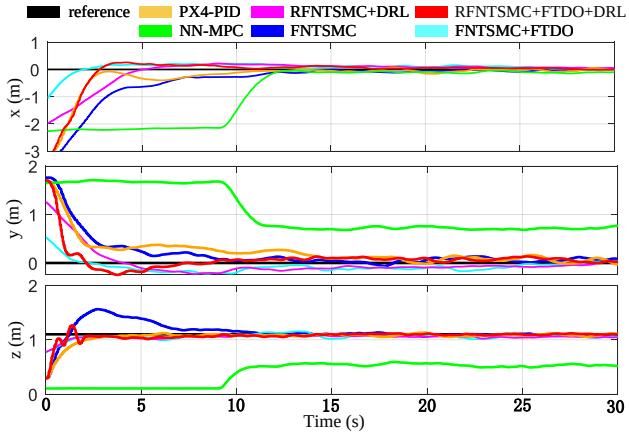


Figure 5.16: State response of the quadrotor in the fixed-point experiment.

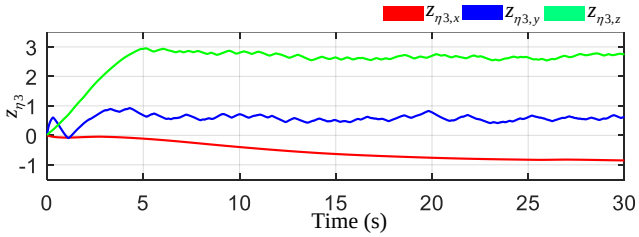


Figure 5.17: OOutputof the observer in the fixed-point experiment.

In Fig. 5.16, "PX4-PID" is the PID controller integrated into the FCU of the quadrotor. Notably, the state response in the x direction is smoother than in the other two directions. This is because the wind-exposed area of the quadrotor in the x-direction is smaller than in the y-direction, and the influence of mass variation on the response in the x-direction is relatively minor. A comprehensive comparison of the four curves shows that the performance of the proposed 'RFNTSMC + FTDO + DRL' control framework surpasses the other three. In particular, the performance of the 'NNMPC' is the least favorable, mainly due to its stringent requirement for an accurate mathematical model, which deviates significantly from the conditions of the current experiment. Correspondingly, the output of the observer in Fig. 5.17 also aligns fully with physical principles. First, since we used elastic ropes to attach the weights beneath the quadrotor, the disturbance becomes a time-varying signal, which explains why it takes approximately 5 seconds to stabilize. Second, the quadrotor has a larger wind-facing area in the y-direction than the x-direction, resulting in slight fluctuations in the observer's output for the y-direction.

The corresponding real-time gains of the proposed SMC tuned by DRL are recorded in Fig. 5.18.

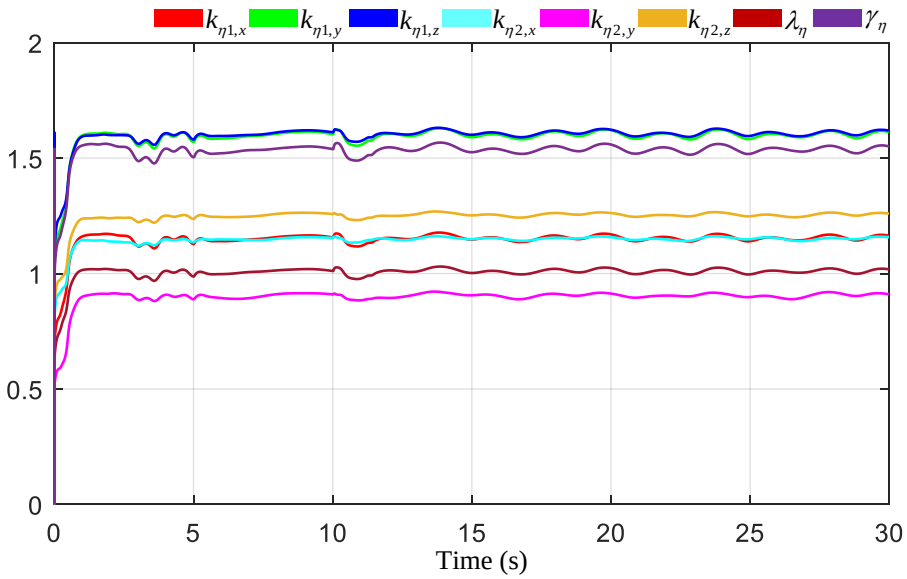


Figure 5.18: hyper-parameters tuned by DRL in the fixed-point experiment.

From the figure, we can observe that the initial values of the parameters are set to zero and subsequently adjusted by the DRL-trained optimizer. The parameters converge within one second, although small oscillations persist. This is attributed to the fixed-point control experiment setup, where two fans are positioned near the target point, and a weight is attached below the drone using a rubber band. As a result, the control parameters require real-time adjustments to counteract strong external disturbances.

5.6.2 Experiment Group 2: Trajectory tracking control

In the trajectory tracking experiment, the reference trajectory is defined as

$$\eta_d = [1.5\cos(0.25\pi t), 1.5\sin(0.25\pi t), 0.2\sin(0.2\pi t)+1]^\top. \tag{5.70}$$

The state response and the output of the disturbance observer are demonstrated in Fig. 5.19 and Fig. 5.20, respectively. Fig. 5.19 corroborates the findings in Fig. 5.16. The control performance of the 'NNMPC' exhibits the poorest state response along the z axis among the six control methods because of the requirement for an accurate mathematical model. In Fig. 5.20, the output of the observer exhibits slight differences compared to those in Fig 5.17. By comparing the reference trajectory with the observer's production, it can be observed that the fluctuation periods of the observer's outputs in the x and y directions closely match the period of the circular reference trajectory. Since the two fans are positioned at fixed locations in the experimental area, the wind disturbances acting on the quadrotor in the x and y directions exhibit periodic variations, with a period approximately matching the

reference trajectories. However, comparing the z-direction curves in Figs 5.17 and 5.20 reveals that their output magnitudes are similar but opposite in sign. This discrepancy arises because we removed the weights in the second set of experiments but increased the quadrotor's mass in the mathematical model by 0.3kg. The observer detected that the actual mass of the quadrotor was smaller than that in the model and, through estimation, provided an "equivalent disturbance" to compensate for the model in the z-direction.

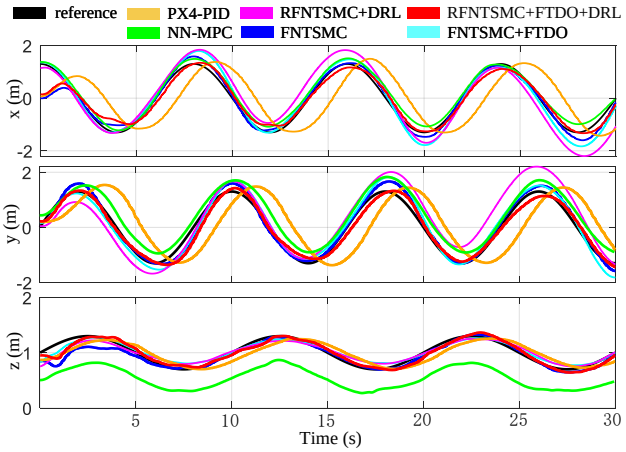


Figure 5.19: State response of the quadrotor in trajectory tracking experiment.

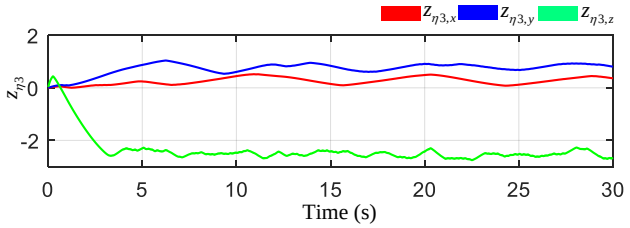


Figure 5.20: OOutputof the observer in trajectory tracking experiment.

The real-time control gains of the RFNTSMC are illustrated in Fig. 5.21. The process depicted in Fig. 5.21 exhibits similarities to those observed in the simulation and the first set of experiments, with the primary distinction being that the parameter fluctuations in Fig. 5.21 are significantly more minor than those in Fig. 5.18. This difference arises because the system operates in a comparatively more 'steady state' in the trajectory tracking experiment due to the reduced disturbance from the fans relative to the set-point experiment despite the real-time variations in the quadrotor's reference position. In the set-point experiment, the quadrotor is additionally burdened with suspended weights, and the fans are strategically positioned near the target point. As a result, the combined effects of the fans and the weights being influenced by the airflow make it challenging for the actual control error to converge precisely to zero, leading to fluctuations within a narrow range. To maintain optimal control performance, the corresponding control gains must be dynamically adjusted in real time to mitigate the strong disturbances originating from all three spatial dimensions. In the trajectory tracking experiment, the control error remained at zero without significant fluctuations because we did not use weights and assumed a more substantial mass for the quadrotor. Most of the time, the fans were positioned farther away from the quadrotor. As a result, the corresponding control gains stabilized and did not require further adjustments after convergence.

Extensive simulations and real-world experimental validations demonstrate that the proposed learning-

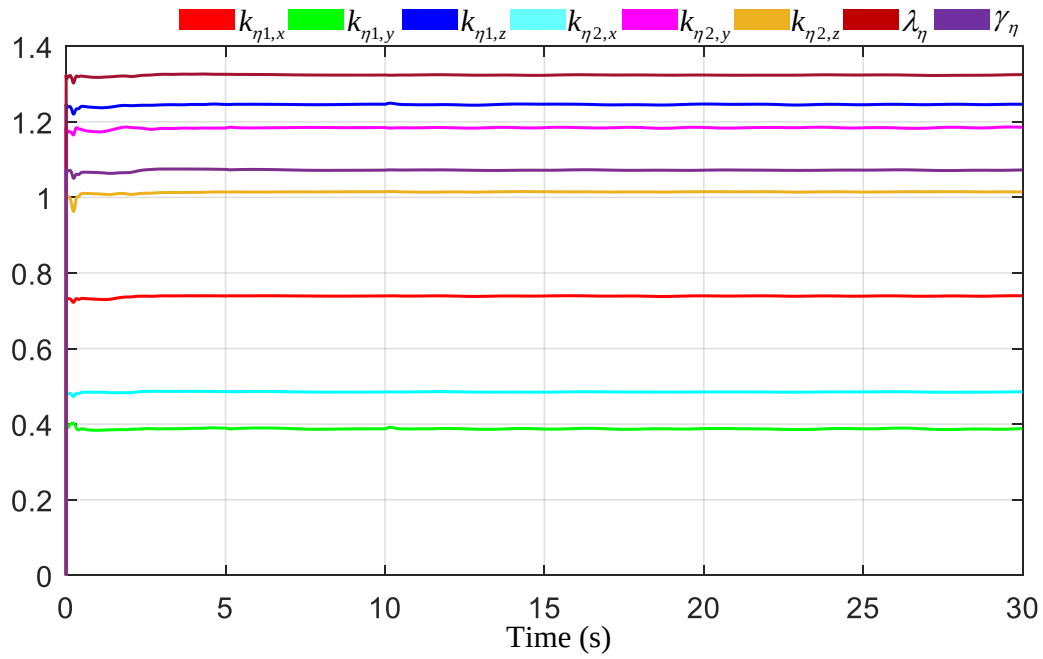


Figure 5.21: hyper-parameters tuned by DRL in the trajectory tracking experiment.

based adaptive control framework for controller gains outperforms traditional control frameworks. This superiority can be attributed to two main factors. First, based on the tracking error and its derivative, the proposed control method dynamically adjusts the control gains in real-time. This enables the system to achieve a rapid response while minimizing overshoot. After the system stabilizes, if it deviates from the equilibrium point due to strong disturbances, the control gains are adjusted again in real-time to ensure the system quickly returns to its origin. Second, the significance of the proposed method lies not only in its ability to tune controller gains in real time but also in its capacity to reduce the burden of manual parameter tuning. Although the control parameters tend to converge to constant values, as evidenced by the simulation and experimental results, these optimal values cannot be determined a priori. Therefore, the proposed method significantly alleviates the difficulty associated with manual parameter adjustment.

In addition, we aim to demonstrate the superiority of our proposed control framework more clearly. We compute the L_1 and L_2 norms of the tracking error of the drones, which is recorded in Table 5.2. It can be observed from Table 2 that the proposed control framework achieves the smallest L_1 and L_2 norms of errors in both fixed-point control and trajectory tracking under strong disturbance conditions. The effectiveness of our proposed 'RFNTSMC +FTDO + DRL control framework is thus highlighted, demonstrating superior performance in swiftly and accurately tracking the reference signal while mitigating the impact of external disturbances.

Table 5.2: Tracking errors of the quadrotor under two groups of experiments.

Experiment Group 1						
	RFNTSMC	RFNTSMC-FTDO	RFNTSMC-DRL	PX4-PID	NNMPC	proposed
$\int \|\|e_\eta\|\|_2$	73.2557	43.3220	50.1110	58.5813	358.0080	20.9075
$\int \|\|e_\eta\|\|_1$	1986.6417	1354.4367	1129.4098	1537.9079	6968.1761	892.3445
Experiment Group 2						
	RFNTSMC	RFNTSMC-FTDO	RFNTSMC-DRL	PX4-PID	NNMPC	proposed
$\int \|\|e_\eta\|\|_2$	61.8285	28.9491	32.6601	101.5166	252.1807	27.1132
$\int \|\|e_\eta\|\|_1$	2868.1603	1468.2348	1130.8006	4958.0295	4840.1805	1106.3265

5.7 Conclusion

This chapter investigates a novel robust control framework for a quadrotor with model uncertainty and external disturbances. First, two FTDOs are proposed to estimate the unknown external disturbance and model uncertainty, which can estimate the unknown components of the system in a fixed time. Then, RFNTSMCs are utilized to stabilize the closed-loop system. Furthermore, the DRL technique is incorporated to optimize the hyper-parameters in the RFNTSMCs to improve control performance further. Both simulation and physical experiments demonstrate the effectiveness and superiority of the proposed control framework. Finally, the stability of the RFNTSMCs, the FTDOs, and the DRL-based training framework are all guaranteed in the Lyapunov sense.

Chapter 6

Fixed-time Adaptive Consensus Control for Multi-Quadrotor Subject to External Disturbances Via Deep Reinforcement Learning

6.1 Research Background

In recent years, multi-quadrotor systems have garnered significant attention due to their wide range of applications across various fields. Unlike single-quadrotor systems, multi-agent systems can accomplish more complex tasks, such as cargo transportation, terrain exploration, and rescue operations.

Sliding mode control is favored for its simple structure, fast convergence, and robustness. However, adjusting the gains in sliding mode control presents a significant challenge. If the controller gains are too small, the system will exhibit slow state transitions, whereas excessively high gains can lead to substantial overshoot. Therefore, designing an appropriate sliding mode reach law and control rate is critical.

As a powerful function approximator, reinforcement learning is well-suited for optimizing sliding mode parameters. This serves as the motivation and primary contribution of this chapter. Based on the analysis above, the main contributions of this chapter can be summarized as follows:

1) A fully distributed Fast Non-singular Terminal Sliding Mode Control(ler) (FNTSMC) is proposed to address the multi-quadrotor consensus control problem, ensuring fixed-time stability in the Lyapunov sense. Compared to the works presented in [88, 89, 153], the proposed FNTSMC enables the quadrotors to accurately track reference trajectories while maintaining formation within a fixed time.

2) A fixed-time disturbance observer (FTDO) is introduced to estimate unknown external disturbances, where the estimation error converges to a neighborhood of the origin within a fixed time. Unlike the observers proposed in [129, 130], this observer does not require the disturbances to vary slowly enough for their time derivatives to be approximately zero.
3) In contrast to the methods proposed in [107, 154–156], the Deep Reinforcement Learning (DRL) technique is utilized to optimize the FNTSMCs, rather than directly replacing the controller with a single-layer linear neural network approximator. The combination of DRL and FNTSMC ensures that the system's fixed-time stability remains intact and enhances the quadrotor formation's robustness and flight performance. Finally, extensive simulations and physical experiments are conducted to verify the effectiveness and superiority of the proposed control framework.

Notations: In what follows, $\mathrm{diag}(\cdot)$ denotes the diagonal matrix, and $\circ$ represents the Hadamard product operator. $C_{(\cdot)}$, $S_{(\cdot)}$, and $T_{(\cdot)}$ denote the cosine, sine, and tangent functions, respectively. For a vector $x \in \mathbb{R}^n$, $|x| = [|x_1|, |x_2|, \cdots, |x_n|]^\top$, and $\lfloor x \rceil^\alpha = |x|^\alpha \circ \mathrm{sgn}(x)$. $\mathbf{I}_n$ represents an n-dimensional identity matrix. The vector $\eta = [x, y, z]^\top$ represents the position of the quadrotor, $\rho = [\phi, \theta, \psi]^\top$ denotes the attitude of the quadrotor, and $\omega = [p, q, r]^\top$ is the angular rate. The parameters k_t and k_r represent the translational and rotational drag coefficients. The inertia tensor matrix of the quadrotor is given by $J = \mathrm{diag}(J_{xx}, J_{yy}, J_{zz})$. The symbol $g = 9.8\mathrm{kg\,m/s^2}$ denotes the gravitational acceleration, $\tau = [\tau_x, \tau_y, \tau_z]^\top$ represents the torque, and u_f is the throttle. Specifically, variables with the subscript 'i' or 'j' represent the corresponding variables of the i-th or j-th quadrotor.

6.2 Preliminaries and Problem Formulation

6.2.1 Fundamental Mathematics

The entire quadrotor group consisting of N quadrotors can be abstracted as a graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$. The set of nodes $\mathscr{V} = \{v_1, v_2, \cdots, v_N\}$ represents the quadrotor group with v_i being the i-th quadrotor. The set of edges $\mathscr{E} = \{(v_i, v_j)\}$ denotes the connections in $\mathscr{G}$, with (v_i, v_j) representing the edge between v_i and v_j. The existence of the edge (v_i, v_j) indicates that information can be transmitted from v_i to v_j. The neighbor set of v_i is defined as $\mathscr{N}_i = \{v_j | (v_j, v_i) \in \mathscr{E}\}$ if the edge (v_j, v_i) exists. Correspondingly, the adjacency matrix is denoted by $\mathscr{A} = [a_{ij}] \in \mathbb{R}^{N\times N}$ with weights $a_{ij} = 1$ if $(v_j, v_i) \in \mathscr{E}$ and $a_{ij} = 0$ if $(v_j, v_i) \notin \mathscr{E}$. Specifically, the graph is undirected if $a_{ij} = a_{ji}, \forall i, j = 1, 2, \cdots, N$, and the graph is directed if there exists at least one tuple (i, j) such that $a_{ij} \neq a_{ji}$. The in-degree matrix of $\mathscr{G}$ is defined as $\mathscr{D} = \mathrm{diag}(d_1, d_2, \cdots, d_N)$ with $d_i = \sum_{j=1}^{N} a_{ji}, i = 1, 2, \cdots, N$. The Laplacian matrix of the graph $\mathscr{G}$ is defined as $\mathscr{L} = [l_{ij}] \in \mathbb{R}^{N\times N} = \mathscr{D} - \mathscr{A}$. For leader-follower control problems, the leader adjacency matrix (or communication matrix) is described as $\mathscr{B} = \mathrm{diag}(b_i), i = 1, 2, \cdots, N$, where $b_i = 1$ if and only if v_i can receive information from the leader node; otherwise, $b_i = 0$. The augmented graph,

which includes the leader node v_b, is denoted as $\mathscr{G}_{\mathscr{B}} = (\mathscr{V}, \mathscr{E}, v_b, b_i)$, $i = 1, 2, \cdots, N$. Without loss of generality, we make the following assumption.

Assumption 6.1 *[157,158] For the graph theory used in the study, the following standard conditions are required:*

1) *The graph $\mathscr{G}$ is undirected.*

2) *There are no self-loops in the graph $\mathscr{G}$. Namely, $a_{ii} = 0, i = 1, 2, \cdots, N$.*

3) *There exists at least one spanning tree with the leader node v_b as the root of graph $\mathscr{G}_{\mathscr{B}}$.*

Assumption 6.2 *[159] The disturbances $\Delta_{\rho,i}$, $\Delta_{\eta,i}$ acted on the N quadrotors are bounded by unknown positive constants, namely, $||\Delta_{\rho,i}|| \leq \overline{\delta}$ and $||\Delta_{\eta,i}|| \leq \overline{\Delta}$.*

Assumption 6.3 *[160] The yaw angle is bounded as $\psi_i \in [-\pi, \pi]$. To avoid singularities, the pitch and roll angles are bounded as $\phi_i, \theta_i \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$.*

For clarity and convenience, some useful lemmas are listed as follows.

Lemma 6.1 *[161] For an undirected graph, the matrix $\mathscr{H} = \mathscr{L} + \mathscr{B}$ is symmetrical positive definite if the graph $\mathscr{G}$ is connected and at least one follower can receive the leader's information.*

Lemma 6.2 *[162] For system $\dot{x} = f(x), f(0) = 0$, where $x \in \mathbb{R}^n$ is the state and $f(x) \in \mathbb{R}^n$ is the system dynamics. If there exists a continuous semi-positive definite function $V(x)$ such that*

$$\dot{V}(x) \leq -\left[a_1 V(x)^{m_1} + a_2 V(x)^{m_2}\right]^k + \vartheta,$$

where $a_1 > 0$, $a_2 > 0$, $m_1 > 0$, $m_2 > 0$, $k > 0$, $km_1 < 1$, and $km_2 > 1$. Then, the origin of the system is practically fixed-time stable. The settling time $\mathscr{T}$ is bounded by

$$\mathscr{T} \leq \mathscr{T}_{max} = \frac{1}{a_1^k \vartheta^k (1 - km_1)} + \frac{1}{a_2^k \vartheta^k (km_2 - 1)},$$

and the residual set is given by

$$\Omega = \left\{x | V(x) \leq \min\left\{a_1^{-\frac{1}{m_1}} \left(\frac{\vartheta}{1-\rho^k}\right)^{\frac{1}{m_1 k}}, a_2^{-\frac{1}{m_2}} \left(\frac{\vartheta}{1-\rho^k}\right)^{\frac{1}{m_2 k}}\right\}\right\},$$

where $\rho \in (0, 1)$.

Lemma 6.3 *[163] $\forall x_i \in \mathbb{R}$, $i = 1, 2, \cdots, n$, $0 < p \leq 1$, there is*

$$\left(\sum_{i=1}^{n} |x_i|\right)^p \leq \sum_{i=1}^{n} |x_i|^p \leq n^{1-p} \left(\sum_{i=1}^{n} |x_i|\right)^p.$$

6.2.2 System Description

According to [63], the dynamic of the i-th quadrotor can be described as

$$\begin{aligned}
\ddot{\eta}_i &= \frac{u_{f,i}}{m_i} A_i(\rho_i) - \mathbf{g} - \frac{k_t}{m_i}\dot{\eta}_i + \frac{\delta_{\eta,i}}{m_i}, \\
\dot{\omega}_i &= J_i^{-1}[-k_r\omega_i - \omega_i \times (J_i\omega_i) + \delta_{\rho,i} + \tau_i], \\
\dot{\rho}_i &= W_i(\rho_i)\omega_i,
\end{aligned} \tag{6.1}$$

where η_i, $u_{f,i}$, k_t, and m_i respectively represent the position, throttle, drag coefficient of the translational loop, and mass of the i-th quadrotor, $\mathbf{g} = [0,0,g]^\top$ is the gravitational acceleration; ω_i, J, k_r, and τ_i respectively denote the angular rate, inertia tensor matrix, drag coefficient of the rotational loop, and torque of the i-th quadrotor; and $A_i \triangleq A_i(\rho_i)$ and $W_i \triangleq W_i(\rho_i)$ are respectively defined as [164]

$$\begin{bmatrix} C_{\varphi,i}C_{\psi,i}S_{\theta,i} + S_{\varphi,i}S_{\psi,i} \\ C_{\varphi,i}S_{\psi,i}S_{\theta,i} - S_{\varphi,i}C_{\psi,i} \\ C_{\theta,i}C_{\varphi,i} \end{bmatrix}, \begin{bmatrix} 1 & S_{\varphi,i}T_{\theta,i} & C_{\varphi,i}T_{\theta,i} \\ 0 & C_{\varphi,i} & -S_{\varphi,i} \\ 0 & S_{\varphi,i}/C_{\theta,i} & C_{\varphi,i}/C_{\theta,i} \end{bmatrix},$$

and $\delta_{\eta,i}$ and $\delta_{\rho,i}$ denote the disturbances acted on the translational and rotational subsystems of the i-th quadrotor.

1) Rotational Subsystem

The tracking error $e_{\rho,i}$ and the 1^{st} and 2^{nd} order derivatives of $e_{\rho,i}$ are given by

$$\begin{aligned}
e_{\rho,i} &= \rho_i - \rho_{d,i} \\
\dot{e}_{\rho,i} &= W_i\omega_i - \dot{\rho}_{d,i}, \\
\ddot{e}_{\rho,i} &= \dot{W}_i\omega_i + W_i\dot{\omega}_i - \ddot{\rho}_{d,i},
\end{aligned} \tag{6.2}$$

where $\rho_{d,i} = \left[\phi_{d,i}, \theta_{d,i}, \psi_{d,i}\right]^\top$ is the reference attitude angle, and

$$\dot{W}_i = \begin{bmatrix} 0 & \dot{\varphi}_i T_{\theta_i} C_{\varphi_i} + \frac{\dot{\theta}_i S_{\varphi_i}}{C_{\theta_i}^2} & -\dot{\varphi}_i S_{\varphi_i} T_{\theta_i} + \frac{\dot{\theta}_i C_{\varphi_i}}{C_{\theta_i}^2} \\ 0 & -\dot{\varphi}_i S_{\varphi_i} & -\dot{\varphi}_i C_{\varphi_i} \\ 0 & \frac{\dot{\varphi}_i C_{\varphi_i} C_{\theta_i} + \dot{\theta}_i S_{\varphi_i} S_{\theta_i}}{C_{\theta_i}^2} & \frac{-\dot{\varphi}_i S_{\varphi_i} C_{\theta_i} + \dot{\theta}_i C_{\varphi_i} S_{\theta_i}}{C_{\theta_i}^2} \end{bmatrix}.$$

By defining $\Delta_{\rho,i} = J_i^{-1}\delta_{\rho,i} - \ddot{\rho}_{d,i}$, $f_{\rho,i} = -J_i^{-1}[k_r\omega_i + \omega_i \times (J_i\omega_i)]$, $A_{\rho,i} = \dot{W}_i\omega_i + W_i f_{\rho,i}$, $B_{\rho,i} = W_i J_i^{-1}$ and doing some manipulations, Eq. (6.2) can be finally simplified as

$$\ddot{e}_{\rho,i} = A_{\rho,i} + B_{\rho,i}\tau_i + \Delta_{\rho,i}. \tag{6.3}$$

Remark 6.1 *Note that the second-order derivative of $\rho_{d,i}$ is known for pure attitude control. However, in the case of position control, the desired attitude commands are generated by the translational subsystem. As a result, $\ddot{\rho}_{d,i}$ is absorbed into $\Delta_{\rho,i}$ and treated as part of the unknown disturbances.*

1) Translational Subsystem

The virtual expected acceleration of the i-th quadrotor can be defined as

$$u_{\eta,i} = [a_{x,i}, a_{y,i}, a_{z,i}]^\top, \tag{6.4}$$

yielding

$$\ddot{\eta}_i = -\frac{k_t}{m_i}\dot{\eta}_i + u_{\eta,i} + \Delta_{\eta,i}, \tag{6.5}$$

where $\Delta_{\eta,i} = \frac{u_{f,i}}{m_i}A_i + \frac{\delta_{\eta,i}}{m_i} - \mathbf{g} - u_{\eta,i}$ is the equivalent disturbance. Thereafter, it can be easily derived that

$$\begin{aligned} u_{f,i} &= m_i\sqrt{a_{x,i}^2 + a_{y,i}^2 + (a_{z,i}+g)^2}, \\ \varphi_{d,i} &= \arcsin\frac{m_i\left[a_{x,i}S_{\psi,i} - a_{y,i}C_{\psi,i}\right]}{u_{f,i}}, \\ \theta_{d,i} &= \arctan\frac{a_{x,i}C_{\psi,i} + a_{y,i}S_{\psi,i}}{a_{z,i}+g}. \end{aligned} \tag{6.6}$$

Based on the derivation above, the consensus tracking error of the i-th quadrotor can be defined as

$$e_{\eta,i} = \sum_{j=1}^{N} a_{ij}\left[(\eta_i - \nu_i) - (\eta_j - \nu_j)\right] + b_i(\eta_i - \eta_d - \nu_i), \tag{6.7}$$

where η_d is the reference geometric center of the quadrotor formation and ν_i is the offset of the i-th quadrotor to the geometric center.

For ease of theoretical derivation, a new variable can be defined as

$$\Lambda_i = b_i\eta_d + (b_i + d_i)\nu_i + \textstyle\sum_{j=1}^{N} a_{ij}(\eta_j - \nu_j). \tag{6.8}$$

Correspondingly, one obtains

$$\begin{aligned}\dot{\Lambda}_i &= b_i\dot{\eta}_d + (b_i+d_i)\dot{v}_i + \sum_{j=1}^{N} a_{ij}(\dot{\eta}_j - \dot{v}_j),\\ \ddot{\Lambda}_i &= b_i\ddot{\eta}_d + (b_i+d_i)\ddot{v}_i + \sum_{j=1}^{N} a_{ij}(\ddot{\eta}_j - \ddot{v}_j).\end{aligned} \tag{6.9}$$

Further, substituting $\ddot{\eta}_j$ into $\ddot{\Lambda}_i$ and doing some manipulations yield

$$\begin{aligned}\ddot{\Lambda}_i &= b_i\ddot{\eta}_d + (b_i+d_i)\ddot{v}_i + \sum_{j=1}^{N} a_{ij}\left(\ddot{\eta}_j - \ddot{v}_j\right)\\ &= b_i\ddot{\eta}_d + (b_i+d_i)\ddot{v}_i + \sum_{j=1}^{N} a_{ij}\left(-\frac{k_{t,j}}{m_j}\dot{\eta}_j + u_{\eta,j} + \Delta_{\eta,j} - \ddot{v}_j\right)\\ &= \Lambda_{i0} + \sum_{j=1}^{N} a_{ij}\Delta_{\eta,j},\end{aligned} \tag{6.10}$$

where $\Lambda_{i0} = b_i\ddot{\eta}_d + (b_i+d_i)\ddot{v}_i + \sum_{j=1}^{N} a_{ij}(-\frac{k_{t,j}}{m_j}\dot{\eta}_j + u_{\eta,j} - \ddot{v}_j)$ is a known variable. Then, the error dynamics of the translational loop can be given by

$$\ddot{e}_{\eta,i} = -\frac{(d_i+b_i)k_{t,i}}{m_i}\dot{\eta}_i + (b_i+d_i)u_{\eta,i} - \Lambda_{i0} + (b_i+d_i)\Delta_{\eta,i} - \sum_{j=1}^{N} a_{ij}\Delta_{\eta,j}. \tag{6.11}$$

Remark 6.2 *In the 'λ_{i0}', η_d and $\ddot{v}_i$ are manually defined. Note that the $u_{\eta,j}$ and $\ddot{v}_j$ have to be accessed from other quadrotors. However, these two items are controlled by the a_{ij}. Specifically, as defined and described in the Lapalce matrix, $a_{ij} = 0$ means these messages cannot transmit from the j-th quadrotor to the i-th quadrotor; otherwise, $a_{ij} = 1$.*

6.2.3 Problem Formulation

Based on the Assumptions 6.1- 6.3, the control objective of this chapter is formulated as follows:

Control Objective: Given a quadrotor group with N agent with undirected graph $\mathcal{G}$ and a set of reference trajectories generated by the virtual leader node with $\eta_d = [x_d, y_d, z_d]^\top$ being the reference position, ψ_d being the reference yaw angle, and $v_i = [v_{x,i}, v_{y,i}, v_{z,i}]^\top$ being the offset to the i-th quadrotor to η_d, design a type of adaptive controller such that for $i = 1, 2, \cdots, N$

$$\begin{aligned}\lim_{t\to T_\eta}\left[\eta_i(t) - (\eta_d + v_i)\right] &= 0,\\ \lim_{t\to T_\rho}\left[\psi_i(t) - \psi_d\right] &= 0,\end{aligned} \tag{6.12}$$

with T_η and T_ρ being the settling time of the translational and rotational subsystem, respectively.

6.3 Controller Design

6.3.1 Rotational subsystem stability

For simplicity, the subscript "i" in the rotational loop controller design is omitted since the quadrotors are all homogeneous, and the design of the FTDO and FNTSMC in the rotational loop does not require the information from other quadrotors. [165].

To begin with, an FTDO can be designed as

$$\begin{aligned}
\dot{z}_{\rho 1} &= \aleph_\rho m_{\rho 1} \left\lfloor \tilde{e}_\rho \right\rceil^{\alpha_1} + (1-\aleph_\rho) n_{\rho 1} \left\lfloor \tilde{e}_\rho \right\rceil^{\beta_1} + z_{\rho 2}, \\
\dot{z}_{\rho 2} &= \aleph_\rho m_{\rho 2} \left\lfloor \tilde{e}_\rho \right\rceil^{\alpha_2} + (1-\aleph_\rho) n_{\rho 2} \left\lfloor \tilde{e}_\rho \right\rceil^{\beta_2} + z_{\rho 3} + A_\rho + B_\rho \tau, \\
\dot{z}_{\rho 3} &= \aleph_\rho m_{\rho 3} \left\lfloor \tilde{e}_\rho \right\rceil^{\alpha_3} + (1-\aleph_\rho) n_{\rho 3} \left\lfloor \tilde{e}_\rho \right\rceil^{\beta_3},
\end{aligned} \tag{6.13}$$

where $z_{\rho 1}$, $z_{\rho 2}$, and $z_{\rho 3}$ are the estimations of e_ρ, $\dot{e}_\rho$, and Δ_ρ, respectively, $\tilde{e}_{\rho,i} = e_\rho - z_{\rho 1}$ is the estimation error of e_ρ. $\aleph_\rho$ is a switching parameter; α_1, α_2, α_3, β_1, β_1, and β_3 are positive constants. Specifically, $\alpha_1 = \frac{3}{4}$, $\alpha_2 = \frac{2}{4}$, $\alpha_3 = \frac{1}{4}$, $\beta_1 = \frac{5}{4}$, $\beta_2 = \frac{6}{4}$, $\beta_3 = \frac{7}{4}$, and

$$\aleph_\rho = \begin{cases} 0 & ||\tilde{e}_\rho||_2 > e_\rho^* \\ 1 & ||\tilde{e}_\rho||_2 < e_\rho^* \end{cases}$$

with e_ρ^* being the threshold of the estimation error. Apart from that, parameters $m_{\rho 1}$, $m_{\rho 2}$, $m_{\rho 3}$, $n_{\rho 1}$, $n_{\rho 2}$, and $n_{\rho 3}$ are designed such that matrices

$$\Gamma_{m,\rho} = \begin{bmatrix} -m_{\rho 1} & 1 & 0 \\ -m_{\rho 2} & 0 & 1 \\ -m_{\rho 3} & 0 & 0 \end{bmatrix} \text{ and } \Gamma_{n,\rho} = \begin{bmatrix} -n_{\rho 1} & 1 & 0 \\ -n_{\rho 2} & 0 & 1 \\ -n_{\rho 3} & 0 & 0 \end{bmatrix}$$

are Hurwitz.

Given observer (6.13) and inspired by Lemma 6.2, the equivalent disturbance of the quadrotors, say, Δ_ρ satisfying Assumption 6.2, can be estimated in a fixed-time $\mathscr{T}_{\rho 1}$, and the estimation error converges to a neighborhood of the origin Ω_ρ [166] and [167].

Remark 6.3 *redAlthough the stability of the observer can be ensured by guaranteeing the stability of matrices $\Gamma_{m,\rho}$ and $\Gamma_{n,\rho}$. In practical applications, however, it is often challenging to determine whether a third-order matrix is Hurwitz simply by inspecting its parameters. Therefore, we propose a method that integrates linear system theory to explicitly compute $m_{\rho 1}$, $m_{\rho 2}$, $m_{\rho 3}$, $n_{\rho 1}$, $n_{\rho 2}$, and $n_{\rho 3}$.*

Take $\Gamma_{m,\rho}$ as an example, solving $|\lambda I_3 - \Gamma_{m,\rho}| = 0$ yields

$$\lambda^3 + m_{\rho 1}\lambda^2 - m_{\rho 2}\lambda + m_{\rho 3} = 0. \tag{6.14}$$

Simultaneously, assuming three negative real roots of a three-order linear equation are $-\omega_1$, $-\omega_2$, and $-\omega_3$ with $\omega_1, \omega_2, \omega_3 > 0$. Then, we have

$$\begin{aligned} &\lambda^3 + m_{\rho 1}\lambda^2 - m_{\rho 2}\lambda + m_{\rho 3} \\ =&(\lambda+\omega_1)(\lambda+\omega_2)(\lambda+\omega_3) \\ =&\lambda^3 + (\omega_1+\omega_2+\omega_3)\lambda^2 + (\omega_1\omega_2+\omega_2\omega_3+\omega_1\omega_3)\lambda + \omega_1\omega_2\omega_3 = 0 \end{aligned} \tag{6.15}$$

which yields

$$\begin{aligned} m_{\rho 1} &= \omega_1 + \omega_2 + \omega_3, \\ m_{\rho 2} &= \omega_1\omega_2 + \omega_2\omega_3 + \omega_1\omega_3, \\ m_{\rho 3} &= \omega_1\omega_2\omega_3. \end{aligned} \tag{6.16}$$

Therefore, we can compute $m_{\rho 1}$, $m_{\rho 2}$, and $m_{\rho 3}$ by selecting appropriate ω_1, ω_2, and ω_3.

Larger values of $\omega_i, i = 1,2,3$ indicate that the observer possesses a higher bandwidth, which implies faster convergence of the observer. However, before convergence, the output may exhibit significant magnitudes and heightened sensitivity to noise. Conversely, smaller $\omega_i, i = 1,2,3$ corresponds to lower bandwidth, suggesting that the observer will not produce substantial overshoot before convergence and will be less sensitive to noise, albeit at the cost of slower convergence. In practical applications, it is essential to strike an appropriate balance between convergence speed and overshoot based on specific requirements.

Thereafter, a fast non-singular terminal sliding mode surface can be defined as

$$s_\rho = e_\rho + k_{\rho 1} e_\rho^{\frac{p_1}{p_2}} + k_{\rho 2}\dot{e}_\rho^{\frac{p_3}{p_4}}, \tag{6.17}$$

where $k_{\rho 1} > 0$, $k_{\rho 2} > 0$. p_1, p_2, p_3, and p_4 are all positive odd numbers satisfying

$$\frac{p_1}{p_2} > \frac{p_3}{p_4} > 1 \text{ and } 2 > \frac{p_3}{p_4} > 1.$$

Assuming there are no disturbances or uncertain terms in the system, an equivalent control law can be given by

$$\tau_{eq} = -B_\rho^{-1}\Big[A_\rho + \frac{p_4}{k_{\rho 2}p_3}\dot{e}_\rho^{2-\frac{p_3}{p_4}} \circ \Big(\mathbf{I}_3 - \frac{k_{\rho 1}p_1}{p_3}e_\rho^{\frac{p_1}{p_2}-1}\Big)\Big]. \tag{6.18}$$

Remark 6.4 *By [64, 130, 159, 168], it can be easily verified that the control matrix B_ρ is full rank and invertible.*

In addition, a switching control law is further required to maintain s_ρ at the origin when there exists disturbances or uncertainty in the system, which is given by

$$\tau_{sw} = -B_\rho^{-1}\left[z_{\rho 3} + k_{\rho 3}\,\mathrm{sgn}\,(s_\rho) + k_{\rho 4} s_\rho^{\frac{p_5}{p_6}}\right], \tag{6.19}$$

where $k_{\rho 3} > 0$, $k_{\rho 4} > 0$ are positive constants. $p_5 > p_6 > 1$ are all positive odd parameters. Then, the complete control law for the rotational loop can be designed as

$$\tau = \tau_{eq} + \tau_{sw}. \tag{6.20}$$

The following theorem can be concluded based on the analysis and derivation aforementioned.

Theorem 6.1 *For the rotational subsystem of the quadrotor (6.3) disturbed by Δ_ρ, the system is fixed-time stable with the FNTSMC designed in (6.20) and the FTDO introduced in (6.13).*

Proof 6.1 *Firstly, we must prove that the sliding mode surface converges to the origin in a fixed time.*

Choose a Lyapunov function candidate as $V_{\rho 1} = \frac{1}{2} s_\rho^\top s_\rho$. Differentiating $V_{\rho 1}$ yields

$$\dot{V}_{\rho 1} = s_\rho^\top \left\{ \dot{e}_\rho + \frac{k_{\rho 1} p_1}{p_2} e_\rho^{\frac{p_1}{p_2}-1} \circ \dot{e}^\rho + \frac{k_{\rho 2} p_3}{p_4} \dot{e}_\rho^{\frac{p_3}{p_4}-1} \circ \left[A_\rho + B_\rho\left(\tau_{eq} + \tau_{sw}\right) + \Delta_\rho\right] \right\} \tag{6.21}$$

Substituting controller (6.20) into $\dot{V}_{\rho 1}$ and doing some manipulations yield

$$\dot{V}_{\rho 1} = s_\rho^\top \left[\frac{k_{\rho 2} p_3}{p_4} \dot{e}_\rho^{\frac{p_3}{p_4}-1} \circ \left(\Delta_\rho - z_{\rho 3} - k_{\rho 3}\,\mathrm{sgn}\,(s_\rho) - k_{\rho 4} s_\rho^{\frac{\alpha_{\rho 5}}{\alpha_{\rho 6}}} \right) \right]. \tag{6.22}$$

Denote $\tilde{\Delta}_\rho = \Delta_\rho - z_{\rho 3}$ as the estimation error of Δ_ρ and $k_{\rho 0} = \frac{k_{\rho 2} p_3}{p_4} \dot{e}_\rho^{\frac{p_3}{p_4}-1}$, which yields

$$\begin{aligned} \dot{V}_{\rho 1} &= -k_{\rho 0}^\top \circ s_\rho^\top \left(k_{\rho 3}\,\mathrm{sgn}\,(s_\rho) + k_{\rho 4} s_\rho^{\frac{p_5}{p_6}} - \tilde{\Delta}_\rho \right) \\ &= -k_{\rho 4} k_{\rho 0}^\top s_\rho^{\frac{p_5+p_6}{p_6}} - k_{\rho,eq} k_{\rho 0}^\top \left| s_\rho \right| \end{aligned} \tag{6.23}$$

where $k_{\rho,eq} = k_{\rho 3} - \left\| \tilde{\Delta}_\rho \right\|_2 > 0$.

Note that all elements in $k_{\rho 0}$ are non-negative. Demoting the minimum element in $k_{\rho 0}$ as $\underline{k}_\rho$ and

using Lemma 6.3 yield

$$\begin{aligned}\dot{V}_{\rho 1} &\leq -k_{\rho 4}\underline{k}_{\rho}||s_{\rho}||_2^{\frac{p_5+p_6}{2p_6}} - k_{\rho,eq}\underline{k}_{\rho}||s_{\rho}||_2^{\frac{1}{2}} \\ &= -k_{\rho 4}\underline{k}_{\rho}2^{\frac{p_5+p_6}{2p_6}}V_{\rho 1}^{\frac{p_5+p_6}{2p_6}} - k_{\rho,eq}\underline{k}_{\rho}\sqrt{2}V_{\rho 1}^{\frac{1}{2}}.\end{aligned} \tag{6.24}$$

Using Lemma 6.2 and the fact $p_5 > p_6$ yield that s_ρ is fixed-time stable, and the settling time $\mathscr{T}_{\rho 2}$ is bounded by

$$\mathscr{T}_{\rho 2} \leq \frac{\sqrt{2}}{k_{\rho,eq}\underline{k}_{\rho}} + \frac{p_6}{k_{\rho 4}\underline{k}_{\rho}(p_5-p_6)}2^{\frac{2p_6}{p_5-p_6}}. \tag{6.25}$$

Secondly, we need to prove that e_ρ converges to the origin in a fixed time when the states are maintained on the sliding mode surface. On the sliding mode surface, there is

$$e_{\rho} + k_{\rho 1}e_{\rho}^{\frac{p_1}{p_2}} + k_{\rho 2}\dot{e}_{\rho}^{\frac{p_3}{p_4}} = 0, \tag{6.26}$$

which yields

$$\dot{e}_{\rho}^{\frac{p_3}{p_4}} = -\frac{1}{k_{\rho 2}}\left(e_{\rho} + k_{\rho 1}e_{\rho}^{\frac{p_1}{p_2}}\right). \tag{6.27}$$

Choose a Lyapunov function candidate as $V_{\rho 2} = \frac{1}{2}e_{\rho}^{\top}e_{\rho}$. Differentiating $V_{\rho 2}$ along the system trajectory and using Lemma 6.3 yield

$$\begin{aligned}\dot{V}_{\rho 2} &= -\left[\frac{1}{k_{\rho 2}}\left(e_{\rho}^{\top}\right)^{\frac{p_3}{p_4}}\left(e_{\rho} + k_{\rho 1}e_{\rho}^{\frac{p_1}{p_2}}\right)\right]^{\frac{p_4}{p_3}} \\ &\leq -\left[\frac{1}{k_{\rho 2}}||e_{\rho}||_2^{\frac{p_3+p_4}{p_4}} + \frac{k_{\rho 1}}{k_{\rho 2}}||e_{\rho}||_2^{\frac{p_1}{p_2}+\frac{p_3}{p_4}}\right]^{\frac{p_4}{p_3}} \\ &= -\left[\kappa_{\rho 1}V_{\rho 2}^{\frac{p_3+p_4}{2p_4}} + \kappa_{\rho 2}V_{\rho 2}^{\frac{1}{2}\left(\frac{p_1}{p_2}+\frac{p_3}{p_4}\right)}\right]^{\frac{p_4}{p_3}},\end{aligned} \tag{6.28}$$

where $\kappa_{\rho 1} = \frac{\sqrt{2^{(p_3+p_4)/p_4}}}{k_{\rho 2}}$ and $\kappa_{\rho 2} = \frac{k_{\rho 1}\sqrt{2^{p_1/p_2+p_3/p_4}}}{k_{\rho 2}}$.

By Lemma 6.2, one concludes e_ρ converges to the origin in a fixed time, and the settling time $\mathscr{T}_{\rho 3}$ can be bounded by

$$\mathscr{T}_{\rho 3} = \frac{2p_3}{(p_3-p_4)\kappa_{\rho 1}^{p_4/p_3}} + \frac{2p_2p_3}{(p_1p_4-p_2p_3)\kappa_{\rho 2}^{p_4/p_3}}. \tag{6.29}$$

Therefore, the convergence time of the system with external disturbance convergences within

$$\mathscr{T}_{\rho} \leq \mathscr{T}_{\rho 1} + \mathscr{T}_{\rho 2} + \mathscr{T}_{\rho 3}. \tag{6.30}$$

The proof is completed.

Remark 6.5 *In Eq. (6.17), it is obvious to conclude that*

$$\frac{p_3 + p_4}{2p_4} \cdot \frac{p_4}{p_3} = \frac{p_3 + p_4}{2p_3} \in (0,1),$$

and

$$\frac{1}{2}\left(\frac{p_1}{p_2} + \frac{p_3}{p_4}\right) \cdot \frac{p_4}{p_3} = \frac{1}{2}\left(\frac{p_1 p_4}{p_2 p_3} + 1\right) > 1$$

hold for $\frac{p_1}{p_2} > \frac{p_3}{p_4} > 1$*, satisfying the conditions required in Lemma 6.2.*

6.3.2 Translational Subsystem Stability

Similarly, the fixed-time disturbance observer can be designed as

$$\begin{aligned}
\dot{z}_{\eta 1,i} &= \aleph_{\eta,i} m_{\eta 1,i} \left\lfloor \tilde{e}_{\eta 1,i} \right\rceil^{\alpha_1} + (1 - \aleph_{\eta,i}) n_{\eta 1,i} \left\lfloor \tilde{e}_{\eta 1,i} \right\rceil^{\beta_1} + z_{\eta 2,i}, \\
\dot{z}_{\eta 2,i} &= \aleph_{\eta,i} m_{\eta 2,i} \left\lfloor \tilde{e}_{\eta 1,i} \right\rceil^{\alpha_2} + (1 - \aleph_{\eta,i}) n_{\eta 2,i} \left\lfloor \tilde{e}_{\eta 1,i} \right\rceil^{\beta_2} + z_{\eta 3,i} - \frac{k_t}{m_i}\dot{\eta}_i + u_{\eta,i}, \\
\dot{z}_{\eta 3,i} &= \aleph_{\eta,i} m_{\eta 3,i} \left\lfloor \tilde{e}_{\eta 1,i} \right\rceil^{\alpha_3} + (1 - \aleph_{\eta,i}) n_{\eta 3,i} \left\lfloor \tilde{e}_{\eta 1,i} \right\rceil^{\beta_3},
\end{aligned} \tag{6.31}$$

where $z_{\eta 1,i}$, $z_{\eta 2,i}$, and $z_{\eta 3,i}$ are the estimation of $e_{\eta,i}$,$\dot{e}_{\eta,i}$,and $\Delta_{\eta,i}$,respectively, $\tilde{e}_{\eta 1,i} = e_{\eta,i} - z_{\eta 1,i}$ is the estimation error of $e_{\eta,i}$. $\aleph_{\eta,i}$ is a switching parameter and

$$\aleph_{\eta,i} = \begin{cases} 0 & ||\tilde{e}_{\eta,i,1}||_2 > e^*_{\eta,i} \\ 1 & ||\tilde{e}_{\eta,i,1}||_2 < e^*_{\eta,i} \end{cases}$$

with $e^*_{\eta,i}$ being the threshold of the estimation error. Apart from that, hyper-parameters $m_{\eta 1,i}$, $m_{\eta 2,i}$, $m_{\eta 3,i}$, $n_{\eta 1,i}$, $n_{\eta 2,i}$, and $n_{\eta 3,i}$ are designed such that matrices

$$\Gamma_{m,\eta,i} = \begin{bmatrix} -m_{\eta 1,i} & 1 & 0 \\ -m_{\eta 2,i} & 0 & 1 \\ -m_{\eta 3,i} & 0 & 0 \end{bmatrix}, \Gamma_{n,\eta,i} = \begin{bmatrix} -n_{\eta 1,i} & 1 & 0 \\ -n_{\eta 2,i} & 0 & 1 \\ -n_{\eta 3,i} & 0 & 0 \end{bmatrix}$$

are Hurwitz. Similarly, $\Delta_{\eta,i}$ can be estimated in fixed-time $\mathscr{T}_{\eta 1,i}$, and the estimation error converges to a neighborhood of the origin $\Omega_{\eta,i}$ [166, 167].

Thereafter, a fast non-singular terminal sliding mode surface for the i-th quadrotor can be defined as

$$s_{\eta,i} = e_{\eta,i} + k_{\eta 1,i} e_{\eta,i}^{\frac{q_1}{q_2}} + k_{\eta 2,i} \dot{e}_{\eta,i}^{\frac{q_3}{q_4}}, \tag{6.32}$$

where $k_{\eta 1,i} > 0$, $k_{\eta 2,i} > 0$. q_1, q_2, q_3, and q_4 are all positive odd numbers satisfying

$$\frac{q_1}{q_2} > \frac{q_3}{q_4} > 1 \text{ and } 2 > \frac{q_3}{q_4} > 1.$$

An equivalent control law is then proposed to maintain $s_{\eta,i}$ on the sliding mode surface, which is given by

$$\begin{aligned} u_{\eta,i,eq} &= -\frac{1}{b_i + d_i} \left(u_{\eta,i,eq1} + u_{\eta,i,eq2} \right), \\ u_{\eta,i,eq1} &= -\frac{(b_1 + d_i) k_{t,i}}{m_i} \dot{\eta}_i - \Lambda_{i0}, \\ u_{\eta,i,eq2} &= \frac{q_4}{q_3 k_{\eta 2,i}} \dot{\eta}_i^{2-\frac{q_3}{q_4}} \circ \left(\mathbf{I}_3 - \frac{q_1 k_{\eta 1,i}}{q_2} e_{\eta,i}^{\frac{q_1}{q_2}-1} \right). \end{aligned} \tag{6.33}$$

Further, considering the uncertain parts and external disturbances acted on the translational subsystem, a switching control law is required, which is designed as

$$u_{\eta,i,sw} = -\frac{1}{b_i + d_i} \left[(b_i + d_i) z_{\eta 3,i} + \sum_{i=1}^{N} a_{ij} z_{\eta 3,j} + k_{\eta 3,i} \operatorname{sgn}(s_{\eta,i}) - k_{\eta 4,i} s_{\eta,i}^{\frac{q_5}{q_6}} \right]. \tag{6.34}$$

Finally, the complete control law is given by

$$u_{\eta,i} = u_{\eta,i,eq} + u_{\eta,i,sw}. \tag{6.35}$$

Similar to the rotational subsystem, a theorem is then illustrated to guarantee the stability of the translational subsystem of the entire quadrotor group.

Theorem 6.2 *For the consensus tracking error of the translational subsystems of the quadrotor formation (6.11) disturbed by Δ_i, the system is fixed-time stable with FNTSMC (6.35) and FTDO (6.31).*

Proof 6.2 *Firstly, we need to prove $s_{\eta,i}$, $i = 1, 2, \cdots, N$ converge to the origin in fixed-time. Choose a Lyapunov function candidate as $V_{\eta 1} = \frac{1}{2} \sum_{i=1}^{N} s_{\eta,i}^{\top} s_{\eta,i}$. Differentiating $V_{\eta 1}$ yields*

$$\dot{V}_{\eta 1} = \sum_{i=1}^{N} s_{\eta,i}^{\top} \left(\dot{e}_{\eta,i} + \frac{q_1 k_{\eta 1,i}}{q_2} e_{\eta,i}^{\frac{q_1}{q_2}-1} \circ \dot{e}_{\eta,i} + + \frac{q_3 k_{\eta 3,i}}{q_4} \dot{e}_{\eta,i}^{\frac{q_3}{q_4}-1} \circ \ddot{e}_{\eta,i} \right). \tag{6.36}$$

Substituting Eq. (6.11) into $\dot{V}_{\eta 1}$, using controller (6.35), and doing some manipulations yield

$$\begin{aligned}\dot{V}_{\eta 1} &= \sum_{i=1}^{N} s_{\eta,i}^{\top}\Bigg\{\dot{e}_{\eta,i} + \frac{q_1 k_{\eta 1,i}}{q_2} e_{\eta,i}^{\frac{q_1}{q_2}-1} \circ \dot{e}_{\eta,i} + \frac{q_3 k_{\eta 3,i}}{q_4}\dot{e}_{\eta,i}^{\frac{q_3}{q_4}-1}\Big[-\frac{(b_i+d_i)k_{t,i}}{m_i}\dot{\eta}_i + (b_i+d_i)u_{\eta,i} \\ &\quad - \Lambda_{i0} + (b_i+d_i)\Delta_{\eta,i} - \sum_{j=i}^{N} a_{ij}\Delta_{\eta,j}\Big]\Bigg\} \\ &= \sum_{i=1}^{N} s_{\eta,i}^{\top}\Bigg\{\frac{q_3 k_{\eta 2,i}}{q_4}\dot{e}_{\eta,i}^{\frac{q_3}{q_4}-1} \circ \Big[(b_i+d_i)\left(\Delta_{\eta,i} - z_{\eta 3,i}\right) + \sum_{j=i}^{N} a_{ij}\left(z_{\eta 3,j} - \Delta_{\eta,j}\right)\Big]\Bigg\}.\end{aligned} \tag{6.37}$$

Define the estimatation error of $\Delta_{\eta,i}$ as $\tilde{\Delta}_{\eta,i} = \Delta_{\eta,i} - z_{\eta 3,i}$ and $k_{\eta 0,i} = \frac{q_3 k_{\eta 2,i}}{q_4}\dot{e}_{\eta,i}^{\frac{q_3}{q_4}-1}$. $\dot{V}_{\eta 1}$ can be simplified as

$$\dot{V}_{\eta 1} = -\sum_{i=1}^{N} k_{\eta 0,i}^{\top} \circ s_{\eta,i}^{\top}\Bigg\{\Big[k_{\eta 3,i}\,\mathrm{sgn}\left(s_{\eta,i}\right) + k_{\eta 4,i} s_{\eta,i}^{\frac{q_5}{q_6}} - (b_i+d_i)\tilde{\Delta}_{\eta,i} - \sum_{j=i}^{N} a_{ij}\tilde{\Delta}_{\eta,j}\Big]\Bigg\}. \tag{6.38}$$

Note that all elements in $k_{\eta 0,i}$ are non-negative. Using Lemma 6.3 and denoting the minimum element in $k_{\eta 0,i}$ as $\underline{k}_{\eta,i}$ yield

$$\begin{aligned}\dot{V}_{\eta 1} &\leq -\sum_{i=1}^{N} \underline{k}_{\eta,i} s_{\eta,i}^{\top}\Bigg\{\Big[k_{\eta 3,i}\,\mathrm{sgn}\, s_{\eta,i} + k_{\eta 4,i} s_{\eta,i}^{\frac{q_5}{q_6}} - (b_i+d_i)\tilde{\Delta}_{\eta,i} - \sum_{j=i}^{N} a_{ij}\tilde{\Delta}_{\eta,j}\Big]\Bigg\} \\ &\leq -\sum_{i=1}^{N} \underline{k}_{\eta,i}\Bigg\{k_{\eta 3,i}||s_{\eta,i}||_2^{\frac{1}{2}} + k_{\eta 4,i}||s_{\eta,i}||_2^{\frac{1}{2}\left(1+\frac{q_5}{q_6}\right)}\Bigg\} \\ &\leq -\underline{k}_{\eta 3}\sum_{i=1}^{N}\sum_{j=1}^{3}\left|s_{\eta,i,j}\right| - \underline{k}_{\eta 4}\sum_{i=1}^{N}\sum_{j=1}^{3}\left|s_{\eta,i,j}\right|^{1+\frac{q_5}{q_6}},\end{aligned} \tag{6.39}$$

where $\underline{k}_{\eta 3} = \min\left(\underline{k}_{\eta,i}, k_{\eta 3,i}\right)$, $\underline{k}_{\eta 4} = \min\left(\underline{k}_{\eta,i}, k_{\eta 4,i}\right)$, and $k_{\eta,i} = k_{\eta 3,i} - (b_i+d_i)||\tilde{\Delta}_{\eta,i}||_2 - \sum_{j=1}^{N} a_{ij}||\tilde{\Delta}_{\eta,j}||_2 > 0$ for $i = 1,2,\cdots,N$. Using Lemma 6.3 again in Eq. (6.39) yields

$$\begin{aligned}\dot{V}_{\eta 1} &\leq -\underline{k}_{\eta 3}\sum_{i=1}^{N}\left(||s_{\eta,i}||_2^2\right)^{\frac{1}{2}} - \underline{k}_{\eta 4}\sum_{i=1}^{N}\left(||s_{\eta,i}||_2^2\right)^{\frac{1}{2}\left(1+\frac{q_5}{q_6}\right)} \\ &= -\underline{k}_{\eta 3}\left(\sum_{i=1}^{N}||s_{\eta,i}||_2^2\right)^{\frac{1}{2}} - \underline{k}_{\eta 4}\left(\sum_{i=1}^{N}||s_{\eta,i}||_2^2\right)^{\frac{1}{2}\left(1+\frac{q_5}{q_6}\right)} \\ &= -\kappa_{\eta 1} V_{\eta 1}^{\frac{1}{2}} - \kappa_{\eta 2} V_{\eta 1}^{\frac{1}{2}\left(1+\frac{q_5}{q_6}\right)},\end{aligned} \tag{6.40}$$

where $\kappa_{\eta 1} = \underline{k}_{\eta 3}\sqrt{2}$ and $\kappa_{\eta 2} = \underline{k}_{\eta 4}\sqrt{2^{(q_6+q_6)/q_6}}$. Using Lemma 6.2 yields that $s_{\eta,i}, i = 1,2,\cdots,N$ are

fixed-time stable, and the settling time $\mathscr{T}_{\eta 2}$ can be bounded by

$$\mathscr{T}_{\eta 2} \leq \frac{2}{\kappa_{\eta 1}} + \frac{2q_6}{\kappa_{\eta 2}(q_5 - q_6)}. \tag{6.41}$$

Secondly, similar to that of the rotational subsystem, we need to prove that the tracking errors of the quadrotor group converge to the origin in a fixed time. On the sliding mode surface, there is

$$s_{\eta,i} = e_{\eta,i} + k_{\eta 1,i} e_{\eta,i}^{\frac{q_1}{q_2}} + k_{\eta 2,i} \dot{e}_{\eta,i}^{\frac{q_3}{q_4}} = 0, \tag{6.42}$$

which yields

$$\dot{e}_{\eta,i}^{\frac{q_3}{q_4}} = -\frac{1}{k_{\eta 2,i}} \left(e_{\eta,i} + k_{\eta 1,i} e_{\eta,i}^{\frac{q_1}{q_2}} \right). \tag{6.43}$$

A Lyapunov function candidate can be defined as $V_{\eta 2} = \frac{1}{2}\sum_{i=1}^{N} e_{\eta,i}^{\top} e_{\eta,i}$. Differentiating $V_{\eta 2}$ yields

$$\begin{aligned} \dot{V}_{\eta 2} &= -\sum_{i=1}^{N} \left[\frac{1}{k_{\eta 2,i}} \left(e_{\eta,i}^{\top} \right)^{\frac{q_3}{q_4}} \left(e_{\eta,i} + k_{\eta 1,i} e_{\eta}^{\frac{p_1}{p_2}} \right) \right]^{\frac{q_4}{q_3}} \\ &= -\sum_{i=1}^{N} \left[\frac{1}{k_{\eta 2,i}} \left(||e_{\eta,i}||_2^2 \right)^{\frac{1}{2}\left(1+\frac{q_3}{q_4}\right)} + \frac{k_{\eta 1,i}}{k_{\eta 2,i}} \left(||e_{\eta,i}||_2^2 \right)^{\frac{1}{2}\left(\frac{q_3}{q_4}+\frac{q_1}{q_2}\right)} \right]^{\frac{q_4}{q_3}}. \end{aligned} \tag{6.44}$$

Using Lemma 6.3 yields

$$\begin{aligned} \dot{V}_{\eta 2} &\leq -\left\{ \sum_{i=1}^{N} \left[\frac{1}{k_{\eta 2,i}} \left(||e_{\eta,i}||_2^2 \right)^{\frac{1}{2}\left(1+\frac{q_3}{q_4}\right)} + \frac{k_{\eta 1,i}}{k_{\eta 2,i}} \left(||e_{\eta,i}||_2^2 \right)^{\frac{1}{2}\left(\frac{q_3}{q_4}+\frac{q_1}{q_2}\right)} \right] \right\}^{\frac{q_4}{q_3}} \\ &\leq -\left[\frac{1}{\overline{k}_{\eta 2}} \left(\sum_{i=1}^{N} ||e_{\eta,i}||_2^2 \right)^{\frac{1}{2}\left(1+\frac{q_3}{q_4}\right)} + \frac{\underline{k}_{\eta 1}}{\overline{k}_{\eta 2}} \left(\sum_{i=1}^{N} ||e_{\eta,i}||_2^2 \right)^{\frac{1}{2}\left(\frac{q_3}{q_4}+\frac{q_1}{q_2}\right)} \right]^{\frac{q_4}{q_3}} \\ &= -\left[\kappa_{\eta 1} V_{\eta 2}^{\frac{1}{2}\left(1+\frac{q_3}{q_4}\right)} + \kappa_{\eta 2} V_{\eta 2}^{\frac{1}{2}\left(\frac{q_3}{q_4}+\frac{q_1}{q_2}\right)} \right]^{\frac{q_4}{q_3}}, \end{aligned} \tag{6.45}$$

where $\overline{k}_{\eta 2} = \max(k_{\eta 1,i}, k_{\eta 2,i}, \cdots, k_{\eta N,i})$, $\kappa_{\eta 1} = 2^{\left(\frac{1}{2}+\frac{q_3}{2q_4}\right)}/\overline{k}_{\eta 2}$, $\underline{k}_{\eta 1} = \min(k_{\eta 1,i}, k_{\eta 2,i}, \cdots, k_{\eta N,i})$, and $\kappa_{\eta 2} = \underline{k}_{\eta 1} 2^{\left(\frac{q_3}{2q_4}+\frac{q_1}{2q_2}\right)}/\overline{k}_{\eta 2}$. Using Lemma 6.2 indicates that $e_{\eta,i}, i = 1, 2, \cdots, N$ converge to the origin in a fixed-time $\mathscr{T}_{\eta 3}$, which can be bounded by

$$\mathscr{T}_{\eta 3} \leq \frac{2q_3}{(q_3 - q_4)\kappa_{\eta 1}^{q_4/q_3}} + \frac{2q_2 q_3}{(q_1 q_4 - q_2 q_3)\kappa_{\eta 2}^{q_4/q_3}}. \tag{6.46}$$

Therefore, the translational subsystem of the quadrotor group is fixed-time stable, and the settling time can be bounded by

$$\mathscr{T}_{\eta} \leq \mathscr{T}_{\eta 1} + \mathscr{T}_{\eta 2} + \mathscr{T}_{\eta 3}. \tag{6.47}$$

The proof is completed.

6.4 DRL for Parameter Optimization

In Section 6.3, a consensus control protocol is designed for the quadrotor formation. However, tuning the hyperparameters remains a critical issue that needs to be addressed. This section uses deep reinforcement learning (DRL) as a hyperparameter optimizer for FNTSMCs to achieve improved control performance.

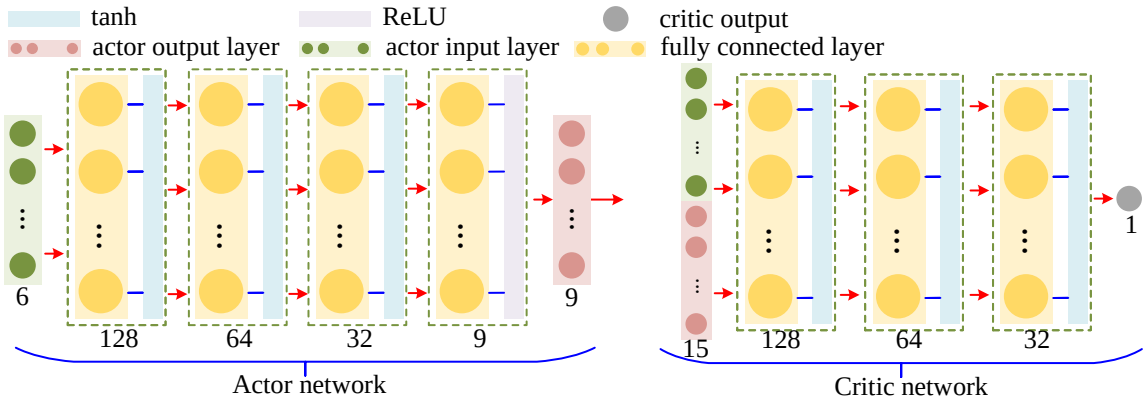


Figure 6.1: The architecture of the actor and the critic networks.

The basic Deep Reinforcement Learning (DRL) algorithm employed in this study is Proximal Policy Optimization (PPO) [146], which is an engineering approximation of Trust Region Policy Gradient (TRPO). Compared to other DRL methods, TRPO aims to achieve monotonic improvement of the policy during the iterative learning process by integrating Policy Gradient (PG) [148], Natural Policy Gradient (NPG) [149], and Conversative Policy Optimization (CPO) [147] into a unified learning framework. However, implementing pure TRPO is challenging due to the complexity of calculating the Hessian matrix in real-time. Therefore, PPO is a more popular choice for real-world applications. Specifically, an improved version of PPO, referred to as PPO with Generalized Advantage Estimation (GAE) [169], is selected as the learning framework. Table 6.1 lists some related parameters of PPO.

In Table 6.1, T_m represents the maximum simulation time of an episode, and dt is the sampling period. std_0, std_{min}, std_d, and std_{dN} are parameters for tuning the standard deviation in the Gaussian exploration policy. N_m is the maximum number of learning episodes, γ is the discount factor, K_{ep} is

Table 6.1: Some related parameters of the PPO optimizer.

Symbol	Value	Symbol	Value
T_m	10	dt	0.01
std_0	0.45	std_{min}	0.2
std_d	0.05	std_{dN}	250
γ	0.99	K_{ep}	10
b_s	$T_m/dt*2$	a_{lr}	10^{-4}
c_{lr}	10^{-3}	c_{en}	0.01
λ	0.95	c_{min}, c_{max}	0.8, 1.2

the number of times the neural network (NN) gradient descends in one learning iteration, and b_s is the buffer size. a_{lr} and c_{lr} denote the learning rates of the actor and critic networks, respectively. c_{en}, λ, c_{min}, and c_{max} are parameters used in the GAE technique, as referenced in [169]. Additionally, $N_m = 1000 \cdot \frac{(std_0 - std_{min})}{std_d} + 1000$ is the maximum training episode. The pseudocode for PPO with GAE is illustrated in Algorithm 8. The fundamentals of the PPO algorithm can be found in [146] and various highly-stared GitHub repositories.

The architecture of the actor and the critic networks are illustrated in Fig. 6.1. The structure of the actor-network is a five-layer fully connected NN whose dimensions of the input and output of the actor-network are 6 and 9, respectively. The input of the actor NN represents the quadrotor's tracking error, and the actor NN's production is the learned hyperparameters of the FNTSMC. Note that the activation function of the actor NN's output layer is ReLU rather than tanh because the hyper-parameters of the FNTSMC should be positive. The input dimension of the critic NN is 15, which is the sum of the dimensions of the input and output of actor NN. The output of the critic NN is a scalar representing the state-action value function of the current input, which serves as an indicator to evaluate the quality of the selected hyper-parameters. The diagram of the learning-based control framework for a single quadrotor is demonstrated in Fig. 6.2. In Fig. 6.2, the rotational and translational loop controls are coupled and connected with the desired roll and pitch angles, denoted as ϕ_d and θ_d. The hyperparameters of the FNTSMCs for both loops are optimized by deep reinforcement learning (DRL) simultaneously and separately.

6.4.1 Rotational Subsystem Parameter Optimizer Training

The controller for the rotational subsystem is utilized in Eq. (6.20) and is tuned by $k_{\rho 1,i}$, $k_{\rho 2,i}$, $k_{\rho 3,i}$, $k_{\rho 4,i}$, and $p_1 \sim p_6$. First, to ensure the rapid convergence of the training process, we select $p_1 = 9$, $p_2 = 7$, $p_3 = 5$, $p_4 = 3$, $p_5 = 7$, and $p_6 = 5$. Second, to reduce the gap between numerical simulations and real-world experiments, we opted not to rely solely on the results learned by DRL. Specifically, the regulation of $k_{\rho 1,i}$, $k_{\rho 2,i}$, and $k_{\rho 4,i}$ is assigned to DRL, while $k_{\rho 3,i}$ is retained further to enhance the robustness of the controller during real-world experiments.

Algorithm 8 PPO with GAE

Input Episode index $e_p = 0$, maximum learning episode N_e, initial critic net $\mathscr{C}()$, initial actor net $\mathscr{A}()$.

while $e_p < N_e$ **do**
- Collect time-sequential data buffer and compute the log-probability of the actor: $\mathscr{A}_{lg}$.
- At a certain timestep, calculate the value function for state s at the current timestep and s' at the next timestep.
$$V = \mathscr{C}(s), V' = \mathscr{C}(s')$$
- Calculate the GAE and advantage function of the actor: V_{gae}, V_{adv}
- Calculate the target value function:
$$V_{tar} = V_{adv} + V$$
- Calculate the distribution of actor $\mathscr{N}_{ac}$, the corresponding entropy of the distribution $\mathscr{N}_{en}$, and the corresponding log-probability of the distribution $\mathscr{N}_{lg}$
- $\mathscr{N}_a = e^{\mathscr{N}_{lg} - \mathscr{A}_{lg}}$
- Calculate the surrogate objective
$$s_1 = \mathscr{N}_a * V_{adv},$$
$$s_2 = V_{adv} * [\mathscr{N}_a.clip(c_{min}, c_{max})]$$
- Calculate loss function for $\mathscr{A}()$.
$$\mathscr{L}_a = -min(s_1, s_2) - c_{en} * \mathscr{N}_{en}$$
- Update actor net weights using backpropagation
$$\mathscr{A}().learn().$$
- Calculate loss function for $\mathscr{C}()$
$$\mathscr{L}_c = \frac{1}{2}(V_{tar} - V)^2.$$
- Update critic net weights using backpropagation
$$\mathscr{C}().learn().$$
- $e_p += 1$.

return $\mathscr{A}()$

Therefore, the input and output of the optimizer for the rotational subsystem are respectively defined as

$$\begin{aligned} \chi_{\rho,i} &= \left[e_{\rho,i}^\top, \dot{e}_{\rho,i}^\top\right]^\top \in \mathbb{R}^6, \\ \Theta_{\rho,i} &= \left[k_{\rho 1,i,x}, k_{\rho 1,i,y}, k_{\rho 1,i,z}, k_{\rho 2,i,x}, k_{\rho 2,i,y}, k_{\rho 2,i,z}, k_{\rho 3,i,x}, k_{\rho 3,i,y}, k_{\rho 3,i,z}\right]^\top \in \mathbb{R}^9. \end{aligned} \tag{6.48}$$

The reward function is defined as

$$J_{\rho,i}(t) = \int_{t=0}^{\infty} e^{-\gamma_\rho (s-t)} \left(\chi_{\rho,i}^\top Q_{\rho,i} \chi_{\rho,i} + \tau^\top R_{\rho,i} \tau\right) ds, \tag{6.49}$$

where $Q_{\rho,i} = \text{diag}(Q_{e_{\rho,i}}, Q_{\dot{e}_{\rho,i}})$ with $Q_{e_{\rho,i}} = \mathbf{I}_3$, $Q_{\dot{e}_{\rho,i}} = 0.01\mathbf{I}_3$, $R_{\rho,i} = 0.01\mathbf{I}_3$, and $\gamma_\rho = 0.99$ is the discount factor.

The curves for reward during the training process are shown in Fig. 6.3, where a multi-stage train-

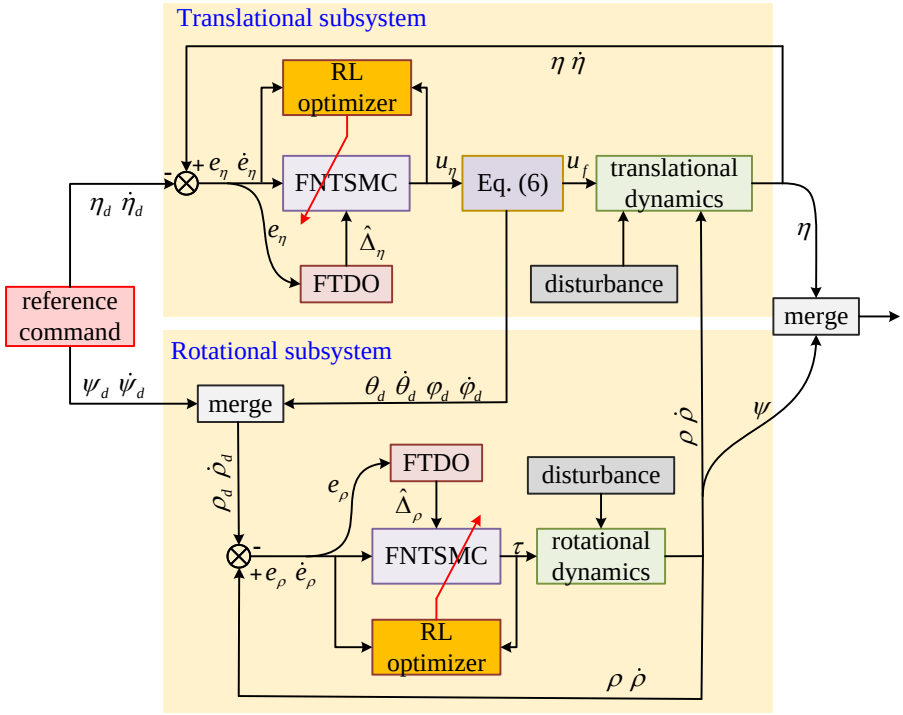


Figure 6.2: Diagram of the learning-based control framework

ing technique [151] is employed to simultaneously accelerate the training process and improve the robustness of the trained neural network (NN). Specifically, during the first training stage, the control performance fluctuates significantly because the NNs have not fully converged. Therefore, in the subsequent three training stages, the initial policies are set to the results from the corresponding previous training stage. Additionally, lower learning rates for the NNs are chosen to reduce fluctuations and enhance the robustness of the learned optimizer.

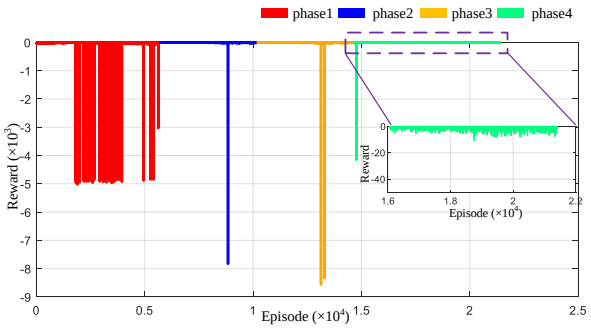


Figure 6.3: Reward of the training process of the rotational subsystem.

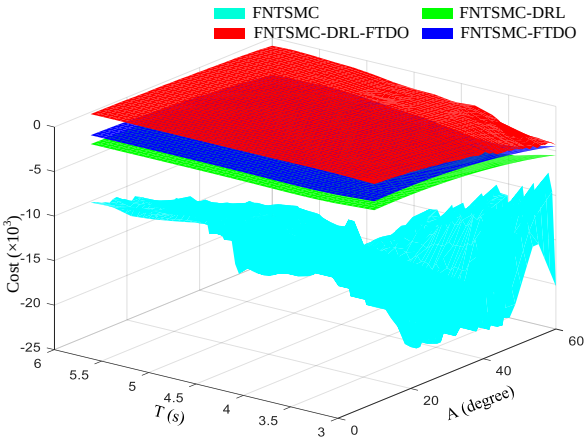


Figure 6.4: Comparative cost surface under different control frameworks and initial conditions for the rotational subsystem.

Moreover, we further collected the costs associated with the rotational subsystem control under various initial conditions and different control frameworks to demonstrate the superiority of our proposed control framework preliminarily. This is shown in Fig. 6.4, which indicates that the proposed FNTSMC-DRL-FTDO control framework outperforms the controllers that do not utilize the DRL technique. Notably, the performance of the pure FNTSMC (without FTDO or DRL) exhibits significant fluctuations due to the strong external disturbances applied to the system, which indirectly highlights the robustness of the proposed control framework.

Remark 6.6 *The neural networks are trained in a single-agent interactive environment to obtain a DRL-based optimizer more quickly, as all quadrotors are homogeneous. The well-trained NN-based optimizer is integrated into the FNTSMC of each quadrotor during numerous validations and physical experiments to tune the hyper-parameters of the FNTSMCs in real time.*

6.4.2 Translational Subsystem Parameter Optimizer Training

Similarly, we set $q_1 = 9$, $q_2 = 7$, $q_3 = 5$, $q_4 = 3$, $q_5 = 7$, and $q_6 = 5$. and $k_{\eta 3,i}$ is retained out of the DRL-based optimization framework. The input and output of the optimizer of the translational subsystem are respectively defined as

$$\begin{aligned} \chi_{\eta,i} &= \left[e_{\eta,i}^{\top}, \dot{e}_{\eta,i}^{\top}\right]^{\top} \in \mathbb{R}^6, \\ \Theta_{\eta,i} &= \left[k_{\eta 1,i,x}, k_{\eta 1,i,y}, k_{\eta 1,i,z}, k_{\eta 2,i,x}, k_{\eta 2,i,y}, k_{\eta 2,i,z}, k_{\eta 3,i,x}, k_{\eta 3,i,y}, k_{\eta 3,i,z}\right]^{\top} \in \mathbb{R}^9. \end{aligned} \tag{6.50}$$

The reward function is defined as

$$J_{\eta,i}(t) = \int_{t=0}^{\infty} e^{-\gamma_\eta (s-t)} \left(\chi_{\eta,i}^{\top} Q_{\eta,i} \chi_{\eta,i} + u_{\eta,i}^{\top} R_{\eta,i} u_{\eta,i}\right) dt, \tag{6.51}$$

where $Q_{\eta,i} = \mathrm{diag}(Q_{e_{\eta,i}}, Q_{\dot{e}_{\eta,i}})$ with $Q_{e_{\eta,i}} = \mathbf{I}_3$, $Q_{\dot{e}_{\eta,i}} = 0.1\mathbf{I}_3$ and $R_{\eta,i} = 0.01\mathbf{I}_3$, and $\gamma_\eta = 0.99$ is the discount factor.

The curves for the reward during the training process are recorded in Fig. 6.5. The trend of the reward curve is very similar to that of the rotational subsystem. The cost surfaces of the translational subsystem control under different initial conditions and various control frameworks are presented in Fig. 6.6, which reveals that the patterns of translational control performance are fundamentally similar to those of the rotational loop. The performance of the proposed FNTSMC-DRL-FTDO control framework outperforms the other three methods. Additionally, the cost of the FNTSMC (represented by the green surface) is lower than that of FNTSMC-DRL (represented by the cyan surface), further indicating the superiority of the DRL technique.

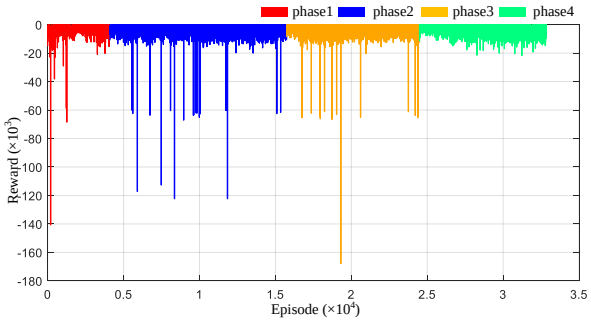


Figure 6.5: Reward of the training process of the translational subsystem.

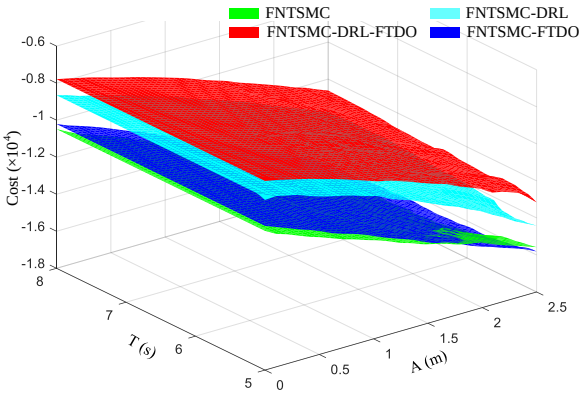


Figure 6.6: Comparative cost surface under different control frameworks and initial conditions for the translational subsystem.

Remark 6.7 *Adaptively tuning the gains of a controller using DRL offers significant advantages over directly learning the controller itself [107, 154–156]. The process of optimizing parameters through reinforcement learning does not compromise the structure of the controller or the stability of the closed-loop system. This significantly enhances the system's robustness during the learning process, thereby accelerating the system's training speed. Extensive experimental validation demonstrates that the trained network can update parameters at a frequency exceeding* 200Hz *in practical experiments, which will not compromise the real-time performance of the system.*

6.5 Simulation

6.5.1 Simulation Group 1

The topological graph is shown in Fig. 6.7. Correspondingly, the adjacent matrix $\mathscr{A}_{s1}$, in-degree matrix $\mathscr{D}_{s1}$, communication matrix $\mathscr{B}_{s1}$, and Laplacian matrix $\mathscr{L}_{s1}$ are respectively defined as $\mathscr{A}_{s1}$=[0, 1, 1, 1, 1, 1; 1, 0, 0, 0, 0, 0; 1, 0, 0, 0, 0, 0; 1, 0, 0, 0, 0, 0; 1, 0, 0, 0, 0, 0; 1, 0, 0, 0, 0, 0], $\mathscr{D}_{s1} = \text{diag}(5,1,1,1,1,1)$, $\mathscr{B}_{s1} = \text{diag}(1,0,0,0,0,0)$, and $\mathscr{L}_{s1} = \mathscr{D}_{s1} - \mathscr{A}_{s1}$.

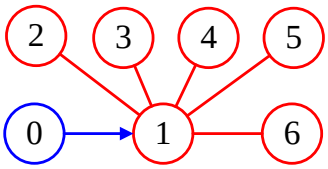


Figure 6.7: Topological graph of simulation group 1.

The equation of the geometric center $\mathscr{O}_d$ is defined as

$$x_d = r_d \sin(0.2\pi t) + 2,\ y_d = r_d \cos(0.2\pi t) + 3,\ z_d = \sin(0.4\pi t) + 2 \tag{6.52}$$

with $r_d = 5m$. The offsets of each quadrotor to $\mathcal{O}_d$, denoted by $\nu_i, i = 1,2,\cdots,6$, are defined as

$$\nu_{i,x} = r_\nu \sin(0.2\pi t + \phi_{x,i}),\ \nu_{i,y} = r_\nu \sin(0.2\pi t + \phi_{y,i}),\ \nu_{i,z} = 0, \tag{6.53}$$

where $\phi_{x,i} = \frac{\pi}{2} + (i-1)\frac{\pi}{3}$, $\phi_{y,i} = (i-1)\frac{\pi}{3}$ and $r_\nu = 2m$.

Fig. 6.8 illustrates the 2-norm of the tracking errors under six different control frameworks, clearly demonstrating that the purple curve exhibits the best performance. The red curve also converges to the origin eventually; however, fluctuations are pretty noticeable at the beginning due to the absence of a parameter adaptive adjustment mechanism. Other controllers can ensure system stability but consistently exhibit relatively large tracking errors.

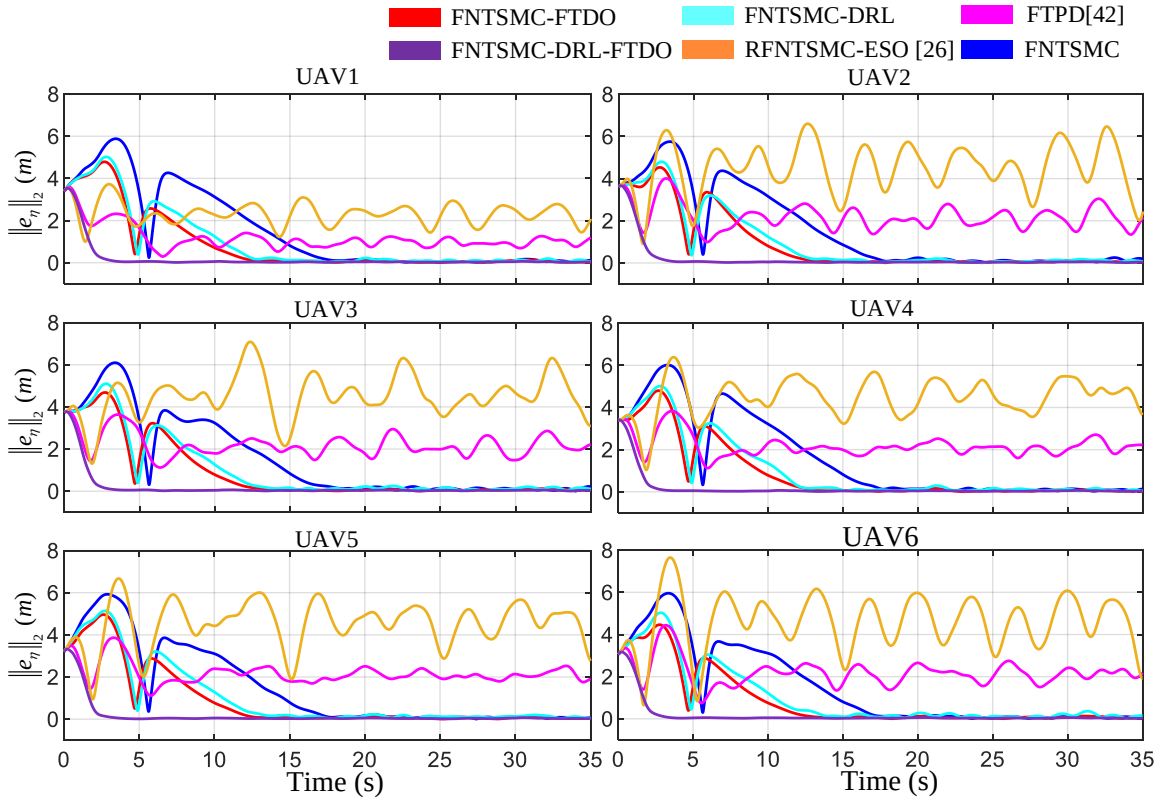


Figure 6.8: $||e_{\eta,i}||_2$ of each quadrotor under different control frameworks.

We further conduct a simulation in which the reference trajectories along the XOY plane remain unchanged while the altitude is set to different constant values. The corresponding position response of the quadrotor group under the control framework "FNTSMC+RL+FTDO" is demonstrated in Fig. 6.9, where the reference trajectories are the superposition of two circles with different radii.

Correspondingly, the output of the FTDOs is illustrated in Fig. 6.10. Fig. 6.10 demonstrates that the FTDO can accurately track the external disturbances in fixed time (about 1 second). The disturbances set in the simulation consist of combinations of sine functions with varying amplitudes, phases, and

periods. Therefore, the FTDO can precisely estimate the unknown items affecting the quadrotors.

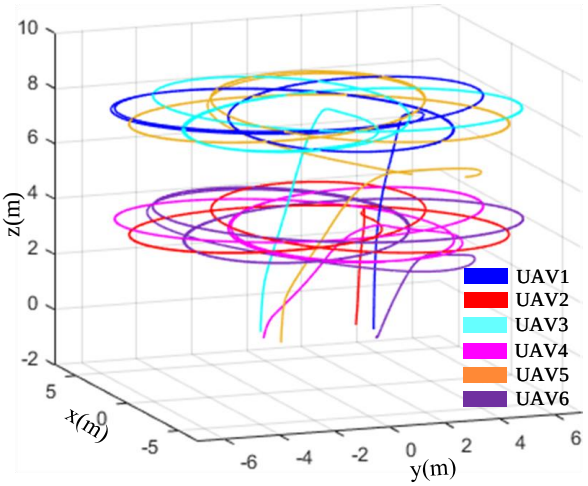


Figure 6.9: Graphic demonstration of the quadrotor formation in a 3D view.

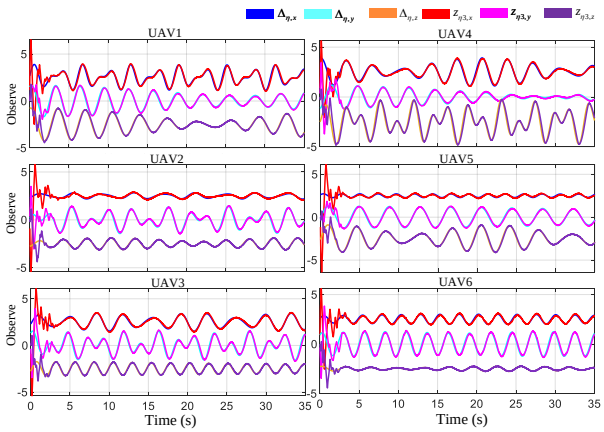


Figure 6.10: Output of the observers.

6.5.2 Simulation Group 2

We further test our algorithm with a more complicated topological graph and a more aggressive reference trajectory. The topological graph of simulation group 2 is shown in Fig. 6.11. Correspondingly, the adjacency matrix $\mathscr{A}_{s2}$, in-degree matrix $\mathscr{D}_{s2}$, communication matrix $\mathscr{B}_{s2}$, and Laplacian matrix $\mathscr{L}_{s2}$ are defined as follows: $\mathscr{A}_{s2}$=[0, 1, 0, 1, 0, 0; 1, 0, 1, 0, 0, 0; 0, 1, 0, 1, 0, 0; 1, 0, 1, 0, 1, 0; 0, 0, 0, 1, 0, 1; 0, 0, 0, 1, 0, 1; 0, 0, 0, 0, 1, 0], $\mathscr{D}_{s2} = \text{diag}(2,2,2,3,2,1)$, $\mathscr{B}_{s2} = \text{diag}(1,0,0,0,0,0)$, and $\mathscr{L}_{s2} = \mathscr{D}_{s2} - \mathscr{A}_{s2}$.

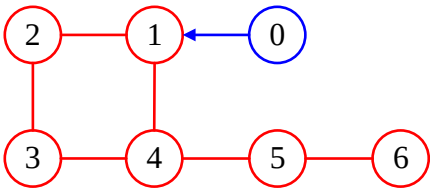


Figure 6.11: Topological graph of simulation group 2.

The equation of the geometric center $\mathscr{O}_d$ is defined as

$$x_d = r_d \cos(0.2\pi t) + 2,\ y_d = r_d \sin(0.4\pi t) + 3,\ z_d = \sin(0.4\pi t) + 2 \tag{6.54}$$

with $r_d = 5m$. The offsets of each quadrotor to $\mathscr{O}_d$, denoted by $v_i, i = 1,2,\cdots,6$, are defined as

$$\begin{aligned} v_1 &= [r_v, 0, 0]^\top, & v_2 &= [r_v \sin(\theta_0), r_v \cos(\theta_0), 0]^\top, \\ v_3 &= [-r_v \sin(\theta_0), r_v \cos(\theta_0), 0]^\top, & v_4 &= [-r_v, 0, 0]^\top, \\ v_5 &= [-r_v \sin(\theta_0), -r_v \cos(\theta_0), 0]^\top, & v_6 &= [r_v \sin(\theta_0), -r_v \cos(\theta_0), 0]^\top, \end{aligned} \tag{6.55}$$

where $\theta_0 = 60°$ and $r_v = 2$m. Fig. 6.12 illustrates the 2-norm of the tracking errors across different

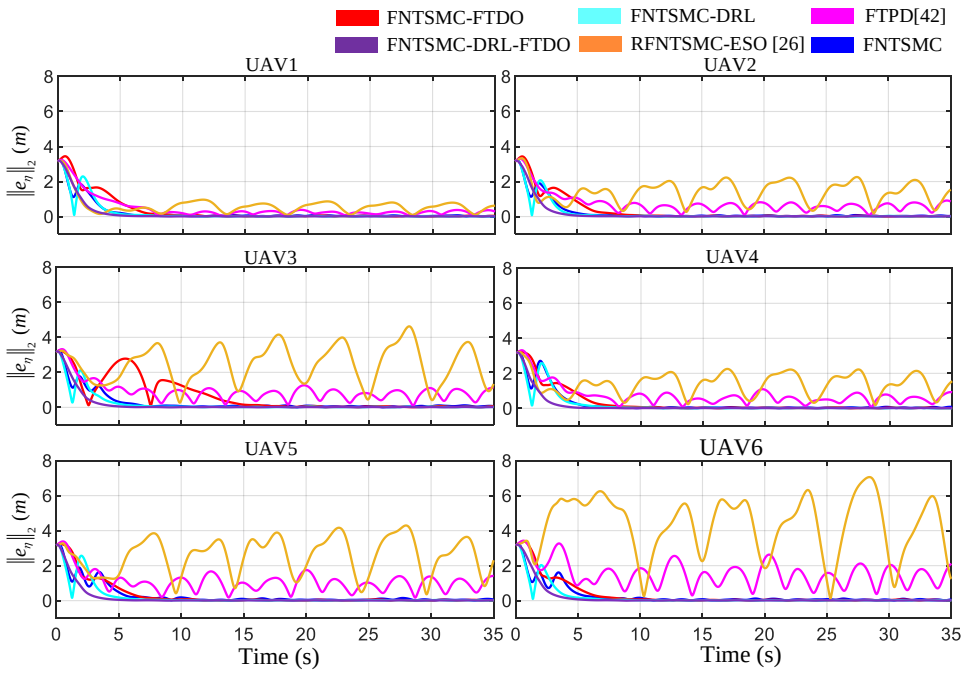


Figure 6.12: $||e_\eta||_2$ of each quadrotor under different control frameworks.

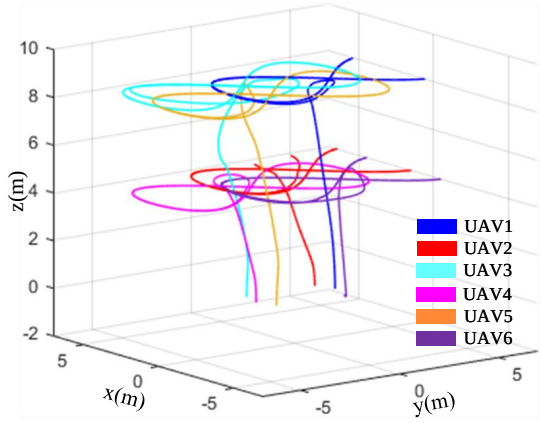


Figure 6.13: Graphic demonstration of the quadrotor formation in a 3D view.

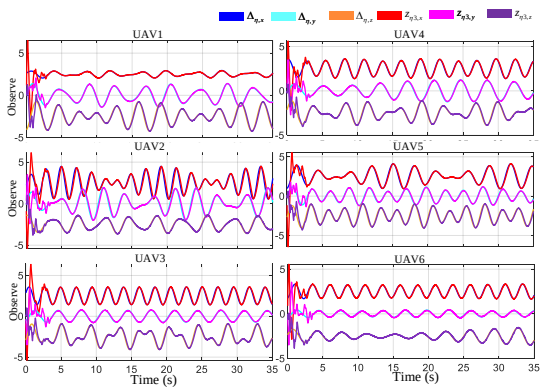


Figure 6.14: Output of the observers.

control frameworks. Correspondingly, the three-dimensional position response of the quadrotor group

is presented in Fig. 6.13, while the output of the FTDOs is recorded in Fig. 6.14. Similarly, in Figs.6.8 and 6.12, the control performances of FNTSMC+RL" (the cyan curves) and FNTSMC+FTDO" (the red curves) are slightly better than that of the traditional FNTSMC (the blue curves), although the errors remain relatively large. In contrast, the tracking error converges rapidly to zero under the control of the FNTSMC+RL+FTDO" framework for both the double-circle" and "∞-shaped" reference trajectories.

6.6 Physical Experiments

This section presents real-world experiments for further validation. The quadrotors used in the experiment are depicted in Fig. 6.15. Each quadrotor has a mass of 0.722kg and a wheelbase of 250mm. We utilize four quadrotors to demonstrate the consensus formation control. To highlight the robustness of our proposed control method under strong disturbances, we introduce two types of disturbances in the experimental environment: high-speed rotating fans and weights suspended by elastic ropes. Additionally, the aerodynamic interactions between different aircraft in the flight formation and discrepancies between the UAV's center of mass and its centroid further complicate quadrotor flight control.

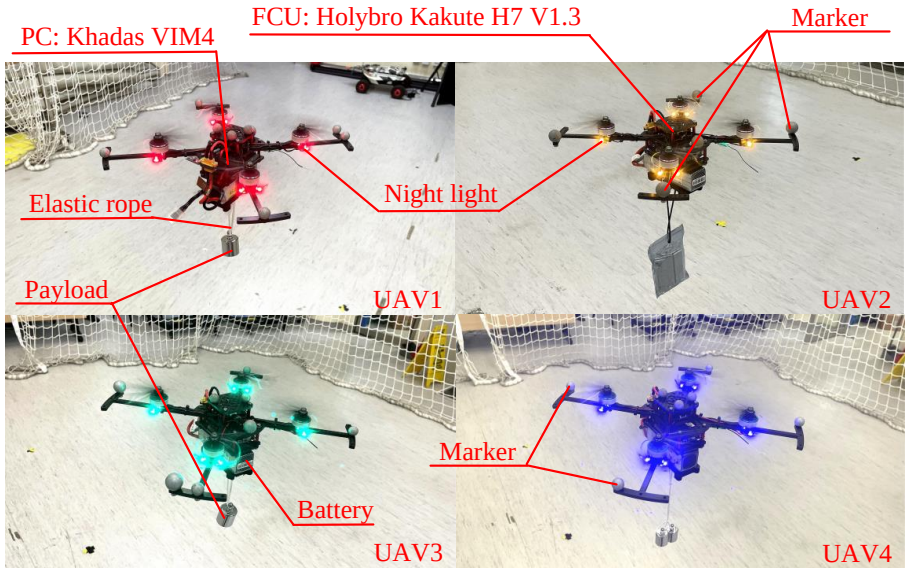


Figure 6.15: The quadrotors used in the experiment.

The complete hardware configuration for the experiment is illustrated in Fig. 6.16. In this figure, four self-designed quadrotors, each featuring different night light colors, form the quadrotor formation. The Flight Control Unit (FCU) of the quadrotors is the Holybro Kakute H7 v1.3, which inherits the open-source PX4-Autopilot firmware. The onboard computer is LattePanda Alpha 864s running Ubuntu 20.04 with ROS Noetic. The FCU is connected to the onboard computer via a USB to

TTL module and utilizes the MAVROS communication protocol for real-time data transmission. The quadrotors are localized by a VICON indoor positioning system equipped with 14 high-resolution optical cameras, which provide precise position and attitude feedback to the quadrotors at a frequency exceeding 200Hz. Velocity feedback is obtained by fusing data from the VICON system and the IMU using a Kalman filter. The ground station computer is used solely to monitor the states of the quadrotors and collect necessary data after the experiments. It does not send any control-related commands to the quadrotor formation, as the control protocol proposed in this chapter is fully distributed and decentralized.

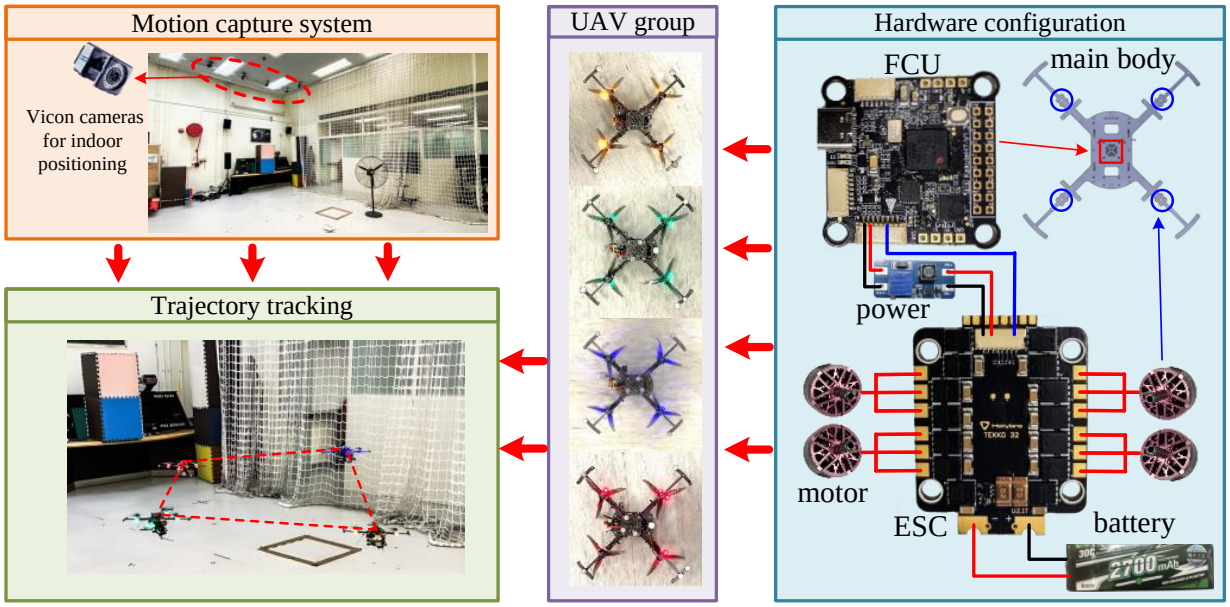


Figure 6.16: The entire experiment configuration.

6.6.1 Experiment Group 1

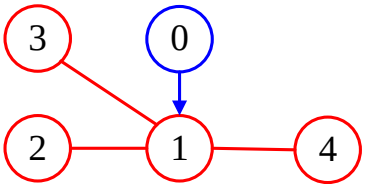


Figure 6.17: Topological graph in Experiment Group 1.

The topological graph of the quadrotor group is shown in Fig. 6.17. The adjacent matrix $\mathscr{A}_{p1}$, in-degree matrix $\mathscr{D}_{p1}$, communication matrix $\mathscr{B}_{p1}$, and Laplacian matrix $\mathscr{L}_{p1}$ are respectively defined as $\mathscr{A}_{p1} = [0,1,1,1;1,0,0,0;1,0,0,0;1,0,0,0]$, $\mathscr{D}_{p1} = \text{diag}(3,1,1,1)$, $\mathscr{B}_{p1} = \text{diag}(1,0,0,0)$, and $\mathscr{L}_{p1} = \mathscr{D}_{p1} - \mathscr{B}_{p1}$. The geometric center $\mathscr{O}_d = [0,0.2,1.5]^\top$ remains unchanged. The offsets of each

quadrotor to $\mathcal{O}_d$, denoted by $v_i, i = 1,2,3,4$, are defined as

$$
\begin{aligned}
v_1 &= [1.3\cos(0.4\pi t), 1.3\sin(0.4\pi t), 0.3\sin(0.2\pi t) + 0.5]^\top, \\
v_2 &= [-1.3\sin(0.4\pi t), 1.3\cos(0.4\pi t), -0.5]^\top, \\
v_3 &= [-1.3\cos(0.4\pi t), -1.3\sin(0.4\pi t), 0.3\sin(0.2\pi t) + 0.5]^\top, \\
v_4 &= [1.3\sin(0.4\pi t), -1.3\cos(0.4\pi t), -0.5]^\top.
\end{aligned}
$$

Fig. 6.18 records the real-time control gains $k_{\eta 1}$, $k_{\eta 2}$, and $k_{\eta 4}$ tuned by DRL. Fig. 6.19 illustrates the 2-norm of the consensus tracking errors under different control frameworks, while the position response in a 3D view is depicted in Fig. 6.20. Firstly, the response curve of the PID controller is

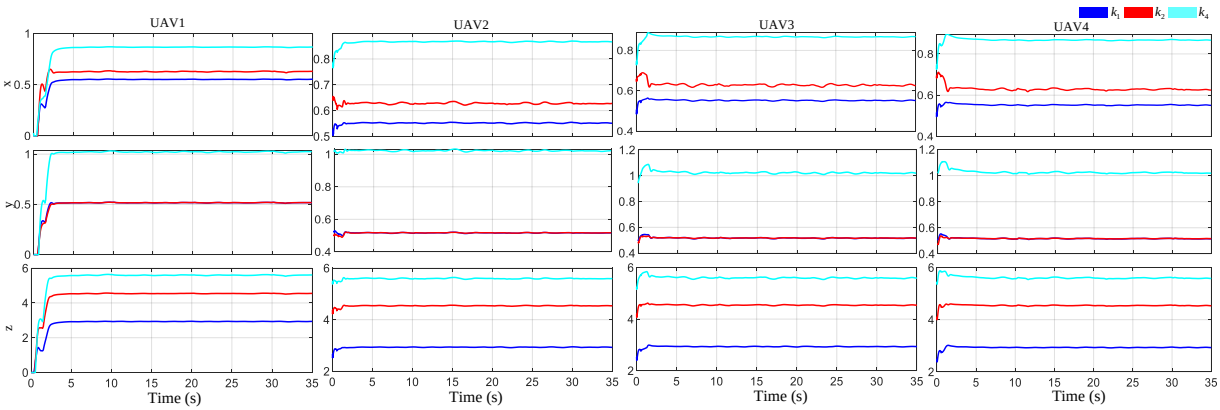


Figure 6.18: Hyperparameters $k_{\eta 1}$, $k_{\eta 2}$, and $k_{\eta 4}$ tuned by DRL in experiment group 1.

notably smooth; however, it exhibits relatively large steady-state errors and phase delays. Secondly, the performance of the FNTSMC-DRL-FTDO method surpasses that of all other controllers. With enhancements in the FTDO and the DRL parameter optimizer, the proposed method stands out for its superior control performance under strong disturbances and uncertainties. Fig. 6.18 reveals that the control gains are time-variant rather than some pre-tuned constants. The initial values of the gains are all set to be 0 and converge to some constants after less than 2 seconds. Eq. (6.50) indicates that the main reason for the convergence of the gains is the convergence of the consensus tracking error e_η and $\dot{e}_\eta$.

Fig. 6.21 illustrates the corresponding output of the FTDO. As shown in the figure, the observed outputs in the x and y directions exhibit noticeable fluctuations, with the oscillation period closely matching that of the reference trajectory for the quadrotor formation. This periodic behavior is attributed to the influence of the fans, which act as an external disturbance on the quadrotors. Additionally, the FTDO output in the z direction corresponds to an equivalent weight that closely matches the weight of the masses suspended below the quadrotor, demonstrating the effectiveness of the FTDO.

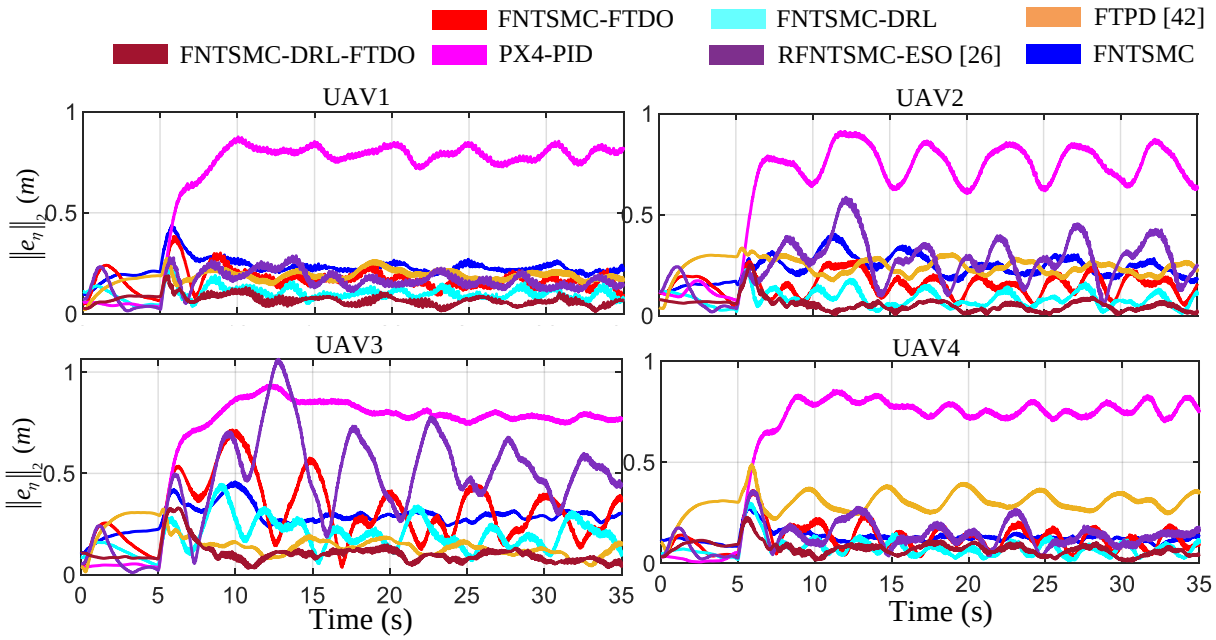


Figure 6.19: $||e_\eta||_2$ of each quadrotor under different control frameworks.

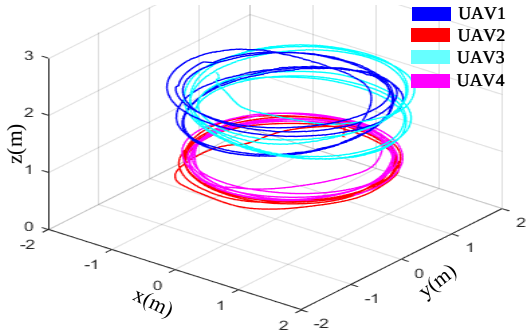


Figure 6.20: Graphic demonstration of the quadrotor formation in a 3D view.

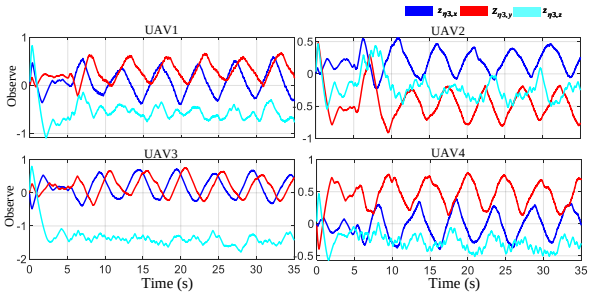


Figure 6.21: Output of the observers in experiment group 1.

6.6.2 Experiment Group 2

Furthermore, we employed a more aggressive reference trajectory to test the performance of the quadrotor formation under more extreme environmental conditions. The topological graph of the quadrotor group is shown in Fig. 6.22. The adjacent matrix $\mathscr{A}_{p2}$, in-degree matrix $\mathscr{D}_{p2}$, communication matrix $\mathscr{B}_{p2}$, and Laplacian matrix $\mathscr{L}_{p2}$ are respectively defined as $\mathscr{A}_{p2} = [0,1,0,1;1,0,1,0;0,1,0,0;1,0,0,0]$, $\mathscr{D}_{p2} = \text{diag}(2,2,1,1)$, $\mathscr{B}_{p2} = \text{diag}(1,0,0,0)$, and $\mathscr{L}_{p2} = \mathscr{D}_{p2} - \mathscr{B}_{p2}$. The equation for the geometric center $\mathscr{O}_d$ is defined as

$$x_d = \cos(0.4\pi t),\ y_d = \sin(0.8\pi t) + 0.2,\ z_d = 1.5. \tag{6.56}$$

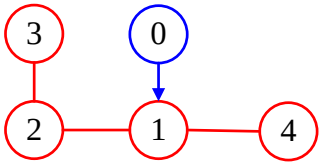


Figure 6.22: Topological graph in Experiment Group 2.

The offsets of each quadrotor to $\mathcal{O}_d$, denoted by $v_i, i = 1,2,3,4$, are defined as

$$v_1 = [0.5, 0, 0.3\sin(0.2\pi t) + 0.5]^\top, \quad v_2 = [0, 0.8, -0.5]^\top,$$
$$v_3 = [-0.5, 0, 0.3\sin(0.2\pi t) + 0.5]^\top, \quad v_4 = [0, -0.8, -0.5]^\top.$$

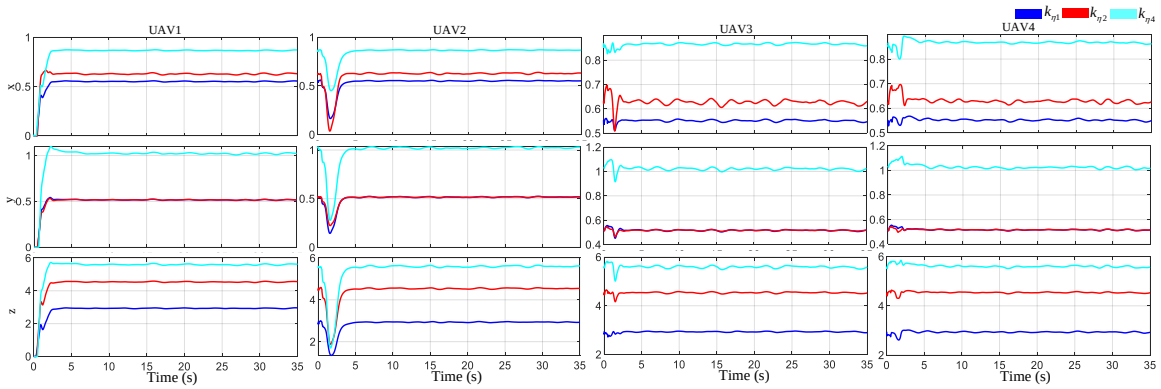


Figure 6.23: Hyperparameters $k_{\eta 1}$, $k_{\eta 2}$, and $k_{\eta 4}$ tuned by DRL in experiment group 2.

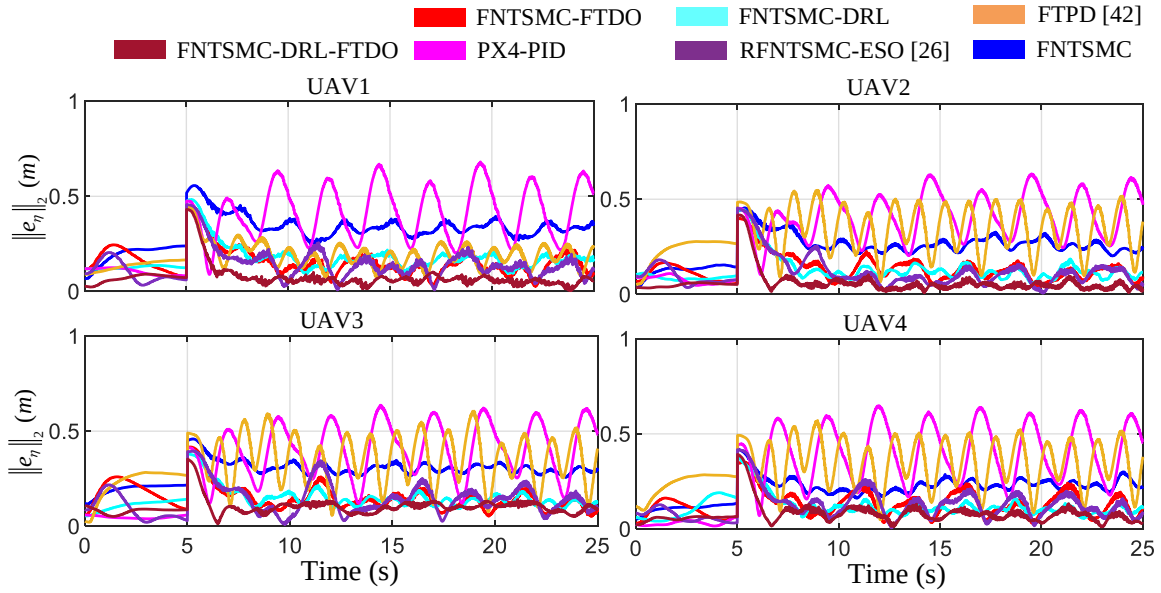


Figure 6.24: $||e_\eta, i||_2$ of each quadrotor under different control frameworks.

Similarly, the gains tuned by DRL, the 2-norm of the tracking errors, the position response in a 3D view, and the corresponding output of the FTDO are recorded in Fig. 6.23, Fig 6.24, Fig. 6.25, and Fig. 6.26, respectively. The error statistical curves from Experiment 2 exhibit the same pattern as those from Experiment 1, indicating that the proposed control framework consistently outperforms other methods. Specifically, we calculated the sum of the L_2 and L_1 norms of the consensus errors for the quadrotor formation, as presented in Table 6.2, which demonstrates the superiority of the FNTSMC-DRL-FTDO method. Finally, as shown in Fig. 6.26, the oscillation frequency of the observer output curve in the y direction is approximately twice that of the x direction, which aligns perfectly with the characteristics of the pre-defined reference trajectory.

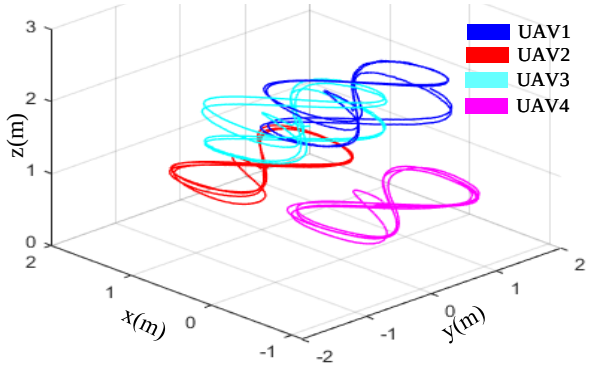


Figure 6.25: Graphic demonstration of the quadrotor formation in a 3D view.

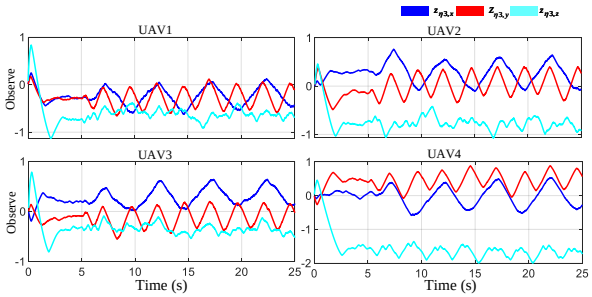


Figure 6.26: Output of the observers in experiment group 2.

Table 6.2: Tracking errors of the quadrotor group under two groups of experiments.

Experiment Group1				
	FNTSMC	FNTSMC-FTDO	FNTSMC-RL	PX4-PID
$\sum_{i=1}^{4}\|\|e_{\eta,i}\|\|_2$	210.8893	202.3448	120.8247	613.5627
$\sum_{i=1}^{4}\|\|e_{\eta,i}\|\|_1\ (\times 10^4)$	1.1409	0.9841	0.5905	3.0267
	[160]	[170]	Proposed	
$\sum_{i=1}^{4}\|\|e_{\eta,i}\|\|_2$	226.4347	261.9122	83.8014	
$\sum_{i=1}^{4}\|\|e_{\eta,i}\|\|_1\ (\times 10^4)$	1.1450	1.2361	0.3897	
Experiment Group2				
	FNTSMC	FNTSMC-FTDO	FNTSMC-RL	PX4-PID
$\sum_{i=1}^{4}\|\|e_{\eta,i}\|\|_2$	224.7086	134.0974	130.7996	274.9714
$\sum_{i=1}^{4}\|\|e_{\eta,i}\|\|_1\ (\times 10^4)$	0.9820	0.5438	0.5319	1.1355
	[160]	[170]	Proposed	
$\sum_{i=1}^{4}\|\|e_{\eta,i}\|\|_2$	263.8589	124.6594	78.5620	
$\sum_{i=1}^{4}\|\|e_{\eta,i}\|\|_1\ (\times 10^4)$	1.0839	0.4707	0.2926	

6.7 Conclusions

This chapter presents a novel robust control framework for quadrotor groups integrating FNTSMCs, DRL techniques, and FTDOs. Initially, FNTSMCs were designed for closed-loop systems. Next, FTDOs are employed to accurately estimate the uncertainties and disturbances affecting the quadrotors in fixed time. The outputs of the observers are integrated into the switching control laws of the system, thereby enhancing the robustness of the controller and improving overall control performance. The fixed-time stability of the multi-agent system is guaranteed in a Lyapunov sense. To further enhance control performance, DRL is utilized to train a parameter optimizer that adaptively tunes certain hyper-parameters in the FNTSMCs based on the tracking errors of the quadrotors. Finally, extensive simulations and physical experiments are conducted to validate the effectiveness and robustness of the proposed control framework.

Chapter 7

Modified Predefined-Time Adaptive Observer-Based Fast Nonsingular Terminal Sliding Mode Control for Multi-Quadrotor Subject to External Disturbances

7.1 Research Background

Similar to but different from the finite-time control method proposed in Chapter 5 and the fixed-time control proposed in Chapter 6, predefined-time control represents a class of techniques that impose stricter requirements on the convergence time of control systems. In these methods, the upper bound of the convergence time can be explicitly expressed, thereby enabling designers to predict the dynamic behavior of the control system more conveniently.

Therefore, this chapter proposes an enhanced predefined-time convergence criterion and leverages this criterion to design a predefined-time observer, a distributed Predefined-time Disturbance Observer (PdTDO), and a distributed Predefined-time Fast Nonsingular Terminal Sliding Mode Control (PdTFNTSMC) framework to achieve consensus control for quadrotor formations.

The main contributions of the chapter can be summarized as follows:

1) A PdT-FNTSMC consensus control protocol is designed for multi-quadrotor formation control. Different from some existing works [171, 172], we focus on multi-quadrotor formation control rather than controlling a single quadrotor. In consensus control for multi-agent formations, individual quadrotors lack access to complete information about the entire formation. Consequently, they must compute control laws based solely on the limited information available to them to ensure the overall system's stability. This inherent constraint significantly increases

the complexity of controller design and analysis. Compared to work proposed in [172], we introduce a more general PdT criterion formation by adding an exponential and linear terms for faster convergence.

2) In contrast to some observers proposed in previous works [173, 174], the PtDDO proposed in this study does not require the differential of the disturbance to be bounded and the upper bound of the disturbance unknown. Additionally, we introduce an adaptive tuning law to simultaneously estimate the disturbance and upper bound of the disturbance, reducing the observer's conservation.

3) Unlike finite or fixed time stable controllers proposed in [175–177], the fully distributed control framework proposed in this chapter achieves predefined time convergence in observing, sliding mode convergence, and tracking error convergence processes. Sufficient comparative simulations and physical experiments are conducted to verify the superiority and effectiveness of the proposed control framework.

Notation: In what follows, $C_{(\cdot)}$, $S_{(\cdot)}$, and $T_{(\cdot)}$ respectively denote the cosine function, sine function, and tangent function. For a column vector $x \in \mathbb{R}^n$, $\text{sgn}(x)$ and $|x|$ respectively represent the signum function and absolute function by element of vector x, $\lfloor x \rceil^{\alpha} = |x|^{\alpha} \circ \text{sgn}(x)$ with $\circ$ being the Hadamard product operator; $||x||$ is the 2-norm of vector x. $x >, <, = 0$ means all elements in x are all positive, negative, zeros. $\mathbf{I}_n$ represents an n-dimensional identity matrix. For mathematical modelling of the i-th quadrotor, $\eta_i = [x_i, y_i, z_i]^{\top}$ and $\Theta_i = [\varphi_i, \theta_i, \psi_i]^{\top}$ respectively denote the position and attitude angle of the i-th quadrotor in world frame, respectively. $\omega_i = [p_i, q_i, r_i]^{\top}$ is the angular rate defined in body frame. The inertia tensor matrix of the quadrotor is denoted by $J_i = \text{diag}(J_{x,i}, J_{y,i} J_{z,i})$, $\tau_i = [\tau_{x,i}, \tau_{y,i}, \tau_{z,i}]^{\top}$ denotes the torque, and $u_{f,i}$ is the throttle of the i-th quadrotor. $g = 9.8(\text{kg m/s}^2)$ is the gravitational acceleration.

7.2 Preliminaries and Problem Formulation

7.2.1 Mathematical fundamental

The quadrotor formation group consists of N quadrotor follower nodes $v_1, v_2, \cdots, v_N$ and one virtual leader node v_0. Graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ is defined to describe all follower nodes with $\mathscr{V} = \{v_i, i = 1, 2, \cdots, N\}$ being the the node set and $\mathscr{E} = \{(v_i, v_j), i, j = 1, 2, \cdots, N\}$ being the edge set. Specifically, (v_i, v_j) represents the edge that connects v_i and v_j. Augmented graph $\bar{\mathscr{G}} = (\mathscr{V}, \mathscr{E}, v_0, b_i)$, $i = 1, 2, \cdots, N$ is defined to describe the entire leader-follow multi-quadrotor system with b_i being the connection among v_0 and other follower nodes. $b_i = 1$ means the information can transform from v_0 to v_i, otherwise, $b_i = 0$. The communication matrix of the leader node v_0 is defined as

$\mathscr{B} = \text{diag}([b_i])$. The neighborhood of v_i is denoted as $\mathscr{N}_i\{v_j|(v_j,v_i) \in \mathscr{E}\}$. The adjacency matrix is defined as $\mathscr{A} = [a_{ij}] \in \mathbb{R}^n$, the in-degree of the follower node v_i is defined as $d_i = \sum_{j=1}^{N} a_{ij}$, the in-degree matrix is defined as $\mathscr{D} = \text{diag}(d_i)$, and the Laplacian matrix of $\mathscr{G}$ is defined as $\mathscr{L} = \mathscr{D} - \mathscr{A}$. Some useful definitions, corollary, assumptions, and lemmas are listed without loss of generality.

Definition 7.1 *(see the Appendix A. of [178]) For $x_1, x_2 \in \mathbb{R}^+$, the Gamma function $\Gamma(x_1)$ and the Beta function $B(x_1,x_2)$ are respectively defined as*

$$\begin{aligned} \Gamma(x_1) &= \int_0^{\infty} \hbar^{x_1-1} e^{-\hbar} \mathrm{d}\hbar, \\ B(x_1,x_2) &= \int_0^{1} \hbar^{x_1-1}(1-\hbar)^{x_2-1} \mathrm{d}\hbar. \end{aligned} \tag{7.1}$$

In addition, $B(x_1,x_2) = \frac{\Gamma(x_1)\Gamma(x_2)}{\Gamma(x_1+x_2)}$.

Corollary 7.1 *An equivalent formation of the Beta function can be given by*

$$B(x_1,x_2) = \int_0^{\infty} \frac{\hbar^{x_1-1}}{(\hbar+1)^{x_1+x_2}} \mathrm{d}\hbar. \tag{7.2}$$

Proof 7.1 *We define $u = \frac{\hbar}{\hbar-1}$. There is $\mathrm{d}\hbar = \frac{1}{(u+1)^2}\mathrm{d}u$. The integration can be reformulated as*

$$\begin{aligned} B(x_1,x_2) &= \int_0^{1} \hbar^{x_1-1}(1-\hbar)^{x_2-1} \mathrm{d}\hbar \\ &= \int_0^{\infty} \left(\frac{u}{u+1}\right)^{x_1-1} \left(\frac{1}{u+1}\right)^{x_2-1} \frac{\mathrm{d}u}{(u+1)^2} \\ &= \int_0^{\infty} \frac{u^{x_1-1}}{(u+1)^{x_1+x_2}} \mathrm{d}u. \end{aligned} \tag{7.3}$$

The proof is completed.

Assumption 7.1 *(see [159]) For the i-th quadrotor, the equivalent disturbances acted on the rotational subsystem, $\Delta_{\Theta,i}$, and translational subsystem, $\Delta_{\eta,i}$, are all bounded by some unknown positive constants, namely, $||\Delta_{\Theta,i}|| \leq \sigma_\Theta$ and $||\Delta_{\eta,i}|| \leq \sigma_\eta$.*

Assumption 7.2 *(see [160]) The yaw angle of the i-th quadrotor is bounded as $-\pi \leq \psi_i \leq \pi$, while the roll and pitch angles are bounded as $-\pi/2 < \varphi_i, \theta_i < \pi/2$.*

Assumption 7.3 *(see [179]) For graph $\mathscr{G}$ and augmented graph $\bar{\mathscr{G}}$, there are*

(1) The graph $\mathscr{G}$ is undirected.

(2) There are no self-loops in $\mathscr{G}$, say, $a_{ii} = 0$, $i = 1,2,\cdots,N$.

(3) There exists at least one spanning tree in augmented graph $\bar{\mathcal{G}}$ with the leader node v_0 being the root of the tree.

Lemma 7.1 *(see [180]) $\forall x_i \geq 0$, $i = 1,2,\cdots,N$, $0 < p \leq 1$, there is*

$$\left(\sum_{i=1}^{n} x_i\right)^p \leq \sum_{i=1}^{n} x_i^p \leq n^{1-p}\left(\sum_{i=1}^{n} x_i\right)^p$$

Lemma 7.2 *$\forall x_i \geq 0$, $i = 1,2,\cdots,N$ and $p \geq 1$, there is*

$$n^{p-1}\sum_{i=1}^{n} x_i^p \geq \left(\sum_{i=1}^{n} x_i\right)^p. \tag{7.4}$$

Proof 7.2 *Using Hölder's inequality, we have*

$$\sum_{i=1}^{n} x_i y_i \leq \left(\sum_{i=1}^{n} x_i^p\right)^{\frac{1}{p}}\left(\sum_{i=1}^{n} y_i^q\right)^{\frac{1}{q}}, \tag{7.5}$$

where $\frac{1}{p}+\frac{1}{q}=1$. Letting $y_i = 1, i = 1,2,\cdots,n$ yields

$$n^{p-1}\sum_{i=1}^{n} x_i^p \geq \left(\sum_{i=1}^{n} x_i\right)^p. \tag{7.6}$$

Then, the proof is completed by selecting $p \geq 1$.

Lemma 7.3 *(see [181]) Consider a differential equation*

$$\dot{x}(t) = -m_1 x(t) - m_2 x^{\mu}(t) + m_3\phi(t), \tag{7.7}$$

where $m_1, m_2, m_3 > 0$, $\mu \geq 1$, and $\phi(t) \geq 0$. Then, $x(t) \geq 0$ if $x(0) \geq 0$ holds.

Lemma 7.4 *(see [181]) For $\forall x_1, x_2 \in \mathbb{R}$, $x_2 \geq x_1$, and $\varepsilon \geq 1$, there is*

$$x_1(x_2 - x_1)^{\varepsilon} \leq \frac{\varepsilon}{\varepsilon+1}\left(x_2^{1+\varepsilon} - x_1^{1+\varepsilon}\right). \tag{7.8}$$

Lemma 7.5 *(see [182]) For $\forall x_1, x_2 \in \mathbb{R}$, $\vartheta, \varpi_1, \varpi_2 \in \mathbb{R}^+$, we have*

$$|x_1|^{\varpi_1}|x_2|^{\varpi_2} \leq \frac{\varpi_1|x_1|^{\varpi_1+\varpi_2}}{\varpi_1+\varpi_2}\vartheta + \frac{\varpi_2|x_2|^{\varpi_1+\varpi_2}}{\varpi_1+\varpi_2}\vartheta^{-\frac{\varpi_1}{\varpi_2}}. \tag{7.9}$$

Lemma 7.6 *For $\forall x_1, x_2, k > 0$ and k is the ratio of two positive odd integers. Then,*

(1): for $k \geq 1, x_2 \leq x_1$ or $0 < k \leq 1, x_2 \geq x_1$, there is

$$(x_1 - x_2)^k \leq x_1^k - x_2^k \tag{7.10}$$

(2): for $k \geq 1, x_2 \geq x_1$ or $0 < k \leq 1, x_2 \leq x_1$, there is

$$(x_1 - x_2)^k \geq x_1^k - x_2^k \tag{7.11}$$

Proof 7.3 *Define $x_2 = m_0 x_1$, $m_0 > 0$, and function $f(x_1) = (x_1 - x_2)^k - (x_1^k - x_2^k)$. Then, $f(x_1)$ can be simplified as*

$$f(x_1) = [(1-m_0)^k - 1 + m_0^k] x_1^k \propto (1-m_0)^k - 1 + m_0^k \triangleq g(m_0). \tag{7.12}$$

It is easy to find that $g(0) = g(1) = 0$ and solving $\frac{\mathrm{d}g(m_0)}{dm_0} = 0$ yields $m_0 = 0.5$. Therefore, if $0 < k \leq 1$, function $g(m_0) \geq 0$ on $(0,1]$ (namely, $x_2 \leq x_1$) and $g(m_0) \leq 0$ on $[1,\infty)$ (namely, $x_2 \geq x_1$). Similarly, if $k \geq 1$, function $g(m_0) \leq 0$ when $x_2 \leq x_1$ and $g(m_0) \geq 0$ when $x_2 \geq x_1$. This completes the proof.

Lemma 7.7 *(see [181]) For a nonlinear system $\dot{x} = f(x)$, the system is practically PdT stable if there exists a Lyapunov function $V(x)$ such that*

$$\dot{V} \leq -\frac{\pi}{(2-2\alpha) T_c \sqrt{\beta_2 \beta_3}} (\beta_2 V^{\alpha} + \beta_2 V^{2-\alpha}) + \rho \tag{7.13}$$

holdes, in which $0 < \alpha < 1$, $\beta_2, \beta_3 > 0$, $T_c > 0$ are positive constants, and $\rho > 0$. The upper bound of the settling time satisfies $T_p = \sqrt{2} T_c$.

7.2.2 System modelling

Based on [183], the mathematical model of the i-th quadrotor can be given by

$$\begin{aligned} \ddot{\eta}_i &= \frac{u_{f,i}}{m_i} A_i(\Theta_i) - \mathbf{g} - \frac{k_t}{m_i} \dot{\eta}_i + \frac{\delta_{\eta,i}}{m_i}, \\ \dot{\omega}_i &= J_i^{-1} \left[-k_r \omega_i - \omega_i \times (J_i \omega_i) + \delta_{\Theta,i} + \tau_i \right], \\ \dot{\Theta}_i &= W_i(\Theta_i) \omega_i, \end{aligned} \tag{7.14}$$

where η_i, $u_{f,i}$, and m_i denote the position, throttle, and mass of the i-th quadrotor. k_t and k_r, respectively, denote the drag coefficients of the translational and rotational loops. ω_i, J_i, Θ_i, and τ_i represent the angular rate, inertial tenser matrix, attitude angle, and torque of the i-th quadrotor, respectively. $\delta_{\eta,i}$ and $\delta_{\Theta,i}$ respectively represent the disturbance acted on the translational and rotational subsystem of the i-th quadrotor. $\mathbf{g} = [0,0,g]^{\top}$ is the gravitational acceleration vector. $A_i \triangleq A_i(\Theta_i)$ and

$W_i \triangleq W_i(\Theta_i)$ are respectively defined as

$$\begin{bmatrix} C_{\varphi,i}C_{\psi,i}S_{\theta,i} + S_{\varphi,i}S_{\psi,i} \\ C_{\varphi,i}S_{\psi,i}S_{\theta,i} - S_{\varphi,i}C_{\psi,i} \\ C_{\theta,i}C_{\varphi,i} \end{bmatrix}, \begin{bmatrix} 1 & S_{\varphi,i}T_{\theta,i} & C_{\varphi,i}T_{\theta,i} \\ 0 & C_{\varphi,i} & -S_{\varphi,i} \\ 0 & S_{\varphi,i}/C_{\theta,i} & C_{\varphi,i}/C_{\theta,i} \end{bmatrix}.$$

1) Rotational Subsystem: Define the expected attitude angle of the i-th quadrotor as $\Theta_{d,i} = [\varphi_{d,i}, \theta_{d,i}, \psi_{d,i}]^\top$. The tracking error $e_{\Theta,i}$ as well as the 1^{st} and 2^{nd} order derivatives of $e_{\Theta,i}$ are given by

$$\begin{aligned} e_{\Theta,i} &= \Theta_i - \Theta_{d,i} \\ \dot{e}_{\Theta,i} &= W_i\omega_i - \dot{\Theta}_{d,i}, \\ \ddot{e}_{\Theta,i} &= \dot{W}_i\omega_i + W_i\dot{\omega}_i - \ddot{\Theta}_{d,i}, \end{aligned} \tag{7.15}$$

where $\dot{W}_i$ can be defined as

$$\dot{W}_i = \begin{bmatrix} 0 & \dot{\varphi}_i T_{\theta_i} C_{\varphi_i} + \frac{\dot{\theta}_i S_{\varphi_i}}{C^2_{\theta_i}} & -\dot{\varphi}_i S_{\varphi_i} T_{\theta_i} + \frac{\dot{\theta}_i C_{\varphi_i}}{C^2_{\theta_i}} \\ 0 & -\dot{\varphi}_i S_{\varphi_i} & -\dot{\varphi}_i C_{\varphi_i} \\ 0 & \frac{\dot{\varphi}_i C_{\varphi_i} C_{\theta_i} + \dot{\theta}_i S_{\varphi_i} S_{\theta_i}}{C^2_{\theta_i}} & \frac{-\dot{\varphi}_i S_{\varphi_i} C_{\theta_i} + \dot{\theta}_i C_{\varphi_i} S_{\theta_i}}{C^2_{\theta_i}} \end{bmatrix}.$$

By defining $\Delta_{\Theta,i} = J_i^{-1}\delta_{\Theta,i} - \ddot{\Theta}_{d,i}$, $f_{\Theta,i} = -J_i^{-1}[k_r\omega_i + \omega_i \times (J_i\omega_i)]$, $A_{\Theta,i} = \dot{W}_i\omega_i + W_i f_{\Theta,i}$, $B_{\Theta,i} = W_i J_i^{-1}$ and doing some manipulations, Eq. (7.15) can be finally simplified as

$$\ddot{e}_{\Theta,i} = A_{\Theta,i} + B_{\Theta,i}\tau_i + \Delta_{\Theta,i}. \tag{7.16}$$

Remark 7.1 *In pure attitude control, the 1^{st} and 2^{nd} order derivatives of $\Theta_{d,i}$ can be predefined or analytically computed. However, obtaining the $2^{nd}-$ order derivatives of the desired angle through differentiation may introduce unnecessary noise in position control. Therefore, $\ddot{\Theta}_{d,i}$ is absorbed into $\Delta_{\Theta,i}$ and treated as part of the unknown disturbances.*

2) Translational Subsystem: Due to the coupling characteristics, we introduce the virtual desired acceleration of the i-th quadrotor in the position control loop as

$$u_{\eta,i} = [a_{x,i}, a_{y,i}, a_{z,i}]^\top, \tag{7.17}$$

yielding

$$\ddot{\eta}_i = -\frac{k_t}{m_i}\dot{\eta}_i + u_{\eta,i} + \Delta_{\eta,i}, \tag{7.18}$$

where $\Delta_{\eta,i} = \frac{u_{f,i}}{m_i}A_i + \frac{\delta_{\eta,i}}{m_i} - \mathbf{g} - u_{\eta,i}$ is the equivalent disturbance. Thereafter, by [184], the expected throttle and pitch and roll angles of the i-th quadrotor can be computed by

$$\begin{aligned} u_{f,i} &= m_i\sqrt{a_{x,i}^2 + a_{y,i}^2 + (a_{z,i}+g)^2}, \\ \varphi_{d,i} &= \arcsin\frac{m_i\left[a_{x,i}S_{\psi,i} - a_{y,i}C_{\psi,i}\right]}{u_{f,i}}, \\ \theta_{d,i} &= \arctan\frac{a_{x,i}C_{\psi,i} + a_{y,i}S_{\psi,i}}{a_{z,i}+g}. \end{aligned} \tag{7.19}$$

Using Eq. (7.18), the consensus tracking error of the i-th quadrotor can be expressed as

$$e_{\eta,i} = \sum_{j=1}^{N} a_{ij}\left[(\eta_i - \nu_i) - (\eta_j - \nu_j)\right] + b_i(\eta_i - \eta_d - \nu_i), \tag{7.20}$$

where $\eta_d = [x_d, y_d, z_d]^\top$ is the expected geometric center of the quadrotor group and $\nu_i = [\nu_{x,i}, \nu_{y,i}, \nu_{z,i}]^\top$ is the offset of the i-th quadrotor to η_d. For ease of further derivation, we define a new variable Λ_i as

$$\Lambda_i = b_i\eta_d + (b_i + d_i)\nu_i + \textstyle\sum_{j=1}^{N} a_{ij}(\eta_j - \nu_j). \tag{7.21}$$

Correspondingly, one obtains

$$\begin{aligned} \dot{\Lambda}_i &= b_i\dot{\eta}_d + (b_i + d_i)\dot{\nu}_i + \textstyle\sum_{j=1}^{N} a_{ij}(\dot{\eta}_j - \dot{\nu}_j), \\ \ddot{\Lambda}_i &= b_i\ddot{\eta}_d + (b_i + d_i)\ddot{\nu}_i + \textstyle\sum_{j=1}^{N} a_{ij}(\ddot{\eta}_j - \ddot{\nu}_j). \end{aligned} \tag{7.22}$$

Substituting Eq. (7.18) into Eq. (7.22) and doing some manipulations yield

$$\begin{aligned} \ddot{\Lambda}_i &= b_i\ddot{\eta}_d + (b_i + d_i)\ddot{\nu}_i + \textstyle\sum_{j=1}^{N} a_{ij}(\ddot{\eta}_j - \ddot{\nu}_j) \\ &= b_i\ddot{\eta}_d + (b_i + d_i)\ddot{\nu}_i + \textstyle\sum_{j=1}^{N} a_{ij}\left(-\frac{k_{t,j}}{m_j}\dot{\eta}_j + u_{\eta,j} + \Delta_{\eta,j} - \ddot{\nu}_j\right) \\ &= \Lambda_{i0} + \textstyle\sum_{j=1}^{N} a_{ij}\Delta_{\eta,j}, \end{aligned} \tag{7.23}$$

where $\Lambda_{i0} = b_i\ddot{\eta}_d + (b_i + d_i)\ddot{\nu}_i + \sum_{j=1}^{N} a_{ij}(-\frac{k_{t,j}}{m_j}\dot{\eta}_j + u_{\eta,j} - \ddot{\nu}_j)$ is a known variable. Then, the error dynamics of the translational loop can be given by

$$\ddot{e}_{\eta,i} = -\frac{(d_i + b_i)k_t}{m_i}\dot{\eta}_i + (b_i + d_i)u_{\eta,i} - \Lambda_{i0} + (b_i + d_i)\Delta_{\eta,i} - \textstyle\sum_{j=1}^{N} a_{ij}\Delta_{\eta,j}. \tag{7.24}$$

7.2.3 Control Objective

Based on Assumptions 7.1- 7.3, the control objective of this chapter is formulated as follows:

Control Objective: Give a quadrotor formation with N quadrotors with graph $\mathscr{G}$ and a set of reference trajectories of the geometric center $\{\eta_d = [x_d, y_d, z_d]^\top, \psi_d\}$ generated by the virtual leader node and the offset of each quadrotor to η_d, say, $\nu_i = [\nu_{x,i}, \nu_{y,i}, \nu_{z,i}^\top]$, $i = 1, 2, \cdot s, N$, design a type of adaptive fully distributed PdT control protocol such that for $i = 1, 2, \cdots, N$, we have

$$\begin{aligned} &\lim_{t \to T_\eta} ||\eta_i(t) - (\eta_d(t) + \nu_i(t))|| = 0, \\ &\lim_{t \to T_\Theta} \psi_i(t) - \psi_d(t) = 0 \end{aligned} \tag{7.25}$$

with T_η and T_Θ being the upper bound of the PdT of the translational and rotational subsystems, respectively.

7.3 Main results

Theorem 7.1 *For a continuous nonlinear system $\dot{x} = f(x)$, $x \in \mathbb{R}^n$ and $f(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ is the system dynamics. The system is PdT stable, and the settling time is bounded by $T_0 > 0$ if there exists a Lyapunov function such that inequality*

$$\dot{V} \leq -\kappa_0 \left(2\beta_1 V + \beta_2 V^\alpha + \beta_3 V^{2-\alpha}\right)^{\kappa_1} \tag{7.26}$$

holds, where $\beta_2, \beta_3 > 0$, $\beta_1^2 \geq \beta_2 \beta_3$, $0 < \alpha < 1$, $\frac{1}{2-\alpha} < \kappa_1 < \frac{1}{\alpha}$ is defined by the ratio of two positive odd numbers, and

$$\kappa_0 = \frac{1}{T_0} \frac{\Gamma\left(\frac{1-\alpha\kappa_1}{2-2\alpha}\right)\Gamma\left(\frac{2\kappa_1-\alpha\kappa_1-1}{2-2\alpha}\right)}{\beta_2^{\kappa_1}(2-2\alpha)\Gamma(\kappa_1)} \left(\frac{\beta_2}{\beta_3}\right)^{\frac{1-\alpha\kappa_1}{2-2\alpha}}.$$

Proof 7.4 *For Eq. (7.26), there is*

$$\begin{aligned} \frac{\mathrm{d}V}{dt} &\leq -\kappa_0 \left(2\beta_1 V + \beta_2 V^\alpha + \beta_3 V^{2-\alpha}\right)^{\kappa_1} \\ \int_{V_0}^{V_f} \mathrm{d}V &\leq -\kappa_0 \int_0^T \left(2\beta_1 V + \beta_2 V^\alpha + \beta_3 V^{2-\alpha}\right)^{\kappa_1} dt \\ T &\leq -\frac{1}{\kappa_0} \int_{V_0}^{V_f} \frac{\mathrm{d}V}{\left(2\beta_1 V + \beta_2 V^\alpha + \beta_3 V^{2-\alpha}\right)^{\kappa_1}}, \end{aligned} \tag{7.27}$$

where T is the settling time, V_0 is the Lyapunov function for $x_0 = x(0)$, and V_f is the Lyapunov function for $x_f = x(T)$.

It is obvious to find out that $\dot{V} < 0$ for all $V > 0$ and $\dot{V} = 0$ if and only if $V = 0$, indicating that V

will decrease monotonically to 0*. Therefore, substituting* $V_f = 0$*, we have*

$$\begin{aligned} T &\leq \frac{1}{\kappa_0}\int_0^{V_0} \frac{\mathrm{d}V}{(2\beta_1 V + \beta_2 V^{\alpha} + \beta_3 V^{2-\alpha})^{\kappa_1}} \\ &= \frac{1}{\kappa_0}\int_0^{V_0} \frac{\mathrm{d}V}{\left[\beta_2 V^{\alpha}\left(1 + 2\frac{\beta_1}{\beta_2}V^{1-\alpha} + \frac{\beta_3}{\beta_2}V^{2-2\alpha}\right)\right]^{\kappa_1}} \\ &= \frac{1}{\kappa_0}\int_0^{V_0} \frac{\mathrm{d}V}{\left[\beta_2 V^{\alpha}(1 + \sqrt{\beta_3/\beta_2}V^{1-\alpha})^2 + \delta_{\beta}\right]^{\kappa_1}}, \end{aligned} \tag{7.28}$$

where $\delta_{\beta} = \left(2\beta_1/\beta_2 - 2\sqrt{\beta_3/\beta_2}\right)V^{1-\alpha}$*. Using Young's inequality, the fact* $\beta_1^2 \geq \beta_2\beta_3$*, and doing some manipulations yields*

$$\begin{aligned} T &\leq \frac{1}{\kappa_0}\int_0^{V_0} \frac{\mathrm{d}V}{\left[\beta_2 V^{\alpha}(1 + \sqrt{\beta_3/\beta_2}V^{1-\alpha})^2\right]^{\kappa_1}} \\ &\leq \frac{1}{\kappa_0\beta_2^{\kappa_1}}\int_0^{V_0} \frac{\mathrm{d}V}{V^{\alpha\kappa_1}\left(1 + \beta_3/\beta_2 V^{2-2\alpha}\right)^{\kappa_1}} \end{aligned} \tag{7.29}$$

Defining $\zeta = V^{2-2\alpha}$ *yields* $\mathrm{d}\zeta = (2-2\alpha)V^{1-2\alpha}\mathrm{d}V$ *and* $\zeta_0 = \zeta(0) = V_0^{2-2\alpha}$*. Then, (7.29) becomes*

$$\begin{aligned} T &\leq \frac{1}{\kappa_0\beta_2^{\kappa_1}}\int_0^{\zeta_0} \frac{\zeta^{\frac{2\alpha-1}{2-2\alpha}}\mathrm{d}\zeta}{(2-2\alpha)\zeta^{\frac{\alpha\kappa_1}{2-2\alpha}}\left(1 + \zeta\beta_3/\beta_2\right)^{\kappa_1}} \\ &= \frac{1}{\kappa_0\beta_2^{\kappa_1}(2-2\alpha)}\int_0^{\zeta_0} \frac{\mathrm{d}\zeta}{\left(1 + \beta_3/\beta_2\zeta\right)^{\kappa_1}\zeta^{\frac{\alpha\kappa_1 - 2\alpha + 1}{2-2\alpha}}}. \end{aligned} \tag{7.30}$$

Defining $\aleph = \zeta\beta_3/\beta_2$ *yields* $\mathrm{d}\aleph = (\beta_3/\beta_2)d\xi$*,* $\aleph_0 = \aleph(0) = \zeta_0\beta_3/\beta_2$ *and*

$$T \leq \frac{(\beta_2/\beta_3)^{\frac{1-\alpha\kappa_1}{2-2\alpha}}}{\kappa_0\beta_2^{\kappa_1}(2-2\alpha)}\int_0^{\aleph_0} \frac{\aleph^{\frac{2\alpha-1-\alpha\kappa_1}{2-2\alpha}}\mathrm{d}\aleph}{(1+\aleph)^{\kappa_1}}. \tag{7.31}$$

Note that the facts $0 < \alpha < 1$ *and all terms in the integration function are positive. By using Corollary 7.1, substituting* κ_0 *into Eq. (7.31), and doing some manipulations, we have*

$$\begin{aligned} T &\leq \frac{(\beta_2/\beta_3)^{\frac{1-\alpha\kappa_1}{2-2\alpha}}}{\kappa_0\beta_2^{\kappa_1}(2-2\alpha)}\int_0^{\infty} \frac{\aleph^{\frac{2\alpha-1-\alpha\kappa_1}{2-2\alpha}}\mathrm{d}\aleph}{(1+\aleph)^{\kappa_1}} \\ &= \frac{T_0\Gamma(\kappa_1)}{\Gamma(a_1)\Gamma(a_2)}B(a_1,a_2) \\ &= \frac{T_0\Gamma(\kappa_1)}{\Gamma(a_1)\Gamma(a_2)}\frac{\Gamma(a_1)\Gamma(a_2)}{\Gamma(\kappa_1)} = T_0, \end{aligned} \tag{7.32}$$

where $a_1 = \frac{1-\alpha\kappa_1}{2-2\alpha}$, $a_2 = \frac{2\kappa_1-\alpha\kappa_1-1}{2-2\alpha}$, *and* $a_1+a_2 = \kappa_1$. *Note that to make Eq. (7.32) holds, the following inequalities should be satisfied*

$$a_1 > 0,\ a_2 > 0,\ \kappa_1 > 0, 0 < \alpha < 1, \tag{7.33}$$

which means

$$\begin{aligned} &0 < \frac{1-\alpha\kappa_1}{2-2\alpha} < 1, \\ &0 < \frac{2\kappa_1-\alpha\kappa_1-1}{2-2\alpha} < 1, \\ &\kappa_1 > 0,\ 0 < \alpha < 1. \end{aligned} \tag{7.34}$$

Solving Eq. (7.34) yields $\frac{1}{2-\alpha} < \kappa_1 < \frac{1}{\alpha}$. *This completes the proof.*

Remark 7.2 *Note that the '*κ_0*' appeared in Theorem 7.1 can be automatically calculated when the predefined time* T_0, α, κ_1, β_2, *and* β_3 *are manually designed. Additionally, the Gamma function that appeared in* κ_0 *can be computed by some numerical computing toolkit, for example, the 'Scipy' package in Python, the 'gammainc' function in Matlab, the 'tgamma' function integrated into the cmath calculation library in C++ et al. Therefore, although the formation of* κ_0 *is complicated, it can be automatically computed using Python, Matlab, or C++.*

However, it is usually hard to make the estimation error exactly converge to zero when a PdT disturbance observer based on Theorem 7.1 is utilized to estimate some time-variant disturbances. Therefore, we further develop the following corollary.

Corollary 7.2 *Based on Theorem 7.1, the system* $\dot{x} = f(x)$ *is said to be practically PdT stable if there exists a Lyapunov function* V *and* $\Delta > 0$ *such that equality*

$$\dot{V} \le -\kappa_0 \left(2\beta_1 V + \beta_2 V^{\alpha} + \beta_3 V^{2-\alpha}\right)^{\kappa_1} + \Delta \tag{7.35}$$

holds, where $\beta_2, \beta_3 > 0$, $\beta_1^2 \ge \beta_2\beta_3$, $0 < \alpha < 1$, $\frac{1}{2-\alpha} < \kappa_1 < \frac{1}{\alpha}$ *is defined by the ratio of two positive odd numbers, and*

$$\kappa_0 = \frac{1}{T_0} \frac{\Gamma\left(\frac{1-\alpha\kappa_1}{2-2\alpha}\right)\Gamma\left(\frac{2\kappa_1-\alpha\kappa_1-1}{2-2\alpha}\right)}{\beta_2^{\kappa_1}(2-2\alpha)\Gamma(\kappa_1)} \left(\frac{\beta_2}{\beta_3}\right)^{\frac{1-\alpha\kappa_1}{2-2\alpha}}.$$

The upper bound of the settling time T_1 *and the residual set* Ω *can be respectively denoted as*

$$\begin{aligned} T_1 &= \min\left\{T_0(1-\delta)^{\frac{1+\alpha\kappa_1-2\kappa_1}{2-2\alpha}}, T_0(1-\delta)^{\frac{\alpha\kappa_1-1}{2-2\alpha}}\right\}, \\ \Omega &= \min\left\{\left[\frac{\Delta}{(\kappa_0\beta_2\delta)^{\kappa_1}}\right]^{\frac{1}{\alpha\kappa_1}}, \left[\frac{\Delta}{(\kappa_0\beta_3\delta)^{\kappa_1}}\right]^{\frac{1}{\alpha\kappa_1}}\right\}, \end{aligned} \tag{7.36}$$

where $\delta \in (0,1)$ is a positive scalar.

Proof 7.5 *Eq. (7.35) can be regiven as*

$$\dot{V} \leq -\kappa_0 \left[2\beta_1 V + \beta_2(1-\delta)V^{\alpha} + \beta_3 V^{2-\alpha} + \delta\beta_2 V^{\alpha}\right]^{\kappa_1} + \Delta \tag{7.37a}$$

$$\dot{V} \leq -\kappa_0 \left[2\beta_1 V + \beta_2 V^{\alpha} + \beta_3(1-\delta)V^{2-\alpha} + \delta\beta_3 V^{2-\alpha}\right]^{\kappa_1} + \Delta. \tag{7.37b}$$

Case (1): For Eq. (7.37a), assuming $-\kappa_0\beta_2\delta V^{\alpha\kappa_1} + \Delta \leq 0$ yields $V \geq \left[\frac{\Delta}{(\kappa_0\beta_2\delta)^{\kappa_1}}\right]^{\frac{1}{\alpha\kappa_1}}$ and

$$\begin{aligned}\dot{V} &\leq -\kappa_0 \left[2\beta_1 V + \beta_2(1-\delta)V^{\alpha} + \beta_3 V^{2-\alpha}\right]^{\kappa_1} - \kappa_0\delta V^{\alpha} + \Delta \\ &\leq -\kappa_0 \left[2\beta_1 V + \beta_2(1-\delta)V^{\alpha} + \beta_3 V^{2-\alpha}\right]^{\kappa_1}.\end{aligned} \tag{7.38}$$

By replacing the 'β_2' in Corollary 7.2 with '$\beta_2/(1-\delta)$', we know that the system is practically PdT stable, and the settling time is bounded by

$$T_{11} = T_0(1-\delta)^{\frac{1+\alpha\kappa_1-2\kappa_1}{2-2\alpha}}. \tag{7.39}$$

Case (2): For Eq. (7.37b), similarly, assuming $-\kappa_0\beta_3\delta V^{\alpha\kappa_1} + \Delta \leq 0$ yields $V \geq \left[\frac{\Delta}{(\kappa_0\beta_3\delta)^{\kappa_1}}\right]^{\frac{1}{\alpha\kappa_1}}$ and

$$\begin{aligned}\dot{V} &\leq -\kappa_0 \left[2\beta_1 V + \beta_2 V^{\alpha} + \beta_3(1-\delta)V^{2-\alpha}\right]^{\kappa_1} - \kappa_0\delta V^{\alpha} + \Delta \\ &\leq -\kappa_0 \left[2\beta_1 V + \beta_2 V^{\alpha} + \beta_3(1-\delta)V^{2-\alpha}\right]^{\kappa_1}.\end{aligned} \tag{7.40}$$

By replacing the 'β_3' in Corollary 7.2 with '$\beta_3/(1-\delta)$', it indicates that the system is practically PdT stable, and the settling time is bounded by

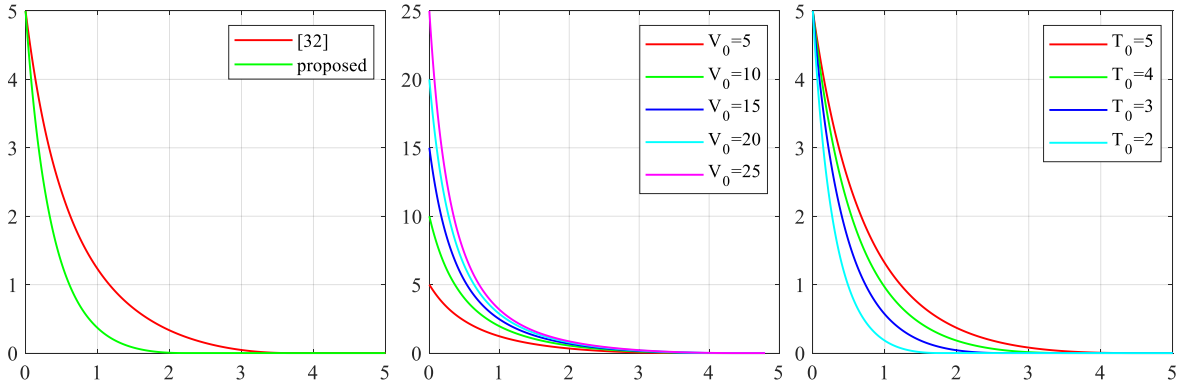
$$T_{12} = T_0(1-\delta)^{\frac{\alpha\kappa_1-1}{2-2\alpha}}. \tag{7.41}$$

Therefore T_1 can be bounded by $T_1 = \min(T_{11}, T_{12})$, and the residual set can be given by

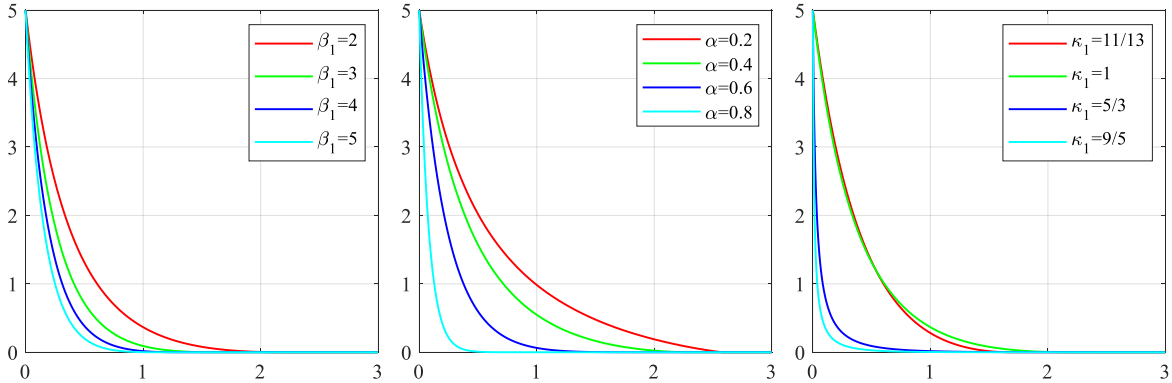
$$\Omega = \min\left\{\left[\frac{\Delta}{(\kappa_0\beta_2\delta)^{\kappa_1}}\right]^{\frac{1}{\alpha\kappa_1}}, \left[\frac{\Delta}{(\kappa_0\beta_3\delta)^{\kappa_1}}\right]^{\frac{1}{\alpha\kappa_1}}\right\}.$$

This completes the proof.

Remark 7.3 *It can be easily computed that both $\frac{1+\alpha\kappa_1-2\kappa_1}{2-2\alpha}$ and $\frac{\alpha\kappa_1-1}{2-2\alpha}$ are negative and $1-\delta \in (0,1)$. Therefore, it indicates that $T_1 = T_{11}$ when $0 < \kappa_1 < 1$ and $T_1 = T_{12}$ when $\kappa_1 > 1$.*



(a) $\beta_1=\beta_2=\beta_3=\kappa_1=1$, $\alpha=0.5$, $T_0=5$, $V_0=5$. (b) $\beta_1=\beta_2=\beta_3=\kappa_1=1$, $\alpha=0.5$, $T_0=5$. (c) $\beta_1=\beta_2=\beta_3=\kappa_1=1$, $\alpha=0.5$, $V_0=5$.



(d) $\beta_2=\beta_3=\kappa_1=1$, $\alpha=0.5$, $T_0=5$, $V_0=5$. (e) $\beta_1=\beta_2=\beta_3=\kappa_1=1$, $T_0=5$, $V_0=5$. (f) $\beta_1=\beta_2=\beta_3=1$, $\alpha=0.5$, $T_0=5$, $V_0=5$.

Figure 7.1: Comparative simulation of Theorem 7.1 with different conditions.

Remark 7.4 *Compared to Lemma 7.7 (Theorem 1 in [181]), the PdT convergence system proposed in this study 1) has a faster convergence speed and 2) has a more general formation. By selecting* $\kappa_1=1$*, the convergence time difference between the two methods can be defined as* $\Delta T=T_0-T_c=\frac{T_0 2(1-\alpha)\sqrt{\beta_2\beta_3}}{\pi}\int_0^{V_0}\frac{1}{2\beta_1 V+\beta_2 V^{\alpha}+\beta_3 V^{2-\alpha}}-\frac{1}{\beta_2 V^{\alpha}+\beta_3 V^{2-\alpha}}\leq 0$*, which indicates that the convergence process introduced in this chapter is faster than that of in [181]. Furthermore,* κ_1 *is not strictly constrained to be equal to* 1 *but is instead restricted within a reasonable range, thereby generalizing the form presented in [181]. Meanwhile,* 1 *is a valid choice because* $0<\alpha<1$*. Designers can achieve improved convergence characteristics by fine-tuning* κ_1 *around* 1*.*

Fig. 7.1 provides some guidance for tuning the system's parameters. Fig. 7.1a indicates that the V converges faster in our proposed method by adding a linear term into the differential equation. In Fig. 7.1b, we can see that the system converges to the origin within the PdT T_0 regardless of the system's initial state. Subsequently, as demonstrated in Figs 7.1c, 7.1d, and 7.1e, the system exhibits faster convergence when T_0 is smaller, β_1 is larger, and α is larger. However, the adjustment pattern

of κ_1 appears slightly different. It is evident that increasing κ_1 will accelerate the convergence of the system when $\kappa > 1$. Nevertheless, the convergence curves corresponding to different κ_1 values exhibit two intersection points when $\kappa_1 < 1$, indicating that the system's performance can be slightly fine-tuned around $\kappa_1 = 1$ to achieve better results.

Note that although Fig. 7.1 demonstrates methods to accelerate system convergence through comparative simulations, indiscriminately minimizing convergence time is not necessarily advisable in practical applications. This is because the maximum acceleration achievable by a physical system is ultimately constrained by its inherent physical limits. Excessively rapid convergence may risk damaging the controller or inducing unnecessary overshoot. Therefore, in practical implementations, it is crucial to adjust the system gains judiciously based on specific requirements. This aspect will be further elaborated in Remark 7.7.

7.4 Controller design

7.4.1 Translational subsystem stability

Considering the homogeneity among quadrotors and the observer design does not rely on information from other quadrotors, the subscript 'i' corresponding to individual quadrotors is omitted in the observer design. Instead, for ease of derivation and clarity, we use the subscript $j = x, y, z$ to represent the components along the x, y, and z axes in the observer design process. By Eq. (7.18), we respectively define z_η as the estimation of $\dot{\eta}$, $\hat{\Delta}_\eta$ as the estimation of Δ_η, and $\hat{\sigma}_\eta$ as the estimation of σ_η. Correspondingly, $\tilde{z}_\eta = \dot{\eta} - z_\eta$, $\tilde{\Delta}_\eta = \Delta_\eta - \hat{\Delta}_\eta$, and $\tilde{\sigma}_\eta = \sigma_\eta - \hat{\sigma}_\eta$ are the corresponding estimation errors. Then, a PdT observer can be designed as

$$\begin{aligned}
\dot{z}_\eta =& \kappa_{\eta 0}\Big(2a_{\eta 1}\beta_{\eta 1}\left\lfloor \tilde{z}_\eta \right\rceil^{2-\frac{1}{\kappa_{\eta 1}}} + a_{\eta 2}\beta_{\eta 2}\left\lfloor \tilde{z}_\eta \right\rceil^{2\alpha_\eta-\frac{1}{\kappa_{\eta 1}}} + a_{\eta 3}\beta_{\eta 3}\left\lfloor \tilde{z}_\eta \right\rceil^{4-2\alpha_\eta-\frac{1}{\kappa_{\eta 1}}}\Big)^{\kappa_{\eta 1}} \\
&+ \hat{\sigma}_\eta \operatorname{sgn}(\tilde{z}_\eta) - \frac{k_t}{m_i}\dot{\eta} + u_\eta, \\
\dot{\hat{\Delta}}_\eta =& \kappa_{\eta 0}\Big(2a_{\eta 1}\beta_{\eta 1}\left\lfloor \tilde{z}_\eta \right\rceil^{2-\frac{1}{\kappa_{\eta 1}}} + a_{\eta 2}\beta_{\eta 2}\left\lfloor \tilde{z}_\eta \right\rceil^{2\alpha_\eta-\frac{1}{\kappa_{\eta 1}}} + a_{\eta 3}\beta_{\eta 3}\left\lfloor \tilde{z}_\eta \right\rceil^{4-2\alpha_\eta-\frac{1}{\kappa_{\eta 1}}}\Big)^{\kappa_{\eta 1}} + \hat{\sigma}_\eta, \\
\dot{\hat{\sigma}}_\eta =& -\kappa_{\eta 0}\Big(2b_{\eta 1}\beta_{\eta 1}\hat{\sigma}_\eta^{2-\frac{1}{\kappa_{\eta 1}}} + b_{\eta 2}\beta_{\eta 2}\hat{\sigma}_\eta^{2-\frac{1}{\kappa_{\eta 1}}} + b_{\eta 3}\beta_{\eta 3}\hat{\sigma}_\eta^{4-2\alpha_\eta-\frac{1}{\kappa_{\eta 1}}}\Big)^{\kappa_{\eta 1}} + \tilde{z}_\eta \operatorname{sgn}(\tilde{z}_\eta),
\end{aligned} \tag{7.42}$$

where positive parameters $a_{\eta 1} = \frac{1}{2}$, $b_{\eta 1} = 1$, $a_{\eta 2} = 2^{-\alpha_\eta}$, $b_{\eta 2} = \frac{1-\alpha_\eta}{2^{\alpha_\eta-1}}$, $a_{\eta 3} = 2^{\alpha_\eta-2}6^{1-\alpha_\eta}$, $b_{\eta 3} = \frac{4-2\alpha_\eta}{3-2\alpha_\eta}2^{\alpha_\eta-2}6^{1-\alpha_\eta}$, $\kappa_{\eta 1} = 1$, $0 < \alpha_\eta < 1$, $\beta_{\eta 2}, \beta_{\eta 3} > 0$, $\beta_{\eta 1}^2 \geq \beta_{\eta 2}\beta_{\eta 3}$, and

$$\kappa_{\eta 0} = \frac{\Gamma\left(\frac{1-\alpha_\eta\kappa_{\eta 1}}{2-2\alpha_\eta}\right)\Gamma\left(\frac{2\kappa_{\eta 1}-\alpha_\eta\kappa_{\eta 1}-1}{2-2\alpha_\eta}\right)}{T_{\eta 0}\beta_{\eta 2}^{\kappa_{\eta 1}}(2-2\alpha_\eta)\Gamma(\kappa_{\eta 1})}\left(\frac{\beta_{\eta 2}}{\beta_{\eta 3}}\right)^{\frac{1-\alpha_\eta\kappa_{\eta 1}}{2-2\alpha_\eta}}.$$

The following theorem can be concluded.

Theorem 7.2 *For system (7.18), we design a PdTDO (7.42) to simultaneously estimate the external disturbances and the upper bound of the disturbances. Then, the following conditions hold:*

(1). The estimation error of σ_η can be bounded as $||\tilde{\sigma}_\eta|| \leq N_{\eta 0,i}$ in a PdT $T_{\eta 1} = \frac{T_{\eta 0}}{\sqrt{1-\delta_\eta}}$, where $\delta_\eta \in (0,1)$ is a positive scalar,

$$\Lambda_\eta = \kappa_{\eta 0}\left[\sigma_\eta^2\beta_{\eta 1} + \frac{(1-\alpha_\eta)\beta_{\eta 2}}{2_\eta^\alpha}\sigma_\eta^2 + 2^{\alpha_\eta - 1}6^{1-\alpha_\eta}\beta_{\eta 3}\sigma_\eta^{4-2\alpha_\eta} + \frac{1-\alpha_\eta}{2^{\alpha_\eta}}\left(\frac{\alpha_\eta}{1-\alpha_\eta}\right)^{\frac{\alpha_\eta}{1-\alpha_\eta}}\right]$$

$$\Omega_\eta = \min\left\{\left(\frac{\Lambda_\eta}{\beta_{\eta 2}\delta_\eta}\right)^{\frac{1}{\alpha_\eta}}, \left(\frac{\Lambda_\eta}{\beta_{\eta 3}\delta_\eta}\right)^{\frac{1}{\alpha_\eta}}\right\},$$

and $N_{\eta 0} = \max\{||x||_2 : x \in \Omega_\eta\}$.

(2). The estimation error of Δ_η can be bounded as $||\tilde{\Delta}_\eta||_2 \leq N_\eta$ in a PdT $T_{\eta 1}$, where

$$N_\eta = 2^{\frac{1}{2}}\left(1+\kappa_{\eta 0}\beta_{\eta 1}\right)N_{\eta 0}^{\frac{1}{2}} + 2^{-\frac{1}{2}}\kappa_{\eta 0}\beta_{\eta 2} + N_{\eta 0}^{\frac{2\alpha_\eta - 1}{2}} + 2^{-\frac{1}{2}}6^{1-\alpha}\kappa_{\eta 0}\beta_{\eta 3}N_{\eta 0}^{\frac{3-2\alpha_\eta}{2}} + \sigma_\eta$$

Proof 7.6 *We define $V_{j1} = \frac{1}{2}\tilde{z}_{\eta j}^2$, $V_{j2} = \frac{1}{2}\,\tilde{\sigma}_{\eta j}^2$. take Lyapunov candidate function as $V = \sum_{j=x,y,z}(V_{j1}+V_{j2})$. Differentiating V, substituting Eqs. (7.18) and (7.42) into $\dot{V}$ and doing some manipulations yields*

$$\begin{aligned}
\dot{V} =& \sum_j \tilde{z}_{\eta j}\Big[\Delta_{\eta j} - \hat{\sigma}_{\eta j}\,\mathrm{sgn}(\tilde{z}_{\eta j}) - \kappa_{\eta 0}\Big(2a_{\eta 1}\beta_{\eta 1}\left\lfloor\tilde{z}_{\eta j}\right\rceil^{2-\frac{1}{\kappa_{\eta 1}}} \\
&+ a_{\eta 2}\beta_{\eta 2}\left\lfloor\tilde{z}_{\eta j}\right\rceil^{2\alpha_\eta - \frac{1}{\kappa_{\eta 1}}} + a_{\eta 3}\beta_{\eta 3}\left\lfloor\tilde{z}_{\eta j}\right\rceil^{4-2\alpha_\eta - \frac{1}{\kappa_{\eta 1}}}\Big)^{\kappa_{\eta 1}}\Big] \\
&- \tilde{\sigma}_{\eta j}\Big[\tilde{z}_{\eta j}\,\mathrm{sgn}(\tilde{z}_{\eta j}) - \kappa_{\eta 0}\Big(2b_{\eta 1}\beta_{\eta 1}\hat{\sigma}_{\eta j}^{2-\frac{1}{\kappa_{\eta 1}}} \\
&+ b_{\eta 2}\beta_{\eta 2}\hat{\sigma}_{\eta j}^{2-\frac{1}{\kappa_{\eta 1}}} + b_{\eta 3}\beta_{\eta 3}\hat{\sigma}_{\eta j}^{4-2\alpha_\eta - \frac{1}{\kappa_{\eta 1}}}\Big)^{\kappa_{\eta 1}}\Big] \\
\leq& \sum_j \Upsilon_{\eta j} + \kappa_{\eta 0}\Big(2b_{\eta 1}\beta_{\eta 1}\Upsilon_{1j}^{\frac{1}{\kappa_{\eta 1}}} + b_{\eta 2}\beta_{\eta 2}\Upsilon_{2j}^{\frac{1}{\kappa_{\eta 1}}} + b_{\eta 3}\beta_{\eta 3}\Upsilon_{3j}^{\frac{1}{\kappa_{\eta 1}}}\Big)^{\kappa_{\eta 1}},
\end{aligned} \tag{7.43}$$

where $\Upsilon_{\eta j} = -\kappa_{\eta 0}\big(2a_{\eta 1}\beta_{\eta 1}\tilde{z}_{\eta j}^2 + a_{\eta 2}\beta_{\eta 2}\tilde{z}_{\eta j}^{2\alpha_\eta} + a_{\eta 3}\beta_{\eta 3}\tilde{z}_{\eta j}^{4-2\alpha_\eta}\big)^{\kappa_{\eta 1}}$, $\Upsilon_{1j} = \tilde{\sigma}_{\eta j}(\sigma_\eta - \tilde{\sigma}_{\eta j})^{2\kappa_{\eta 1}-1}$, $\Upsilon_{2j} = \tilde{\sigma}_{\eta j}(\sigma_\eta - \tilde{\sigma}_{\eta j})^{2\kappa_{\eta 1}-1}$, and $\Upsilon_{3j} = \tilde{\sigma}_{\eta j}(\sigma_\eta - \tilde{\sigma}_{\eta j})^{(4-2\alpha_\eta)\kappa_{\eta 1}-1}$.

In order to use Lemma 7.3 and Lemma 7.4, the following inequalities have to be satisfied

$$2\kappa_{\eta 1} - 1 \geq 1,\ (4-2\alpha_\eta)\kappa_{\eta 1} - 1 \geq 1,\ 0 < \alpha_\eta < 1. \tag{7.44}$$

Solving (7.44) yields $\kappa_{\eta 1} \geq 1$. *Using Lemma 7.3 and Lemma 7.4 in* Υ_{1j} *and* Υ_{3j} *yields*

$$\begin{aligned} \Upsilon_{1j} &\leq \frac{2\kappa_{\eta 1}-1}{2\kappa_{\eta 1}}\left(\sigma_{\eta}^{2\kappa_{\eta 1}}-\tilde{\sigma}_{\eta j}^{2\kappa_{\eta 1}}\right), \\ \Upsilon_{3j} &\leq \frac{(4-2\alpha_{\eta})\kappa_{\eta 1}-1}{(4-2\alpha_{\eta})\kappa_{\eta 1}}\left[\sigma_{\eta}^{(4-2\alpha_{\eta})\kappa_{\eta 1}}-\tilde{\sigma}_{\eta j}^{(4-2\alpha_{\eta})\kappa_{\eta 1}}\right], \end{aligned} \tag{7.45}$$

Using Eq. (7.45) in Eq. (7.43), we have

$$\begin{aligned} \dot{V} \leq \sum_{j}\Upsilon_{\eta j}+\kappa_{\eta 0}\Bigg\{ & 2b_{\eta 1}\beta_{\eta 1}\big(1-\frac{1}{\kappa_{\eta 1}}\big)^{\frac{1}{\kappa_{\eta 1}}}\times\left(\sigma_{\eta}^{2\kappa_{\eta 1}}-\tilde{\sigma}_{\eta j}^{2\kappa_{\eta 1}}\right)^{\frac{1}{\kappa_{\eta 1}}}+b_{\eta 2}\beta_{\eta 2}\Upsilon_{2j}^{\frac{1}{\kappa_{\eta 1}}} \\ & +b_{\eta 3}\beta_{\eta 3}\Big[1-\frac{1}{(4-2\alpha_{\eta})\kappa_{\eta 1}}\Big]^{\frac{1}{\kappa_{\eta 1}}}\times\left[\sigma_{\eta}^{(4-2\alpha_{\eta})\kappa_{\eta 1}}-\tilde{\sigma}_{\eta j}^{(4-2\alpha_{\eta})\kappa_{\eta 1}}\right]^{\frac{1}{\kappa_{\eta 1}}}\Bigg\}^{\kappa_{\eta 1}}, \end{aligned} \tag{7.46}$$

Remark 7.5 *Note that to make the inequality* $(\sigma_{\eta}^{2\bar{n}}-\tilde{\sigma}_{\eta j}^{2\bar{n}})^{\frac{1}{\kappa_{\eta 1}}} \leq (\sigma_{\eta}^{2\bar{n}})^{\frac{1}{\kappa_{\eta 1}}}-(\tilde{\sigma}_{\eta j}^{2\bar{n}})^{\frac{1}{\kappa_{\eta 1}}}$ *holds for all* $\sigma_{\eta}^{2\bar{n}},\tilde{\sigma}_{\eta j}^{2\bar{n}} \in \mathbb{R}^{+}$ *with* $\bar{n}=\kappa_{\eta 1},(2-\alpha_{\eta})\kappa_{\eta 1}$ *and* $j=x,y,z$. *With the help of Lemma 7.6, we know conditions* $\frac{1}{\kappa_{\eta 1}} \geq 1$ *and* $\frac{1}{\kappa_{\eta 1}} \leq 1$ *must be satisfied simultaneously. Therefore, we only have* $\kappa_{\eta 1}=1$.

For Lemma 7.5, let

$$\begin{aligned} & x_1=1,\ x_2=\tilde{\sigma}_{\eta j}^{2},\ \varpi_1=\frac{(3-2\alpha_{\eta})\kappa_{\eta 1}-1}{2\kappa_{\eta 1}},\ \varpi_2=1-\varpi_1, \\ & \vartheta=\Big[\frac{(2\alpha_{\eta}-1)\kappa_{\eta 1}+1}{(3-2\alpha_{\eta})\kappa_{\eta 1}-1}\Big]^{\frac{(2\alpha_{\eta}-1)\kappa_{\eta 1}+1}{(3-2\alpha_{\eta})\kappa_{\eta 1}-1}}. \end{aligned}$$

Substituting $\kappa_{\eta 1}=1$ *into Lemma 7.5 and doing some manipulations, we have* $\varpi_1=1-\alpha_{\eta}$, $\varpi_2=\alpha_{\eta}$, $\vartheta=(\frac{\alpha_{\eta}}{1-\alpha_{\eta}})^{\frac{\alpha_{\eta}}{1-\alpha_{\eta}}}$, $\Upsilon_{2j}=\tilde{\sigma}_{\eta j}(\sigma_{\eta}-\tilde{\sigma}_{\eta j})$ *and*

$$(\tilde{\sigma}_{\eta j}^{2})^{\alpha_{\eta}} \leq (1-\alpha_{\eta})\tilde{\sigma}_{\eta j}^{2}+(1-\alpha_{\eta})\big(\frac{\alpha_{\eta}}{1-\alpha_{\eta}}\big)^{\frac{\alpha_{\eta}}{1-\alpha_{\eta}}}. \tag{7.47}$$

Using Young's inequality on Υ_{2j} *and substituting Eq. (7.47) into it yield*

$$\begin{aligned} \Upsilon_{2j} &= \tilde{\sigma}_{\eta j}(\sigma_{\eta}-\tilde{\sigma}_{\eta j}) \\ &\leq \frac{1}{2}\sigma_{\eta}^{2}-\frac{1}{2}\tilde{\sigma}_{\eta j}^{2} \\ &\leq \frac{1}{2}\sigma_{\eta}^{2}-\frac{1}{2(1-\alpha_{\eta})}(\tilde{\sigma}_{\eta j}^{2})^{\alpha_{\eta}}+\frac{1}{2}\big(\frac{\alpha_{\eta}}{1-\alpha_{\eta}}\big)^{\frac{\alpha_{\eta}}{1-\alpha_{\eta}}}. \end{aligned} \tag{7.48}$$

For convenience of derivation, we define $\Lambda_{\eta j}=\kappa_{\eta 0}\big[b_{\eta 1}\beta_{\eta 1}\sigma_{\eta}^{2}+b_{\eta 2}\beta_{\eta 2}\big(\frac{1}{2}\sigma_{\eta}^{2}+\frac{1}{2}\big(\frac{\alpha_{\eta}}{1-\alpha_{\eta}}\big)^{\frac{\alpha_{\eta}}{1-\alpha_{\eta}}}\big)+$

$\frac{3-2\alpha_\eta}{4-2\alpha_\eta} b_{\eta 3}\beta_{\eta 3}\sigma_\eta^{4-2\alpha_\eta}]$, *substitute* $\kappa_{\eta 1}=1$, *Eq. (7.45) and Eq. (7.48) back to Eq. (7.43), and do some manipulations on* $\dot{V}$, *and then there is*

$$\begin{aligned}\dot{V} \leq & -\kappa_{\eta 0}\sum_j \left[4a_{\eta 1}\beta_{\eta 1}V_{j1}+2^{\alpha_\eta}a_{\eta 2}\beta_{\eta 2}V_{j1}^{\alpha_\eta}+2^{2-\alpha_\eta}a_{\eta 3}\beta_{\eta 3}V_{j1}^{2-\alpha}\right] \\ & -\kappa_{\eta 0}\sum_j \left[2b_{\eta 1}\beta_{\eta 1}V_{j2}+\frac{2^{\alpha_\eta-1}}{1-\alpha_\eta}b_{\eta 2}\beta_{\eta 2}V_{j2}^{\alpha_\eta}+\frac{3-2\alpha_\eta}{4-2\alpha_\eta}2^{2-\alpha_\eta}b_{\eta 3}\beta_{\eta 3}V_{j2}^{2-\alpha_\eta}\right]+\Lambda_{\eta j}.\end{aligned} \tag{7.49}$$

Inspired by the left side of Lemma 7.1, we set

$$a_{\eta 1}=\frac{1}{2},\ a_{\eta 2}=2^{-\alpha_\eta},\ b_{\eta 1}=1,\ b_{\eta 2}=\frac{1-\alpha_\eta}{2^{\alpha_\eta-1}}. \tag{7.50}$$

Substituting a_i *and* b_i, $i=1,2$ *back to Eq. (7.49) and using Lemma 7.1,* $\dot{V}$ *can be further simplified as*

$$\begin{aligned}\dot{V} \leq & -\kappa_{\eta 0}\left(2\beta_1 V+\beta_2 V^{\alpha_\eta}\right)-\kappa_{\eta 0}\sum_j 2^{2-\alpha_\eta}a_{\eta 3}\beta_{\eta 3}V_{j1}^{2-\alpha_\eta} \\ & +\frac{3-2\alpha_\eta}{4-2\alpha_\eta}2^{2-\alpha_\eta}b_{\eta 3}\beta_{\eta 3}V_{j2}^{2-\alpha_\eta}+\Lambda_{\eta j}.\end{aligned} \tag{7.51}$$

Thereafter, to make Theorem 7.2 hold, inspired by Lemma 7.2, we set

$$a_{\eta 3}=2^{\alpha_\eta-2}6^{1-\alpha_\eta},\ b_{\eta 3}=\frac{4-2\alpha_\eta}{3-2\alpha_\eta}2^{\alpha_\eta-2}6^{1-\alpha_\eta} \tag{7.52}$$

By substituting a_3 *and* b_3 *back to* $\dot{V}$, *we have*

$$\dot{V} \leq -\kappa_{\eta 0}\left(2\beta_{\eta 1}V+\beta_{\eta 2}V^{\alpha_\eta}+\beta_{\eta 3}V^{2-\alpha_\eta}\right)^{\kappa_{\eta 1}}+||\Lambda_\eta||_1 \tag{7.53}$$

with $\kappa_{\eta 1}=1$, $\kappa_{\eta 0}=\frac{\pi\sqrt{\beta_{\eta 2}/\beta_{\eta 3}}}{T_0\beta_{\eta 1}(2-2\alpha_\eta)}$, *and* $\Lambda_{\eta j}=\kappa_{\eta 0}[\sigma_\eta^2\beta_{\eta 1}+\frac{(1-\alpha_\eta)\beta_{\eta 2}}{2_\eta^\alpha}\sigma_\eta^2+2^{\alpha_\eta-1}6^{1-\alpha_\eta}\beta_{\eta 3}\sigma_\eta^{4-2\alpha_\eta}+\frac{1-\alpha_\eta}{2^{\alpha_\eta}}(\frac{\alpha_\eta}{1-\alpha_\eta})^{\frac{\alpha_\eta}{1-\alpha_\eta}}]$, *satisfying the formation introduced in Corollary 7.2. Therefore, the estimation error of* $\dot{\eta}$ *and* σ_η *converge to a neighborhood of the origin* Ω_η *in a PdT* $T_{\eta 1}$, *where* Ω_η *and* $T_{\eta 1}$ *are respectively defined by*

$$\Omega_\eta=\min\left\{\left(\frac{\Lambda_\eta}{\beta_{\eta 2}\delta_\eta}\right)^{\frac{1}{\alpha_\eta}},\left(\frac{\Lambda_\eta}{\beta_{\eta 3}\delta_\eta}\right)^{\frac{1}{\alpha_\eta}}\right\},T_{\eta 1}=\frac{T_{\eta 0}}{\sqrt{1-\delta_\eta}},$$

where $\delta_\eta\in(0,1)$ *is a positive scalar. The boundary of* $\Omega_{\eta 0}$ *can be given by* $N_{\eta 0}=\max\{||x||_2 : x\in\Omega_{\eta 0}\}$. *Then, it clearly indicates that* $||\tilde{z}_\eta||,||\tilde{\sigma}_\eta||\leq\sqrt{2N_{\eta 0}}$. *Use observer (7.42), we have*

$$\tilde{\Delta}_\eta=-\kappa_{\eta 0}(\beta_{\eta 1}\left\lfloor\tilde{z}_\eta\right\rceil+2^{-\alpha_\eta}\beta_{\eta 2}\left\lfloor\tilde{z}_\eta\right\rceil^{2\alpha_\eta-1}+2^{\alpha_\eta-2}6^{1-\alpha_\eta}\beta_{\eta 3}\left\lfloor\tilde{z}_\eta\right\rceil^{3-2\alpha_\eta})+\Delta_\eta-\hat{\sigma}_\eta. \tag{7.54}$$

Then, there is

$$\begin{aligned}||\tilde{\Delta}_\eta|| \leq & \kappa_{\eta 0}(\beta_{\eta 1}||\tilde{z}_\eta|| + 2^{-\alpha_\eta}\beta_{\eta 2}||\tilde{z}_\eta||^{2\alpha_\eta - 1} + 2^{\alpha_\eta - 2}6^{1-\alpha_\eta}\beta_{\eta 3}||\tilde{z}_\eta||^{3-2\alpha_\eta}) + \sigma_\eta + \sqrt{2N_{\eta 0}} \\ \leq & 2^{\frac{1}{2}}\left(1+\kappa_{\eta 0}\beta_{\eta 1}\right)N_{\eta 0}^{\frac{1}{2}} + 2^{-\frac{1}{2}}\kappa_{\eta 0}\beta_{\eta 2} + N_{\eta 0}^{\frac{2\alpha_\eta - 1}{2}} + 2^{-\frac{1}{2}}6^{1-\alpha}\kappa_{\eta 0}\beta_{\eta 3}N_{\eta 0}^{\frac{3-2\alpha_\eta}{2}} + \sigma_\eta \triangleq N_\eta,\end{aligned} \tag{7.55}$$

which obviously indicates that $\tilde{\Delta}_\eta$ is bounded. This completes the proof.

Note that the topology of the quadrotor group is generally not a fully-connected graph. We have to analyze the stability of all quadrotors in the group simultaneously rather than separately. Therefore, we must distinguish different quadrotors using a subscript i, $i = 1, 2, \cdots, N$ in the controller design process.

After designing the observer, a PdT fast nonsingular terminal sliding mode surface is defined as

$$s_{\eta,i} = \dot{e}_{\eta,i} + \kappa_{\eta 2}\left(\lambda_{s1}\left\lfloor e_{\eta,i}\right\rceil^{2-\frac{1}{\kappa_{\eta 3}}} + \frac{\lambda_{s2}}{2}\left\lfloor e_{\eta,i}\right\rceil^{2\alpha_{s\eta}-\frac{1}{\kappa_{\eta 3}}} + \frac{\lambda_{s3}}{2}\left\lfloor e_{\eta,i}\right\rceil^{4-2\alpha_{s\eta}-\frac{1}{\kappa_{\eta 3}}}\right)^{\kappa_{\eta 3}}, \tag{7.56}$$

where $\lambda_{s2}, \lambda_{s3} > 0$, $\lambda_{s1} \geq \sqrt{\lambda_{s2}\lambda_{s3}}$, $0 < \alpha_{s\eta} < 1$, $\frac{1}{2-\alpha_{s\eta}} < \kappa_{\eta 3} < \frac{1}{\alpha_{s\eta}}$, and

$$\kappa_{\eta 2} = \frac{\Gamma\left(\frac{1-\alpha_{s\eta}\kappa_{\eta 3}}{2-2\alpha_{s\eta}}\right)\Gamma\left(\frac{2\kappa_{\eta 3}-\alpha_{s\eta}\kappa_{\eta 3}-1}{2-2\alpha_{s\eta}}\right)}{T_{\eta 2}\lambda_{s2}^{\kappa_{\eta 3}}(2-2\alpha_{s\eta})\Gamma(\kappa_{\eta 3})}\left(\frac{\lambda_{s2}}{\lambda_{s3}}\right)^{\frac{1-\alpha_{s\eta}\kappa_{\eta 3}}{2-2\alpha_{s\eta}}}.$$

For clarity and simplicity, in further derivations, we define

$$\mathscr{M}_{\eta,i} = \kappa_{\eta 2}\left(\lambda_{s1}\left\lfloor e_{\eta,i}\right\rceil^{2-\frac{1}{\kappa_{\eta 3}}} + \frac{\lambda_{s2}}{2}\left\lfloor e_{\eta,i}\right\rceil^{2\alpha_{s\eta}-\frac{1}{\kappa_{\eta 3}}} + \frac{\lambda_{s3}}{2}\left\lfloor e_{\eta,i}\right\rceil^{4-2\alpha_{s\eta}-\frac{1}{\kappa_{\eta 3}}}\right)^{\kappa_{\eta 3}}. \tag{7.57}$$

We have

$$\begin{aligned}\dot{\mathscr{M}}_{\eta,i} = & \kappa_{\eta 2}\kappa_{\eta 3}\left(\lambda_{s1}\left\lfloor e_{\eta,i}\right\rceil^{2-\frac{1}{\kappa_{\eta 3}}} + \frac{\lambda_{s2}}{2}\left\lfloor e_{\eta,i}\right\rceil^{2\alpha_{s\eta}-\frac{1}{\kappa_{\eta 1}}} + \frac{\lambda_{s3}}{2}\left\lfloor e_{\eta,i}\right\rceil^{4-2\alpha_{s\eta}-\frac{1}{\kappa_{\eta 3}}}\right)^{\kappa_{\eta 3}-1} \\ & \times\left[\frac{\lambda_{s1}(2\kappa_{\eta 3}-1)}{\kappa_{\eta 3}}\left\lfloor e_{\eta,i}\right\rceil^{\frac{\kappa_{\eta 3}-1}{\kappa_{\eta 3}}}\dot{e}_{\eta,i} + \frac{\lambda_{s2}(2\alpha_{s\eta}\kappa_{\eta 3}-1)}{2\kappa_{\eta 3}}\left\lfloor e_{\eta,i}\right\rceil^{\frac{(2\alpha_{s\eta}-1)\kappa_{\eta 3}-1}{\kappa_{\eta 3}}}\dot{e}_{\eta,i}\right. \\ & \left. + \frac{\lambda_{s3}(4-2\alpha_{s\eta})\kappa_{\eta 3}-1}{2\kappa_{\eta 3}}\left\lfloor e_{\eta,i}\right\rceil^{\frac{(3-2\alpha_{s\eta})\kappa_{\eta 3}-1}{\kappa_{\eta 3}}}\dot{e}_{\eta,i}\right].\end{aligned} \tag{7.58}$$

We assume no external disturbances are acting on the quadrotors. An equivalent control law needs to be designed to pull $s_{\eta,i}$ to the sliding mode surface, which can be given by

$$u_{eq,i} = -\frac{1}{b_i+d_i}\left[-\frac{(b_i+d_i)k_t}{m_i}\dot{\eta}_i - \Lambda_{i0} + \dot{\mathscr{M}}_{\eta,i}\right]. \tag{7.59}$$

Considering the effect of disturbances $\Delta_{\eta,i}$, a switching control law is further required to maintain $s_{\eta,i}$

on the sliding mode surface, which can be put forth as

$$\begin{aligned} u_{sw,i} = &-\frac{1}{b_i+d_i}\Big[(b_i+d_i)\hat{\Delta}_{\eta,i} - \sum_{j=1}^{N} a_{ij}\hat{\Delta}_{\eta,j} + k_{d,i}\,\mathrm{sgn}\,s_{\eta,i} \\ &+ \kappa_{\eta 4}\Big(\lambda_{u1}\left\lfloor s_{\eta,i}\right\rceil^{2-\frac{1}{\kappa_{\eta 5}}} + \frac{\lambda_{u2}}{2}\left\lfloor s_{\eta,i}\right\rceil^{2\alpha_{u\eta}-\frac{1}{\kappa_{\eta 3}}} + \frac{\lambda_{u3}}{2}\left\lfloor s_{\eta,i}\right\rceil^{4-2\alpha_{u\eta}-\frac{1}{\kappa_{\eta 5}}}\Big)^{\kappa_{\eta 5}}\Big], \end{aligned} \tag{7.60}$$

where $\lambda_{u2},\lambda_{u3}>0$, $\lambda_{u1}\geq\sqrt{\lambda_{u2}\lambda_{u3}}$, $0<\alpha_{u\eta}<1$, $\frac{1}{2-\alpha_{u\eta}}<\kappa_{\eta 5}<\frac{1}{\alpha_{u\eta}}$,

$$\kappa_{\eta 4} = \frac{\Gamma\left(\frac{1-\alpha_{u\eta}\kappa_{\eta 5}}{2-2\alpha_{u\eta}}\right)\Gamma\left(\frac{2\kappa_{\eta 5}-\alpha_{u\eta}\kappa_{\eta 5}-1}{2-2\alpha_{u\eta}}\right)}{T_{\eta 3}\lambda_{u2}^{\kappa_{\eta 5}}(2-2\alpha_{u\eta})\Gamma(\kappa_{\eta 5})}\left(\frac{\lambda_{u2}}{\lambda_{u3}}\right)^{\frac{1-\alpha_{u\eta}\kappa_{\eta 5}}{2-2\alpha_{u\eta}}},$$

and $k_{d,i}\geq(b_i+d_i+N)N_\eta$ is a positive scalar with N being the number of the quadrotor.

Then, the complete control law can be designed as

$$u_{\eta,i} = u_{eq,i} + u_{sw,i}. \tag{7.61}$$

Then, the following theorem can be given to guarantee the PdT stability of the translational subsystems of the quadrotor group.

Theorem 7.3 *For the consensus error dynamics given as Eq. (7.24), we design a PdTDO (7.42) and a PdT control law (7.61). Then, the translational subsystems of the quadrotor group are PdT stable, and the settling time can be bounded by* $T_\eta = T_{\eta 1} + T_{\eta 2} + T_{\eta 3}$.

Proof 7.7 *To begin with, we need to prove that* $s_{\eta,i}$ *converge to the origin in a PdT* $T_{\eta 1}$. *Define a Lyapunov function candidate as*

$$V_s = \sum_{i=1}^{N}\frac{1}{2}s_{\eta,i}^{\top}s_{\eta,i}. \tag{7.62}$$

Differentiating V_s *along the system trajectory and substituting* $u_{\eta,i}$ *into* $\dot{V}_s$ *yield*

$$\dot{V}_s = \sum_{i=1}^{N} s_{\eta,i}^{\top}\Big[-\frac{(d_i+b_i)k_t}{m_i}\dot{\eta}_i + (b_i+d_i)u_{\eta,i} - \Lambda_{i0} + (b_i+d_i)\Delta_{\eta,i} - \sum_{j=1}^{N} a_{ij}\Delta_{\eta,j} + \mathscr{M}_{\eta,i}\Big] \tag{7.63}$$

Substituting $u_{\eta,i}$ into $\dot{V}_s$ and doing some manipulations, we have

$$\begin{aligned}
\dot{V}_s &= \sum_{i=1}^{N} s_{\eta,i}^{\top}\Big[(b_i+d_i)\Delta_{\eta,i} - \sum_{j=1}^{N} a_{ij}\Delta_{\eta,j} + (b_i+d_i)u_{sw,i}\Big] \\
&= \sum_{i=1}^{N} s_{\eta,i}^{\top}\Big[(b_i+d_i)\Delta_{\eta,i} - \sum_{j=1}^{N} a_{ij}\Delta_{\eta,j} - (b_i+d_i)\hat{\Delta}_{\eta,i} + \sum_{j=1}^{N} a_{ij}\hat{\Delta}_{\eta,j} - k_{d,i}\,\mathrm{sgn}(s_{\eta,i}) \\
&\quad - \kappa_{\eta 4}\Big(\lambda_{u1}\left\lfloor s_{\eta,i}\right\rceil^{2-\frac{1}{\kappa_{\eta 5}}} + \frac{\lambda_{u2}}{2}\left\lfloor s_{\eta,i}\right\rceil^{2\alpha_{u\eta}-\frac{1}{\kappa_{\eta 5}}} + \frac{\lambda_{u3}}{2}\left\lfloor s_{\eta,i}\right\rceil^{4-2\alpha_{u\eta}-\frac{1}{\kappa_{\eta 5}}}\Big)^{\kappa_{\eta 5}}\Big] \\
&= -\sum_{i=1}^{N} \kappa_{\eta 4}\Big(\lambda_{u1}||s_{\eta,i}||_2^2 + \frac{\lambda_{u2}}{2}||s_{\eta,i}||_2^{2\alpha_{u\eta}} + \frac{\lambda_{u3}}{2}||s_{\eta,i}||_2^{4-2\alpha_{u\eta}}\Big)^{\kappa_{\eta 5}} \\
&\quad + \sum_{i=1}^{N} s_{\eta,i}^{\top}\Big[(b_1+d_i)\tilde{\Delta}_{\eta,i} - \sum_{j=1}^{N} a_{ij}\tilde{\Delta}_{\eta,j} - k_d\,\mathrm{sgn}(s_{\eta,i})\Big].
\end{aligned} \tag{7.64}$$

Using the fact $k_{d,i} \geq (b_i+d_i+N)N_d$ and Lemma 7.1, there is

$$\begin{aligned}
\dot{V}_s &\leq -\sum_{i=1}^{N} \kappa_{\eta 4}\Big(\lambda_{u1}||s_{\eta,i}||_2^2 + \frac{\lambda_{u2}}{2}||s_{\eta,i}||_2^{2\alpha_{u\eta}} + \frac{\lambda_{u3}}{2}||s_{\eta,i}||_2^{4-2\alpha_{u\eta}}\Big)^{\kappa_{\eta 5}} \\
&\leq -\kappa_{\eta 4}\Big(\lambda_{u1}\sum_{i=1}^{N}||s_{\eta,i}||_2^2 + \frac{\lambda_{u2}}{2}\sum_{i=1}^{N}||s_{\eta,i}||_2^{2\alpha_{u\eta}} + \sum_{i=1}^{N}||s_{\eta,i}||_2^{4-2\alpha_{u\eta}}\Big)^{\kappa_{\eta 5}} \\
&= -\kappa_{\eta 4}\big(2\lambda_{u1}V_s + \lambda_{u2}V_s^{\alpha_{u\eta}} + \lambda_{32}V_s^{2-\alpha_{u\eta}}\big)^{\kappa_{\eta 5}}.
\end{aligned} \tag{7.65}$$

We can see from Eq. (7.65) that V_s converges to the origin in a PdT $T_{\eta 2}$

Secondly, we need to prove that $e_{\eta,i}$ all converge to zero within a PdT $T_{\eta 2}$ after $s_{\eta,i} = 0$. On the sliding mode surface $s_{\eta,i} = 0$, there is

$$\dot{e}_{\eta,i} = -\kappa_{\eta 2}\Big(\lambda_{s1}\left\lfloor e_{\eta,i}\right\rceil^{2-\frac{1}{\kappa_{\eta 3}}} + \frac{\lambda_{s2}}{2}\left\lfloor e_{\eta,i}\right\rceil^{2\alpha_{s\eta}-\frac{1}{\kappa_{\eta 3}}} + \frac{\lambda_{s3}}{2}\left\lfloor e_{\eta,i}\right\rceil^{4-2\alpha_{s\eta}-\frac{1}{\kappa_{\eta 1}}}\Big)^{\kappa_{\eta 3}} \tag{7.66}$$

Define a Lyapunov function candidate as $V_e = \sum_{i=1}^{n}\frac{1}{2}e_{\eta,i}^{\top}e_{\eta,i}$. Differentiating V_e, using Lemma 7.1, and doing some manipulations yields

$$\begin{aligned}
\dot{V}_e &= -\kappa_{\eta 2}\sum_{i=1}^{N}\Big(\lambda_{s1}||e_{\eta,i}||_2^2 + \frac{\lambda_{s2}}{2}||e_{\eta,i}||_2^{2\alpha_{s\eta}} + \frac{\lambda_{s3}}{2}||e_{\eta,i}||_2^{4-2\alpha_{s\eta}}\Big)^{\kappa_{\eta 3}} \\
&\leq -\kappa_{\eta 2}\Big(2\lambda_{\eta 1}\sum_{i=1}^{N}||e_{\eta,i}||_2^2 + \lambda_{\eta 2}\sum_{i=1}^{N}||e_{\eta,i}||_2^{2\alpha_{s\eta}} + \lambda_{\eta 3}\sum_{i=1}^{N}||e_{\eta,i}||_2^{4-2\alpha_{s\eta}}\Big)^{\kappa_{\eta 3}} \\
&= -\kappa_{\eta 2}\big(2\lambda_{\eta 1}V_e + \lambda_{\eta}V_e^{\alpha_{s\eta}} + \lambda_{\eta 3}V_e^{2-\alpha_{s\eta}}\big)^{\kappa_{\eta 3}}.
\end{aligned} \tag{7.67}$$

Using Theorem 7.1 indicates that the consensus tracking errors $e_{\eta,i}$ converge the origin within a PdT

$T_{\eta 3}$.

Therefore, the closed-loop system is PdT stable, and the settling time can be bounded as $T_\eta = T_{\eta 1} + T_{\eta 2} + T_{\eta 3}$. *This completes the proof.*

Remark 7.6 *It is noteworthy that in Eq. 7.57, 7.58, and 7.60, the exponential-related terms may become negative when* α_η, $k_{\eta 3}$, *and* $k_{\eta 5}$ *are selected within certain ranges, thereby inducing singular phenomena. Inspired by the work proposed in [185], the singularity issues can be avoided by introducing the following piecewise continuous function.*

$$f(|x|^\alpha) = \begin{cases} |x|^\alpha & if\ |x| > \varepsilon \\ \sin(\frac{\pi}{2\varepsilon})|x|^{|\alpha|} & else, \end{cases} \tag{7.68}$$

where $\varepsilon > 0$ *is a small positive scalar.*

7.4.2 Rotational subsystem stability

Note that controlling the rotational subsystems does not require any information from other quadrotors. Namely, the rotational subsystem control is a single-agent control problem. Therefore, we omit the subscript '*i*' in the following controller and observer design process for clarity and convenience.

We define define z_Θ as the estimation of $\dot{e}_\Theta$, $\hat{\Delta}_\Theta$ as the estimation of Δ_Θ, and $\hat{\sigma}_\Theta$ as the estimation of σ_Θ. Correspondingly, $\tilde{z}_\Theta = \dot{e}_\Theta - z_\Theta$, $\tilde{\Delta}_\Theta = \Delta_\Theta - \hat{\Delta}_\Theta$, and $\tilde{\sigma}_\Theta = \sigma_\Theta = \hat{\sigma}_\Theta$ are the estimation errors.

Similarly, using error dynamics (7.16), the PdT observer can be designed as

$$\begin{aligned}
\dot{z}_\Theta =& \kappa_{\Theta 0}\Big(2a_{\Theta 1}\beta_{\Theta 1}\lfloor\tilde{z}_\Theta\rceil^{2-\frac{1}{\kappa_{\Theta 1}}} + a_{\Theta 2}\beta_{\Theta 2}\lfloor\tilde{z}_\Theta\rceil^{2\alpha_\Theta-\frac{1}{\kappa_{\Theta 1}}} + a_{\Theta 3}\beta_{\Theta 3}\lfloor\tilde{z}_\Theta\rceil^{4-2\alpha_\Theta-\frac{1}{\kappa_{\Theta 1}}}\Big)^{\kappa_{\Theta 1}} \\
&+ \hat{\sigma}_\Theta \operatorname{sgn}(\tilde{z}_\Theta) + A_\Theta + B_\Theta\tau, \\
\dot{\hat{\Delta}}_\Theta =& \kappa_{\Theta 0}\Big(2a_{\Theta 1}\beta_{\Theta 1}\lfloor\tilde{z}_\Theta\rceil^{2-\frac{1}{\kappa_{\Theta 1}}} + a_{\Theta 2}\beta_{\Theta 2}\lfloor\tilde{z}_\Theta\rceil^{2\alpha_\Theta-\frac{1}{\kappa_{\Theta 1}}} + a_{\Theta 3}\beta_{\Theta 3}\lfloor\tilde{z}_\Theta\rceil^{4-2\alpha_\Theta-\frac{1}{\kappa_{\Theta 1}}}\Big)^{\kappa_{\Theta 1}} + \hat{\sigma}_\Theta, \\
\dot{\hat{\sigma}}_\Theta =& -\kappa_{\Theta 0}\Big(2b_{\Theta 1}\beta_{\Theta 1}\hat{\sigma}_\Theta^{2-\frac{1}{\kappa_{\Theta 1}}} + b_{\Theta 2}\beta_{\Theta 2}\hat{\sigma}_\Theta^{2-\frac{1}{\kappa_{\Theta 1}}} + b_{\Theta 3}\beta_{\Theta 3}\hat{\sigma}_\Theta^{4-2\alpha_\Theta-\frac{1}{\kappa_{\Theta 1}}}\Big)^{\kappa_{\Theta 1}} + \tilde{z}_\Theta \operatorname{sgn}(\tilde{z}_\Theta),
\end{aligned} \tag{7.69}$$

where positive parameters $a_{\Theta 1} = \frac{1}{2}$, $b_{\Theta 1} = 1$, $a_{\Theta 2} = 2^{-\alpha_\Theta}$, $b_{\Theta 2} = \frac{1-\alpha_\Theta}{2^{\alpha_\Theta - 1}}$, $a_{\Theta 3} = 2^{\alpha_\Theta - 2}6^{1-\alpha_\Theta}$, $b_{\Theta 3} = \frac{4-\alpha_\Theta}{3-\alpha_\Theta}2^{\alpha_\Theta-2}6^{1-\alpha_\Theta}$, $\kappa_{\Theta 1} = 1$, $0 < \alpha_\Theta < 1$, $\beta_{\Theta 2}, \beta_{\Theta 3} > 0$, $\beta_{\Theta 1}^2 \geq \beta_{\Theta 2}\beta_{\Theta 3}$, and

$$\kappa_{\Theta 0} = \frac{\Gamma\left(\frac{1-\alpha_\Theta\kappa_{\Theta 1}}{2-2\alpha_\Theta}\right)\Gamma\left(\frac{2\kappa_{\Theta 1}-\alpha_\Theta\kappa_{\Theta 1}-1}{2-2\alpha_\Theta}\right)}{T_{p0}\beta_{\Theta 2}^{\kappa_{\Theta 1}}(2-2\alpha_\Theta)\Gamma(\kappa_{\Theta 1})}\left(\frac{\beta_{\Theta 2}}{\beta_{\Theta 3}}\right)^{\frac{1-\alpha_\Theta\kappa_{\Theta 1}}{2-2\alpha_\Theta}}.$$

Then, the following Theorem can be put forth to guarantee the PdT stability of the rotational subsystem observer.

Theorem 7.4 *For error dynamics (7.16), we design an observer (7.69). The estimation error of* Δ_Θ *converges to a neighborhood of the origin* Ω_Θ *in a PdT* $T_{\Theta 1} = \frac{T_{\Theta 0}}{\sqrt{(1-\delta_\Theta)}}$ *with* $\delta_\Theta \in (0,1)$ *being a positive scalar.*

Proof 7.8 *The proof process is very similar to that of Theorem 7.2 and omitted here.*

Then, by following a similar derivation process, a PdT sliding mode surface is defined as

$$\begin{aligned} s_\Theta =& \dot{e}_\Theta + \mathscr{M}_\Theta \\ \mathscr{M}_\Theta =& \kappa_{\Theta 3}\Big(\gamma_{s1}\lfloor e_\Theta \rceil^{2-\frac{1}{\kappa_{\Theta 3}}} + \frac{\gamma_{s2}}{2}\lfloor e_\Theta \rceil^{2\alpha_{s\Theta}-\frac{1}{\kappa_{\Theta 3}}} \\ &+ \frac{\gamma_{s3}}{2}\lfloor e_\Theta \rceil^{4-2\alpha_{s\Theta}-\frac{1}{\kappa_{\Theta 3}}}\Big)^{\kappa_{\Theta 3}}. \end{aligned} \tag{7.70}$$

where $\gamma_{s2}, \gamma_{s3} > 0$, $\gamma_{s1} \geq \sqrt{\gamma_{s2}\gamma_{s3}}$, $0 < \alpha_{s\Theta} < 1$, $\frac{1}{2-\alpha_{s\Theta}} < \kappa_{\Theta 3} < \frac{1}{\alpha_{s\Theta}}$, and

$$\kappa_{\Theta 2} = \frac{\Gamma\left(\frac{1-\alpha_{s\Theta}\kappa_{\Theta 3}}{2-2\alpha_{s\Theta}}\right)\Gamma\left(\frac{2\kappa_{\Theta 3}-\alpha_{s\Theta}\kappa_{\Theta 3}-1}{2-2\alpha_{s\Theta}}\right)}{T_{\Theta 1}\gamma_{s2}^{\kappa_{\Theta 3}}(2-2\alpha_{s\Theta})\Gamma(\kappa_{\Theta 3})}\left(\frac{\gamma_{s2}}{\gamma_{s3}}\right)^{\frac{1-\alpha_{s\Theta}\kappa_{\Theta 3}}{2-2\alpha_{s\Theta}}}$$

. We have

$$\begin{aligned} \dot{\mathscr{M}}_\Theta =& \kappa_{\Theta 2}\kappa_{\Theta 1}\Big(\gamma_{s1}\lfloor e_\Theta \rceil^{2-\frac{1}{\kappa_{\Theta 3}}} + \frac{\gamma_{s2}}{2}\lfloor e_\Theta \rceil^{2\alpha_{s\Theta}-\frac{1}{\kappa_{\Theta 3}}} + \frac{\gamma_{s3}}{2}\lfloor e_\Theta \rceil^{4-2\alpha_{s\Theta}-\frac{1}{\kappa_{\Theta 3}}}\Big)^{\kappa_{\Theta 3}-1} \\ &\times\Big[\frac{\gamma_{s1}(2\kappa_{\Theta 3}-1)}{\kappa_{\Theta 3}}\lfloor e_\Theta \rceil^{\frac{\kappa_{\Theta 3}-1}{\kappa_{\Theta 3}}}\dot{e}_\Theta + \frac{\gamma_{s2}(2\alpha_{s\Theta}\kappa_{\Theta 3}-1)}{2\kappa_{\Theta 3}}\lfloor e_\Theta \rceil^{\frac{(2\alpha_{s\Theta}-1)\kappa_{\Theta 3}-1}{\kappa_{\Theta 3}}}\dot{e}_\Theta \\ &+ \frac{\gamma_{s3}(4-2\alpha_{s\Theta})\kappa_{\Theta 3}-1}{2\kappa_{\Theta 3}}\lfloor e_\Theta \rceil^{\frac{(3-2\alpha_{s\Theta})\kappa_{\Theta 3}-1}{\kappa_{\Theta 3}}}\dot{e}_\Theta\Big], \end{aligned} \tag{7.71}$$

Assuming no external disturbances are acting on the rotational loop. An equivalent control law can be designed as

$$\tau_{eq} = -B_\Theta^{-1}(A_\Theta + \dot{\mathscr{M}}_\Theta). \tag{7.72}$$

Furthermore, a switching control law is required to maintain s_Θ on the sliding mode surface when disturbances Δ_Θ exist, which can be given by

$$\begin{aligned} \tau_{sw} =& -B_\Theta^{-1}\Big[\hat{\Delta}_\Theta + k_\Theta \operatorname{sgn} s_\Theta + \kappa_{\Theta 4}\Big(\gamma_{u1}\lfloor s_\Theta \rceil^{2-\frac{1}{\kappa_{\Theta 5}}} \\ &+ \frac{\gamma_{u2}}{2}\lfloor s_\Theta \rceil^{2\alpha_{u\Theta}-\frac{1}{\kappa_{\Theta 5}}} + \frac{\gamma_{u3}}{2}\lfloor s_\Theta \rceil^{4-2\alpha_{u\Theta}-\frac{1}{\kappa_{\Theta 5}}}\Big)^{\kappa_{\Theta 5}}\Big], \end{aligned} \tag{7.73}$$

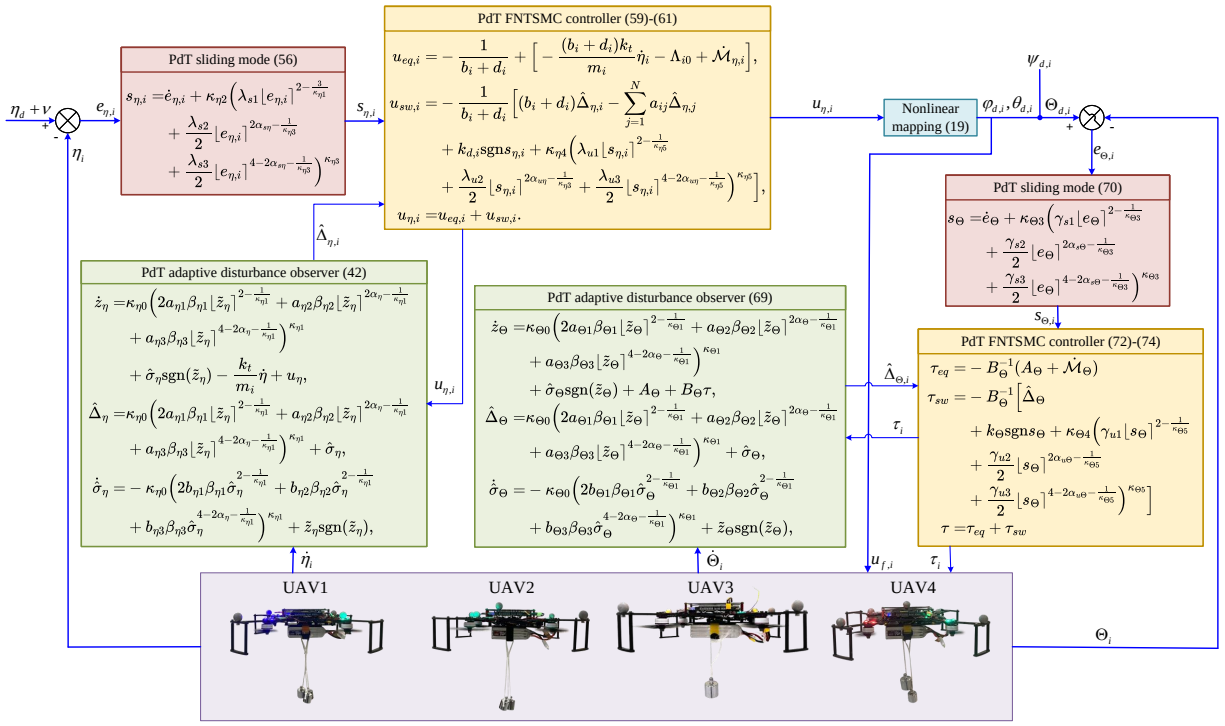


Figure 7.2: Diagram of the entire control framework.

where $\gamma_{u2}, \gamma_{u3} > 0$, $\gamma_{u1} \geq \sqrt{\gamma_{u2}\gamma_{u3}}$, $0 < \alpha_{u\Theta} < 1$, $\frac{1}{2-\alpha_{u\Theta}} < \kappa_{\Theta 5} < \frac{1}{\alpha_{u\Theta}}$,

$$\kappa_{\Theta 4} = \frac{\Gamma\left(\frac{1-\alpha_{u\Theta}\kappa_{\Theta 5}}{2-2\alpha_{u\Theta}}\right)\Gamma\left(\frac{2\kappa_{\Theta 5}-\alpha_{u\Theta}\kappa_{\Theta 5}-1}{2-2\alpha_{u\Theta}}\right)}{T_{\Theta 2}\gamma_{u2}^{\kappa_{\Theta 5}}(2-2\alpha_{u\Theta})\Gamma(\kappa_{\Theta 5})}\left(\frac{\gamma_{u2}}{\gamma_{u3}}\right)^{\frac{1-\alpha_{u\Theta}\kappa_{\Theta 3}}{2-2\alpha_{u\Theta}}}$$

, and $k_\Theta \geq N_\Theta$ is a positive scalar with $N_\Theta = \max\{||x||_2 : x \in \Omega_\Theta\}$.

Then, the complete control law can be designed as

$$\tau = \tau_{eq} + \tau_{sw}. \tag{7.74}$$

Thereafter, a theorem can be illustrated to guarantee the predefined time stability of the system.

Theorem 7.5 *For error dynamics (7.16) disturbed by Δ_Θ, we design an observer as (7.69) and a controller as 7.74. Then, the error e_Θ converges to the origin in a PdT, and the settling time can be bounded as $T_\Theta = T_{\Theta 1} + T_{\Theta 2} + T_{\Theta 3}$.*

Proof 7.9 *The proof process is similar to that of Theorem 7.3. Therefore, we omit it here.*

Remark 7.7 *Although the proposed control framework in this study ensures predefined-time stability of the system, parameter tuning remains a critical issue that warrants attention. Taking the translational loop control as an example, in the sliding mode surface (7.56), the parameters α_η, $k_{\eta 3}$, λ_{s1},*

λ_{s2}, λ_{s3}, and the predefined time $T_{\eta 2}$ are responsible for adjusting the weighting of the terms related to $e_{\eta,i}$ in $s_{\eta,i}$. Evidently, a larger $k_{\eta 2}$ indicates that the sliding mode variable places greater emphasis on $e_{\eta,i}$ rather than $\dot{e}_{\eta,i}$. The three terms in (7.56) associated with $e_{\eta,i}$ respectively correspond to the 'V', 'V^{α}', and '$V^{2-\alpha}$' in Theorem 7.1. The summation of these three components can be adjusted by tuning the values of α_{η}, λ_{s1}, λ_{s2}, and λ_{s3}. Furthermore, if the designer considers the three components to be equally important, it is advisable to set $\lambda_{s2} = \lambda_{s3}$ and $\lambda_{s1} = \sqrt{\lambda_{s2}\lambda_{s3}}$. Additionally, The tuning rule for $k_{\eta 3}$ follows the same principle as revealed in Fig. 7.1 and Remark 7.4. The tuning methodology for the parameters in the switching control law (7.60) is fundamentally similar to that of tuning $s_{\eta,i}$. It is important to note that $k_{\eta 4}$ is specifically designed to compensate for the observation errors of the observer. Consequently, $k_{\eta 4}$ should be designed to be as small as possible within an appropriate range to mitigate chattering in the controller.

7.5 Numerical validation

In this section, some comparative simulations are conducted to verify the superiority of the proposed control method. The framework of the entire multi-quadrotor simulation program is illustrated in Fig. 7.2. Specifically, as for the mathematical model described in Eq. (7.14), $m_i = 0.8\text{kg}$, $g = 9.8\text{m/s}^2$, $J_i = \text{diag}([4.212, 4.212, 8.255]) \times 10^{-3}\text{kg}\cdot\text{m}^2$, $k_r = k_t = 10^{-3}$, and the sampling period is $\Delta t = 0.01\text{s}$. The external disturbances are designed as a combination of constant, sinusoidal, ramp, and sinusoidal functions induced by other sinusoidal functions. The parameters of the controllers and translational loop observers are recorded in Table 7.1 and Table 7.2.

Table 7.1: Parameters of the controllers in the translational loop in simulation.

Param.	Value	Param.	Value	Param.	Value	Param.	Value	Param.	Value
λ_{s1}	5	$\alpha_{s\eta}$	$11/13$	λ_{u1}	7	$\alpha_{u\eta}$	$11/13$	k_d	4
λ_{s2}	1	$\kappa_{\eta 3}$	$11/9$	λ_{u2}	1	$T_{\eta 3}$	10		
λ_{s3}	1	$T_{\eta 2}$	5	λ_{u3}	1	$\kappa_{\eta 5}$	$9/11$		

Table 7.2: Parameters of the observers in the translational loop in simulation.

Param.	Value	Param.	Value	Param.	Value	Param.	Value	Param.	Value
$\beta_{\eta 1}$	2	$\beta_{\eta 2}$	1	$\beta_{\eta 3}$	1	α_{η}	0.5	$\kappa_{\eta 1}$	1
$T_{\eta 0}$	1								

7.5.1 Simulation group 1

Four quadrotors are adopted to establish the quadrotor formation. The reference trajectory of the geometric center follows a counterclockwise circular path with a radius of 5m and a period of 10s.

Each quadrotor's offset from the geometric center is a counterclockwise circular path with a radius of 2m and a period of 5s, characterized by distinct initial phase angles. Additionally, the reference trajectory in the z-direction is a sinusoidal signal with an amplitude of 2m and a period of 10s. The geometric center can be mathematically described by

$$\eta_d = \left[5\sin\frac{2\pi t}{5}, 5\cos\frac{2\pi t}{5}, 2\sin\frac{2\pi t}{5}+6\right]^\top \tag{7.75}$$

and the offsets are defined as

$$\begin{aligned} \nu_1 &= \left[2\cos\tfrac{\pi t}{5}, 2\sin\tfrac{\pi t}{5}, 0\right]^\top, & \nu_2 &= \left[-2\sin\tfrac{\pi t}{5}, 2\cos\tfrac{\pi t}{5}, 0\right]^\top, \\ \nu_3 &= \left[-2\cos\tfrac{\pi t}{5}, -2\sin\tfrac{\pi t}{5}, 0\right]^\top, & \nu_4 &= \left[2\sin\tfrac{\pi t}{5}, -2\cos\tfrac{\pi t}{5}, 0\right]^\top. \end{aligned} \tag{7.76}$$

The adjacent matrix $\mathscr{A}$, in-degree matrix $\mathscr{D}$, communication matrix $\mathscr{B}$, and Laplacian matrix $\mathscr{L}$ are respectively defined as $\mathscr{A} = [0,1,1,1;1,0,0,0;1,0,0,0;1,0,0,0]$, $\mathscr{D} = \text{diag}([0,1,1,1])$, $\mathscr{B} = \text{diag}([1,0,0,0])$, and $\mathscr{L} = \mathscr{D} - \mathscr{A}$. Correspondingly, the topological graph is shown in Fig. 7.3.

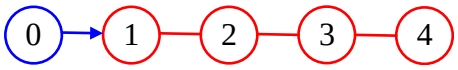


Figure 7.3: Topological graph of simulation group 1.

The simulation results are respectively illustrated in Figs. 7.4- 7.7. Specifically, Fig. 7.4 records the 2-norm of the consensus tracking error, say, $||e_{\eta,i}||$ under different control frameworks. Fig. 7.5 is a 3D demonstration of the quadrotor formation, and Fig. 7.6 records the output of the proposed PdTDOs. Finally, we also give a comparative simulation with different 'predefined convergence times' under our proposed PdT-FNTSMC-PdTDO control framework.

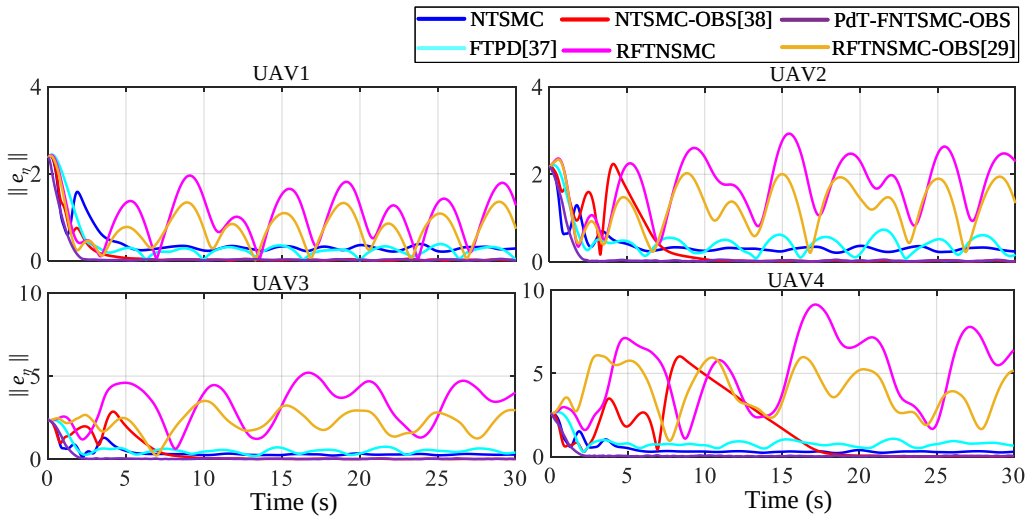


Figure 7.4: 2-norm of the consensus tracking error $||e_{\eta,i}||$ in simulation group 1.

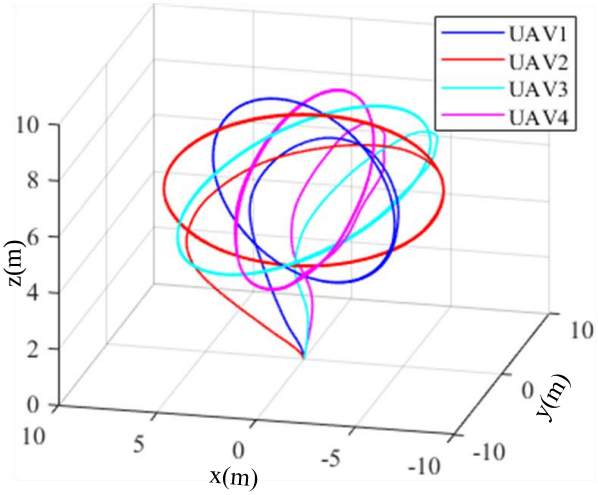


Figure 7.5: A 3D demonstration of the quadrotor group in simulation group 1.

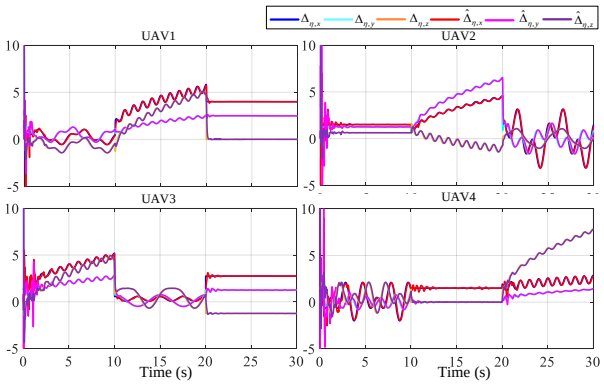


Figure 7.6: Output of the observers in simulation group 1.

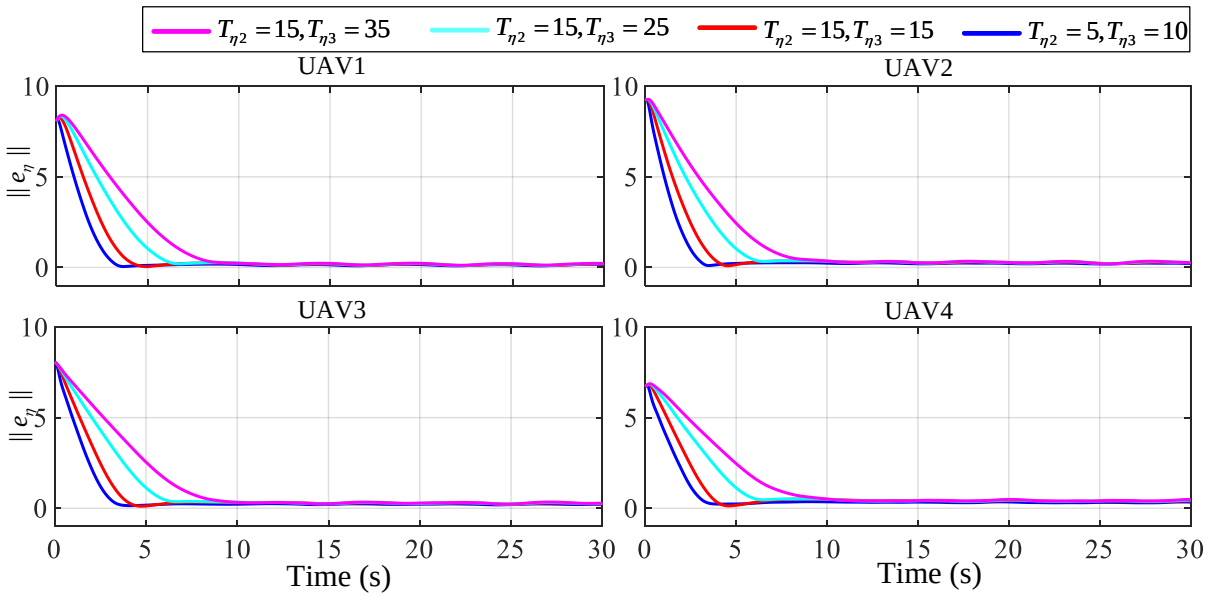


Figure 7.7: Comparative simulations under different 'predefined convergence times' in simulation group 1.

Fig 7.4 demonstrates that the introduction of observers, as shown in [160] and [186] reduces the final tracking error of the system. However, the tracking error remains significantly more significant than the proposed algorithm. Similarly, the fixed-time PD control method employed in [170] achieves system stabilization but still exhibits considerable control errors. Therefore, this set of comparative experiments highlights the superiority of the proposed PDT-FNTSMC-OBS control framework over other methods. Fig. 7.6 presents the output results of the proposed PdT-stable observer. As shown in the figure, the proposed observer can track rapidly varying external disturbances within the predefined time. Fig. 7.7 demonstrates the two norms of the tracking error of the quadrotors under different

'predefined convergence times'. The results shown in Fig. 7.7 are consistent with those discussed earlier in Fig. 7.1. The blue curve exhibits the fastest response due to its shortest predefined time. In contrast, the pink curve has the slowest convergence because its predefined time is the longest.

7.5.2 Simulation group 2

We further test our algorithm with a more complicated topological graph and aggressive reference trajectories. Specifically, $\mathscr{A}$, $\mathscr{D}$, $\mathscr{B}$, and $\mathscr{L}$ are respectively defined as $\mathscr{A} = [0,1,1,0;1,0,0,0;1,0,0,1;0,0,1,0]$, $\mathscr{D} = \text{diag}([0,1,1,1])$, $\mathscr{B} = \text{diag}([1,0,0,0])$, and $\mathscr{L} = \mathscr{D} - \mathscr{A}$. The topological graph is shown in Fig. 7.8. The reference trajectories are designed as counterclockwise '∞-shape'

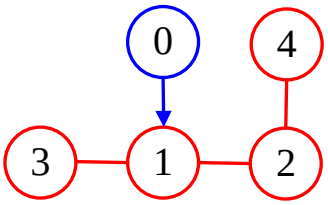


Figure 7.8: Topological graph of simulation group 2.

curves, which are

$$\eta_d = \left[5\cos\frac{\pi t}{5} + 2, 5\sin\frac{2\pi t}{5} + 3, \sin\frac{2\pi t}{5} + 6\right]^\top \tag{7.77}$$

and the offsets are selected to be

$$\begin{aligned} \nu_1 &= [2,0,0]^\top, \quad & \nu_2 &= [0,2,0]^\top, \\ \nu_3 &= [-2,0,0]^\top, \quad & \nu_4 &= [0,-2,0]^\top. \end{aligned} \tag{7.78}$$

Similarly, the 2-norm of the consensus tracking error, the 3D demonstration, the output of the proposed observer, and the comparative simulation under different 'predefined convergence times' are respectively recorded in Figs. 7.9, 7.10, 7.11, and 7.12.

The patterns and characteristics presented in the second set of simulations are similar to those of the first set. Firstly, Fig. 7.9 demonstrates the superiority of the proposed method compared to other approaches. Similarly, as shown in Fig. 7.11, the proposed observer can rapidly track external disturbances. However, the phenomenon revealed in Fig. 7.12 slightly differs from that observed in Fig.7.7. This is because the reference trajectory for the quadrotor formation in simulations group 2 is an ∞-shaped curve with sharper ends compared to a circular trajectory. The tracking error at the sharp ends of the reference trajectory cannot be eliminated due to the inherent characteristics of the quadrotors. Still, the superiority of the proposed PdT-FNTSMC-OBS method is evident from the comparison curves illustrated in Fig. 7.12.

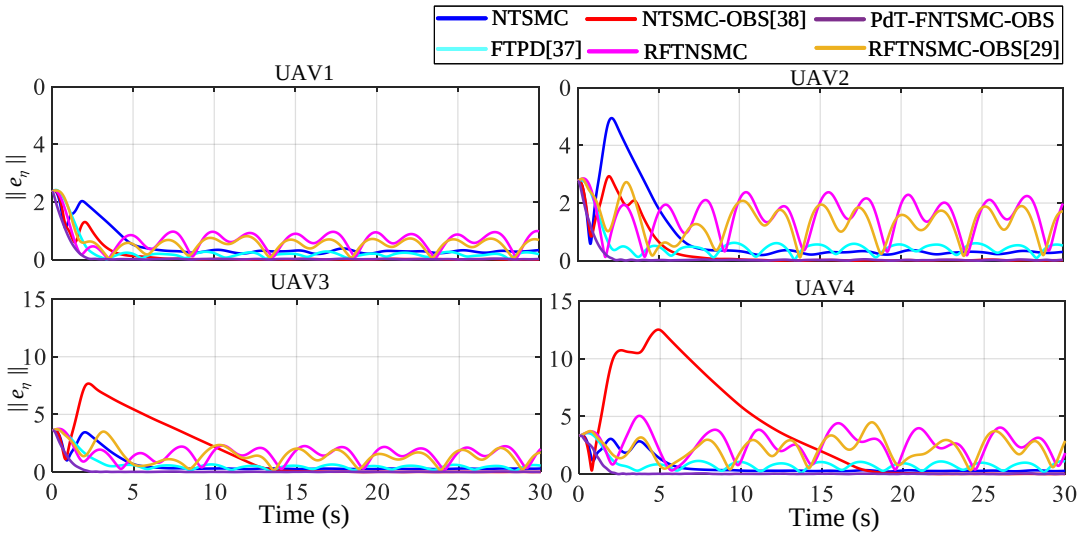


Figure 7.9: 2-norm of the consensus tracking error $||e_{\eta,i}||$ in simulation group 2.

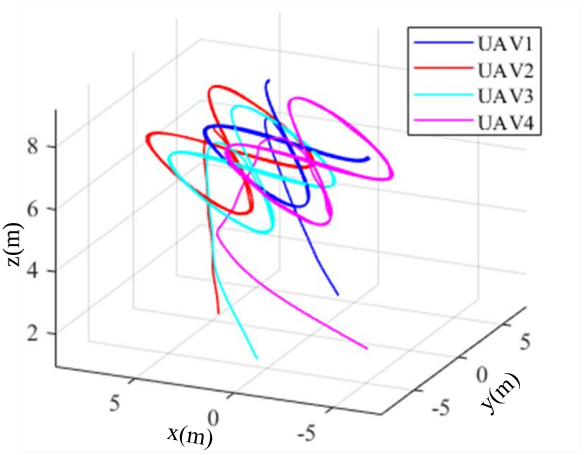


Figure 7.10: A 3D demonstration of the quadrotor group in simulation group 2.

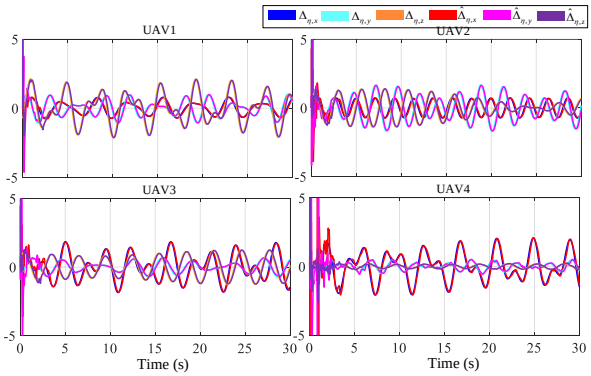


Figure 7.11: Output of the observers in simulation group 2.

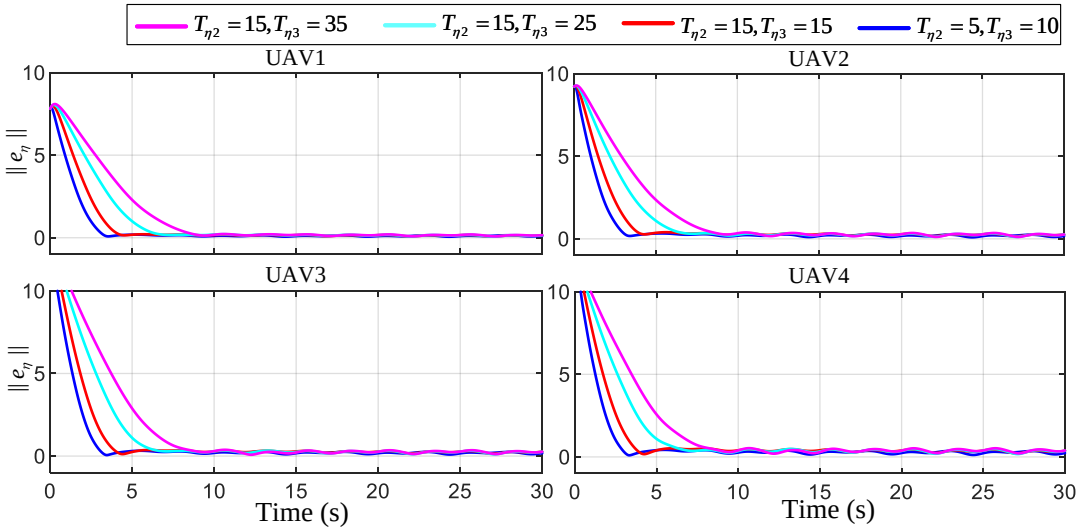


Figure 7.12: Comparative simulations under different 'predefined convergence times' in simulation group 2.

7.6 Real-world experiment

This section presents comparative real-world experiments to highlight the practicality of the proposed control framework. The mass of the quadrotors is $m = 0.85$kg, and the wheel-based is 280mm. Two comparative experiments with four quadrotors with topological graphs identical to those in simulations are conducted. Two high-speed rotating fans and weights suspended by elastic ropes are put into the environment as external disturbances. The complete hardware configuration of the quadrotor formation is shown in Fig. 7.13.

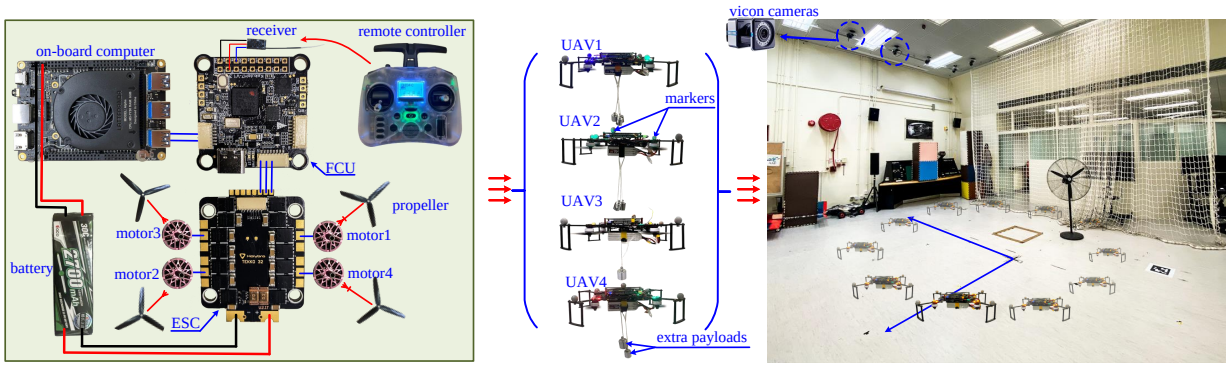


Figure 7.13: Hardware configuration of the experiments.

In Fig. 7.13, Holybro Kakute H7 V1.3 is utilized to be the quadrotors' flight control unit (FCU) that inherits the open-source PX4-Autopilot firmware. The onboard computer, connected to the FCU via a USB to TTL module, is the LattePanda Alpha 864s with Ubuntu 20.04 and ROS Noetic operating system. The data transformation between the FCU and the onboard computer follows the MAVLink communication protocol. The quadrotors are all localized by a VICON indoor positioning system equipped with 14 high-resolution optical cameras, which provide sub-millimeter accuracy with a frequency exceeding 200Hz. The velocity feedback of the quadrotors is obtained by fusing data from the motion capture system and the IMU in the FCU using the Kalman filter. In addition, the ground station computer is solely responsible for monitoring the flight status of the quadrotors. It does not send any control commands, as the proposed control framework is fully decentralized and distributed. This means all control commands are autonomously computed and generated by the onboard computers of the quadrotors.

The mass of each quadrotor is 0.85kg; the translational drag coefficient is $k_t = 0.001$. The hyper-parameters of the controllers and observers of the quadrotors in the physical experiments are illustrated in Tables 7.3 and 7.4. For convenience and ease of comparison, the topological graph defined in Chapters 7.6.1 and 7.6.2 are respectively designed to be identical to the ones used in Chapters 7.5.1 and 7.5.2.

Table 7.3: Parameters of the controllers in the translational loop in physical experiments.

Param.	Value	Param.	Value	Param.	Value	Param.	Value	Param.	Value
λ_{s1}	5	$\alpha_{s\eta}$	$3/5$	λ_{u1}	3	$\alpha_{u\eta}$	$3/5$	k_d	1.5
λ_{s2}	3	$\kappa_{\eta 3}$	$9/11$	λ_{u2}	3	$T_{\eta 3}$	10		
λ_{s3}	3	$T_{\eta 2}$	10	λ_{u3}	3	$\kappa_{\eta 5}$	$9/11$		

Table 7.4: Parameters of the observers in the translational loop in physical experiments.

Param.	Value	Param.	Value	Param.	Value	Param.	Value	Param.	Value
$\beta_{\eta 1}$	$2\sqrt{2}$	$\beta_{\eta 2}$	2	$\beta_{\eta 3}$	2	α_{η}	$3/5$	$\kappa_{\eta 1}$	1
$T_{\eta 0}$	1								

7.6.1 Experiment group 1

Limited by the available space we can use to validate the physical experiments. The geometric center of the quadrotor formation remains unchanged. The trajectories of the offsets of the four quadrotors to the geometric center are circles with equal radii but different phases at different altitudes. Specifically, the location of the geometric center is defined as

$$\eta_d = [0, 0.2, 1.5]^\top. \tag{7.79}$$

The offsets of the four quadrotors to η_d is defined as

$$\begin{aligned}
\nu_1 &= \left[1.2\cos\frac{\pi t}{5}, 1.2\sin\frac{\pi t}{5}, 0.2\sin\frac{\pi t}{5} + 0.5\right]^\top, \\
\nu_2 &= \left[-1.2\sin\frac{\pi t}{5}, 1.2\cos\frac{\pi t}{5}, 0.2\sin\frac{\pi t}{5} - 0.5\right]^\top, \\
\nu_3 &= \left[-1.2\cos\frac{\pi t}{5}, -1.2\sin\frac{\pi t}{5}, 0.2\sin\frac{\pi t}{5} + 0.5\right]^\top, \\
\nu_4 &= \left[1.2\sin\frac{\pi t}{5}, -1.2\cos\frac{\pi t}{5}, 0.2\sin\frac{\pi t}{5} - 0.5\right]^\top.
\end{aligned} \tag{7.80}$$

Similarly, the 2-norm of the consensus tracking errors, 3D demonstration of the quadrotor formation, the output of the observers, and the comparative performance under different 'predefined convergence times' are respectively illustrated in Figs. 7.14- 7.17.

Fig. 7.14 clearly illustrates the differences in formation control performance among various quadrotor control methods. Undoubtedly, the proposed PdT-FNTSMC-OBS control framework demonstrates the best robustness under strong disturbances during quadrotor formation control, and the PID controller exhibits the poorest dynamic tracking performance. Fig. 7.15 records the response curves of four quadrotors, where it can be observed that the quadrotors successfully track the predefined reference trajectories. Fig. 7.16 shows the output of the observer, indicating that the proposed

observer can rapidly and accurately estimate external disturbances. However, it should be noted that slight oscillations are present in the z direction. This is attributed to using an elastic cord to suspend weights in experiment group 1. Finally, Fig. 7.17 compares the formation responses under different PdTs and confirms the conclusions drawn from the theoretical analysis: longer preset times result in slower responses. In extreme cases, when the PdTs are excessively long, the 30-second experimental duration is insufficient for the tracking error to converge to zero.

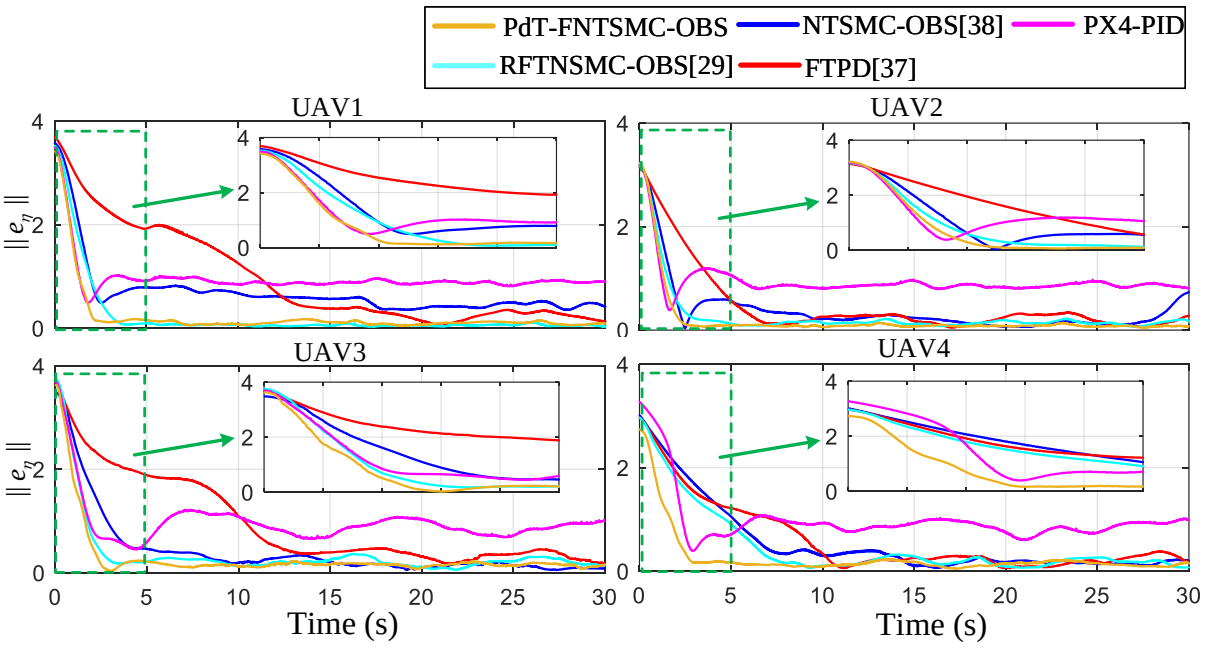


Figure 7.14: 2-norm of the consensus tracking error $||e_{\eta,i}||$ in experiment group 1.

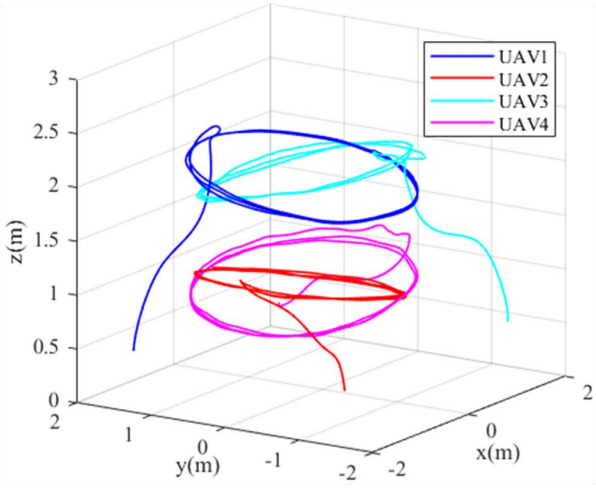


Figure 7.15: A 3D demonstration of the quadrotor group in experiment group 1.

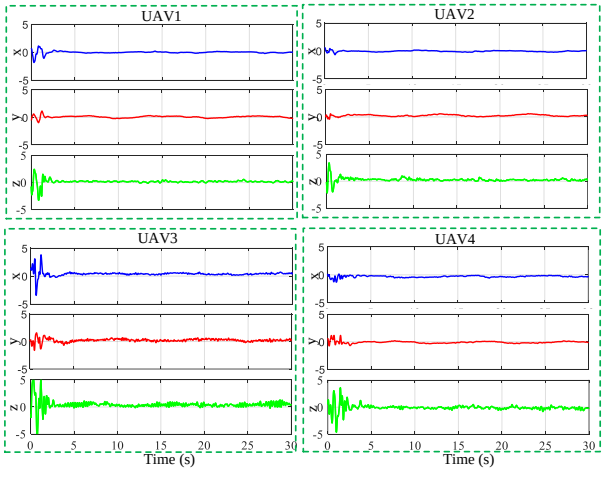


Figure 7.16: Output of the observers in experiment group 1.

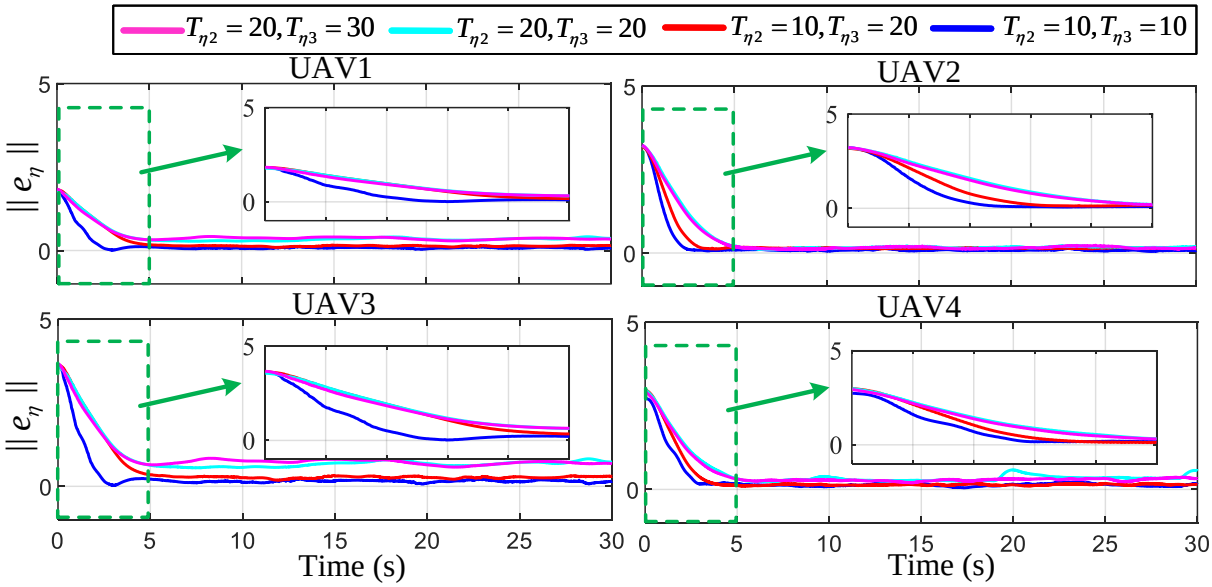


Figure 7.17: Comparative experiment under different 'predefined convergence times' in experiment group 1.

7.6.2 Experiment group 2

Thereafter, we further test the performance of our proposed control framework with a more complicated reference trajectory. The geometric center of the quadrotor formation is designed to be an ∞-shaped trajectory, and the offsets of the quadrotors to the geometric center are defined to be different constants. Specifically.

$$\eta_d = [\cos(0.2\pi t), 0.5\sin(0.4\pi t) + 0.2, 1.5]^\top \tag{7.81}$$

and

$$\begin{aligned} \nu_1 &= [0.5, 0, 0.5]^\top, \quad & \nu_2 &= [0, 0.8, -0.5]^\top, \\ \nu_3 &= [-0.5, 0., 0.5]^\top, \quad & \nu_4 &= [0., -0.8, -0.5]^\top. \end{aligned} \tag{7.82}$$

Correspondingly, Fig. 7.18 records the 2-norm of the consensus tracking error, Fig. 7.19 illustrates a 3D demonstration, Fig. 7.20 shows the output of the observer, and Fig. 7.21 presents the comparative performance among different 'predefined convergence times'.

The patterns revealed in the experiment group 2 are similar to those in the 1. However, it is observed that the tracking errors in the experiment group 2 are slightly larger than in the first. This is because the reference trajectories in the second set impose higher demands on the dynamic characteristics of the quadrotors. For practical applications, this requires a more extended adjustment period. The 3D visualization in Fig. 7.19 illustrates four quadrotors drawing identical ∞-shapes at different locations,

demonstrating that the proposed controller can effectively track highly dynamic reference trajectories even under strong disturbances.

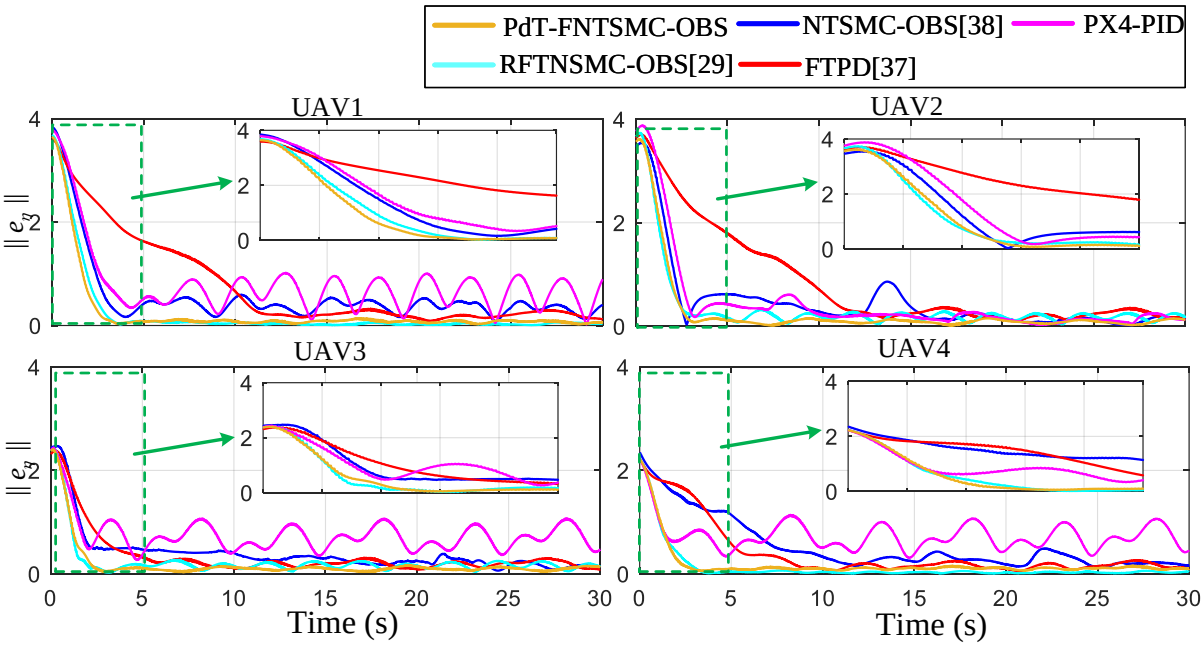


Figure 7.18: 2-norm of the consensus tracking error $||e_{\eta,i}||$ in experiment group 2.

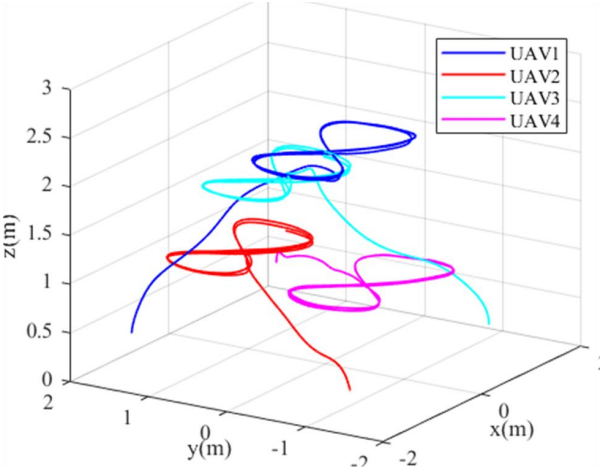


Figure 7.19: A 3D demonstration of the quadrotor group in experiment group 2.

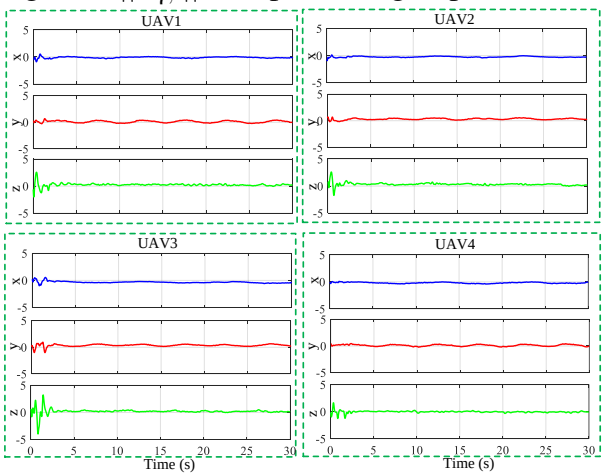


Figure 7.20: Output of the observers in experiment group 2.

Further, to demonstrate the effectiveness and superiority of our proposed control framework, we conduct numerical analyses of the L_1- and L_2-norm of the consensus tracking errors for the two groups' experiments, and the results are recorded in Fig. 7.22, in where we can clearer find out that the proposed PdT-FNTSMC-OBS control framework (the orange bars in Fig. 7.22) outperforms the other four comparative control methods.

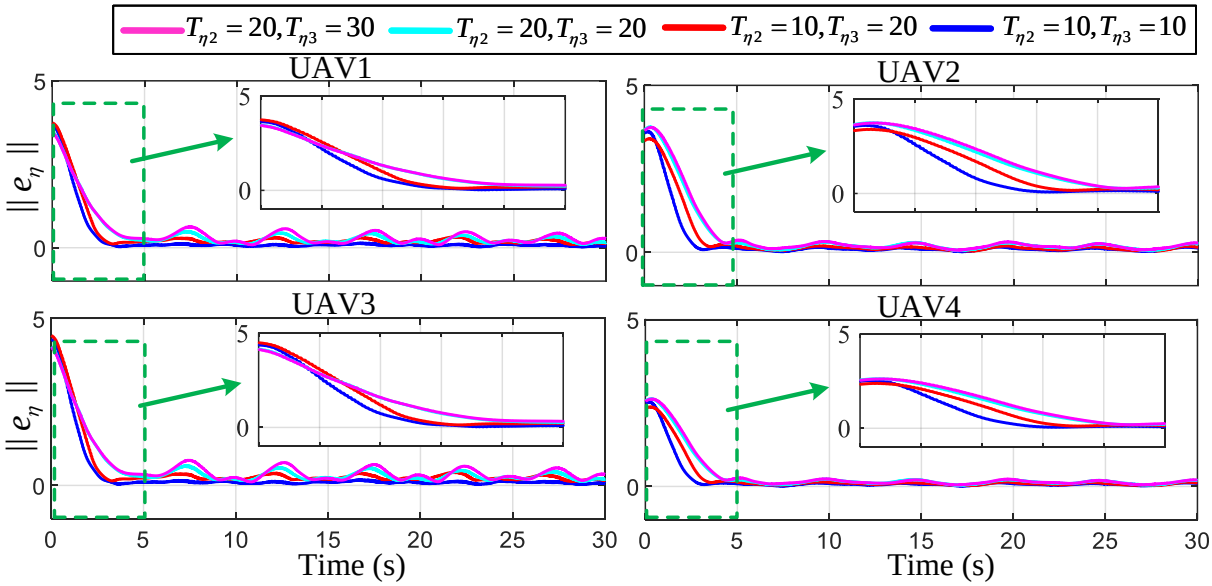


Figure 7.21: Comparative experiment under different 'predefined convergence times' in experiment group 2.

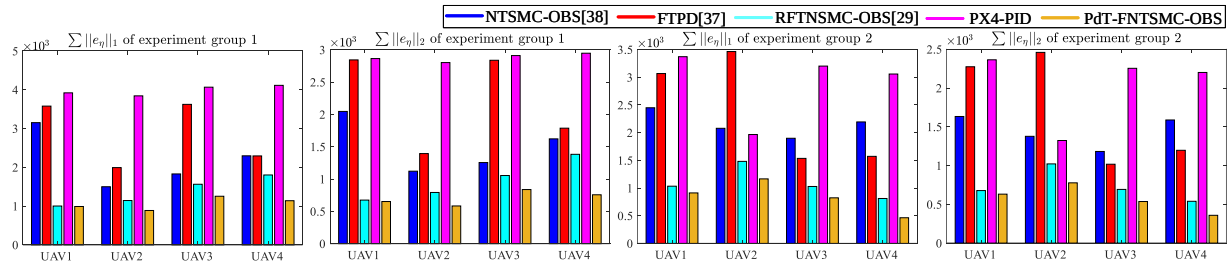


Figure 7.22: L_1- and L_2-norms of e_η of the two groups' experiments under different control frameworks.

7.7 Conclusions

This chapter proposes a modified PdT convergence criterion for multi-quadrotor consensus tracking control problems. Initially, the modified PdT converging law is proposed to simultaneously guarantee the predefined-time convergence of the system and accelerate the convergence process. PdT observers and distributed PdT consensus tracking protocol are presented to control the quadrotor formation system. The proposed disturbance observers can accurately estimate the external disturbance in a PdT, and the entire system is proved to be PdT stable. Finally, extensive simulations and physical experiments are conducted to validate the effectiveness and robustness of the proposed control framework. Our future work will focus on using deep reinforcement learning to adaptively tune the hyperparameters in the proposed control method, aiming to reduce the tuning time and achieve better performance.

Chapter 8

Conclusions and Future Work

In this thesis, we introduced several novel algorithms for robotics path planning and control to address existing challenges in autonomous robotic control. The proposed methods have been thoroughly verified to outperform several state-of-the-art techniques through extensive simulations and real-world experiments.

This research begins with the development of two self-established simulation platforms. The first is a PPSP, which offers more advanced features than many existing platforms. Specifically, we incorporated multi-edge polygons and ellipses into the platform and developed a comprehensive geometric operation function library, whereas many traditional platforms only include circular and rectangular obstacles. For the DRLSP, we integrated over ten state-of-the-art DRL algorithms, such as TD3, PPO, SAC, among others, as well as more than ten control plants, including UAVs, UGVs, CartPole, and second-order integration systems. Additionally, the RL algorithm and environment components are fully decoupled, significantly simplifying the algorithm design process. The simulations presented from Chapter 4 to Chapter 6 were all conducted using these two simulation platforms.

Chapter 4 introduces a comprehensive robotics trajectory planning framework, including a global path planner and a local obstacle avoidance motion planner. The global planner addresses the inefficiency of the traditional RRT algorithm, particularly when navigating complex maps. To enhance its performance, we utilize the geometric properties of the obstacles. The obstacles are grouped into convex hulls, and the RRT sampling region is confined to a series of convex rings and the connectors between them. For local planning, we employ the network decoupling technique in conjunction with the traditional TD3 algorithm, which improves both the training process and the success rate of obstacle avoidance. The proposed framework is validated through extensive simulations and physical experiments using a two-wheel differential ground vehicle, demonstrating its effectiveness in real-world applications.

In Chapter 5, we focus on controlling a QUAV, addressing the critical trajectory tracking problem essential in robotics control and autonomous systems. This chapter leverages the strengths of

traditional control methods and AI-based approaches. While conventional control theory provides rigorous mathematical guarantees for system stability, learning-based methods have the potential to enhance overall performance. Thus, combining these two approaches is an optimal strategy. Specifically, we employ RFNTSMC as the core controller to stabilize the closed-loop system. A FTDO is then introduced to estimate the equivalent disturbances acting on the drone. Additionally, DRL is utilized to dynamically tune the hyper-parameters of the FNTSMC in real-time. The stability of the closed-loop system is further analyzed using the ADP framework. This chapter concludes with extensive demonstrations of numerical simulations and physical experiments, showcasing the effectiveness of the proposed control framework.

In Chapter 6, we build upon the results from Chapter 5 by extending the control framework to a multi-drone system. Specifically, we design a fully distributed control protocol to stabilize a group of drones. Distributed FNTSMCs and FTDOs are employed to achieve this stabilization, with the system's stability rigorously guaranteed in a Lyapunov sense using fundamental graph theory principles. Additionally, the PPO algorithm is implemented to optimize the gains in the FNTSMCs, further enhancing system performance. To validate the effectiveness of the proposed control framework, we conduct simulations with a six-drone QUAV group and perform physical experiments with a four-drone QUAV group. These results demonstrate the robustness and adaptability of the control methodology.

Furthermore, in Chapter 7, we propose an enhanced predefined-time stable control framework for multi-quadrotor formations. This improved predefined-time control framework offers a more generalized form than existing predefined-time or fixed-time convergence criteria, providing designers greater flexibility in tuning controller performance. First, we design a predefined-time stable observer based on the improved predefined-time convergence criterion. This observer can simultaneously estimate disturbances and their upper bounds while ensuring that the estimation errors converge to a neighborhood within the predefined time. Subsequently, we develop a distributed predefined-time control framework, which guarantees that the consensus tracking errors of the entire quadrotor formation converge to zero within the predefined time.

Based on the research finished in this thesis, we further summarize some future work and problems that need to be addressed.

1) DRL is a powerful tool for parameter optimization, but the training process can be time-consuming. In our study, obtaining a satisfactory parameter optimizer took over fifteen minutes. Therefore, future research will focus on improving training efficiency and exploring the possibility of using a real-time reinforcement learning optimizer.
2) The topological graph used in the multi-agent control is time-invariant. However, in real-world applications, switching topologies are more common. Developing distributed control protocols that can handle switching graphs and communication delays is crucial for practical implemen-

tations.

3) To simplify the training process, we trained the DRL-based parameter optimizer in a single-agent environment since all drones were homogeneous. However, this approach overlooks the influence of the topological graph itself. In future studies, we plan to use multi-agent DRL methods, such as Multi-agent DDPG (MA-DDPG) or Multi-agent PPO (MA-PPO), to distribute and separately train different parameter optimizers for each drone in the group.
4) In the predefined-observer presented in Chapter 7, to satisfy certain inequalities arising from the theoretical derivation, we conclude that the 'κ_1' in Theorem 7.1 must be set to 1. However, this condition is not necessary and sufficient. Therefore, in future work, we aim to explore the possibility of relaxing this constraint while still achieving the same theoretical conclusions.

Bibliography

[1] M. M. ElFaham, A. M. Mostafa, and G. Nasr, "Unmanned aerial vehicle (uav) manufacturing materials: Synthesis, spectroscopic characterization and dynamic mechanical analysis (dma)," *Journal of Molecular Structure*, vol. 1201, p. 127211, 2020.

[2] P. K. Singh and A. Sharma, "An intelligent wsn-uav-based iot framework for precision agriculture application," *Computers and Electrical Engineering*, vol. 100, p. 107912, 2022.

[3] M. S. Alam and J. Oluoch, "A survey of safe landing zone detection techniques for autonomous unmanned aerial vehicles (uavs)," *Expert Systems with Applications*, vol. 179, p. 115091, 2021.

[4] B. Fan, Y. Li, R. Zhang, and Q. Fu, "Review on the technological development and application of uav systems," *Chinese Journal of Electronics*, vol. 29, no. 2, pp. 199–207, 2020.

[5] S. Islam, S. Badsha, I. Khalil, M. Atiquzzaman, and C. Konstantinou, "A triggerless backdoor attack and defense mechanism for intelligent task offloading in multi-uav systems," *IEEE Internet of Things Journal*, vol. 10, no. 7, pp. 5719–5732, 2022.

[6] J. Lindsay, J. Ross, M. L. Seto, E. Gregson, A. Moore, J. Patel, and R. Bauer, "Collaboration of heterogeneous marine robots toward multidomain sensing and situational awareness on partially submerged targets," *IEEE Journal of Oceanic Engineering*, vol. 47, no. 4, pp. 880–894, 2022.

[7] D. Omeiza, H. Webb, M. Jirotka, and L. Kunze, "Explanations in autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 10 142–10 162, 2021.

[8] S. Sonkar, P. Kumar, D. Philip, and A. Ghosh, "Low-cost smart surveillance and reconnaissance using vtol fixed wing uav," in *2020 IEEE Aerospace conference*. IEEE, 2020, pp. 1–7.

[9] C. Yan, C. Wang, X. Xiang, Z. Lan, and Y. Jiang, "Deep reinforcement learning of collision-free flocking policies for multiple fixed-wing uavs using local situation maps," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, pp. 1260–1270, 2021.

[10] W. Xue, J. Qi, G. Shao, Z. Xiao, Y. Zhang, and P. Zhong, "Low-rank approximation and multiple sparse constraint modeling for infrared low-flying fixed-wing uav detection," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 4150–4166, 2021.

[11] H. Kim, K. K. F. So, and J. Wirtz, "Service robots: Applying social exchange theory to better understand human–robot interactions," *Tourism Management*, vol. 92, p. 104537, 2022.

[12] B. Rahmadya, R. Sun, S. Takeda, K. Kagoshima, and M. Umehira, "A framework to determine secure distances for either drones or robots based inventory management systems," *IEEE Access*, vol. 8, pp. 170 153–170 161, 2020.

[13] H. M. Balanji, A. E. Turgut, and L. T. Tunc, "A novel vision-based calibration framework for industrial robotic manipulators," *Robotics and Computer-Integrated Manufacturing*, vol. 73, p. 102248, 2022.

[14] A. Pal, V. Restrepo, D. Goswami, and R. V. Martinez, "Exploiting mechanical instabilities in soft robotics: Control, sensing, and actuation," *Advanced Materials*, vol. 33, no. 19, p. 2006939, 2021.

[15] A. E. Hramov, V. A. Maksimenko, and A. N. Pisarchik, "Physical principles of brain–computer interfaces and their applications for rehabilitation, robotics and control of human brain states," *Physics Reports*, vol. 918, pp. 1–133, 2021.

[16] A. T. Taylor, T. A. Berrueta, and T. D. Murphey, "Active learning in robotics: A review of control principles," *Mechatronics*, vol. 77, p. 102576, 2021.

[17] E. Baccour, A. Erbad, A. Mohamed, M. Hamdi, and M. Guizani, "Multi-agent reinforcement learning for privacy-aware distributed cnn in heterogeneous iot surveillance systems," *Journal of Network and Computer Applications*, vol. 230, p. 103933, 2024.

[18] N. Gómez, N. Peña, S. Rincón, S. Amaya, and J. Calderon, "Leader-follower behavior in multi-agent systems for search and rescue based on pso approach," in *SoutheastCon 2022*. IEEE, 2022, pp. 413–420.

[19] C. Zhai, Z. Wang, and J. Dou, "Multi-agent coverage control for enhanced geohazard monitoring: a brief review," *Control Theory and Technology*, vol. 19, no. 3, pp. 418–420, 2021.

[20] F. Ho, R. Geraldes, A. Gonçalves, B. Rigault, B. Sportich, D. Kubo, M. Cavazza, and H. Prendinger, "Decentralized multi-agent path finding for uav traffic management," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 997–1008, 2020.

[21] A. Amirkhani and A. H. Barshooi, "Consensus in multi-agent systems: a review," *Artificial Intelligence Review*, vol. 55, no. 5, pp. 3897–3935, 2022.

[22] Y. Liu, J. Liu, Z. He, Z. Li, Q. Zhang, and Z. Ding, "A survey of multi-agent systems on distributed formation control," *Unmanned Systems*, vol. 12, no. 05, pp. 913–926, 2024.

[23] G. M. Atınç, D. M. Stipanović, and P. G. Voulgaris, "A swarm-based approach to dynamic coverage control of multi-agent systems," *Automatica*, vol. 112, p. 108637, 2020.

[24] R. N. Haksar, S. Trimpe, and M. Schwager, "Spatial scheduling of informative meetings for multi-agent persistent coverage," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3027–3034, 2020.

[25] H. Li and Q. Wei, "Data-driven optimal output cluster synchronization control of heterogeneous multi-agent systems," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 3, pp. 3910 – 3920, 2023.

[26] M. Hasanzadeh, M. Baradarannia, and F. Hashemzadeh, "Distributed fixed-time rotating encirclement control of linear multi-agent systems with moving targets," *Journal of the Franklin Institute*, p. 106970, 2024.

[27] S. Wang, M. Chen, X. Liu, C. Yin, S. Cui, and H. V. Poor, "A machine learning approach for task and resource allocation in mobile-edge computing-based networks," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1358–1372, 2020.

[28] L. C. Garaffa, M. Basso, A. A. Konzen, and E. P. de Freitas, "Reinforcement learning for mobile robotics exploration: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 8, pp. 3796–3810, 2021.

[29] A. Francis, A. Faust, H.-T. L. Chiang, J. Hsu, J. C. Kew, M. Fiser, and T.-W. E. Lee, "Long-range indoor navigation with prm-rl," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1115–1134, 2020.

[30] P. J. Werbos, "Consistency of hdp applied to a simple reinforcement learning problem," *Neural networks*, vol. 3, no. 2, pp. 179–189, 1990.

[31] E. W. Dijkstra, "A note on two problems in connexion with graphs," in *Edsger Wybe Dijkstra*, 1st ed., K. R. Apt and T. Hoare, Eds. New York, NY, USA: ACM, July 2022, pp. 287–290.

[32] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[33] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," *Ann Arbor*, vol. 1001, no. 48105, pp. 18–80, 2008.

[34] S. Sedighi, D.-V. Nguyen, and K.-D. Kuhnert, "Guided hybrid a-star path planning algorithm for valet parking applications," in *2019 5th International Conference on Control, Automation and Robotics (ICCAR)*. Beijing, China: IEEE, Apr. 2019, pp. 570–575.

[35] A. Le, V. Prabakaran, V. Sivanantham, and R. Mohan, "Modified a-star algorithm for efficient coverage path planning in tetris inspired self-reconfigurable robot with integrated laser sensor," *Sensors*, vol. 18, no. 8, p. 2585, Aug. 2018.

[36] C. Liu, Q. Mao, X. Chu, and S. Xie, "An improved a-star algorithm considering water current, traffic separation and berthing for vessel path planning," *Applied Sciences*, vol. 9, no. 6, p. 1057, Mar. 2019.

[37] L. Kavraki and J.-C. Latombe, "Randomized preprocessing of configuration for fast path planning," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. San Diego, CA, USA: IEEE Computer Society Press, 1994, pp. 2138–2145.

[38] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the rrt*," in *2011 IEEE International Conference on Robotics and Automation*. Shanghai, China: IEEE, May 2011, pp. 1478–1483.

[39] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, Mar. 1986.

[40] J. Kuffner and S. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2. San Francisco, CA, USA: IEEE, 2000, pp. 995–1001.

[41] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, June 2011.

[42] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Chicago, IL, USA: IEEE, Sept. 2014, pp. 2997–3004.

[43] Y. Li, R. Cui, Z. Li, and D. Xu, "Neural network approximation based near-optimal motion planning with kinodynamic constraints using rrt," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 11, pp. 8718–8729, Nov. 2018.

[44] J. Wang, W. Chi, C. Li, C. Wang, and M. Q.-H. Meng, "Neural rrt*: Learning-based optimal path planning," *IEEE Transactions Automatation Scienci and Engineering*, vol. 17, no. 4, pp. 1748–1758, Oct. 2020.

[45] O. Matei and P. Pop, "An efficient genetic algorithm for solving the generalized traveling salesman problem," in *Proceedings of the 2010 IEEE 6th International Conference on Intelligent Computer Communication and Processing*. Cluj-Napoca, Romania: IEEE, Aug. 2010, p. 5606458.

[46] M. Pablo, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," *Caltech concurrent computation program, C3P Report*, vol. 826, 1989.

[47] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4. Perth, WA, Australia: IEEE, 1995, pp. 1942–1948.

[48] D. Marco, "Ant colony optimization for vehicle routing problem," Ph.D. dissertation, Politecnico di Milano, Milan, Italy, 1992.

[49] Á. Madridano, A. Al-Kaff, D. Martín, and A. De La Escalera, "Trajectory planning for multi-robot systems: Methods and applications," *Expert Systems with Applications*, vol. 173, p. 114660, July 2021.

[50] M. Mohanan and A. Salgoankar, "A survey of robotic motion planning in dynamic environments," *Robotics and Autonomous Systems*, vol. 100, pp. 171–185, Feb. 2018.

[51] Y. Zhao, Z. Zheng, and Y. Liu, "Survey on computational-intelligence-based uav path planning," *Knowledge-Based Systems*, vol. 158, pp. 54–64, Oct. 2018.

[52] S. Aggarwal and N. Kumar, "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges," *Computer Communications*, vol. 149, pp. 270–299, Jan. 2020.

[53] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics and Automation Magazine*, vol. 4, no. 1, pp. 23–33, Mar. 1997.

[54] C. Warren, "Global path planning using artificial potential fields," in *Proceedings, 1989 International Conference on Robotics and Automation*. Scottsdale, AZ, USA: IEEE Computer Society Press, 1989, pp. 316–321.

[55] Z. Yongzhe, B. Ma, and C. K. Wai, "A practical study of time-elastic-band planning method for driverless vehicle for auto-parking," in *2018 International Conference on Intelligent Autonomous Systems (ICoIAS)*. Singapore: IEEE, Mar. 2018, pp. 196–200.

[56] B. Li, Q. Li, Y. Zeng, Y. Rong, and R. Zhang, "3d trajectory optimization for energy-efficient uav communication: A control design perspective," *IEEE Transactions on Wireless Communications*, vol. 21, no. 6, pp. 4579–4593, June 2022.

[57] G. J. Ducard and M. Allenspach, "Review of designs and flight control techniques of hybrid and convertible vtol uavs," *Aerospace Science and Technology*, vol. 118, p. 107035, Nov. 2021.

[58] S. Wang, A. Polyakov, and G. Zheng, "Quadrotor stabilization under time and space constraints using implicit pid controller," *Journal of the Franklin Institute*, vol. 359, no. 4, pp. 1505–1530, Mar. 2022.

[59] S. Sun, A. Romero, P. Foehn, E. Kaufmann, and D. Scaramuzza, "A comparative study of nonlinear mpc and differential-flatness-based control for quadrotor agile flight," *IEEE Transactions Robotics*, vol. 38, no. 6, pp. 3357–3373, Dec. 2022.

[60] S. Gros and M. Zanon, "Data-driven economic nmpc using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 636–648, Feb. 2020.

[61] M. Greeff and A. P. Schoellig, "Exploiting differential flatness for robust learning-based tracking control using gaussian processes," *IEEE Control Systems Letters*, vol. 5, no. 4, pp. 1121–1126, Oct. 2021.

[62] X. Zhang, Y. Wang, G. Zhu, X. Chen, Z. Li, C. Wang, and C.-Y. Su, "Compound adaptive fuzzy quantized control for quadrotor and its experimental verification," *IEEE Transactions Cybernetics*, vol. 51, no. 3, pp. 1121–1133, Mar. 2021.

[63] Z. Hou, P. Lu, and Z. Tu, "Nonsingular terminal sliding mode control for a quadrotor uav with a total rotor failure," *Aerospace Science and Technology*, vol. 98, p. 105716, Mar. 2020.

[64] M. Labbadi and M. Cherkaoui, "Robust adaptive backstepping fast terminal sliding mode controller for uncertain quadrotor uav," *Aerospace Science and Technology*, vol. 93, p. 105306, Oct. 2019.

[65] M. R. Cohen, K. Abdulrahim, and J. R. Forbes, "Finite-horizon lqr control of quadrotors on $SE_2(3)$," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5748–5755, Oct. 2020.

[66] T. Zioud, J. Escareno, and O. Labbani-Igbida, "Real-time of-based trajectory control of a uas rotorcraft based on integral extended-state lqg," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. Mexico City, Mexico: IEEE, Aug. 2022, pp. 1423–1430.

[67] Z. Zhang, Y. Niu, and J. Song, "Input-to-state stabilization of interval type-2 fuzzy systems subject to cyberattacks: An observer-based adaptive sliding mode approach," *IEEE Transactions Fuzzy Systems*, vol. 28, no. 1, pp. 190–203, Jan. 2020.

[68] H. Razmi and S. Afshinfar, "Neural network-based adaptive sliding mode control design for position and attitude control of a quadrotor uav," *Aerospace Science and Technology*, vol. 91, pp. 12–27, Aug. 2019.

[69] M. Vahdanipour and M. Khodabandeh, "Adaptive fractional order sliding mode control for a quadrotor with a varying load," *Aerospace Science and Technology*, vol. 86, pp. 737–747, Mar. 2019.

[70] S. Lian, W. Meng, Z. Lin, K. Shao, J. Zheng, H. Li, and R. Lu, "Adaptive attitude control of a quadrotor using fast nonsingular terminal sliding mode," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 2, pp. 1597–1607, Feb. 2022.

[71] M. Labbadi and M. Cherkaoui, "Robust adaptive nonsingular fast terminal sliding-mode tracking control for an uncertain quadrotor uav subjected to disturbances," *ISA Transactions*, vol. 99, pp. 290–304, Apr. 2020.

[72] O. Mofid and S. Mobayen, "Adaptive finite-time backstepping global sliding mode tracker of quad-rotor uavs under model uncertainty, wind perturbation, and input saturation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 1, pp. 140–151, Feb. 2022.

[73] B. Li, W. Gong, Y. Yang, B. Xiao, and D. Ran, "Appointed fixed time observer-based sliding mode control for a quadrotor uav under external disturbances," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 1, pp. 290–303, Feb. 2022.

[74] J.-J. Xiong, N.-H. Guo, J. Mao, and H.-D. Wang, "Self-tuning sliding mode control for an uncertain coaxial octorotor uav," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 2, pp. 1160–1171, Feb. 2023.

[75] V. K. Tripathi, A. K. Kamath, L. Behera, N. K. Verma, and S. Nahavandi, "An adaptive fast terminal sliding-mode controller with power rate proportional reaching law for quadrotor position and altitude tracking," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 6, pp. 3612–3625, June 2022.

[76] Z. Guo, H. Li, H. Ma, and W. Meng, "Distributed optimal attitude synchronization control of multiple quavs via adaptive dynamic programming," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 6, pp. 8053–8063, June 2024.

[77] C. Li, J. Ding, F. L. Lewis, and T. Chai, "A novel adaptive dynamic programming based on tracking error for nonlinear discrete-time systems," *Automatica*, vol. 129, p. 109687, July 2021.

[78] S. Li, L. Ding, H. Gao, Y.-J. Liu, L. Huang, and Z. Deng, "Adp-based online tracking control of partially uncertain time-delayed nonlinear system and application to wheeled mobile robots," *IEEE Transactions Cybernetics*, vol. 50, no. 7, pp. 3182–3194, July 2020.

[79] S. Cai, L. Dou, and X. Su, "Distributed adaptive dynamic programming formation control of multiple quadrotor-uav system," in *2021 40th Chinese Control Conference (CCC)*. Shanghai, China: IEEE, July 2021, pp. 4984–4989.

[80] X. Yi, B. Luo, and Y. Zhao, "Adaptive dynamic programming-based visual servoing control for quadrotor," *Neurocomputing*, vol. 504, pp. 251–261, Sept. 2022.

[81] L. Dou, X. Su, X. Zhao, Q. Zong, and L. He, "Robust tracking control of quadrotor via on-policy adaptive dynamic programming," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 7, pp. 2509–2525, May 2021.

[82] E. Nuño, A. Loría, and E. Panteley, "Leaderless consensus formation control of cooperative multi-agent vehicles without velocity measurements," *IEEE Control Systems Letters*, vol. 6, pp. 902–907, 2021.

[83] H. Rezaee and F. Abdollahi, "Adaptive leaderless consensus control of strict-feedback nonlinear multiagent systems with unknown control directions," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 10, pp. 6435–6444, 2020.

[84] M. Rehan, M. Tufail, and S. Ahmed, "Leaderless consensus control of nonlinear multi-agent systems under directed topologies subject to input saturation using adaptive event-triggered mechanism," *Journal of the Franklin Institute*, vol. 358, no. 12, pp. 6217–6239, 2021.

[85] J. Mei, W. Ren, and Y. Song, "A unified framework for adaptive leaderless consensus of uncertain multiagent systems under directed graphs," *IEEE Transactions on Automatic Control*, vol. 66, no. 12, pp. 6179–6186, 2021.

[86] S. Wang, H. Zhang, S. Baldi, and R. Zhong, "Leaderless consensus of heterogeneous multiple euler–lagrange systems with unknown disturbance," *IEEE Transactions on Automatic Control*, vol. 68, no. 4, pp. 2399–2406, 2022.

[87] L. Zhang, J. Sun, and Q. Yang, "Distributed model-based event-triggered leader–follower consensus control for linear continuous-time multiagent systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 10, pp. 6457–6465, 2020.

[88] J. Huang, W. Wang, C. Wen, J. Zhou, and G. Li, "Distributed adaptive leader–follower and leaderless consensus control of a class of strict-feedback nonlinear systems: A unified approach," *Automatica*, vol. 118, p. 109021, 2020.

[89] L. Gu, Z. Zhao, J. Sun, and Z. Wang, "Finite-time leader–follower consensus control of multiagent systems with mismatched disturbances," *Asian Journal of Control*, vol. 24, no. 2, pp. 722–731, 2022.

[90] H. Fan, K. Zheng, L. Liu, B. Wang, and W. Wang, "Event-triggered finite-time consensus of second-order leader–follower multiagent systems with uncertain disturbances," *IEEE Transactions on Cybernetics*, vol. 52, no. 5, pp. 3783–3793, 2020.

[91] J. Li, L. Yuan, T. Chai, and F. L. Lewis, "Consensus of nonlinear multiagent systems with uncertainties using reinforcement learning based sliding mode control," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 1, pp. 424–434, 2022.

[92] A. Mousavi, A. H. Markazi, and E. Khanmirza, "Adaptive fuzzy sliding-mode consensus control of nonlinear under-actuated agents in a near-optimal reinforcement learning framework," *Journal of the Franklin Institute*, vol. 359, no. 10, pp. 4804–4841, 2022.

[93] P. WERBOS, "Approximate dynamic programming for real-time control and neural modeling," *Handbook of intelligent control*, pp. 493–525, 1992.

[94] D. Bertsekas and J. Tsitsiklis, "Neuro-dynamic programming: An overview," in *Proceedings of 1995 34th IEEE Conference on Decision and Control*, vol. 1. New Orleans, LA, USA: IEEE, 1995, pp. 560–564.

[95] D. Liu and H. Zhang, "A neural dynamic programming approach for learning control of failure avoidance problems," *Citeseer*, vol. 10, no. 1, pp. 21–32, 2005.

[96] P. Werbos, "Building and understanding adaptive systems: A statistical numerical approach to factory automation and brain research," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 17, no. 1, pp. 7–20, Jan. 1987.

[97] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[98] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," *AAAI*, vol. 30, no. 1, Mar. 2016.

[99] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, "Dueling network architectures for deep reinforcement learning," 2015.

[100] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015.

[101] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," June 2016.

[102] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," Aug. 2017.

[103] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," 2018.

[104] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, Nov. 2017.

[105] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *FNT in Machine Learning*, vol. 11, no. 3-4, pp. 219–354, 2018.

[106] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: A survey," *Artif Intell Rev*, vol. 55, no. 2, pp. 895–943, Feb. 2022.

[107] J. Wang, J. An, M. Chen, N. Zhan, L. Wang, M. Zhang, and T. Gan, "From model to implementation: a network algorithm programming language," *Science China Information Sciences*, vol. 63, pp. 1–17, 2020.

[108] H. You, Y. Hu, Z. Pan, and N. Liu, "Density-based user clustering in downlink noma systems," *Science China Information Sciences*, vol. 65, no. 5, p. 152303, 2022.

[109] A. Fahim, "K and starting means for k-means algorithm," *Journal of Computational Science*, vol. 55, p. 101445, 2021.

[110] H. Li and J. Wang, "Capkm++ 2.0: An upgraded version of the collaborative annealing power k-means++ clustering algorithm," *Knowledge-Based Systems*, vol. 262, p. 110241, 2023.

[111] E. W. Dijkstra, "A note on two problems in connexion with graphs," in *Edsger Wybe Dijkstra: his life, work, and legacy*, 2022, pp. 287–290.

[112] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[113] M. Kleinbort, K. Solovey, Z. Littlefield, K. E. Bekris, and D. Halperin, "Probabilistic completeness of rrt for geometric and kinodynamic planning with forward propagation," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. i–vii, 2018.

[114] L. Dong, X. Yuan, and C. Sun, "Event-triggered receding horizon control via actor-critic design," *Science China Information Sciences*, vol. 63, pp. 1–15, 2020.

[115] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Advances in neural information processing systems*, vol. 12, 1999.

[116] M. Pflueger, A. Agha, and G. S. Sukhatme, "Rover-irl: Inverse reinforcement learning with soft value iteration networks for planetary rover path planning," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1387–1394, 2019.

[117] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan, "Rrt*-smart: Rapid convergence implementation of rrt* towards optimal solution," in *2012 IEEE international conference on mechatronics and automation*. IEEE, 2012, pp. 1651–1656.

[118] Z. Tang, X. Xu, F. Wang, X. Jiang, and H. Jiang, "Coordinated control for path following of two-wheel independently actuated autonomous ground vehicle," *IET Intelligent Transport Systems*, vol. 13, no. 4, pp. 628–635, 2019.

[119] S. Dankwa and W. Zheng, "Twin-delayed ddpg: A deep reinforcement learning technique to model a continuous movement of an intelligent robot agent," in *Proceedings of the 3rd international conference on vision, image and signal processing*, 2019, pp. 1–5.

[120] C. Qiu, Y. Hu, Y. Chen, and B. Zeng, "Deep deterministic policy gradient (ddpg)-based energy harvesting wireless communications," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8577–8588, 2019.

[121] C. Mu and Y. Zhang, "Learning-based robust tracking control of quadrotor with time-varying and coupling uncertainties," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 1, pp. 259–273, Jan. 2020.

[122] M. M. Fouda, S. Sakib, Z. M. Fadlullah, N. Nasser, and M. Guizani, "A lightweight hierarchical ai model for uav-enabled edge computing with forest-fire detection use-case," *IEEE Network*, vol. 36, no. 6, pp. 38–45, Dec. 2022.

[123] M. Atif, R. Ahmad, W. Ahmad, L. Zhao, and J. J. P. C. Rodrigues, "Uav-assisted wireless localization for search and rescue," *IEEE Systems Journal*, vol. 15, no. 3, pp. 3261–3272, Sept. 2021.

[124] B. Li, Q. Li, Y. Zeng, Y. Rong, and R. Zhang, "3d trajectory optimization for energy-efficient uav communication: A control design perspective," *IEEE Transactions on Wireless Communications*, vol. 21, no. 6, pp. 4579–4593, 2022.

[125] G. J. Ducard and M. Allenspach, "Review of designs and flight control techniques of hybrid and convertible vtol uavs," *Aerospace Science and Technology*, vol. 118, p. 107035, 2021.

[126] M. Labbadi and M. Cherkaoui, "Robust adaptive backstepping fast terminal sliding mode controller for uncertain quadrotor uav," *Aerospace Science and Technology*, vol. 93, p. 105306, 2019.

[127] Z. Zhang, Y. Niu, and J. Song, "Input-to-state stabilization of interval type-2 fuzzy systems subject to cyberattacks: An observer-based adaptive sliding mode approach," *IEEE Transactions Fuzzy Systems*, vol. 28, no. 1, pp. 190–203, 2020.

[128] H. Razmi and S. Afshinfar, "Neural network-based adaptive sliding mode control design for position and attitude control of a quadrotor uav," *Aerospace Science and Technology*, vol. 91, pp. 12–27, 2019.

[129] J. Jia, K. Guo, X. Yu, W. Zhao, and L. Guo, "Accurate high-maneuvering trajectory tracking for quadrotors: a drag utilization method," *Mechanical Systems and Signal Processing*, vol. 7, no. 3, pp. 6966–6973, 2022.

[130] B. Wang, X. Yu, L. Mu, and Y. Zhang, "Disturbance observer-based adaptive fault-tolerant control for a quadrotor helicopter subject to parametric uncertainties and external disturbances," *Mechanical Systems and Signal Processing*, vol. 120, pp. 727–743, 2019.

[131] X. Yi, B. Luo, and Y. Zhao, "Adaptive dynamic programming-based visual servoing control for quadrotor," *Neurocomputing*, vol. 504, pp. 251–261, 2022.

[132] L. Dou, X. Su, X. Zhao, Q. Zong, and L. He, "Robust tracking control of quadrotor via on-policy adaptive dynamic programming," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 7, pp. 2509–2525, 2021.

[133] N. O. Lambert, D. S. Drew, J. Yaconelli, S. Levine, R. Calandra, and K. S. J. Pister, "Low-level control of a quadrotor with deep model-based reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4224–4230, 2019.

[134] G. Wen, W. Hao, W. Feng, and K. Gao, "Optimized backstepping tracking control using reinforcement learning for quadrotor unmanned aerial vehicle system," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 8, pp. 5004–5015, 2022.

[135] J. Yoo, D. Jang, H. J. Kim, and K. H. Johansson, "Hybrid reinforcement learning control for a micro quadrotor flight," *IEEE Control Systems Letters*, vol. 5, no. 2, pp. 505–510, 2021.

[136] Y. Yang, T. Huang, T. Wang, and C.-Y. Wen, "A robust sliding-mode control framework for quadrotors subject to model uncertainty and external disturbances," in *2024 American Control Conference (ACC)*, 2024, pp. 3809–3814.

[137] Z. Hou, P. Lu, and Z. Tu, "Nonsingular terminal sliding mode control for a quadrotor uav with a total rotor failure," *Aerospace Science and Technology*, vol. 98, p. 105716, 2020.

[138] Z. YU, Y. Zhang, B. Jiang, J. Fu, and Y. Jin, "A review on fault-tolerant cooperative control of multiple unmanned aerial vehicles," *Chinese Journal of Aeronautics*, vol. 35, no. 1, pp. 1–18, 2022.

[139] E. Bernuau, D. Efimov, W. Perruquetti, and A. Polyakov, "On homogeneity and its application in sliding mode control," *Automatica*, vol. 351, no. 4, pp. 1866–1901, 2014.

[140] Y. Liu, H. Li, R. Lu, Z. Zuo, and X. Li, "An overview of finite/fixed-time control and its application in engineering systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 12, pp. 2106–2120, 2022.

[141] Z. Zuo, J. Tang, R. Ke, and Q.-L. Han, "Hyperbolic tangent function-based protocols for global/semi-global finite-time consensus of multi-agent systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 11, no. 6, pp. 1381–1397, 2024.

[142] W. Perruquetti, T. Floquet, and E. Moulay, "Finite-time observers: Application to secure communication," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 356–260, 2008.

[143] V. Andrieu, L. Praly, and A. Astolfi, "Homogeneous approximation, recursive observer design, and output feedback," *SIAM Journal on control and optimization*, vol. 47, no. 4, pp. 1814–1850, 2008.

[144] H. Feng, Q. Song, S. Ma, W. Ma, C. Yin, D. Cao, and H. Yu, "A new adaptive sliding mode controller based on the rbf neural network for an electro-hydraulic servo system," *ISA Transactions*, vol. 129, pp. 472–484, 2022.

[145] K. G. Vamvoudakis and F. L. Lewis, "Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, vol. 46, no. 5, pp. 878–888, 2010.

[146] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[147] S. Kakade and J. Langford, "Approximately optimal approximate reinforcement learning," in *Proceedings of the Nineteenth International Conference on Machine Learning*, 2002, pp. 267–274.

[148] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Advances in neural information processing systems*, vol. 12, 1999.

[149] S. M. Kakade, "A natural policy gradient," *Advances in neural information processing systems*, vol. 14, 2001.

[150] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[151] T. Huang, Y. Liang, X. Ban, J. Zhang, and X. Huang, "The control of magnetic levitation system based on improved q-network," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 191–197.

[152] H. Liu, B. Li, B. Xiao, D. Ran, and C. Zhang, "Reinforcement learning-based tracking control for a quadrotor unmanned aerial vehicle under external disturbances," *International Journal of Robust and Nonlinear Control*, vol. 33, no. 7, pp. 10 360–10 377, 2022.

[153] Z. Shang, Y. Jiang, B. Niu, G. Zong, X. Zhao, and H. Li, "Adaptive finite-time consensus tracking control for nonlinear multi-agent systems: An improved tan-type nonlinear mapping function method," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 4, pp. 5434–5444, 2024.

[154] G. Wen and B. Li, "Optimized leader-follower consensus control using reinforcement learning for a class of second-order nonlinear multiagent systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 9, pp. 5546–5555, 2021.

[155] L. Chen, C. Dong, and S.-L. Dai, "Adaptive optimal consensus control of multiagent systems with unknown dynamics and disturbances via reinforcement learning," *IEEE Transactions on Artificial Intelligence*, 2023.

[156] X. Yang, H. Zhang, and Z. Wang, "Data-based optimal consensus control for multiagent systems with policy gradient reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 8, pp. 3872–3883, 2021.

[157] Z. Hou and I. Fantoni, "Interactive leader–follower consensus of multiple quadrotors based on composite nonlinear feedback control," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 5, pp. 1732–1743, 2017.

[158] T. Yan, Z. Xu, and S. X. Yang, "Consensus formation tracking for multiple auv systems using distributed bioinspired sliding mode control," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1081–1092, 2022.

[159] K. Liu, R. Wang, X. Wang, and X. Wang, "Anti-saturation adaptive finite-time neural network based fault-tolerant tracking control for a quadrotor uav with external disturbances," *Aerospace Science and Technology*, vol. 115, p. 106790, 2021.

[160] L. Chen, Z. Liu, Q. Dang, W. Zhao, and G. Wang, "Robust trajectory tracking control for a quadrotor using recursive sliding mode control and nonlinear extended state observer," *Aerospace Science and Technology*, vol. 128, p. 107749, 2022.

[161] Y. Hong, J. Hu, and L. Gao, "Tracking control for multi-agent consensus with an active leader and variable topology," *Automatica*, vol. 42, no. 7, pp. 1177–1182, 2006.

[162] Z. Zuo, B. Tian, M. Defoort, and Z. Ding, "Fixed-time consensus tracking for multiagent systems with high-order integrator dynamics," *IEEE Transactions on Automatic Control*, vol. 63, no. 2, pp. 563–570, 2017.

[163] C. Qian and W. Lin, "A continuous feedback approach to global strong stabilization of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 46, no. 7, pp. 1061–1079, 2001.

[164] L.-X. Xu, H.-J. Ma, D. Guo, A.-H. Xie, and D.-L. Song, "Backstepping sliding-mode and cascade active disturbance rejection control for a quadrotor uav," *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 6, pp. 2743–2753, 2020.

[165] B. Li, W. Gong, Y. Yang, and B. Xiao, "Distributed fixed-time leader-following formation control for multiquadrotors with prescribed performance and collision avoidance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 5, pp. 7281–7294, 2023.

[166] L. Zhang, Y. Xia, G. Shen, and B. Cui, "Fixed-time attitude tracking control for spacecraft based on a fixed-time extended state observer," *Science China Information Sciences*, vol. 64, no. 11, p. 212201, 2021.

[167] M. T. Angulo, J. A. Moreno, and L. Fridman, "Robust exact uniformly convergent arbitrary order differentiator," *Automatica*, vol. 49, no. 8, pp. 2489–2495, 2013.

[168] M. Khodaverdian, S. Hajshirmohamadi, A. Hakobyan, and S. Ijaz, "Predictor-based constrained fixed-time sliding mode control of multi-uav formation flight," *Aerospace Science and Technology*, vol. 148, p. 109113, 2024.

[169] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[170] H. Wang, J. Shan, and H. Alkomy, "Fully distributed edge-based dynamic event-triggered control for multiple quadrotors," *IEEE/ASME Transactions on Mechatronics*, vol. 29, no. 4, pp. 3203–3214, 2024.

[171] X. Qin, Z. Zhao, P. Huang, and J. Li, "Multiple feedback recurrent neural network based super-twisting predefined-time nonsingular terminal sliding mode control for quad-rotor uav," *Aerospace Science and Technology*, p. 109282, 2024.

[172] H. Xu, D. Yu, and Y.-J. Liu, "Observer-based fuzzy adaptive predefined time control for uncertain nonlinear systems with full-state error constraints," *IEEE Transactions on Fuzzy Systems*, 2023.

[173] Q. Li, Y. Chen, and K. Liang, "Predefined-time formation control of the quadrotor-uav cluster'position system," *Applied Mathematical Modelling*, vol. 116, pp. 45–64, 2023.

[174] W. Gong, B. Li, C. K. Ahn, and Y. Yang, "Prescribed-time extended state observer and prescribed performance control of quadrotor uavs against actuator faults," *Aerospace Science and Technology*, vol. 138, p. 108322, 2023.

[175] J. Wang, K. A. Alattas, Y. Bouteraa, O. Mofid, and S. Mobayen, "Adaptive finite-time backstepping control tracker for quadrotor uav with model uncertainty and external disturbance," *Aerospace Science and Technology*, vol. 133, p. 108088, 2023.

[176] M. Lv, C. K. Ahn, B. Zhang, and A. Fu, "Fixed-time anti-saturation cooperative control for networked fixed-wing unmanned aerial vehicles considering actuator failures," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 6, pp. 8812 – 8825, 2023.

[177] B. Li, W. Gong, Y. Yang, and B. Xiao, "Distributed fixed-time leader-following formation control for multi-quadrotors with prescribed performance and collision avoidance," *IEEE Transactions on Aerospace and Electronic Systems*, 2023.

[178] F. Mainardi, *Fractional calculus and waves in linear viscoelasticity: an introduction to mathematical models*. World Scientific, 2022.

[179] Y. Xu, Y. Qu, D. Luo, H. Duan, and Z. Guo, "Distributed predefined-time estimator-based affine formation target-enclosing maneuver control for cooperative underactuated quadrotor uavs with fault-tolerant capabilities," *Chinese Journal of Aeronautics*, 2024.

[180] K. Liu, P. Yang, R. Wang, L. Jiao, T. Li, and J. Zhang, "Observer-based adaptive fuzzy finite-time attitude control for quadrotor uavs," *IEEE Transactions on Aerospace and Electronic Systems*, 2023.

[181] S. Xie, Q. Chen, and Q. Yang, "Adaptive fuzzy predefined-time dynamic surface control for attitude tracking of spacecraft with state constraints," *IEEE Transactions on Fuzzy Systems*, vol. 31, no. 7, pp. 2292–2304, 2022.

[182] O. Mofid and S. Mobayen, "Adaptive finite-time backstepping global sliding mode tracker of quad-rotor uavs under model uncertainty, wind perturbation, and input saturation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 1, pp. 140–151, 2021.

[183] Z. Zuo, C. Liu, Q.-L. Han, and J. Song, "Unmanned aerial vehicles: Control methods and future challenges," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 4, pp. 601–614, 2022.

[184] Z. Ma, Q. Wang, and H. Chen, "A joint guidance and control framework for autonomous obstacle avoidance in quadrotor formations under model uncertainty," *Aerospace Science and Technology*, vol. 138, p. 108335, 2023.

[185] Z. Zuo, "Nonsingular fixed-time consensus tracking for second-order multi-agent networks," *Automatica*, vol. 54, pp. 305–309, 2015.

[186] Z. Hou, P. Lu, and Z. Tu, "Nonsingular terminal sliding mode control for a quadrotor uav with a total rotor failure," *Aerospace Science and Technology*, vol. 98, p. 105716, 2020.