· UNIVERSITY OF OXFORD ·

· DEPARTMENT OF ENGINEERING SCIENCE ·

Safe and Stable Reinforcement Learning with State-wise Guarantees for Control

Liqun Zhao

Wolfson College



A thesis submitted for the degree of

Doctor of Philosophy

Supervised by Dr. Konstantinos Gatsis and Prof. Antonis Papachristodoulou

January 2026

This thesis is dedicated to

Yumei Fang (房玉梅) and Youhui Zhao (赵友辉),

my beloved Mum and Dad.

Acknowledgements

It is a great pleasure for me to have reached this significant milestone in my life. First and foremost, I would like to express my sincere gratitude to my supervisors, Dr Konstantinos Gatsis and Professor Antonis Papachristodoulou, for their insightful guidance and continuous support throughout the past four years of my DPhil. When I first arrived at Oxford, I was deeply curious about research but unsure of where to begin. With their effective guidance and encouragement, I gradually developed a solid understanding of my research area and began my journey as an independent researcher. I am particularly grateful for the academic freedom they granted me, which allowed me to explore and devote myself to the intersection of control theory and learning methods during my time at Oxford. Without their support, this thesis would not have been possible. I feel very fortunate to have had them as my supervisors.

Beyond my supervisors, I would also like to thank my collaborators. In particular, I am grateful to Hongpeng Cao at the Technical University of Munich for his prompt assistance and inspiring collaboration, despite the geographical distance. I am thankful to Jichen Zhang, my colleague sitting next to me at Oxford. Without the collaboration, it is very likely that I would not have entered the field of diffusion models. I greatly enjoyed the discussions and collaborations with all of them. I am also deeply honored to acknowledge the Chiang Chen Industrial Charity Foundation for its generous financial support through the Chiang Chen Overseas Fellowship, which greatly enabled me to pursue my research in the United Kingdom. This thesis is also written in memory of Dr Chiang, in recognition of his remarkable virtue.

I am also very fortunate to have been part of the Oxford Control Group and to have worked alongside many wonderful colleagues. I would like to thank Boyu Pang and Han Wang for the stimulating daily discussions over the past years, as well as Yuwen Chen, Zuxun Xiong, Ruohong Liu, Jingyi Huang, Keyan Miao, Allan Andre Do Nascimento, and many others, without whom I would have missed countless joyful moments during my time at Oxford.

Finally, I would like to thank my family for their unwavering support, encouragement, and unconditional love through both smooth and challenging times. A special thank you goes to my Mum, Yumei Fang, and my Dad, Youhui Zhao. I dedicate this thesis to you.

Abstract

Reinforcement learning (RL) has demonstrated impressive performance in a wide range of problems involving complex sequential decision-making and data-driven control; however, its deployment in many safety-critical tasks remains fundamentally limited due to its inherent randomness and the lack of rigorous safety and stability analysis. This thesis develops a coherent pipeline of control-theoretic and learning-based frameworks for safe and stable control with state-wise guarantees under mild assumptions via reinforcement learning, progressing from single-agent systems to general multi-agent tasks, and from model-based to fully model-free settings.

The thesis begins by establishing a foundational framework, which integrates Lyapunov certificates and control barrier functions into actor–critic reinforcement learning, serving as the first contribution. This framework enables the simultaneous enforcement of stability and safety constraints through an augmented Lagrangian formulation in white-box settings, where nominal system dynamics and predefined safety certificates are available. It is subsequently extended to gray-box settings, in which system dynamics and certificates are learned from data, leading to the second framework.

Beyond single-agent control, the thesis advances the state of the art in multi-agent reinforcement learning by introducing the third framework, which is designed to cope with challenges arising from non-stationarity and partial observability. This framework targets general multi-agent systems with independent objectives, and is developed for black-box, fully model-free RL settings, thereby accommodating scenarios in which no system dynamics are required and certificates are not known *a priori*. State-wise safety and stability are enhanced through neural barrier certificates and input-to-state stability analysis. Convergence to a feasible local Nash equilibrium is theoretically established, together with an explicit investigation of infeasibility arising from conflicting state-wise requirements.

To further demonstrate scalability and improve sample efficiency, the thesis proposes a final framework, which derives a theoretical bound that justifies decentralized training and decentralized execution paradigm under limited information, and incorporates neural ordinary differential equation–based policy refinement mechanism. Extensive empirical evaluations across cooperative and non-cooperative benchmarks demonstrate that the proposed method consistently achieves strong performance in terms of reward, while significantly reducing safety violations compared to baselines and exhibiting improved sample efficiency.

Overall, this thesis presents a family of four principled and progressively developed learning frameworks that can be selected and adapted for practical use according to the availability of system models and prior knowledge, as well as specific task settings. By bridging control theory and reinforcement learning, the thesis helps to enable reliable learning-based control in various complex and safety-critical systems.

Contents

Acknowledgements . . . iii
Abstract . . . iv

CHAPTER 1: Introduction

1.1 Motivation . . . 1
1.2 Contribution to the Literature . . . 4
1.3 Thesis Outline . . . 9

CHAPTER 2: Literature Review and Preliminaries

2.1 Reinforcement Learning (RL) . . . 11
2.1.1 Fundamentals . . . 11
2.1.2 Model-based Reinforcement Learning (MBRL) . . . 13
2.1.3 Model-free Reinforcement Learning (MFRL) . . . 16
2.2 Constrained Reinforcement Learning (CRL) . . . 16
2.2.1 Constrained Markov Decision Process (CMDP) . . . 17
(i) Reward Shaping Method . . . 18
(ii) Primal-dual Method . . . 19
(iii) Primal Method . . . 20
(iv) Others . . . 22
(v) Constrained Multi-agent Reinforcement Learning Based on CMDP Settings . . . 24
2.2.2 State-wise Constraints . . . 26
(i) Control Lyapunov Function . . . 26
(ii) Control Barrier Function . . . 28
(iii) Neural Certificates . . . 29
2.3 Neural Ordinary Differential Equations (NODEs) . . . 31

CHAPTER 3: State-Wise Safe and Stable Reinforcement Learning in White-box Settings

3.1 Introduction . . . 32
3.2 Problem Statement . . . 35
3.2.1 Definition of Safety . . . 36
3.2.2 Definition of Stability . . . 37
3.2.3 Definition of the Safe and Stable Control Problem . . . 38
3.3 Framework Design . . . 39
3.3.1 Value Function of the Cost . . . 39
3.3.2 Augmented Lagrangian Formulation for Parameter Updating . . . 41
3.3.3 Backup Controller Design . . . 44
3.4 Simulations . . . 47
3.4.1 Unicycle . . . 47
3.4.2 Simulated Car Following . . . 50
3.5 Conclusions . . . 51

CHAPTER 4: State-Wise Safe and Stable Reinforcement Learning for Gray-box Settings

4.1 Introduction . . . 53
4.2 Preliminaries and Problem Statement . . . 56
4.2.1 Preliminaries . . . 56
(i) Markov Decision Process . . . 56
(ii) Safety . . . 56
(iii) Stability . . . 58
(iv) Neural Ordinary Differential Equations (NODEs) for System Modeling 58
4.2.2 Definition of the Safe and Stable Control Problem . . . 59
4.3 Training with Predefined Control Barrier Functions . . . 59
4.3.1 Controller Update via ALM . . . 59
4.3.2 Backup Controller Design with Predefined Control Barrier Functions. . . . 62
4.4 Training with Learned Barrier Certificate . . . 63
4.4.1 Neural Barrier Certificates . . . 64
4.4.2 Theoretical Analysis. . . . 66
4.4.3 Combination with the Lyapunov-based Constraint . . . 70
4.5 Practical Implementation with SAC and NODEs . . . 72
4.6 Experiments . . . 77
4.7 Conclusions . . . 91

CHAPTER 5: State-Wise Safe and Stable Reinforcement Learning for General Multi-agent Settings in a Totally Model-free Manner

5.1 Introduction . . . 92
5.2 Problem Formulation and Preliminaries . . . 94
5.3 Methodology. . . . 96
5.3.1 State-Wise Safety via Neural Barrier Certificates for General MARL . . . 96
5.3.2 State-Wise Stability via ISS and Infeasibility under Multiple Constraints. . . 100
5.4 Experiments . . . 103
5.5 Conclusions . . . 107
5.6 Appendix . . . 108
5.6.1 A. Loss Functions for Updates . . . 108
5.6.2 B. Assumptions . . . 109
5.6.3 C. Appendix for State-Wise Safety via Neural Barrier Certificates. . . . 113
(i) Formal Version and Proof for Lemma 5.1. . . . 113
(ii) Proof for Proposition 5.1 . . . 114
(iii) Formal Version and Proof for Theorem 5.1 . . . 115
5.6.4 D. Appendix for State-Wise Stability via ISS and Infeasibility under Multiple Constraints . . . 127
(i) Proof for Theorem 5.2 . . . 127
(ii) Formal Version and Proof for Proposition 5.2 . . . 130
(iii) Proof for Corollary 5.1 . . . 132
5.6.5 E. Pseudocode for MAS^{3}AC . . . 132
5.6.6 F. Baselines . . . 134
5.6.7 G. Benchmark Environments . . . 134
(i) Cooperative Ant. . . . 134
(ii) Cooperative HalfCheetah . . . 135

(iii) Cooperative Coupled HalfCheetah . . . 136
(iv) Non-cooperative Coupled HalfCheetah . . . 138
(v) Non-cooperative Electrical Bidding Task . . . 139
(vi) Non-cooperative Ant . . . 141
(vii) Hyperparameters Setting . . . 142
5.6.8 H. Additional Experimental Results . . . 143
(i) The Evaluation on the Electrical Bidding Task . . . 143
(ii) Additional Results on Neighbor Size . . . 144
(iii) Additional Results on Lyapunov Function Input . . . 146
5.6.9 I. Further Extra Experiments and Ablation Studies . . . 147
(i) Robustness against External Disturbances . . . 148
(ii) Scaling to Larger N with Partial Observability . . . 148
(iii) Comparison with the Additional Baseline (Modified Scal-MAPPO-L) . 150
(iv) Performance on New Decentralized Ant Task . . . 151
(v) Ablation: Max-over-time vs. Sum-over-time Certificates . . . 152
(vi) Ablation: Impact of Input-to-State Stability (ISS) Part . . . 152
(vii) Ablation: No Reward Signal . . . 154
(viii) Ablation: Necessity of the Squared Term . . . 155
(ix) Lagrangian Multipliers Evolution . . . 156
(x) Plots of Differences between the Desired State . . . 157

CHAPTER 6: Improvements regarding Flexibility, Scalability, and Sample Efficiency

6.1 Introduction . . . 158
6.2 Methodology . . . 160
6.2.1 Restated State-Wise Stability Constraints and Convergence Analysis . . . 160
6.2.2 Scalability Analysis for the Decentralized Training Decentralized Execution Paradigm . . . 164
6.2.3 Accelerating the Training with NODE-based Design . . . 169
6.2.4 Practical Implementation . . . 174
6.3 Experiments . . . 174
6.3.1 Baselines . . . 176
6.3.2 Environment Settings . . . 176
6.3.3 Experimental Setup . . . 177
6.3.4 Results . . . 178
6.4 Conclusions . . . 186
6.5 Appendix . . . 187
6.5.1 A. Loss Functions for Updates . . . 187
6.5.2 B. Assumptions . . . 188
().1 Proof . . . 189
6.5.3 C. Formal Version and Proof for Theorem 6.2 . . . 193
6.5.4 D. Formal Version and Proof for Theorem 6.3 . . . 215
6.5.5 E. Additional Experimental Results . . . 227
(i) Additional Main Results on the Non-cooperative Ant task . . . 227
(ii) Additional Ablation Studies on Neighbor Size . . . 228

CHAPTER 7: Summary and Future Research Directions

7.1 Chapter 3: State-Wise Safe and Stable Reinforcement Learning in White-box Settings 230
7.2 Chapter 4: State-Wise Safe and Stable Reinforcement Learning for Gray-box Settings 232
7.3 Chapter 5: State-Wise Safe and Stable Reinforcement Learning for General Multi-agent Settings in a Totally Model-free Manner 233
7.4 Chapter 6: Improvements regarding Flexibility, Scalability, and Sample Efficiency . 234
Bibliography 235

CHAPTER 1

Introduction

1.1 Motivation

Reinforcement learning (RL) has achieved lots of exciting outcomes in solving many sophisticated sequential decision-making problems like Atari games [1], Go [2, 3], and Starcraft [4]. Its good performance in games further encouraged researchers to probe deeper into its applications to many real-world problems like robotics [5], including autonomous driving [6, 7] and surgical robotics [8]. However, due to the trial-and-error nature of RL, agents may sometimes try behaviors resulting in dangerous or harmful consequences during learning, and the lack of safety and stability is a major limitation that hinders the simulation achievements from being transferred to real-world applications. Although typically RL agents are trained mostly, if not entirely, in simulation platforms, real-world training and deployment inevitably introduce challenges such as real-world uncertainties [9] and strict safety requirements. For example, autonomous driving cars that are moving on the road should never cause collisions while exploring. Hence, constrained reinforcement learning is of significant importance for many systems like robotics [10].

In order to maintain the safety for RL-based control of agents during the training and/or deployment processes, the concept of *safe reinforcement learning* was proposed, and it can be defined as the process of learning policies that maximize the expectation of the return while respecting safety constraints [11]. Many existing safe reinforcement learning problems are constructed on the basis of a constrained Markov decision process (CMDP) [12], and various techniques have been proposed, including trust region policy optimization [13, 14], and primal-dual update [15–17]. However, these approaches based on CMDP define safety to

be requiring the cumulative discounted constraint violations along trajectories to stay below a predefined threshold, while many real-world applications require enforcing safety at each individual state [18]. State-wise safety is particularly important in safety-critical applications such as autonomous driving and robotics, where even a single violation can result in irreversible consequences [19]. For instance, in autonomous driving tasks, unlike the safety constraints used in CMDPs, which allow temporary violations as long as the cumulative discounted constraint violations remain within a threshold, collision avoidance must be enforced at every time step to ensure safe operation. Other examples include robotic manipulation, where enforcing state-wise constraints is essential for preventing damage to hardware, ensuring human safety, and maintaining operational constraints. A robot handling fragile objects must release them only when placed on a stable surface; otherwise, improper release may cause irreversible damage [18]. Additionally, in drone navigation, preventing unsafe conditions such as sudden altitude drops necessitates the strict enforcement of instantaneous safety constraints. Motivated by these cases, this thesis focuses on enforcement of state-wise constraints instead of relying on CMDP settings, leading to an RL-based controller satisfying requirement at every single state along trajectories over time.

Another problem is ensuring safety while preserving other properties. Besides safety, stability is also significant in many tasks, as it indicates convergence of the controlled system to the predefined goal state (desired equilibrium). Furthermore, for tasks where rewards are coupled with convergence toward the goal (e.g., navigation), stability can efficiently help to improve returns and prevent significant divergence, therefore mitigating severe performance deterioration during learning. The Lyapunov framework has found application in learning [20, 21] for stability. In [22], Lyapunov functions are used in model-free RL to help guarantee stability, though facing the potential problem of having relatively low sample efficiency. Taken together, ensuring both state-wise safety and stability with RL-based controllers is essential for achieving reliable performance in many systems. However, most existing approaches in constrained RL focus on either safety or stability in isolation. Purely safety-focused RL methods can lead to overly conservative policies [9], resulting in lower rewards, extended trajectories, and potentially suboptimal task completion. Conversely, stability-focused RL methods risk violating safety

constraints, which can have severe consequences in safety-critical applications. A long-standing challenge in combining both aspects is the issue of infeasibility caused by imposing both safety and stability constraints simultaneously [23], and addressing this challenge efficiently is crucial for enabling more reliable RL applications across a wide range of tasks. Inspired by these requirements, this thesis centers on efficient and simultaneous enforcement of both state-wise safety and stability constraints across a wide variety of tasks, and investigates practical implementation aspects and/or theoretical analysis regarding the issue of infeasibility.

While constrained reinforcement learning for single-agent tasks has become a focal point in recent research and many methods have been proposed, its appearance in multi-agent settings remains relatively limited, and many relevant challenges are still underexplored. The inherent challenges of multi-agent reinforcement learning (MARL) may hinder the direct extension of constrained reinforcement learning methods developed for single-agent tasks to multi-agent systems. For instance, RL has been integrated with Lyapunov functions to help guarantee stability for single-agent tasks [22], but this relies on a final unique stationary distribution, which may limit its extension to MARL tasks where non-stationarity is a well-known issue that may prevent the learning from converging to a final fixed joint policy [24]. Additionally, the curse of dimensionality can severely limit the direct application of methods developed for single-agent tasks to large-scale multi-agent tasks. To address these challenges, this thesis proposes the first framework that helps to obtain a joint controller with post-convergence state-wise safety and stability guarantees under mild assumptions for general MARL tasks with unknown dynamics, supported by both theoretical analysis and comprehensive empirical validations.

In addition, training efficiency is another critical aspect for RL. To be specific, both sample efficiency and computational cost should be taken into consideration when designing RL algorithms. It is common that model-free RL algorithms often exhibit lower sample efficiency compared to model-based approaches. Moreover, reliable models of systems can offer valuable guidance for maintaining both safety and stability in controlled agents. Thus, when a good nominal model is available, or the system is not too sophisticated to be approximated well by data-driven methods like Gaussian process regression and neural networks, it can be beneficial to adopt model-based methods instead of model-free ones to improve the sample efficiency,

leading to a smaller demand for data and less trial-and-error, which can be highly valuable for real-world tasks. Further, many existing constrained RL methods rely on the traditional dual ascent update, which is primarily applied to ensure constraint satisfaction for CMDPs. However, this approach may require sophisticated hyperparameter tuning and a long training process to obtain satisfactory results, mainly because of its sensitivity to hyperparameters like learning rates [25], and possibly weak and insufficient enforcement of constraints along long trajectories. This may result in many safety violations during learning, and therefore is also undesirable. Additionally, even if RL studies usually use multilayer perceptrons (MLPs) to construct policy networks, investigations on developing different structures of policy networks by using various learning techniques has become an active research area, and has the potential to improve the overall sample efficiency for RL frameworks. Meanwhile, computational cost is another long-standing issue for RL tasks, especially when large-scale settings like multi-agent systems are investigated [26]. To overcome these difficulties, this thesis develops several practical designs, together with specific theoretical analysis, targeting to improve the training efficiency of the proposed frameworks that can be chosen for practical use based on the task considered.

1.2 Contribution to the Literature

This thesis contributes to the literature on reinforcement learning (RL) and learning-based control by developing a family of principled frameworks that provide *state-wise safety and stability guarantees under mild assumptions*. While RL has achieved remarkable empirical success, its application to many tasks remains limited due to the lack of safety and stability, especially in settings involving multiple state-wise constraints, multiple interacting agents, limited system knowledge, and a high requirement for training efficiency, etc. This thesis addresses these challenges by systematically bridging control-theoretic concepts and modern RL techniques to propose different frameworks, which are supported by both rigorous theoretical analysis and extensive practical validations, and tailored to a range of practically relevant task settings.

State-Wise Safe and Stable Reinforcement Learning in White-box Settings

We first consider the problem of safe and stable control via reinforcement learning in *white-box* settings, where nominal system dynamics and predefined safety certificates are available. Differ-

ent from expectation-based formulations commonly adopted in constrained Markov decision process (CMDP) approaches, this work enforces *state-wise* safety and stability by integrating predefined control barrier functions (CBFs) and Lyapunov certificates into actor-critic reinforcement learning. Specifically, safety and stability constraints are incorporated into the Soft Actor-Critic (SAC) algorithm [27] via the augmented Lagrangian formulation, enabling the learning algorithm to explicitly and simultaneously account for both properties during policy training. For infeasible states where safety and stability constraints cannot be satisfied simultaneously, a backup controller based on CBF constraints is introduced, providing a practical mechanism for maintaining safety while encouraging convergence toward the desired equilibrium by incorporating the Lyapunov constraint into the objective function. While unifying safety and stability has been explored in prior work [28, 29], this contribution represents the first systematic integration of actor-critic reinforcement learning with CBF and Lyapunov constraints treated as separate conditions through an augmented Lagrangian formulation. The resulting Barrier-Lyapunov Actor-Critic (BLAC) framework demonstrates practically improved performance compared to baseline methods in case studies. These results have been published in the following paper.

Zhao, L., Gatsis, K. and Papachristodoulou, A., 2023. Stable and safe reinforcement learning via a barrier-lyapunov actor-critic approach. In 2023 62nd IEEE Conference on Decision and Control (CDC) (pp. 1320-1325). IEEE.

State-Wise Safe and Stable Reinforcement Learning for Gray-box Settings

Departing from the white-box framework, we relax the requirement for known system dynamics and predefined safety certificates, extending state-wise safe and stable RL to gray-box settings. While the first framework relies on nominal models and given CBFs, this part develops a method that can learn both system dynamics and safety certificates directly from data. First, in non-CMDP settings where predefined CBFs are still available, an augmented Lagrangian method–based policy optimization is proposed to enforce state-wise safety and stability simultaneously based on the learned system dynamics, while reducing sensitivity to hyperparameter tuning. For tasks in which no predefined CBFs are available, this contribution further introduces learnable

neural barrier certificates. It is proven that the cumulative discounted value of a specifically defined signal constitutes a valid barrier certificate, enabling safety constraints to be enforced through a trainable value-based formulation without requiring expert demonstrations. A policy update rule based on this learned barrier certificate is developed, together with theoretical analysis showing that the set of states where safety is maintained is monotonically non-shrinking, and the monotonic improvement of control performance in terms of the cumulative discounted reward. The interaction between learned safety constraints and Lyapunov-based stability requirements is also analyzed, demonstrating that the policy naturally learns to avoid infeasible states during training. These ideas are realized in a practical algorithm, Neural ordinary differential equations–based Lyapunov–Barrier Actor–Critic (NLBAC), which integrates the proposed method with Soft Actor-Critic, and NODE-based system modeling to alleviate the problem of large error propagation that is challenging for model-based RL. Empirical evaluations in four widely used tasks demonstrate that NLBAC achieves satisfactory performance in reducing safety violations and driving the controlled system towards the destination with higher sample efficiency. These results have been published as follows.

Zhao, L., Miao, K., Cao, H., Gatsis, K. and Papachristodoulou, A., 2025. NLBAC: A neural ODE-based algorithm for state-wise stable and safe reinforcement learning. Neurocomputing, 638, p.130041.

Zhao, L., Miao, K., Gatsis, K. and Papachristodoulou, A., 2024. Stable and safe human-aligned reinforcement learning through neural ordinary differential equations. In The Workshop on Human-aligned Reinforcement Learning for Autonomous Agents and Robots at ICRA 2024.

State-Wise Safe and Stable Reinforcement Learning for General Multi-agent Settings in a Totally Model-free Manner

Besides the single-agent frameworks developed for white-box and gray-box settings, the third contribution extends state-wise safe and stable RL to general multi-agent settings in a totally model-free way without requiring system dynamics. In contrast to existing safe MARL methods, which predominantly focus on cooperative tasks and expectation-based constraints, we address general-sum MARL problems with explicit state-wise safety and stability requirements. This

contribution introduces a model-free learning framework that integrates neural barrier certificates into MARL to enforce state-wise safety, and provides theoretical analysis on its convergence to a feasible local Nash equilibrium under modified dual ascent updates. In addition, stability of the overall multi-agent system (MAS) is addressed through input-to-state stability (ISS) analysis. Local ISS conditions are derived for each agent, showing that stable control can be achieved using only local state information and limited scalar exchanges from neighboring agents. This result provides a theoretical justification for decentralized training with reduced communication. The framework further investigates the issue of infeasibility arising from conflicting state-wise safety and stability constraints in MAS, and demonstrates that this issue does not prevent the framework from finding safe and stable joint controllers for MAS. These ideas are instantiated as Multi-Agent Safe and Stable Soft Actor-Critic (MAS^{3}AC), a unified framework to solve model-free multi-agent safe and stable control problems under centralized or decentralized training paradigms. We also provide environments for benchmarking safe and stable MARL algorithms. Empirical evaluations demonstrate that MAS^{3}AC consistently and efficiently achieves a strong balance between reward maximization and constraint satisfaction, outperforming baselines in both cooperative and non-cooperative settings. One conference submission, which contains these results and is currently under review, is listed as follows.

Zhao, L., Zhang, J., Gatsis, K. and Papachristodoulou, A. Safe and Stable Control of General Multi-Agent Systems via Reinforcement Learning under State-Wise Constraints. To be submitted to the Artificial Intelligence Journal, 2026.

Improvements regarding Flexibility, Scalability, and Sample Efficiency

Building upon the model-free multi-agent framework developed in the previous subsection, the fourth contribution further investigates challenges related to flexibility, scalability, and sample efficiency in safe and stable MARL. First, the framework restates the local conditions for each agent to achieve stable control of networked MAS, equipped with adaptive coefficients that enhance flexibility in stability enforcement. Then, we derive an upper bound on the update error of a surrogate policy gradient estimator arising from the use of local observations instead of global information, under certain assumptions on transition dynamics and policies.

This provides a theoretical justification for the off-policy decentralized training decentralized execution paradigm, where each agent updates its policy using only detected information under limited communication and computation resources, leading to scalability to large-scale multi-agent tasks. Furthermore, we design a neural ordinary differential equation (NODE)-aided policy network, incorporating the heavy-ball method, to improve the sample efficiency of the training, such that it has a better applicability to MAS where data collection is costly or desired to be limited. Finally, we propose Multi-Agent Safe Stable Scalable and Sample-efficient Neural ordinary differential equation–based Actor–Critic (MAS^{4}NAC), a unified framework for model-free multi-agent safe and stable control under centralized or decentralized training paradigms with improvements regarding flexibility, scalability, and sample efficiency. Empirical results across cooperative and general-sum tasks show that MAS^{4}NAC consistently performs greatly to maximize reward and reduce constraint violations efficiently. Moreover, ablation studies show that incorporating the NODE-based policy refinement design effectively improves sample efficiency, highlighting the framework' s superior applicability. One journal manuscript to be submitted, including all of these theoretical analyses and empirical results, is listed below.

Zhao, L., Zhang, J., Gatsis, K. and Papachristodoulou, A. Neural ODE–Driven Multi-Agent Reinforcement Learning: Unifying Safety, Stability, and Scalability. To be submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence, 2026.

Other Publications

Other work during my DPhil studies that is not included in this thesis has led to the following submissions and publications:

Cao, H.*, **Zhao, L.***, Sha, L. and Caccamo, M. AutoSafe: Safe Online Learning via Smooth Safety-Structured Policy Composition. Submitted to Transactions on Machine Learning Research (2026, **equal contribution**).

Zhang, J., **Zhao, L.**, Papachristodoulou, A. and Umenberger, J. Constrained Diffusers for Safe Planning and Control. In The Thirty-ninth Annual Conference on Neural Information Processing Systems, 2025.

Miao, K., **Zhao, L.**, Wang, H., Gatsis, K. and Papachristodoulou, A., 2025. Opt-ODENet:

Neural ODE Controller Design with Differentiable Optimization Layers for Safety and Stability. Proceedings of Machine Learning Research vol, 283, pp.1-13.

Xiong, Z., Wang, H., **Zhao, L.** and Papachristodoulou, A., 2024. Data-driven stable neural feedback loop design. In 2024 IEEE 63rd Conference on Decision and Control (CDC) (pp. 7344-7349). IEEE.

1.3 Thesis Outline

We now describe the structure of this thesis。

Chapter 2 presents the literature review and introduces the necessary preliminaries. It reviews fundamental reinforcement learning concepts, constrained reinforcement learning, as well as related topics including control Lyapunov function (CLF), control barrier function (CBF), neural certificates, and neural ordinary differential equations (NODEs). Mathematical formulations are provided where appropriate.

Chapter 3 presents the first contribution of the thesis, which addresses safe and stable reinforcement learning in white-box settings. This chapter introduces an actor–critic reinforcement learning framework that integrates control barrier functions and Lyapunov certificates via the augmented Lagrangian formulation, together with a safety-prioritized backup controller. The proposed Barrier–Lyapunov Actor–Critic framework is evaluated on two representative control tasks to demonstrate its effectiveness in enforcing state-wise constraints.

Chapter 4 develops a framework for gray-box settings. This chapter removes the requirement for known nominal system dynamics and predefined CBFs as safety certificates, and introduces learnable barrier certificates with theoretical guarantees under mild assumptions. Issues of infeasibility arising from conflicting safety and stability constraints are analyzed, and a Neural ODE–based Lyapunov–Barrier Actor–Critic algorithm is presented with better modeling precision and sample efficiency.

Chapter 5 generalizes state-wise safe and stable reinforcement learning to general multi-agent settings without utilizing system dynamics. This chapter develops a model-free multi-agent RL framework with state-wise safety constraints enforced via neural barrier certificates

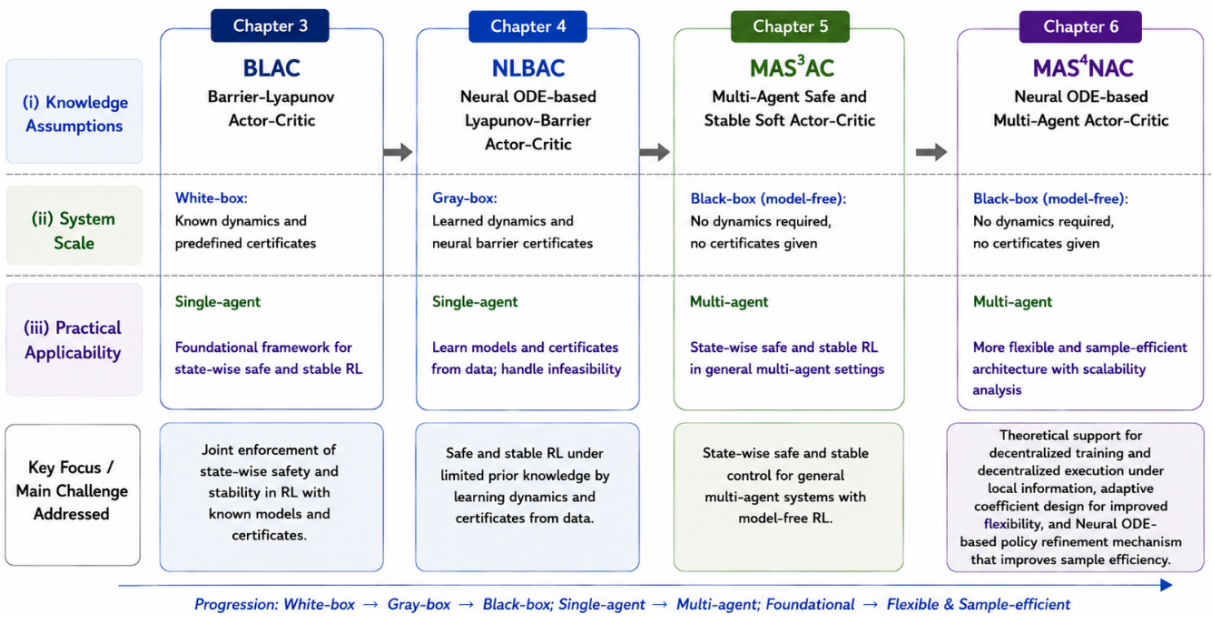


Figure 1.1: Roadmap of the thesis and evolution of the proposed frameworks. The figure illustrates the relationships among Chapters 3–6, the progression of problem settings, and the principal challenges addressed by each framework.

and stability constraints established through input-to-state stability analysis. Theoretical convergence to a feasible local Nash equilibrium is established, and the framework is validated on both cooperative and non-cooperative benchmark tasks.

Chapter 6 further investigates flexibility, scalability, and sample efficiency in safe and stable multi-agent reinforcement learning. This chapter derives the theoretical bound that justifies the decentralized training decentralized execution paradigm under limited information, and introduces a neural ODE–based policy refinement mechanism to improve sample efficiency. The resulting framework is evaluated on diverse multi-agent tasks, demonstrating its practical applicability.

Chapter 7 summarizes the main contributions of the thesis, and outlines directions for future research.

Figure 1.1 summarizes the overall progression of the proposed frameworks developed in this thesis. The figure illustrates how the research evolves from white-box to model-free settings, from single-agent to multi-agent systems, and from foundational safe-and-stable reinforcement learning frameworks to more flexible and sample-efficient architectures with scalability-oriented analysis, while maintaining the common objective of state-wise safety and stability.

CHAPTER 2

Literature Review and Preliminaries

This chapter reviews the literature most relevant to this thesis and introduces the essential concepts required for the subsequent chapters. We cover reinforcement learning fundamentals, constrained reinforcement learning, state-wise safety and stability constraints, and neural ordinary differential equations, which together form the methodological foundation of this work. The chapter also establishes common notation and basic mathematical preliminaries used in the thesis. More detailed preliminaries, assumptions, and definitions specific to each setting are deferred to the corresponding chapters where they are directly applied.

2.1 Reinforcement Learning (RL)

2.1.1 Fundamentals

Reinforcement Learning (RL) is a field of machine learning used to design actions for the system in a specific environment to maximize a cumulative discounted reward. RL is different from supervised learning since it does not need labelled input and output data and can learn the policy automatically with the observation and reward signal provided by the environment [30].

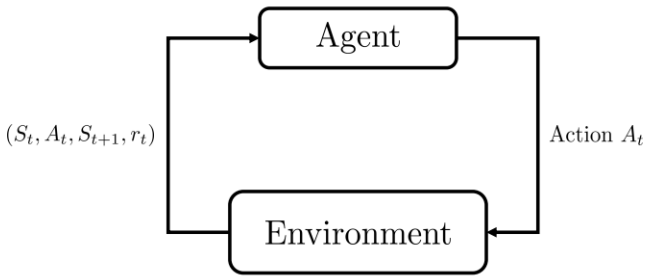


Figure 2.1: Simple illustration of basic reinforcement learning (adapted from[31]).

Most commonly, RL problems take the form of Markov decision processes (MDPs) where all

states are Markovian. To present the mathematical preliminaries of fundamental reinforcement learning, here we first adopt notation that is standard in the RL literature. In later sections, where control-theoretic constraints such as control barrier functions (CBFs) are involved, notation commonly used in control theory is specifically employed instead.

A Markov Decision Processes (MDP) $\mathcal{M}$ is defined by the tuple (S, A, P, r, γ) [32], where S is the set of states, A is the set of available actions, P is the state transition probability $P^a_{ss'} = \mathbb{P}[S_{t+1} = s'|S_t = s, A_t = a]$, r is the reward function $r^a_s = \mathbb{E}[r_t|S_t = s, A_t = a]$ where r_t is the reward obtained at the time-step t when the state is s and action a is taken, and $\gamma \in (0, 1)$ is the discount factor for future rewards. Algorithms are developed to maximize the expected return of reward following the policy π starting from an initial state distribution $\rho : S \mapsto [0, 1]$. Usually, a stochastic policy π is a probability distribution over actions and $\pi(a|s)$ represents the probability of taking action a when the state is s and therefore defines the behavior of the agent. For each state s, the state-value function of policy π is the expected return from the state s and can be denoted by $V_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s]$, where the return $G_t = r_t + \gamma r_{t+1} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$ is the cumulative discounted reward from time-step t. If we take action a into consideration, the action-value function $q_\pi(s, a)$ which is the expected return starting from state s where action a is chosen can be defined as $q_\pi(s, a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a]$. Furthermore, the value function satisfies an important relation called Bellman's equation which incorporates the immediate reward and the discounted value of the successor state: $V_\pi(s) = \mathbb{E}_\pi[r_t + \gamma V_\pi(S_{t+1})|S_t = s]$. Similarly, the action-value function can also be split as $q_\pi(s, a) = \mathbb{E}_\pi[r_t + \gamma q_\pi(S_{t+1}, A_{t+1})|S_t = s, A_t = a]$. The goal of the training is to maximize the expectation of the return. If we are given the initial state distribution $\rho : S \mapsto [0, 1]$, the RL problem can be expressed as:

$$\max_\pi J_\pi = \mathbb{E}_{\pi, s\sim\rho}[\sum_{t=0}^{\infty} \gamma^t r_t|S_0 = s] = \sum_{s\in S} \rho(s)V_\pi(s). \tag{2.1}$$

Several RL algorithms are designed to solve this optimization problem. Depending on whether the agent uses the predictions of the system state after one action a is taken during the learning process, we can classify current RL algorithms into two categories: model-free reinforcement learning (MFRL) where the agent learns the value function and policy by purely interacting

with the environment without making any predictions, and model-based reinforcement learning (MBRL) where predictions of system states are required to train the value function and policy.

2.1.2 Model-based Reinforcement Learning (MBRL)

For a Markov Decision Process $\mathcal{M}$ (S, A, P, r, γ), model-based reinforcement learning (MBRL) algorithms are characterized by having a nominal model or learning the dynamics of the system for the training, which means the state transition probability P and/or the reward function R are known before the training or are learned during the training process. If the model is precise enough, we can directly apply dynamic programming (DP) to solve the RL problem since we can evaluate a given policy π by iteratively applying the Bellman expectation equation[32]:

$$q_{\pi,k+1}(s,a) = r_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s') q_{\pi,k}(s', a'), \tag{2.2}$$

where r_s^a is the reward provided by the environment, s' is the successor state and a' is the action chosen at state s', and then improve the policy by acting greedily with respect to q_π by:

$$\pi'(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{a \in A} q_*(s,a), \\ 0 & \text{otherwise,} \end{cases} \tag{2.3}$$

where the optimal action-value function $q_*(s,a) = \max_\pi q_\pi(s,a)$ is the maximum action-value function over all policies. This means that we should choose the action leading to the maximum action-value. In many real-world applications, particularly in robotics, it is often possible to obtain at least a nominal model of the system. This makes it feasible to apply model-based methods rather than relying solely on direct interaction with the environment without predictive information about future states, rewards, or costs [33]. In general, there are two main approaches to obtaining system models. The first is to learn the system dynamics from scratch using collected data. The second is to start from a provided nominal model and progressively improve it using data-driven techniques. Two examples are provided as follows:

Dyna Algorithms: Dyna is a representative class of algorithms [34] for learning system models from scratch. In Dyna-style methods, training alternates between two main steps. First, the

current policy is used to interact with the environment and collect real experience, which is then used to learn or update a dynamics model. Second, the policy is improved using both real data and simulated data generated by the learned model. Dyna can learn value functions and policies by exploiting rich imaginary experience. Its combination with Q-learning [35] is shown below:

Algorithm 2.1 Dyna-Q Algorithm [32]

Initialization: action-value function $q(s, a)$, and $Model(s, a)$ which is an initialized neural network with state s and action a as the inputs, and predicted reward r and predicted next state $s^{'}$ as the outputs.
for $k = 1, \ldots, K$ **do**
  $s \leftarrow$ current state
  $a \leftarrow \epsilon$-greedy(s, q)
  Execute action a, and observe resultant reward r, and next state $s^{'}$
  $q(s, a) \leftarrow q(s, a) + \alpha[r + \gamma \max_{a'} q(s^{'}, a^{'}) - q(s, a)]$
  $Model(s, a) \leftarrow r, s^{'}$ (assuming deterministic environment)
  for $n = 1, \ldots, N$ **do**
    $s \leftarrow$ random previously observed state
    $a \leftarrow$ random action previously taken in s
    $r, s^{'} \leftarrow Model(s, a)$
    $q(s, a) \leftarrow q(s, a) + \alpha[r + \gamma \max_{a'} q(s^{'}, a^{'}) - q(s, a)]$
  end for
end for

Nominal Model with Gaussian Process: When a nominal model is available, it can be refined to obtain more accurate predictions. Gaussian processes (GPs) are widely used in this context to model the discrepancy between the nominal model and the true system dynamics, as well as to quantify uncertainty [36]. A Gaussian process (GP) is a nonparametric regression method for estimating functions and their uncertain distribution from data [37]. A GP can provide both a mean estimate $\mu_d(s)$, and a variance estimate $\sigma_d^2(s)$ of the difference between the real and nominal model. The difference $d(s)$ lies within the confidence interval:

$$\mu_d(s) - k_\delta \sigma_d(s) \leq d(s) \leq \mu_d(s) + k_\delta \sigma_d(s), \tag{2.4}$$

with probability $(1 - \delta)$ where k_δ is a parameter that determines δ (e.g. about 95% confidence is achieved at $k_\delta = 2$). For vector-valued disturbances, the confidence interval is applied component-wise. As more interaction data are collected, the GP produces increasingly accurate

estimates while reducing uncertainty. Suppose we have a nonlinear affine system model $s_{t+1} = f(s_t) + g(s_t)a_t + d(s_t)$, where s_{t+1} is the next state, a_t is the action taken by the agent, f is the drift vector field, g is a matrix, and $d(s_t) \in \mathbb{R}^{n_s}$ represents the unknown model discrepancy. Since the disturbance term is generally vector-valued, we decompose it as

$$d(s_t) = \begin{bmatrix} d_1(s_t) & d_2(s_t) & \cdots & d_{n_s}(s_t) \end{bmatrix}^\top,$$

and fit an independent scalar Gaussian process to each component $d_i(s_t)$. For notational simplicity, the following derivation is presented for an arbitrary disturbance component $d_i(s_t)$: The corresponding measurement is

$$\hat{d}_i(s_t) = s_{t+1}^{(i)} - f^{(i)}(s_t) - g^{(i)}(s_t)a_t,$$

where $f^{(i)}(s_t)$ denotes the i-th component of $f(s_t)$ and $g^{(i)}(s_t)$ denotes the i-th row of $g(s_t)$. Therefore, after collecting samples from the interaction, we obtain

$$y_n = [\hat{d}_i(s_1), \hat{d}_i(s_2), \ldots, \hat{d}_i(s_n)]^\top,$$

subject to additive independently identically distributed Gaussian noise $\mathcal{N}(0, \sigma_{noise}^2)$. The posterior mean $\mu_{d_i}(s_*)$ and variance $\sigma_{d_i}^2(s_*)$ at state s_* are given by:

$$\begin{aligned} \mu_{d_i}(s_*) &= k_*^T (K + \sigma_{noise}^2 I)^{-1} y_n, \\ \sigma_{d_i}^2(s_*) &= k(s_*, s_*) - k_*^T (K + \sigma_{noise}^2 I)^{-1} k_*. \end{aligned} \tag{2.5}$$

where $K \in \mathbb{R}^{n \times n}$ is the kernel matrix with entries $K_{p,q} = k(s_p, s_q)$, $k_* = [k(s_1, s_*), \ldots, k(s_n, s_*)]^\top$. Here, $k(\cdot, \cdot)$ denotes the scalar kernel function used for the Gaussian process of the i-th disturbance component, and I is the identity matrix of compatible dimension. As additional data are collected, $\mu_{d_i}(s)$ provides an increasingly accurate estimate of the disturbance component $d_i(s)$, while the corresponding uncertainty quantified by $\sigma_{d_i}^2(s)$ decreases. The complete disturbance estimate can then be obtained by combining the independently learned Gaussian process models for all disturbance components.

MBRL is believed to be much more sample-efficient than model-free RL algorithms introduced below, if the model is accurate. Thus, when an acceptable nominal model is available, or the system dynamics are not too complicated, MBRL has huge advantages over the model-free algorithms in terms of sample efficiency, safety, and stability, which are at the core of this thesis.

2.1.3 Model-free Reinforcement Learning (MFRL)

Compared to MBRL algorithms, model-free reinforcement learning algorithms (MFRL) are currently more widely applied since they neither require a nominal model of the system, nor need to train a model by using data collected during the learning process. Usually, MFRL algorithms need to evaluate a target policy $\pi(a|s)$ while following a behavior policy $\mu(a|s)$ to make decisions and collect the corresponding samples. Depending on whether the algorithm uses the policy as in (2.2) to make decisions so as to interact with the environment and sample the data (namely whether $\pi(a|s)$ equals $\mu(a|s)$), model-free algorithms can be split into two categories: on-policy algorithms and off-policy algorithms. On-policy algorithms attempt to directly evaluate the policy which is used to make decisions (therefore $\pi(a|s) = \mu(a|s)$), while off-policy algorithms evaluate a policy different from that used to make decisions ($\pi(a|s) \neq \mu(a|s)$). The table below summarizes the most commonly used MFRL algorithms.

Table 2.1: Commonly Used Model-free RL Algorithms (adapted from [38]).

Algorithm	Full Name	Category	Action Space	Operator
DQN [1]	Deep Q Network	Off-policy	Discrete	Q-value
DDPG [39]	Deep Deterministic Policy Gradient	Off-policy	Continuous	Q-value
TRPO [40]	Trust Region Policy Optimization	On-policy	Continuous or Discrete	Advantage
PPO [41]	Proximal Policy Optimization	On-policy	Continuous or Discrete	Advantage
TD3 [42]	Twin Delayed Deep Deterministic Policy Gradient	Off-policy	Continuous	Q-value
SAC [43]	Soft Actor-Critic	Off-policy	Continuous	Q-value

It is also easy to quantify the progress of control with RL due to the widely used benchmarks [44, 45].

2.2 Constrained Reinforcement Learning (CRL)

Although reinforcement learning has demonstrated impressive performance in various areas, such as video games and robotics, ensuring safety and stability, which are two critical properties from a control perspective, remains a significant challenge due to the inherent randomness of the learning process. Thus, constrained reinforcement learning (CRL) has become a focal

point in recent research [9, 46]. Many existing constrained RL approaches are developed based on the constrained Markov decision process (CMDP) settings. However, such settings enforce constraints only in expectation, allowing constraint violations to occur at individual states along a trajectory as long as the expected cumulative violation remains within a predefined bound. To obtain policies that satisfy state-wise safety and stability constraints at every individual state, thereby imposing stricter conditions that lie at the core of this thesis, we introduce, in the later part, control-theoretic techniques that can be applied to enforce state-dependent constraints.

2.2.1 Constrained Markov Decision Process (CMDP)

We first start from the constrained Markov decision process (CMDP). A CMDP extends a standard Markov decision process by incorporating an additional cost signal and is defined by the tuple (S, A, P, r, c, γ) [12], where c represents the cost associated with constraint violations. The objective is to maximize the expected return of rewards while ensuring that the expected return of costs incurred by following a policy π, starting from an initial state distribution $\rho : S \mapsto [0, 1]$, remains within a prescribed bound:

$$\begin{aligned} &\max_{\pi} \; J_\pi = \mathbb{E}_{\pi, s\sim\rho}[\sum_{t=0}^{\infty} \gamma^t r_t | S_0 = s] = \sum_{s\in S} \rho(s) V_\pi(s), \\ &s.t. \;\; C_\pi = \mathbb{E}_{\pi, s\sim\rho}[\sum_{t=0}^{\infty} \gamma^t c_t | S_0 = s] \leq d. \end{aligned} \tag{2.6}$$

Here, C_π denotes the cumulative discounted cost, c_t is the cost signal representing constraint violations incurred at the time-step t, $\gamma \in (0, 1)$ is the discount factor, and d is a predefined threshold, often referred to as a "safety budget" [47]. However, it is important to note that, the constraint in (2.6) only enforces that the expected return of costs remains below a prescribed threshold d, rather than ensuring state-wise constraints at every individual state along a trajectory to achieve zero constraint violation. Such state-wise guarantees are crucial in many application domains, including autonomous driving. Various techniques have been proposed to solve the constrained optimization problem (2.6), and these methods can be grouped into four categories: 1) reward shaping method; 2) primal-dual method; 3) primal method; 4) others. Furthermore, constrained multi-agent reinforcement learning based on CMDP settings is also

an active research direction. The detailed introductions are provided below:

(i) Reward Shaping Method

Reward shaping [48] is a classic method applied in the field of constrained RL, although it does not strictly solve the constrained optimization problem (2.6). In order to take constraint into consideration while maximizing the discounted reward, this method combines the original objective function $J_\pi = \mathbb{E}_{\pi,s\sim\rho}[\sum_{t=0}^{\infty}\gamma^t r_t|S_0 = s]$ with the constraint on the cost signal $\mathbb{E}_{\pi,s\sim\rho}[\sum_{t=0}^{\infty}\gamma^t c_t|S_0 = s] - d$ directly by treating the latter as the penalty function and using a hand-crafted penalty factor to have the following problem:

$$\max_{\pi} J_\pi - \lambda(C_\pi - d) = \mathbb{E}_{\pi,s\sim\rho}[\sum_{t=0}^{\infty}\gamma^t r_t|S_0 = s] - \lambda\left(\mathbb{E}_{\pi,s\sim\rho}[\sum_{t=0}^{\infty}\gamma^t c_t|S_0 = s] - d\right), \quad (2.7)$$

where λ is the penalty factor and is positive. This method is simple since it can be integrated to the existing RL algorithms like TRPO and PPO easily by combining the original reward signal and the original cost signal with a predefined factor λ to form a new reward signal, and then applying gradient ascent to update the policy which is approximated by a neural network with parameter θ[1]. However, the drawbacks of this method are:

1. Usually, researchers need to be highly familiar with the specific task to design a good penalty factor that combines the reward and cost signals to approximately solve the constrained optimization problem. Also, tuning the value of this factor is labor-consuming.

2. This method may not respond to the constraint enforcement promptly since the factor is fixed during training. For example, if the environment changes, the agent may not be able to update its policy in time due to the delay caused by the fixed penalty factor.

Although reward shaping typically outperforms unconstrained algorithms such as PPO and TRPO, which do not explicitly account for constraints, it cannot be shown to consistently outperform other more principled constrained methods such as constrained policy optimization (CPO) [13]. Usually, the reward shaping method is used as the baseline for comparison to evaluate the performance of other constrained methods, and is currently not an area of focus.

[1]Usually, both value functions and policies in RL are represented by neural networks which are universal approximators. Here we use θ to denote the parameter of the neural network representing the policy.

(ii) Primal-dual Method

Unlike the reward shaping method where the penalty factor is hand-crafted by researchers and perhaps cannot respond to changes promptly, the primal-dual method can update the factor throughout the training process and therefore remove the need of human experience and manually tuning of the penalty factor. Also, this method can respond quickly to any new data the agent obtains during the learning and hence has better performance. The primal–dual method reformulates the original problem (2.6) using a Lagrangian formulation as follows, in which a variable λ, also referred to as the Lagrange multiplier, is introduced:

$$L_\pi = -J_\pi + \lambda(C_\pi - d) = -\mathbb{E}_{\pi,s\sim\rho}[\sum_{t=0}^{\infty}\gamma^t r_t|S_0 = s] + \lambda\left(\mathbb{E}_{\pi,s\sim\rho}[\sum_{t=0}^{\infty}\gamma^t c_t|S_0 = s] - d\right), \tag{2.8}$$

At every iteration, the primal-dual method

1. Firstly fixes the factor λ, which is the Lagrangian multiplier, and updates the policy represented by a neural network with parameter θ by gradient descent; and
2. Then fixes the policy and updates λ by gradient ascent while ensuring the constraint $\lambda \geq 0$ is met.

Through repeated iterations, the policy parameters and the Lagrange multiplier are updated gradually and expected to converge to a locally optimal solution that is a local saddle point of the Lagrangian. This solution is typically feasible and acceptable in terms of improving the cumulative discounted reward while satisfying the constraints, although several other important issues also exist and therefore should be considered when applying primal–dual methods. These include whether strong duality holds and how large the duality gap is for a specific problem, whether the algorithm can be guaranteed to converge to an acceptable local optimum and how quickly it does so, and whether constraint guarantees apply only to the final learned policy, or also to intermediate policies obtained throughout the learning process.

A substantial body of literature has investigated primal–dual methods and approaches for addressing the challenges listed above. The representative work is *Reward Constrained Policy*

Optimization (RCPO) [15] which provides a proof of convergence to a local saddle point of the Lagrangian function under some assumptions. Later, subsequent methods are developed. For example, *Natural Policy Gradient Primal-Dual* (NPGPD) [49] was proposed to solve the discounted infinite-horizon constrained Markov decision processes (CMDPs) problem in a primal-dual way with the help of natural policy gradient [50], and *Optimistic Primal-Dual Proximal Policy Optimization* (OPDOP) [51] estimates the value function by combining the least-squares policy evaluation and an additional bonus term for safe exploration. *Accelerated Primal-Dual Optimization* (APDO) [52] incorporates an off-policy trained dual variable in the dual update procedure while updating the policy in primal space with on-policy likelihood ratio gradient. A proof of strong duality of the constrained optimization problem under some mild assumptions is given in [16, 53] which provides theoretical support to the approaches introduced above by establishing that the constrained optimization problem has zero duality gap if the policy is described by a good parameterization (e.g., neural networks). Furthermore, some concepts from other research areas have also been combined with the primal-dual method to solve the problem. For example, in [54] the Lagrangian multiplier is updated in a PID fashion which achieves favorable learning dynamics to improve the performance and ensure the safety without being more sophisticated to derive and implement compared to the traditional Lagrangian approach.

(iii) Primal Method

Although primal–dual methods generally achieve good performance, they also suffer from several limitations:

1. Since the primal-dual method needs to jointly update the dual variable together with the policy, the choice of the initial value of the dual variable (Lagrangian multiplier) λ can affect the convergence behavior of the algorithm.
2. Since the dual variable can only be updated gradually by gradient ascent, using its value to determine whether constraints are violated can be misleading. For example, even when a constraint is already satisfied, the corresponding Lagrange multiplier may remain positive for some time since its value is updated gradually from a positive number to 0,

potentially leading to incorrect assessments of constraint violations if the value of λ is used to determine whether the constraint is violated or not ($\lambda = 0$ when the constraint is not violated, and $\lambda > 0$ when the constraint is violated).

3. Due to the delay of the update of the dual variable, the intermediate policy obtained during the learning process may not satisfy constraints since the λ may also not react promptly to the feedback from the environment in time. Therefore, the policy updated correspondingly during training may fail to drive the agent safely in the environment.

The primal method solves the constrained optimization problem without keeping and updating the dual variable at every iteration, and is designed to improve the performance in terms of rewards while limiting the cumulative discounted cost by only updating the policy once when given a new batch of data. Therefore, primal methods are usually combined with on-policy algorithms, while primal-dual methods can naturally leverage an off-policy algorithm to use the data more efficiently. The representative work of primal method is *Constrained Policy Optimization* (CPO) [13] which is developed based on the Trust Region Policy Optimization (TRPO) method. CPO applies a first-order approximation to both the objective function and the cost constraints, together with a second-order approximation on the Kullback–Leibler divergence to simplify the optimization problem. To be more specific, the original problem is:

$$\begin{aligned} \pi_{k+1} = arg\max_{\pi}\ & \mathbb{E}_{a\sim\pi, s_0\sim\rho}[A_{\pi_k}(s,a)|S_0 = s_0], \\ s.t.\ & C_{\pi_k,i} + \frac{1}{1-\gamma}\mathbb{E}_{a\sim\pi, s_0\sim\rho}[A^{C_i}_{\pi_k}(s,a)|S_0 = s_0] \leq d_i \quad \forall i, \\ & \bar{D}_{KL}(\pi||\pi_k) \leq \delta, \end{aligned} \tag{2.9}$$

where $A_{\pi_k}(s,a)$ is the advantage function [32] calculated based on the reward, $A^{C_i}_{\pi_k}(s,a)$ is the advantage function calculated based on the ith cost, $C_{\pi_k,i}$ is the cumulative discounted value of the ith cost if the agent follows the policy π_k, γ is the discounted factor, d_i is the threshold of the ith constraint, $\bar{D}_{KL}$ is the Kullback–Leibler divergence between the current policy π_k, used to collect the data, and the decision variable π, which is finally the output of the optimization problem and therefore used as the new policy π_{k+1}. $\delta > 0$ is the threshold for the Kullback–

Leibler divergence. CPO simplifies this problem to the following by applying approximations:

$$\begin{aligned} \theta_{k+1} = arg\max_{\theta}\ & g^T(\theta - \theta_k), \\ s.t.\ \ & c_i + b_i^T(\theta - \theta_k) \leq 0 \quad \forall i, \\ & \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \leq \delta, \end{aligned} \tag{2.10}$$

where g is the gradient of the objective function, b_i is the gradient of the constraint on cost and $c_i = C_{\pi_k,i} - d_i$, H is the Fisher information matrix which is the Hessian of the KL divergence, and the policies π_k and π_{k+1} are parameterized by θ_k and θ, respectively. This problem is easier to be solved, and its analytical solution can be derived for quick computation if the problem has only one constraint on the cumulative discounted cost.

Several extensions of CPO have subsequently been proposed. For instance, *projection-based constrained policy optimization* (PCPO) [14] splits the update into two processes:

1. Reward improvement step, which performs a local reward improvement update without considering constraints on the cost to obtain an intermediate policy $\pi_{k+\frac{1}{2}}$ based on π_k.
2. Projection step which projects the intermediate policy $\pi_{k+\frac{1}{2}}$ onto the constraint set by minimizing the distance measure between $\pi_{k+\frac{1}{2}}$ and the solution π_{k+1}.

In addition, the *First Order Constrained Optimization in Policy Space* (FOCOPS) [55] method first finds the optimal update policy by solving a constrained optimization problem in the non-parameterized policy space, and then projects the update policy back into the parametric policy space. In summary, primal methods are typically implemented within a trust-region optimization framework, in which a locally approximated constrained optimization problem is solved at each policy update. This design encourages constraint satisfaction not only for the final policy but also for intermediate policies encountered during training, possibly leading to improved constraint satisfaction throughout the learning process [56].

(iv) Others

In addition to the methods discussed above, there are also other methods which are applied to solve the constrained optimization problem for constrained reinforcement learning: [21, 57]

define and present a method that uses the concept of Lyapunov stability to provide a way to ensure the safety of the policy. Other studies include [58], where a human overseer (or a blocker mimicking the human) is applied to prevent unsafe actions a and replace them with safe actions. Similarly, *Distributive Exploration Safety Training Algorithm* (DESTA) [59] introduces a safety agent that is applied to minimize safety violations, as well as a task agent whose goal is to maximize the reward. In this framework, the safety agent takes control of the system when safety violation may happen; otherwise, the task agent executes its policy. *SAUTE RL* [47, 60] addresses the problem by introducing Safety Augmented (Saute) MDPs where the safety constraints are not considered separately since the algorithm augments them into the state space without changing the algorithm itself, but creating a wrapper around the environment.

Here we provide Table 2.2 summarizing all introduced methods. It is important to note

Method	Algorithm (with name if available)
Reward Shaping	[48]
Primal-dual	RCPO[15] NPGPD[49] OPDOP[51] APDO[52] [16][53] [54]
Primal	CPO[13] PCPO[14] FOCOPS[55]
Others	[57][21] [58] DESTA[59] SAUTE RL[47][60]

Table 2.2: Summary of model-free constrained reinforcement learning algorithms.

that all the algorithms listed in the table are model-free RL algorithms since they do not use the model of the system to obtain predictions for training their value function and policy. Moreover, the constrained Markov decision process (CMDP) setting aims to help satisfy constraints on the expected cumulative cost, achieving reasonable empirical performance in a variety of tasks. However, there are two main drawbacks of these algorithms:

1. Expectation-based constraints are not suitable for enforcing state-dependent requirements, which specify constraints at every specific state rather than over entire trajectories. In many safety-critical applications, such as autonomous driving, state-wise constraints are

essential to achieve zero-violation performance. Existing CMDP-based approaches usually enforce constraints on the cumulative discounted cost $C_\pi = \mathbb{E}_{\pi,s\sim\rho}[\sum_{t=0}^{\infty}\gamma^t c_t|S_0 = s] \leq d$ over trajectories starting from the initial distribution ρ under policy π, and this expectation-based constraint is only defined over complete trajectories starting from $t = 0$ to infinity instead of every specific state s [61, 62]. To illustrate, while the expected number of collisions between an autonomous car and obstacles over a complete trajectory from the starting point to the destination may be bounded to be smaller than one predefined value, the vehicle may still enter unsafe states when operating close to obstacles, as no explicit state-wise constraint is imposed.

2. These model-free constrained RL algorithms may suffer from low sample efficiency since they rely entirely on real interactions with the environment to gain the data required for training the value function and the policy. In real-world domains such as robotics, data collection can be expensive and time-consuming. This limitation is particularly severe for on-policy algorithms [40, 41], which exploit the most recently collected data. Consequently, improving sample efficiency remains a major challenge in constrained RL.

These limitations motivate the use of model-based RL approaches if possible. When a model of the system is available, control-theoretic tools commonly used in the control community can be integrated into learning algorithms to improve constraint satisfaction. For instance, control barrier functions [28] can be combined with RL to enforce state-wise safety during both training and deployment processes as a safety filter [36] if a system model is provided, without relying solely on expectation-based constraints in CMDP settings. In addition, model-based reinforcement learning methods are typically more sample-efficient than model-free approaches. Algorithms such as Dyna leverage learned or known models to generate simulated experience, allowing agents to train using both real and simulated data. When the model is sufficiently accurate, this can substantially reduce the amount of real-world data required for training.

(v) Constrained Multi-agent Reinforcement Learning Based on CMDP Settings

A natural extension of single-agent reinforcement learning is to consider multi-agent settings, leading to the development of multi-agent reinforcement learning (MARL) algorithms. For

brevity, clarity, and ease of reference, detailed problem formulations and preliminaries for MARL are provided in Chapter 5, where MARL-specific notation is introduced and used throughout the subsequent analysis. MARL has already achieved notable success in solving complex decision-making problems for multi-agent systems (MAS) across diverse domains, such as video games [63]. Encouraged by these achievements, recent research has increasingly focused on applying MARL to real-world safety-critical applications such as autonomous driving [64] and power systems [65], where agents must operate under certain constraints.

Within existing settings, cooperative MARL tasks, where agents are required to collaborate to accomplish shared objectives while satisfying local and/or system-level constraints, have received particular attention [17, 66]. Meanwhile, among the various types of constraints considered, safety has been most extensively studied in recent MARL research [46]. Inspired by single-agent safe RL methods, many safe MARL algorithms also formulate the problem within the CMDP setting [17, 66–69]. In this formulation, safety constraints are still imposed on the expected cumulative discounted costs along trajectories, requiring them to remain below a predefined threshold. Although this provides a principled optimization framework, expectation-based constraints do not guarantee state-wise safety and allow constraint violations at individual states. However, many real-world multi-agent systems require stronger properties than those offered by CMDP-based formulations. In safety-critical applications, zero safety violations at every time step are often required to avoid catastrophic consequences, rather than merely bounding constraint violations in expectation over entire trajectories [18]. Moreover, the traditional dual ascent update, which is an optimization technique commonly adopted in this context [17, 67], may suffer from inefficient training and poor convergence behavior in multi-agent settings [66]. In addition, comparatively little attention has been paid to general MARL problems [70], where agents may have distinct objectives rather than sharing a single joint team reward, and where access to global information may be unavailable due to competition among agents. Nevertheless, many real-world domains are more naturally modeled under this general MARL setting.

Taken together, these challenges highlight the limitations of existing CMDP-based constrained MARL approaches and motivate the development of more advanced constrained MARL frameworks that can provide stronger conditions for safety and stability, improved

training efficiency, and applicability to general multi-agent scenarios, which are also the central focus of this thesis.

2.2.2 State-wise Constraints

There has been increasing interest in integrating control-theoretic approaches with RL to ensure desired properties of learning-based controllers [71], such as safety and stability. This subsection first introduces the control Lyapunov function as a widely used stability condition, providing a foundation for stable control via RL. It then introduces control barrier functions, which can serve as predefined safety certificates and are employed in the subsequent chapter to enforce state-wise safety constraints. Later, it reviews prior work utilizing neural certificates, which are learned from data and can be used when predefined analytical Lyapunov and/or barrier functions are unavailable. As this subsection focuses on control-theoretic constraints, notation commonly used in control theory is adopted throughout.

(i) Control Lyapunov Function

Firstly we give the definition of a control Lyapunov function. We start with the definition of Lyapunov stability.

Definition 2.1. *[72] The equilibrium point $x^* = 0$ of a system $\dot{x} = f(x)$, where $x \in \mathcal{X} \subset \mathbb{R}^n$ and f is locally Lipschitz with $f(0) = 0$, is:*

1. *Lyapunov stable (commonly referred to as "stable") if for each $\epsilon > 0$ there exists $\delta > 0$ such that $||x^0|| < \delta \Rightarrow ||x_t|| < \epsilon, \forall t \geq 0$, where x^0 denotes the initial state;*
2. *Unstable if it is not stable;*
3. *Asymptotically stable if it is stable and $\exists \delta > 0$ such that $||x^0|| < \delta \Rightarrow \lim_{t \to \infty} x_t = 0$;*
4. *Exponentially stable if it is stable and there exist constants $\alpha, \beta, \delta > 0$ such that for each $||x^0|| < \delta$, $||x_t|| \leq \alpha ||x^0|| e^{-\beta t}$.*

To analyze the stability of an equilibrium of a system, a common approach is to use a *Lyapunov function* $V(x)$ satisfying the following conditions.

Theorem 2.1 (Lyapunov Stability Criterion)**.** *Suppose there exists a differentiable function V :*

$\mathcal{X} \to \mathbb{R}$ *such that:*

1. $V(x) = 0$ *if and only if* $x = 0$*;*
2. $V(x) > 0$ *for* $x \neq 0$*;*
3. *The time derivative of* V *along system trajectories satisfies* $\dot{V}(x) = \frac{d}{dt}V(x) = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} f_i(x) = \nabla V \cdot f(x) \leq$ *for all* $x \neq 0$*.*

Then the equilibrium $x^* = 0$ *is (Lyapunov) stable. Moreover, if the inequality in 3 is strict, i.e.,* $\dot{V}(x) < 0$ *for* $x \neq 0$*, then the equilibrium is asymptotically stable.*

A control Lyapunov function (CLF) extends this concept to controlled systems by certifying the existence of a control policy that stabilizes the closed-loop system.

Definition 2.2 (Control Lyapunov Function [73])**.** *Consider a nonlinear affine system model* $\dot{x} = f(x) + g(x)u$*. A continuous differential function* $V : \mathcal{X} \to \mathbb{R}$ *is a control Lyapunov function if there exist positive constants* $c_1, c_2 > 0$ *such that for all* $x \in \mathcal{X}$*, the following inequalities hold:*

$$c_1||x||^2 \leq V(x) \leq c_2||x||^2,$$
$$\inf_{u \in U}[\mathcal{L}_f V(x) + \mathcal{L}_g V(x)u] \leq 0,$$

where $\mathcal{L}$ *denotes the Lie derivative operator, i.e.* $\mathcal{L}_f V(x) = \frac{\partial V(x)}{\partial x} f(x)$*.*

Consider the set defined by the Lyapunov function $V(x)$: $K_{clf}(x) = \{u \in U : \mathcal{L}_f V(x) + \mathcal{L}_g V(x)u \leq 0\}$, then any locally Lipschitz continuous controller $u : \mathcal{X} \to U$ in $K_{clf}(x)$ can stabilize the system. However, asymptotic stability (AS) may be not enough for fast convergence when applied in real applications. Therefore, exponentially stabilizing control Lyapunov functions (ES-CLF) [74] were proposed to obtain an exponentially stabilizing control law.

Definition 2.3 (Exponentially Stabilizing Control Lyapunov Function [74])**.** *A continuously differentiable function* $V : X \to \mathbb{R}$ *is an exponentially stabilizing control Lyapunov function (ES-CLF) if there exists positive constants* $c_1, c_2, c_3 > 0$ *such that for all* $x \in X$*, the following*

inequalities hold,

$$\begin{aligned} &c_1||x||^2 \leq V(x) \leq c_2||x||^2, \\ &\inf_{u\in U}[\mathcal{L}_f V(x) + \mathcal{L}_g V(x)u + c_3 V(x)] \leq 0. \end{aligned} \tag{2.11}$$

(ii) Control Barrier Function

In [75], the notion of control barrier function (CBF) was introduced, as well as approaches for safe controller design. Safety is closely related to the concept of forward invariance.

Definition 2.4 (Forward Invariance)**.** *Consider a system $\dot{x} = f(x)$ where $x \in \mathcal{X} \subset \mathbb{R}^n$ and f is locally Lipschitz. Suppose that for any initial condition $x^0 := x(t_0) \in \mathbb{R}^n$, there exists a maximum time interval $I(x^0) = [t_0, \tau_{max})$ such that $x(t)$ is the unique solution to $\dot{x} = f(x)$ on $I(x^0)$. Then, a set $\mathcal{S}$ is called forward invariant if for every $x^0 \in \mathcal{S}$, $x(t) \in \mathcal{S}$ for all $t \in I(x^0)$.*

The CBF is motivated by the formulation of the CLF. Just as stability requires a unique equilibrium point x^*, the safety property requires a safe set. We can define a set $\mathcal{C} \subset \mathbb{R}^n$ as:

$$\mathcal{C} = \{x \in \mathbb{R}^n : h(x) \geq 0\}, \partial\mathcal{C} = \{x \in \mathbb{R}^n : h(x) = 0\}, \text{Int}(\mathcal{C}) = \{x \in \mathbb{R}^n : h(x) > 0\}, \tag{2.12}$$

where $h : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function. For *safety* with respect to $\mathcal{C}$:

$$\exists u \; s.t. \; \dot{h}(x) \geq 0 \Rightarrow \mathcal{C} \text{ is } \textit{forward invariant}. \tag{2.13}$$

For a nonlinear affine system model $\dot{x} = f(x) + g(x)u$ where f and g are locally Lipschitz, we have the following definition.

Definition 2.5 (Control Barrier Function [28])**.** *The function h is called a control barrier function (CBF), if there exists an extended class-$\mathcal{K}_\infty$ function κ such that for the control system $\dot{x} = f(x) + g(x)u$, $\sup_{u\in U}\{\mathcal{L}_f h(x) + \mathcal{L}_g h(x)u + \kappa(h(x))\} \geq 0$, for all $x \in \mathcal{C}$. Based on this, we can define the feasible control law set: $K_{cbf}(x) = \{u \in U : \mathcal{L}_f h(x) + \mathcal{L}_g h(x)u + \kappa(h(x)) \geq 0\}$.*

Control barrier functions provide a systematic mechanism to enforce forward invariance of a designated safe set, and have already been applied widely in robotics [76] and perception-

based systems [29, 77]. Research concerning the combination of machine learning and CBF also appeared in recent years [78, 79]. However, in many existing approaches [36, 80, 81], a physics-based control-affine nominal model of the system is required. Moreover, requiring the predefined CBF to be affine with respect to the state [36], or adopting the continuous version of the CBF despite computing control actions in discrete time [80, 82], could potentially limit the applicability of these algorithms.

(iii) Neural Certificates

In many practical settings, predefined analytical control barrier functions (CBFs) are unavailable prior to training, especially when the task setting is sophisticated or domain knowledge is limited. Motivated by the success of CBFs in control theory, a growing body of work has explored the use of neural barrier certificates, which are learned from data and can enforce safety constraints without requiring closed-form expressions [83–86]. These approaches have demonstrated promising performance in reducing safety violations, and such methods are effective in scenarios involving adversarial attacks, which require a robust training process [87], as well as in tasks demanding probabilistic safety guarantees under model uncertainties [88]. Despite these advances, many existing methods require access to system dynamics, precise labeling of safe and unsafe states, and the condition that time derivatives should be negative to learn neural barrier certificates based on data. Meanwhile, safety labels may be obtained by human demonstrations [83, 84] that can be rough in practice, and in [89], an extra constraint decay function (CDF) and a separate safe policy should be learned in advance to train neural certificates, which can lead to additional computational overhead. The use of neural barrier certificates has also been extended to multi-agent settings to address state-wise safety constraints [90, 91]. However, these approaches either assume access to system dynamics [92], focus on cooperative tasks with a joint team reward/cost [91], or restrict attention to the centralized training scheme [90], limiting their applicability to general multi-agent reinforcement learning scenarios with state-wise constraints.

Beyond safety, stability is another critical property in many control tasks. Several studies have investigated the integration of learning with neural Lyapunov certificates [93, 94]. Never-

theless, compared to safety, stability in RL, particularly in multi-agent settings, remains relatively unexplored. [95] applies Lyapunov certificates in RL to provide stability guarantees; however, the requirement of a final unique stationary distribution may not easily be satisfied in MARL since the non-stationarity may make it difficult for the update to converge towards a final fixed joint policy. Outside RL, some works propose stable neural network-based controllers for multi-agent settings [20], but they still require system dynamics. To jointly ensure both safety and stability, some studies use control Lyapunov barrier functions [83, 85]. However, such approaches still rely on system dynamics, or expectation-based formulations. These limitations highlight the need for learning frameworks that can automatically train neural safety and stability certificates in a model-free manner, particularly for general multi-agent tasks.

Recent years have witnessed remarkable success of domain randomization. By exposing policies to diverse environments and disturbances during training, domain randomization has proven highly effective for improving robustness and sim-to-real transfer, enabling impressive performance in many challenging control tasks. Nevertheless, domain randomization and constrained reinforcement learning address complementary objectives. Constrained reinforcement learning seeks to explicitly incorporate desired properties into the learning process. In this thesis, learning is formulated as a constrained optimization problem, allowing safety and stability requirements to be represented directly through constraint functions. Although these constraints are inspired by control-theoretic concepts such as barrier certificates and Lyapunov functions, the corresponding certificates are learned from data and integrated into model-free reinforcement learning pipelines. Such a design combines the flexibility of modern reinforcement learning with structured mechanisms for promoting desired system properties. In applications where requirements such as safety are of particular importance, explicit constraint enforcement provides a principled alternative to relying solely on robustness obtained through unconstrained training. Furthermore, the underlying constrained-learning paradigm is not restricted to robotic control and may potentially be applied to other machine learning tasks where additional requirements, such as safety, reliability, or interpretability, are desired.

2.3 Neural Ordinary Differential Equations (NODEs)

Residual Networks (ResNets), proposed by [96], revolutionized neural networks by overcoming depth limitations using skip connections, significantly boosting their performance. This period also saw the rise in considering neural network hidden layers as states within a dynamical system. [97] expanded on this notion by introducing an ordinary differential equation (ODE) to model continuous dynamics within neural networks. The hidden state of a neural ODE can be defined by the solution of

$$\frac{dh(t)}{dt} = f\big(h(t), I(t), t, \theta\big), \tag{2.14}$$

where $h(t)$ denotes the hidden states, $I(t)$ denotes the external input, t is time, and f is a neural network parameterized by θ. Vanilla neural ODEs typically do not include the term $I(t)$ since their focus lies in the application of neural ODEs to classic machine learning problems, such as classification, considering it as an autonomous system. However, given Neural ODEs' origin in dynamical systems, they naturally lend themselves well to non-autonomous dynamical systems where external control variable I is incorporated.

Neural ODEs can be effectively utilized in tasks like trajectory planning and system identification [98, 99]. Moreover, this concept has also been applied to RL: [100] and [101] leverage NODEs to approximate the real dynamics of controlled systems for RL training, and [102] proposes a ODE-based RL algorithm developed to smooth action fluctuations. In this thesis, NODEs are employed for system modeling and policy refinement, improving modeling precision and sample efficiency for safe and stable control via reinforcement learning.

CHAPTER 3

State-Wise Safe and Stable Reinforcement Learning in White-box Settings

In this chapter, we consider state-wise safe and stable reinforcement learning in white-box settings, where nominal system dynamics and predefined safety certificates are available, serving as the starting point of the progressive study developed in this thesis. We propose an actor–critic reinforcement learning framework that integrates control barrier functions and Lyapunov certificates via an augmented Lagrangian formulation to enforce both safety and stability. A safety-prioritized backup controller is further introduced for infeasible states where safety and stability constraints conflict. Simulation results[1] demonstrate that the proposed framework enables systems to approach the desired state while incurring fewer safety violations compared to baseline algorithms.

3.1 Introduction

The remarkable success of reinforcement learning (RL) in solving complex sequential decision-making problems has inspired researchers to explore its potential in real-world applications, such as robotics [5]. However, the trial-and-error nature of RL may lead agents to exhibit actions resulting in dangerous consequences during learning. For example, although RL agents are typically trained in simulation platforms, it is crucial to take the challenges that arise in real-world scenarios, such as real-world uncertainties [103, 104], into consideration when training RL agents directly in real environments where safety is fundamental. Furthermore, regardless of the controller design method employed, stability is one of the paramount properties that need

[1]The code can be found in the GitHub repository: `https://github.com/LiqunZhao/A-Barrier-Lyapunov-Actor-Critic-Reinforcement-Learning-Approach-for-Safe-and-Stable-Control`.

to be guaranteed given a control system. Stability means the system will stay close or converge to the equilibrium, and it bears a close relationship with the performance of the control system. A lack of stability renders the system ineffective and even dangerous [22].

Safety in RL has been gaining attention from researchers. Existing safe RL approaches are mainly based on the constrained Markov decision process (CMDP) [12], and methods such as traditional dual ascent update [15, 16], and trust region policy optimization [13, 14] have been employed. However, in previous studies, safety constraints are usually defined based on the cumulative cost of an entire trajectory, rather than on the individual cost signals at each timestep along the trajectory. Consequently, safety violations at specific timesteps are wrapped by the trajectory expectation, and certain states may be permitted to be unsafe [19, 61, 62].

Recently, there has been growing interest in combining control-theoretic methods with RL to ensure the safety for the system [71]. In [105], a learning-based model predictive control method is proposed to provide safety guarantees with high probability. Besides, other concepts such as safe set algorithm (SSA) [62, 106] and control barrier function (CBF) [36, 80, 107, 108] have also been applied as safety constraints to help RL training maintain safety. However, according to [80], the supervised learning of the CBF layer in [36] has the potential to introduce approximations that may adversely impact RL training. Also, applying a filter to aid the RL-based controller in choosing a safe action at each timestep can lead to a jerky output, or trajectory, which is often undesirable in many applications. In [80], when the time interval between consecutive timesteps is not small enough, there might be a misalignment between the constrained optimization part, where continuous CBFs are used as constraints, and the discrete-time system dynamics yielding the state transitions for the Markov decision process (MDP).

The Lyapunov function, a popular tool in the control community, has recently been applied to RL as well [109]. Some previous studies develop ways to construct Lyapunov functions for RL training [57], while [33] proposes a method using Lyapunov functions to guide safe exploration, which is however difficult to be applied to high-dimensional systems due to the curse of dimensionality caused by the discretization of the state space. [22] and [110] apply Lyapunov certificates to model-free RL to help to guarantee the stability for various systems.

However, considering the sample efficiency, and the fact that usually at least a nominal model is available in real applications like robotic arms, model-based RL methods can be used since they are more sample efficient than model-free RL methods.

In this chapter, our focus is on helping to guarantee safety and stability for systems that can be expressed by MDP via RL-based controllers. The main work and contributions can be summarized as follows:

1. An RL-based controller is proposed by combining CBF and Lyapunov certificate with the Soft Actor-Critic (SAC) algorithm [27], and the augmented Lagrangian formulation is used to solve the corresponding constrained optimization problem. The RL-based controller assists in guaranteeing both safety and stability simultaneously with separate CBF and Lyapunov-based constraints, respectively, and the augmented Lagrangian formulation can update the controller parameters efficiently.

2. A backup controller is proposed to replace the RL-based controller when no feasible solution exists to satisfy both safety and stability constraints simultaneously. The inequality constraints of the backup controller are constructed based on CBFs, which help the system achieve and maintain safety. The Lyapunov certificate is incorporated into the objective function to prevent the system from diverging too far from equilibrium while satisfying CBF constraints. This safety-critical design is well-suited for real-world applications where safety is of paramount importance, and accelerates the system's approach towards its desired state in practice.

3. A framework called Barrier-Lyapunov Actor-Critic (BLAC) is proposed by combining the RL-based and backup controllers, and we test it on two simulation tasks. Our results show that the framework helps to achieve better results in terms of returns and constraint satisfaction compared to baselines.

While unifying safety and stability is common in many other studies [28, 29, 111], to the best of the authors' knowledge, this is the first time to combine actor-critic RL with CBF and Lyapunov certificate as separate constraints to train an RL-based controller. [112] uses RL to learn uncertainties in the CLF, CBF, and other constraints to generate control signals using

quadratic programming, and thus is different from the framework proposed in this chapter. Further, since control-theoretic constraints such as CBFs are involved, the notation commonly used in control theory is employed here.

3.2 Problem Statement

The Markov decision process (MDP) with control-affine dynamics can be defined by the tuple $\mathcal{M}$, which is $(\mathcal{X}, \mathcal{U}, f, g, d, r, c, \gamma, \gamma_c)$. $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{U} \subset \mathbb{R}^m$ are state and control signal spaces, and the state transitions for the MDP are obtained by the following control-affine system:

$$x_{t+1} = f(x_t) + g(x_t)u_t + d(x_t). \tag{3.1}$$

Here, $x_t \in \mathcal{X}$ is the state at timestep t, $u_t \in \mathcal{U}$ is the control signal at timestep t, and with an RL-based controller π, it is sampled from a distribution $\pi(u_t|x_t)$. $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ define the known nominal model of the system. $d : \mathbb{R}^n \to \mathbb{R}^n$ denotes the unknown model. r, c are the reward and cost, respectively. γ and γ_c are the discount factors.

Remark 3.1. *In practical applications, the nominal model that is known beforehand may perform poorly, which necessitates the need to learn a better model. Similar to [36, 80], we use a Gaussian process (GP), a kernel-based nonparametric regression model, to estimate the unknown dynamics d from data. Consequently, when constructing safety and stability constraints in Section 3.3, we replace d using the mean and variance given by the GP. Due to its poor scalability with a large number of data points, we cease GP updates after a certain number of episodes.*

Here we present some additional notations that will be used later. Based on (3.1), the transition probability can be denoted as $P(x_{t+1}|x_t, u_t) \triangleq I_{\{x_{t+1}=f(x_t)+g(x_t)u_t+d(x_t)\}}$ where $I_{\{x_{t+1}=f(x_t)+g(x_t)u_t+d(x_t)\}}$ is an indicator function that equals 1 if x_{t+1} satisfies (3.1) given x_t and u_t, and 0 otherwise. Similar to [22], the closed-loop transition probability is denoted as $P_\pi(x_{t+1}|x_t) \triangleq \int_{\mathcal{U}} \pi(u_t|x_t)P(x_{t+1}|x_t, u_t)du_t$. Moreover, the closed-loop state distribution at timestep t is denoted by $\upsilon(x_t|\rho, \pi, t)$, which can be calculated iteratively using the closed-loop transition probability: $\upsilon(x_{t+1}|\rho, \pi, t+1) = \int_{\mathcal{X}} P_\pi(x_{t+1}|x_t)\upsilon(x_t|\rho, \pi, t)dx_t, \ \ \forall t \in \mathbb{N}$, and $\upsilon(x_0|\rho, \pi, 0) = \rho$ is the initial state distribution.

3.2.1 Definition of Safety

In this subsection, we present the condition that the controller should satisfy to maintain system safety. Assuming there are k different safety constraints that need to be satisfied, the system is considered safe if

$$h_i(x_t) \geq 0, \quad \forall t \geq 0, \tag{3.2}$$

holds for each $i = 1, \ldots, k$. Here $h_i : \mathbb{R}^n \to \mathbb{R}$ is a function defined for the ith safety constraint, and a safe set $\mathcal{C}_i \subset \mathbb{R}^n$ can be defined by the super-level set of h_i as follows:

$$\mathcal{C}_i = \{x \in \mathbb{R}^n | h_i(x) \geq 0\}. \tag{3.3}$$

A safe set $\mathcal{C} \subset \mathbb{R}^n$ can therefore be defined as the intersection of all $\mathcal{C}_i$:

$$\mathcal{C} = \bigcap_{i=1}^{k} \mathcal{C}_i = \bigcap_{i=1}^{k} \{x \in \mathbb{R}^n | h_i(x) \geq 0\}. \tag{3.4}$$

We require the system state to remain within this set $\mathcal{C}$, i.e., the safe set $\mathcal{C}$ should be forward invariant. Therefore, the system is required to be safe at every timestep, and thus the constraint here is stricter than that constructed by the expected return of costs, which is widely used in previous studies. A control barrier function can be used to ensure forward invariance of the safe set, and therefore can be naturally applied here to help maintain system safety.

Definition 3.1 (Discrete-time Control Barrier Function [36])**.** *Given a set $\mathcal{C}_i \subset \mathbb{R}^n$ defined by (3.3), the function h_i is called a discrete-time control barrier function (CBF) for system (3.1) if there exists $\eta \in [0, 1]$ such that*

$$\sup_{u_t \in \mathcal{U}} \left\{ h_i\big(f(x_t) + g(x_t)u_t + d(x_t)\big) - h_i(x_t) \right\} \geq -\eta h_i(x_t), \tag{3.5}$$

holds for all $x_t \in \mathcal{C}_i$.

The existence of a CBF means the existence of a controller such that the set $\mathcal{C}_i$ is forward invariant. Consequently, safety is maintained if there exists a controller such that $\forall i \in [1, k]$, $h_i\big(f(x_t) + g(x_t)u_t + d(x_t)\big) - h_i(x_t) \geq -\eta h_i(x_t)$ holds for all $x_t \in \mathcal{C}$.

3.2.2 Definition of Stability

In this subsection, we first introduce the cost function, and then give the definition of stability used in this chapter.

In a stabilization task, our goal is to find a controller that can drive the system state to the equilibrium, i.e., the desired state, eventually. To achieve this goal, given the state x_t and control signal u_t, we define the instantaneous cost signal, which is one element in the tuple $\mathcal{M}$, to be $c(x_t, u_t) = \|x_{t+1} - x_{\text{desired}}\|$ where x_{t+1} is the next state following (3.1), and x_{desired} denotes the desired state (i.e., equilibrium). Alternatively, in a tracking task, the control signal can be denoted as $c(x_t, u_t) = \|x_{t+1} - r_s\|$ where r_s is the reference signal the system needs to track. Since we investigate the stability of a closed-loop system under a nondeterministic RL-based controller π, we combine $\pi(u_t|x_t)$, which is a Gaussian distribution in this chapter, with the cost signal $c(x_t, u_t)$ to define the cost function under the controller π as

$$c_\pi(x_t) = \mathbb{E}_{u_t \sim \pi} c(x_t, u_t) = \mathbb{E}_{u_t \sim \pi}[\|x_{t+1} - x_{\text{desired}}\|]. \tag{3.6}$$

The cost function $c_\pi(x_t)$ represents the expected value of the norm of the difference between the next state x_{t+1} following (3.1), and the desired state x_{desired}, over u_t sampled from the distribution $\pi(u_t|x_t)$. It is natural to expect that the value of $c_\pi(x_t)$ should decrease as t increases to drive the system state towards the equilibrium, and we hope eventually $c_\pi(x_t) = 0$, which means the state reaches the equilibrium. However, as the control signal is sampled from a Gaussian distribution, the state at timestep t is also distributed. We therefore adopt the definition of stability as presented in [22] for the framework proposed in this chapter.

Definition 3.2 (Stability in Mean Cost)**.** *Let $\upsilon(x_t|\rho, \pi, t)$ denote the closed-loop state distribution at timestep t. The equilibrium of a system is said to be stable in mean cost if there exists a positive constant b such that for any initial state $x_0 \in \{x_0|c_\pi(x_0) < b\}$, the condition*

$$\lim_{t \to \infty} \mathbb{E}_{x_t \sim \upsilon}\big[c_\pi(x_t)\big] = 0 \tag{3.7}$$

holds. If b is arbitrarily large, the equilibrium is globally stable in mean cost.

Remark 3.2. *It is important to note that the term "stability in mean cost" differs from the*

commonly used concept of Lyapunov stability. The condition (3.7) is the convergence condition required for asymptotic stability. Therefore, the condition of Lyapunov stability is not involved in Definition 3.2. For linear systems, convergence implies stability in the sense of Lyapunov. However, in general cases, this does not hold - see Vinograd's counterexample. Considering this, in Section 3.3, we apply an exponentially stabilizing Lyapunov-based constraint to achieve exponential stability, which implies Lyapunov stability for the equilibrium.

3.2.3 Definition of the Safe and Stable Control Problem

Based on the previous subsections, similar to [83], we give the formal definition of the safe and stable control problem:

Problem 3.1 (Safe and Stable Control Problem)**.** *Given a control-affine system* $x_{t+1} = f(x_t) + g(x_t)u_t + d(x_t)$*, a unique desired state (equilibrium)* $x_{desired}$*, a set* $\mathcal{X}_b = \{x_0 | c_\pi(x_0) < b\}$ *where* b *is an arbitrarily large positive number and* x_0 *denotes the initial state, a set of unsafe states* $\mathcal{X}_{unsafe} \subseteq \mathcal{X}$*, and a set of safe states* $\mathcal{X}_{safe} \subseteq \mathcal{X}$ *such that* $x_{desired} \in \mathcal{X}_{safe}$ *and* $\mathcal{X}_{safe} \cap \mathcal{X}_b \neq \emptyset$*, find a controller* π *generating control signal* u_t *such that all trajectories satisfying* $x_{t+1} = f(x_t) + g(x_t)u_t + d(x_t)$ *and* $x_0 \in \mathcal{X}_{safe} \cap \mathcal{X}_b$ *have the following properties:*

1. ***Safety****:* $x_t \in \mathcal{X}_{safe} \quad \forall t \geq 0$*.*
2. ***The Equilibrium Is Stable in Mean Cost:*** $\lim_{t\to\infty} \mathbb{E}_{x_t \sim \upsilon}\left[c_\pi(x_t)\right] = 0$ *where* $\upsilon(x_t|\rho, \pi, t)$ *is the closed-loop state distribution at timestep* t*.*

Therefore, the controller is used to help the system arrive at the desired state x_{desired} while avoiding the unsafe states.

Remark 3.3. *Since* b *is arbitrarily large in Problem 3.1,* $x_{desired}$ *is globally stable in mean cost and is the only equilibrium point. Thus, given this equilibrium is stable in mean cost, we can say that the system is stable in mean cost. For brevity, we will use the term "stability" to refer to "stability in mean cost" in the rest of this chapter, and when we say the system is stable, it means the system is stable in mean cost.*

Remark 3.4. *Given* b *is arbitrarily large, the problem formulation requires that from every initial state* $x_0 \in \mathcal{X}_{safe}$ *the system should arrive at the unique equilibrium* $x_{desired}$ *eventually. Therefore, this chapter excludes tasks where it is impossible for the system to reach the unique*

equilibrium $x_{desired}$ starting from some safe states, or modifications should be made to preclude those safe but undesirable states.

3.3 Framework Design

In this section, we first provide the value function of the cost and give a condition for stability. Then, we employ the augmented Lagrangian formulation to update the RL-based controller parameters, aiming to satisfy safety and stability conditions and thus help to guarantee both properties for the system. Finally, considering the possible infeasibility of the constrained optimization problem, a backup controller is proposed to replace the RL-based controller when no feasible solution exists to satisfy both safety and stability constraints simultaneously. The overall Barrier-Lyapunov Actor-Critic (BLAC) framework is included at the end of this section.

3.3.1 Value Function of the Cost

To assist in maintaining stability for the system, we provide the value function of the cost at the state x_t:

$$L_\pi(x_t) = \mathbb{E}_{\tau\sim\pi}\big[\sum_{i=0}^{\infty}\gamma_c^i c_\pi(x_{t+i})\big]. \tag{3.8}$$

Here $\tau = \{x_t, x_{t+1}, x_{t+2}, \cdots\}$ is a trajectory under controller π starting from the initial state x_t. Based on this definition, $L_\pi(x_t)$ can also be approximated by a neural network, and since $c_\pi(x_t)$ is non-negative for any $t \geq 0$, we may choose to apply the rectified linear unit (ReLU) as the output activation function to make $L_\pi(x_t) \geq 0$ for each x_t. Other methods to ensure non-negativity can be found in [22], [113].

Based on (3.8), to achieve stability, a natural approach is to consider imposing a condition in the algorithm that helps to ensure that the value of $L_\pi(x_t)$ decreases along the trajectory τ. Inspired by the concept of exponentially stabilizing CLF and [22], we first make two assumptions for the MDP:

Assumption 3.1. *The state and control signal are sampled from compact sets, and the reward and cost values obtained at any timestep are bounded by r_{max} and c_{max}, respectively. Therefore, the value function of the reward and cost are upper bounded by $\frac{r_{max}}{1-\gamma}$ and $\frac{c_{max}}{1-\gamma_c}$, respectively.*

Remark 3.5. *Assumption 3.1 should be interpreted as a property of the problem formulation*

rather than a consequence of stability. In many control and reinforcement learning settings, the state space and action space are explicitly defined as bounded domains (e.g., box), which are compact in finite-dimensional Euclidean spaces. Furthermore, the reward and cost functions considered throughout this thesis are directly constructed from the corresponding states and actions, such as tracking errors, distances, and norms. Consequently, when evaluated over compact state and action spaces, these quantities are naturally bounded, implying bounded rewards, costs, and value functions.

Assumption 3.2 (Ergodicity)**.** *The Markov chain induced by controller π is ergodic with a unique stationary distribution $q_\pi(x) = \lim_{t\to\infty} \upsilon(x_t = x|\rho, \pi, t)$, where $\upsilon(x_t|\rho, \pi, t)$ is the closed-loop state distribution.*

These assumptions are widely used in previous RL research. For more content on the relationship between models of control systems and Markov chains, as well as ergodicity, we refer to the book [114]. Then, we introduce the Lemma 3.1 invoked by [22] as follows:

Lemma 3.1. *Under Assumptions 3.1 and 3.2, the system defined in (3.1) is stable in mean cost if there exist positive constants α_1, α_2, β, and a controller π, such that*

$$\alpha_1 c_\pi(x) \leq L_\pi(x) \leq \alpha_2 c_\pi(x), \tag{3.9}$$

$$\mathbb{E}_{x\sim\mu_\pi, x'\sim P_\pi}\big[L_\pi(x') - L_\pi(x)\big] \leq -\beta \mathbb{E}_{x\sim\mu_\pi}\big[L_\pi(x)\big], \tag{3.10}$$

hold for all $x \in \mathcal{X}$. Here L_π defined in (3.8) is the value function of the cost under the controller π, and

$$\mu_\pi(x) \triangleq \lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N} \upsilon(x_t = x|\rho, \pi, t) \tag{3.11}$$

is the sampling distribution, where $\upsilon(x_t|\rho, \pi, t)$ is the closed-loop state distribution at timestep t.

Proof. The proof of this lemma closely resembles that of Theorem 1 in [22], where further details can be found. As a sketch, the proof combines the Lyapunov candidate construction and the mean-cost stability result from [22], and the boundedness from [115]. Specifically,

Appendix B.2 of [115] shows that the discounted cost value function L_π satisfies the bounds in (3.9) under Assumption 3.1. Theorem 1 of [22] then establishes that Lyapunov function satisfying the expected decrease condition (3.10) implies stability in mean cost. Applying Theorem 1 of [22] to the Lyapunov candidate L_π yields the result. □

According to the proof presented in [115], L_π naturally satisfies the constraints (3.9) under Assumption 3.1. In the next subsection, we present a method to obtain a controller that results in an L_π meeting stricter state-wise constraints and thus satisfying (3.10). We therefore can call L_π an exponentially stabilizing Lyapunov certificate.

3.3.2 Augmented Lagrangian Formulation for Parameter Updating

In the previous sections, we proposed conditions (3.5) and (3.10) that a controller should satisfy to help guarantee safety and stability. We now turn our attention to updating an RL-based controller π to meet these conditions, namely finding a qualified controller. This can be thought of as a constrained optimization problem with the above conditions as constraints. It is common to employ the traditional dual ascent method to update the RL-based controller parameters in previous studies. However, this approach often requires sophisticated hyperparameter tuning for adjusting values such as the learning rate, which might be challenging, and a long training process to obtain satisfactory results, mainly because of its sensitivity to hyperparameters and possibly weak and insufficient enforcement of constraints along long trajectories. Here we adopt the augmented Lagrangian formulation, inspired by its effectiveness in solving constrained optimization problems and [116], to update the parameters of the RL-based controller.

Practically, we need to construct the conditions based on data collected during the learning process. At each timestep, the system applies a control signal and receives feedback, including reward and cost. The resulting transition pair $(x_t, u_t, r_t, c_t, x_{t+1})$ is stored in the replay buffer, and we sample a batch of these transition pairs, denoted as $\mathcal{D}$, randomly from this replay buffer at each timestep to construct the CBF and Lyapunov-based constraints with the system model

for safety and stability, respectively:

$$\begin{aligned} &h_i(\hat{x}_{t+1}) - h_i(x_t) \geq -\eta h_i(x_t), && \forall i \in [1, k], \\ &L_\pi(\hat{x}_{t+1}) - L_\pi(x_t) \leq -\beta L_\pi(x_t), \end{aligned} \tag{3.12}$$

for each $x_t \in \mathcal{D}$. $\hat{x}_{t+1} = f(x_t) + g(x_t)u + d(x_t)$ is the predicted next state, and $d(x_t)$ is estimated and replaced by the mean value given by GP. Thus, the system model is used in RL training. Note that u here is the control signal generated by the current controller instead of being the historical control signal u_t generated by a previous controller and then stored in a transition pair in $\mathcal{D}$, and therefore, these constraints are functions of the controller π. To apply the augmented Lagrangian formulation, these inequality constraints are converted to equality constraints using ReLU[116] for each $x_t \in \mathcal{D}$:

$$\begin{aligned} &ReLU\big(h_i(x_t) - h_i(\hat{x}_{t+1}) - \eta h_i(x_t)\big) = 0, && \forall i \in [1, k], \\ &ReLU\big(L_\pi(\hat{x}_{t+1}) - L_\pi(x_t) + \beta L_\pi(x_t)\big) = 0. \end{aligned} \tag{3.13}$$

Remark 3.6. *In ideal settings, the above formulation should be an idealized constrained optimization problem defined over the full state space. In practical implementations, however, the expectations and constraints involved in the optimization are approximated using sampled trajectories collected through interactions with the environment, together with replay-buffer-based learning and neural function approximation. Consequently, the implemented algorithm should be viewed as a sampled approximation of the idealized optimization problem rather than an exact realization of it. The practical algorithm seeks to approximate the idealized constrained optimization problem for corresponding desired properties through data-driven learning.*

Then the actor-critic approach is used to learn the RL-based controller. We represent the parameters of the RL-based controller and two action-value networks used in SAC by θ and $\phi_i, i = 1, 2$, respectively. Moreover, we use L_ν, which is called Lyapunov network, to approximate L_{π_θ} defined in (3.8) with parameters ν. Using these notations, we formulate a

new constrained optimization problem as follows:

$$\begin{aligned}
&\min_{\theta} \ -V^{\pi_\theta} \\
&s.t. \ \mathbb{E}_{x_t\sim\mathcal{D},u_\theta\sim\pi_\theta}\big[ReLU\big(h_i(x_t)-h_i(\hat{x}_{t+1})-\eta h_i(x_t)\big)\big]=0, \ \ \forall i\in[1,k] \\
&\quad \mathbb{E}_{x_t\sim\mathcal{D},u_\theta\sim\pi_\theta}\big[ReLU\big(L_\nu(\hat{x}_{t+1})-L_\nu(x_t)+\beta L_\nu(x_t)\big)\big]=0.
\end{aligned} \tag{3.14}$$

Here expected values are calculated to construct constraints since we sample a batch of x_t from the replay buffer. The objective function $V^{\pi_\theta}(x_t)$ is:

$$V^{\pi_\theta}=\mathbb{E}_{x_t\sim\mathcal{D},}\big[\min_{j=1,2} Q_{\phi_j}(x_t,u_\theta(x_t))-\alpha\log\pi_\theta(u_\theta(x_t)|x_t)\big], \tag{3.15}$$

which is the same as the commonly-used objective function in SAC [27] with only small differences in notation. The reparameterized action generated from the stochastic policy $\pi_\theta(\cdot|x_t)$ is represented as $u_\theta(x_t)=\tanh(\mu_\theta(x_t)+\sigma_\theta(x_t)\odot\xi)$, where $\xi\sim\mathcal{N}(0,I)$,and μ_θ and σ_θ denote the mean and standard deviation of the controller π_θ, which is a Gaussian distribution, and $\odot$ represents element-wise multiplication. Additionally, loss functions of the action-value networks $Q_{\phi_i},i=1,2$, coefficient α, and Lyapunov network L_ν are:

$$\begin{aligned}
J_Q(Q_{\phi_i})=&\mathbb{E}_{(x_t,u_t,r_t,x_{t+1})\sim\mathcal{D},\xi\sim\mathcal{N}}\Big[\big[r_t+\gamma\big(\min_{j=1,2}Q_{targ,\phi_j}(x_{t+1},\\
&\tilde{u}_\theta(x_{t+1},\xi))-\alpha\log\pi_\theta(\tilde{u}_\theta(x_{t+1},\xi)|x_{t+1})\big)-Q_{\phi_i}(x_t,u_t)\big]^2\Big],
\end{aligned} \tag{3.16}$$

$$J_\alpha(\alpha)=-\alpha\times\mathbb{E}_{x_t\sim\mathcal{D},\xi\sim\mathcal{N}}\big[\log\pi_\theta(\tilde{u}_\theta(x_t,\xi)|x_t)+\mathcal{H}\big], \tag{3.17}$$

$$J_L(L_\nu)=\mathbb{E}_{(x_t,c_t,x_{t+1})\sim\mathcal{D}}\Big[\big[c_t+\gamma_c L_{targ,\nu}(x_{t+1})-L_\nu(x_t)\big]^2\Big], \tag{3.18}$$

where $Q_{targ,\phi_i},i=1,2$ are the target action-value networks, and $\mathcal{H}$ is a designed threshold set to be the lower bound of the entropy of the controller π_θ. The constrained optimization problem (3.14) can be solved by the augmented Lagrangian formulation, and the augmented

Lagrangian function is:

$$\begin{aligned}\mathcal{L}_A(\theta, \lambda_i, \zeta; \rho_{\lambda_i}, \rho_\zeta) = &- V^{\pi_\theta} + \sum_{i=1}^{k} \lambda_i \times \mathbb{E}_{x_t \sim \mathcal{D}, u \sim \pi} \big[ReLU\big(h_i(x_t) - h_i(\hat{x}_{t+1}) - \eta h_i(x_t)\big)\big] \\ &+ \sum_{i=1}^{k} \frac{\rho_{\lambda_i}}{2} \big[\mathbb{E}_{x_t \sim \mathcal{D}, u \sim \pi}\big[ReLU\big(h_i(x_t) - h_i(\hat{x}_{t+1}) - \eta h_i(x_t)\big)\big]\big]^2 \\ &+ \zeta \times \mathbb{E}_{x_t \sim \mathcal{D}, u \sim \pi} \big[ReLU\big(L_\nu(\hat{x}_{t+1}) - L_\nu(x_t) + \beta L_\nu(x_t)\big)\big] \\ &+ \frac{\rho_\zeta}{2} \big[\mathbb{E}_{x_t \sim \mathcal{D}, u \sim \pi} \big[ReLU\big(L_\nu(\hat{x}_{t+1}) - L_\nu(x_t) + \beta L_\nu(x_t)\big)\big]\big]^2,\end{aligned} \tag{3.19}$$

where λ_i and ζ are the Lagrangian multipliers for CBF and Lyapunov-based constraints, respectively, and ρ_{λ_i} and ρ_ζ are the corresponding coefficients for the additional quadratic terms. Since it is usually not easy to solve the problem

$$\theta_{k+1} = \arg\min_\theta \mathcal{L}_A(\theta, \lambda_{i,k}, \zeta_k; \rho_{\lambda_{i,k}}, \rho_{\zeta_k}), \tag{3.20}$$

directly in RL, we still apply gradient descent to update θ, and gradient ascent to update λ_i and ζ, while increasing the value of ρ_{λ_i} and ρ_ζ gradually to find a solution for the constrained optimization problem (3.14). The pseudocode is provided as a part of Algorithm 3.1.

3.3.3 Backup Controller Design

Due to the existence of several constraints, the feasibility of the constrained optimization problem (3.14) becomes a crucial problem during the learning process. Typically, there are two scenarios where infeasibility can cause problems:

1. The Lyapunov-based constraint for stability is violated, for example, the agent is required to cross an obstacle to reach the equilibrium by the stability constraint, but this is prevented by the safety constraint, and thus the agent is trapped in some specific positions near the obstacle.

2. The CBF constraint for safety is violated, for example, the desired state is not fixed but a reference signal and falls in a danger region, which means satisfying the Lyapunov-based constraint will breach the safety constraint eventually.

The frequent invalid control signals provided by the RL-based controller due to the infeasibility may prevent the system from approaching its desired state quickly, or violate the safety constraints severely. Given that safety takes priority when safety and stability constraints cannot be satisfied simultaneously, similar to [36, 80], we propose to design a backup controller by formulating an additional constrained optimization problem that leverages CBFs as constraints to achieve and maintain safety. However, compared to previous studies, the objective function of this backup controller is designed not only to minimize the difference between the actual control signal and nominal control signal, but also to prevent the system from deviating too much from its equilibrium by incorporating the Lyapunov certificate. We first establish a QP-based controller as follows:

$$\begin{aligned}
&\min_{u_{\text{modi}},\epsilon} \ \frac{1}{2}\, u_{\text{modi}}^T\, Q\, u_{\text{modi}} + k_{\epsilon,i}\epsilon_i^2 - \kappa \nabla_x L_\nu(x_t) \cdot g(x_t) u_{\text{modi}} \\
&\quad s.t.\ h_i\big(f(x_t) + g(x_t)(u_{\text{nominal}} - u_{\text{modi}}) + d(x_t)\big) - h_i(x_t) \geq -\eta h_i(x_t) - \epsilon_i, \quad \forall i \in [1, k],
\end{aligned} \tag{3.21}$$

where x_t is the current state of the system, Q is a symmetric positive semidefinite matrix, and similar to [36, 80], $d(x_t)$ is estimated and replaced by the mean and variance given by GP. ϵ_i is the slack variable introduced to enforce the feasibility of the constrained optimization problem with the coefficient $k_{\epsilon,i}$. $u_{\text{modi}} = u_{\text{nominal}} - u_{\text{actual}}$, where u_{modi} is the optimization variable, u_{nominal} is the nominal control signal, and u_{actual} is the actual control signal to perform. Additionally, $\nabla_x L_\nu(x)$ is the gradient of the Lyapunov network with respect to the state, and therefore the Lyapunov certificate is incorporated into the objective function with the coefficient κ.

The choices of coefficients, nominal control signal, and the condition for using the backup controller depend on the systems to which the backup controller is applied. This QP-based controller is tested and shown to work well in tasks where the CBFs are affine with respect to the control signal. When the CBFs are not affine, it is possible to apply local first-order linearization to the CBFs to obtain approximate affine surrogate CBFs, or other types of constrained optimization problems can be utilized to construct the backup controller, for example, using a quadratically constrained quadratic program (QCQP) when the CBF takes a quadratic form with respect to the control signal. Two detailed examples corresponding to the two typical

scenarios listed earlier in this subsection are included in Section 3.4.

In summary, the framework combining the RL-based and backup controllers can be summarized as Algorithm 3.1.

Algorithm 3.1 Barrier-Lyapunov Actor-Critic (BLAC)

Initialization: RL-based controller network π_θ, coefficient α, action-value networks $Q_{\phi_i}, i = 1, 2$, Lyapunov network L_ν, Lagrange multipliers λ_i and ζ, replay buffer $\mathcal{B}$, coefficients of quadratic terms ρ_{λ_i} and ρ_ζ, learning rates η_1, η_2, and η_3, quadratic term coefficient factor $C_\rho \in (1, \infty)$
for $k = 1, \ldots, K$ **do**
  if Backup controller should be used according to the condition specific to the task **then**
    Solve the constrained optimization problem (3.21)
    Apply the control signal u_{actual}
  else
    Sample and apply control signal u_t
    Store the transition pair $(x_t, u_t, r_t, c_t, x_{t+1})$ in $\mathcal{B}$
    Sample a batch of transition pairs randomly from $\mathcal{B}$, and construct CBF and Lyapunov-based constraints with the system model
    Update the Lyapunov network and action-value networks by using (3.18) and (3.16) according to

$$\nu_{k+1} \leftarrow \nu_k - \eta_1 \nabla_\nu J_L(L_{\nu_k})$$
$$\phi_{i_{k+1}} \leftarrow \phi_{i_k} - \eta_1 \nabla_{\phi_i} J_Q(Q_{\phi_{i_k}})$$

    Update the controller network and coefficient α by using (3.19) and (3.17) according to

$$\theta_{k+1} \leftarrow \theta_k - \eta_2 \nabla_\theta \mathcal{L}_A(\theta_k, \lambda_{i,k}, \zeta_k; \rho_{\lambda_{i,k}}, \rho_{\zeta_k})$$
$$\alpha_{k+1} \leftarrow \alpha_k - \eta_2 \nabla_\alpha J_\alpha(\alpha_k)$$

    Update the Lagrangian multipliers using (3.19) according to

$$\lambda_{i,k+1} \leftarrow \lambda_{i,k} + \eta_3 \nabla_{\lambda_i} \mathcal{L}_A(\theta_{k+1}, \lambda_{i,k}, \zeta_k; \rho_{\lambda_{i,k}}, \rho_{\zeta_k})$$
$$\zeta_{k+1} \leftarrow \zeta_k + \eta_3 \nabla_\zeta \mathcal{L}_A(\theta_{k+1}, \lambda_{i,k}, \zeta_k; \rho_{\lambda_{i,k}}, \rho_{\zeta_k})$$

    Update coefficients of quadratic terms by $\rho_{\lambda_{i,k+1}} \leftarrow C_\rho \rho_{\lambda_{i,k}}$, $\rho_{\zeta_{k+1}} \leftarrow C_\rho \rho_{\zeta_k}$, and GP model
  end if
end for
return π_θ, $Q_{\phi_i}, i = 1, 2$, and L_ν.

Remark 3.7. *Stronger conclusions can be obtained in the idealized white-box setting where the disturbance term vanishes and exact system dynamics are available. In particular, for the safety-prioritized backup controller proposed in this chapter, the control barrier function constraint can be applied using the true dynamics and imposed directly as the only constraint. Therefore, provided that the system is initialized within the safe set, the barrier function is properly designed, and*

the corresponding constrained optimization problem is feasible, the safe set is forward invariant and safety is guaranteed by construction. For the reinforcement-learning-based controller, however, exact system knowledge does not eliminate the possibility of infeasibility arising from the simultaneous enforcement of safety and stability constraints. Consequently, stronger guarantees remain dependent on the feasibility of the underlying constrained optimization problem and the quality of the selected barrier functions. This observation motivates the later chapters of the thesis, which investigate the learning of barrier certificates and safe-and-stable control in settings where accurate models and predefined certificates are unavailable.

3.4 Simulations

In this section, we test the framework on two tasks to answer the following questions:

1. Does the BLAC framework help to ensure the stability for the system when compared to other baseline algorithms? Since in these tasks high rewards will be given when the system approaches and achieves the desired state (equilibrium), we use the cumulative reward as a measure, where a higher cumulative reward indicates that the system can converge to the equilibrium more efficiently.
2. Does the BLAC framework cause fewer safety violations compared to baseline algorithms?

We use LAC [22], CPO [13], PPO-Lagrangian and TRPO-Lagrangian [117] as baselines. These baselines use the Lyapunov-based method, trust region policy optimization, and primal-dual method, respectively, and are representative works in the literature of constrained reinforcement learning. Furthermore, to demonstrate whether the Lyapunov-based constraint contributes to maintaining stability and therefore improves performance, we remove the Lyapunov-based constraint in the BLAC framework to create an additional algorithm Barrier Actor-Critic (BAC) as another baseline. Two custom environments built based on [36, 80] are used since model knowledge is required for the proposed method.

3.4.1 Unicycle

This experiment is modified from the first environment in [80]. In this experiment, a unicycle is required to arrive at the desired location, i.e., destination, while avoiding collisions with

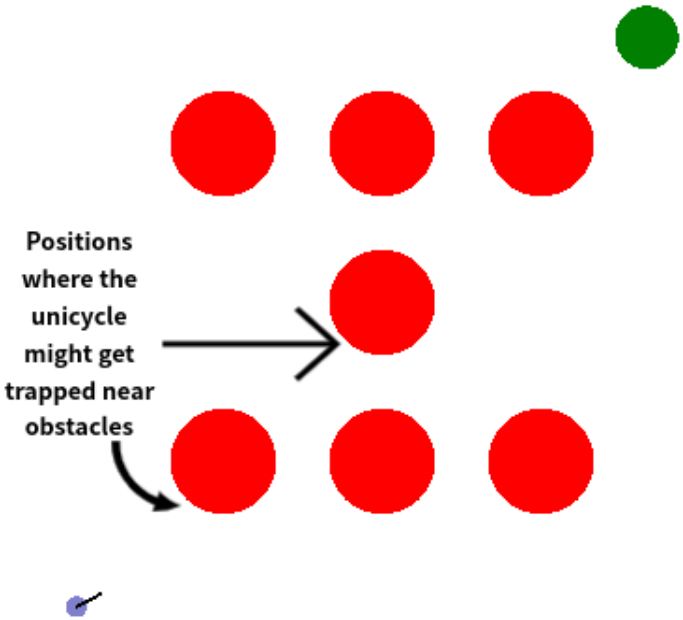


Figure 3.1: The unicycle environment. The blue point is the unicycle system, the green circle is the destination (desired state), and the red circles are obstacles to avoid. Some positions where the unicycle can get trapped due to the infeasibility of the constrained optimization problem with multiple CBF and Lyapunov-based constraints are shown.

obstacles. The model of the unicycle is given as:

$$x_{t+1} = x_t + \begin{bmatrix} \Delta T \cos(\theta_t) & 0 \\ \Delta T \sin(\theta_t) & 0 \\ 0 & \Delta T \end{bmatrix} (u_t + u_{d,t}).$$

Here $x_t = [x_{1t}, x_{2t}, \theta_t]^T$ where x_{1t} and x_{2t} are the X-coordinate and Y-coordinate of the unicycle at the timestep t, θ_t is the angle between the X-coordinate and the direction of the unicycle's movement at t. $u_t = [v_t, \omega_t]^T$ is the control signal where v_t and ω_t are the linear and angular velocities, respectively. ΔT represents the time interval. $u_{d,t} = -0.1[\cos(\theta_t), 0]^T$ is unknown to the nominal model, and therefore is the unknown part and GPs can be used. Then, similarly to [80], to formulate collision-free safety constraints, a point at a distance $l_p \geq 0$ ahead of the unicycle is considered, and we define the function $p : \mathbb{R}^3 \to \mathbb{R}^2$ to be

$$p(x_t) = \begin{bmatrix} x_{1t} \\ x_{2t} \end{bmatrix} + l_p \begin{bmatrix} \cos(\theta_t) \\ \sin(\theta_t) \end{bmatrix}.$$

The reward signal is defined as $-K_1(v_t - v_s)^2 + K_2\Delta d$, where v_s is the predefined velocity, Δd is the decrease in the distance between the unicycle and destination in two consecutive timesteps, and K_1 and K_2 are coefficients set to 0.1 and 30, respectively. The cost signal is $\|p(x_{t+1}) - p(x_{\text{desired}})\|$ where $p(x_{\text{desired}}) = [x_{1\text{desired}}, x_{2\text{desired}}]^T$ denotes the position of the desired location. CBFs are defined as $h_i(x_t) = \frac{1}{2}\big((p(x_t) - p_{\text{obs}_i})^2 - \delta^2\big)$ where p_{obs_i} is the position of the ith obstacle, and δ denotes the minimum required distance between the unicycle and obstacles. When stability constraint is violated if safety and stability constraints cannot be satisfied simultaneously, the unicycle can get trapped near obstacles, and then the RL-based controller is replaced by the backup controller where u_{nominal} is set to be the maximum allowable control signal to encourage exploration while maintaining safety. The RL-based controller will resume when the unicycle moves away from the trapped position for a long distance, or when the predetermined time threshold for using the backup controller is exceeded.

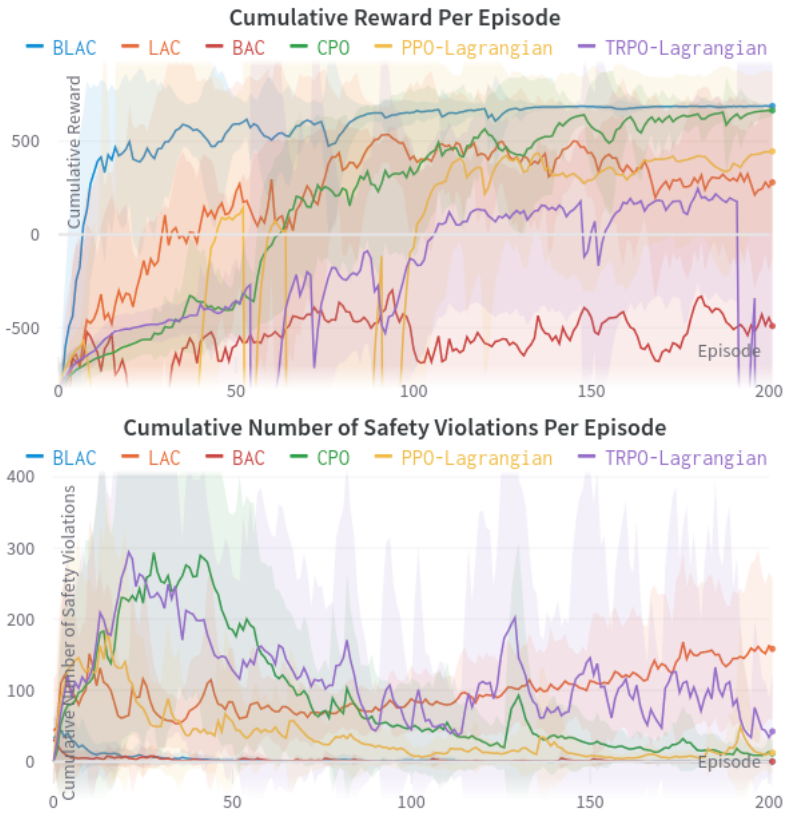


Figure 3.2: The cumulative reward and cumulative number of safety violations of each episode in the unicycle environment are compared for the proposed BLAC (in blue) and other baselines. Each plot shows the mean of ten experiments using different seeds, with the shading representing the standard deviation.

Simulation results are shown in Figure 3.2. Compared to other baselines, our framework enables the system to achieve higher cumulative reward in fewer episodes, indicating that the

framework helps the unicycle approach and successfully reach its destination (equilibrium) after a shorter training process. Additionally, the fluctuations of the cumulative reward are smaller than those of any other baseline algorithm, suggesting that the system performance in obtaining high rewards can quickly recover to its original high level after deterioration. Regarding safety, as evidenced by the cumulative number of safety violations per episode, our framework results in much fewer violations compared to LAC, CPO, PPO-Lagrangian, and TRPO-Lagrangian where CBFs are not used, which means CBFs can greatly help maintain safety, and our framework is thus suitable for safety-critical applications in the real world.

3.4.2 Simulated Car Following

This environment, which involves a chain of five cars following each other on a straight road, is adapted from [36, 80]. The goal is to control the velocity of the 4th car to keep a desired distance from the 3rd car while avoiding collisions with other cars. The model of cars except for the 4th one is:

$$x_{t+1,i} = x_{t,i} + \begin{bmatrix} v_{t,i} \\ 0 \end{bmatrix} \Delta T + \begin{bmatrix} 0 \\ 1 + d_i \end{bmatrix} a_{t,i} \Delta T, \forall i \in \{1, 2, 3, 5\}.$$

Each state of the system is denoted by $x_{t,i} = [p_{t,i}, v_{t,i}]^T$, where $p_{t,i}$ and $v_{t,i}$ are the position and velocity of the ith car at the timestep t, $d_i = 0.1$ is unknown to the nominal model and therefore is the unknown part. ΔT represents the time interval. The 1st car has a velocity $v_{t,1} = v_s - 4\sin(t)$, where $v_s = 3.0$ is the predefined velocity. Its acceleration is given by $a_{t,1} = k_v(v_s - v_{t,1})$ where $k_v = 4.0$ is a constant. Car 2 and 3 have accelerations given by:

$$a_{t,i} = \begin{cases} k_v(v_s - v_{t,i}) - k_b(p_{t,i-1} - p_{t,i}) & \text{if } |p_{t,i-1} - p_{t,i}| < 6.5 \\ k_v(v_s - v_{t,i}) & \text{otherwise,} \end{cases}$$

where $k_b = 20.0$. The 5th car has the following acceleration:

$$a_{t,5} = \begin{cases} k_v(v_s - v_{t,5}) - k_b(p_{t,3} - p_{t,5}) & \text{if } |p_{t,3} - p_{t,5}| < 13.0 \\ k_v(v_s - v_{t,5}) & \text{otherwise.} \end{cases}$$

The model of the 4th car is as follows:

$$x_{t+1,4} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} x_{t,4} + \begin{bmatrix} 1 \\ \frac{1}{\Delta T} \end{bmatrix} u_t \Delta T,$$

where u_t is the velocity of the 4th car, and also the control signal generated by the controller.

The reward signal is defined to minimize the difference between u_t and v_s, and an additional reward of 1.5 is given at the timesteps when $d_t = p_{t,3} - p_{t,4}$, which is the distance between the 3rd and 4th car, is within $[9.0, 10.0]$ defined as the region where d_t is expected to be. The cost signal is $\|d_{t+1} - d_{\text{desired}}\|$, where $d_{\text{desired}} = 9.5$. CBFs are defined as $h_1(x_t) = p_{t,3} - p_{t,4} - \delta$ and $h_2(x_t) = p_{t,4} - p_{t,5} - \delta$, where δ is the minimum required distance between the cars. When a safety constraint is violated if two types of constraints cannot be simultaneously satisfied, the 4th car may be too close to the 5th car to make d_t be within $[9.0, 10.0]$. In this case, the backup controller where u_{nominal} is simply set to be zero to check whether safety can be achieved as quickly as possible and then maintained is activated. The RL-based controller will be reinstated once the 4th car leaves the dangerous area, namely beyond the proximity of the 5th car.

Based on the results presented in Figure 3.3, the proposed framework consistently yields the highest cumulative reward, indicating its superior performance in regulating the distance d_t within the range $[9.0, 10.0]$. Moreover, the much smaller cumulative number of safety violations compared to baselines demonstrates the effectiveness of the proposed framework in helping the system achieve safety with the assistance of CBFs. These results suggest that the proposed framework has promising potential for practical applications.

3.5 Conclusions

In this chapter, we propose the BLAC framework, which combines separate CBF and Lyapunov-based constraints with the actor-critic RL method to help to guarantee both the safety and stability of the controlled system. This framework imposes safety constraints for each step in the trajectory instead of the trajectory expectation which is widely used in previous research. Thus, this framework imposes stricter safety constraints, which is crucial in real-world safety-critical applications. Moreover, our framework contributes to guaranteeing the stability of the system,

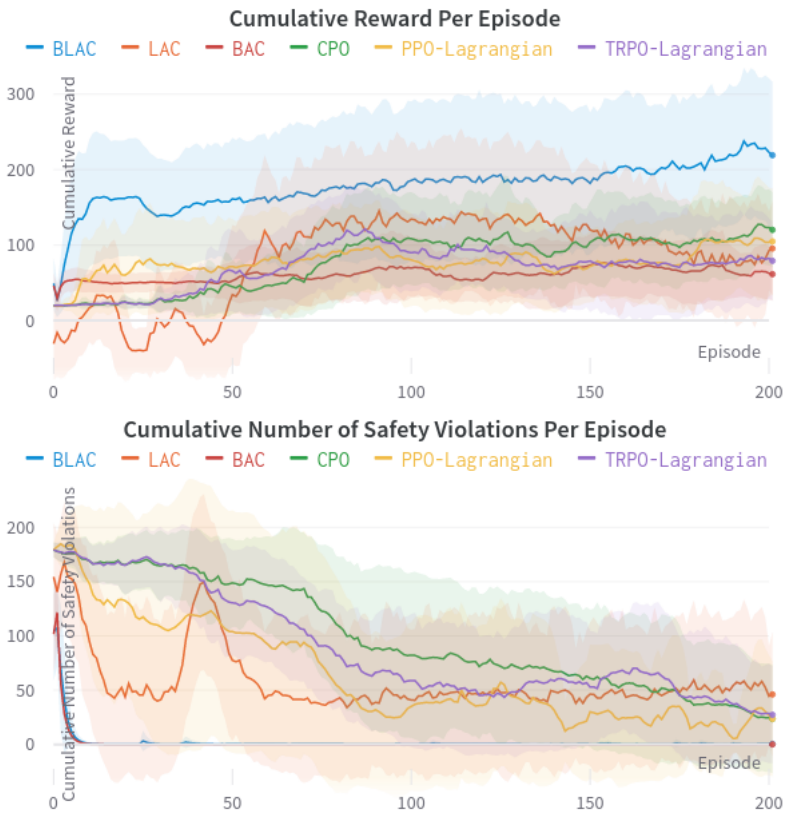


Figure 3.3: Comparisons are made between the cumulative reward and cumulative number of safety violations of each episode for the proposed BLAC (in blue) and other baselines in the simulated car following environment. Each plot shows the mean of ten experiments using different seeds, with the shading representing the standard deviation. Graph comparing the cumulative number of safety violations shows that the curves for BLAC and BAC decrease rapidly to zero and do not exhibit significant fluctuations afterwards.

facilitating the system to approach the desired state (equilibrium) and obtain higher cumulative reward in tasks where high rewards are offered when the system gets closer to or reaches the desired state, such as navigation tasks. With the augmented Lagrangian formulation and backup controller, higher cumulative reward and fewer safety constraint violations are realized in the experiments.

Despite these advantages, the proposed framework has several limitations. First, the current formulation is evaluated only on tasks where the relative degree of the CBFs is one; extending the framework to handle higher relative-degree CBFs remains an important task. Second, the framework assumes the availability of predefined CBFs prior to training, which may be difficult to construct in many sophisticated applications. Third, the method relies on a nominal control-affine system model, which may not always be available in practice. These limitations motivate the developments in the next chapter, where we develop a framework for more general and less restrictive settings.

CHAPTER 4

State-Wise Safe and Stable Reinforcement Learning for Gray-box Settings

4.1 Introduction

In the previous chapter, state-wise safety and stability were enforced for white-box settings, where nominal system dynamics and predefined safety certificates were available. In this chapter, we move beyond this setting and study state-wise safe and stable reinforcement learning in more general settings where such knowledge can no longer be available. Specifically, we introduce learnable barrier certificates and Lyapunov-based constraints that are trained jointly with the policy, helping to achieve safety and stability directly from data while retaining theoretical performance analysis.

It is noteworthy that some previous studies [116, 118] investigate state-wise constraints by adapting common CMDPs to special subclasses of CMDPs. However, these methods are still developed within a CMDP framework, limiting constraints to be formulated solely based on the instantaneous cost of the state-action-transition tuple. To address the limitations of existing methods, this chapter departs from CMDP-based RL approaches by directly incorporating state-wise constraints like discrete-time CBFs with high relative degree, which requires multiple future states, and neural Lyapunov and barrier certificates in non-CMDP settings. Compared to other CBF-based methods that often rely on specific assumptions about the constraints' form, such as linearity for quadratic programming solvers [36, 80], our method accommodates tasks with diverse dynamics and barrier-based constraints without such restrictions. Importantly, the infeasibility issue that arises from applying multiple state-wise constraints simultaneously is

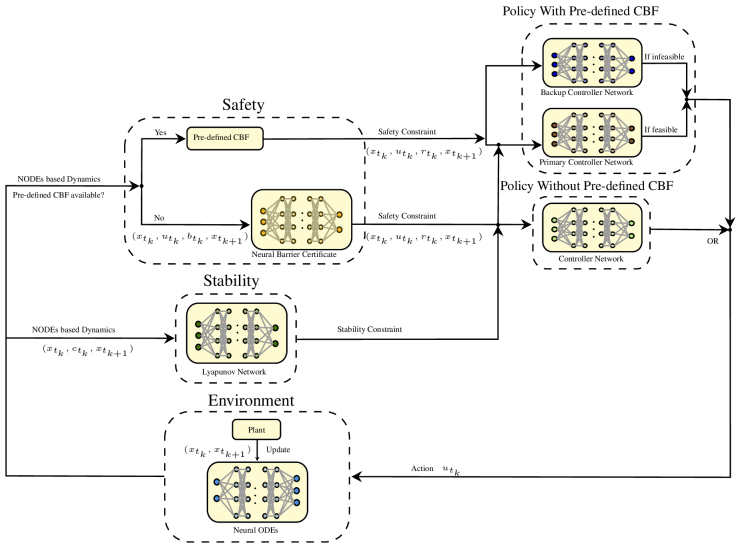


Figure 4.1: Schematic of the proposed NLBAC algorithm, which includes a Lyapunov network for stability, predefined CBFs or neural barrier certificates for safety, and neural ODEs for modeling. The algorithm operates as follows: If predefined CBF is available, both safety and stability constraints are applied during the training of the primary controller network, while the backup controller network is trained with only the safety constraint to handle the issue of infeasibility. If no predefined CBF is available, the safety constraint is imposed by using a neural barrier certificate, and the policy without predefined CBF is trained together with the stability constraint. The output junction denotes that action signals are computed by one of these policies, depending on the availability of predefined CBF.

explicitly investigated in this work. The schematic of the proposed algorithm is shown in Fig. 4.1. Detailed contributions made in this chapter are:

1. With Lyapunov certificates and predefined CBFs, we propose a method to achieve state-wise stable and safe RL in non-CMDP settings by using the augmented Lagrangian method (ALM, rather than relying solely on the augmented formulation of the Lagrangian function (3.19) in Chapter 3). Also, we investigate the infeasible states from which the controlled system's approach towards the desired state, driven by the Lyapunov-based constraint, will cause immediate safety violations due to the task setting, and then design an additional RL-based backup controller to achieve and maintain safety while encouraging exploration at such infeasible states.

2. For tasks without predefined CBFs, we first prove that the cumulative discounted value of a specifically defined signal is a barrier certificate. We then propose a policy update method whose constraint is such a barrier certificate which is easy to train in a principled way without requiring additional expert demonstrations. We further theoretically prove that

the set of states where safety is maintained monotonically expands if the control policy is updated according to the proposed method because of the barrier certificate, and show the monotonic improvement of the control performance in terms of the cumulative discounted reward, which will finally converge to an optimal solution under certain assumptions. Moreover, the infeasibility caused by the addition of the Lyapunov-based constraint is also studied, and we show that the controlled system will learn to avoid those infeasible states as training proceeds.

3. We provide a practical implementation called Neural ordinary differential equations-based Lyapunov-Barrier Actor-Critic (NLBAC) by combining the proposed method with Soft Actor-Critic (SAC) [27] where policy, value, Lyapunov and barrier (if no predefined CBFs are available) functions are approximated by multi-layer neural networks. This algorithm alleviates the problem of large error propagation that is challenging for model-based RL by applying NODEs for modeling and using a shorter prediction horizon compared to those in many previous model-based RL studies. We further evaluate our algorithm in four widely-used tasks. The results indicate that NLBAC achieves satisfactory performance in reducing safety violations and driving the controlled system towards the destination with higher sample efficiency. Ablation studies show better performance of the ALM for non-CMDP constrained RL problems compared to the widely-used dual ascent update.

Compared to the previous Chapter 3, this chapter removes the need for a nominal model, and we make changes to the update of Lagrangian multipliers to avoid extra hyperparameter tuning. The QP-based backup controller in [81] is replaced with an RL-based backup controller when predefined CBFs are used, making the algorithm directly applicable to tasks with nonconvex constraints. Also, different from chapter 3 and our workshop paper [119], an integration with neural barrier certificates is provided here, together with theoretical analysis. We also present comparisons to baselines in more tasks.

4.2 Preliminaries and Problem Statement

4.2.1 Preliminaries

(i) Markov Decision Process

Consider an MDP defined by the tuple $\mathcal{M} = (\mathcal{X}, \mathcal{U}, \mathcal{T}, r, \gamma)$. $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{U} \subset \mathbb{R}^m$ are state and control input spaces which are both bounded, and $x_{t_k} \in \mathcal{X}$ is the state at time t_k, $u_{t_k} \in \mathcal{U}$ is the control signal at time t_k. The reward is denoted as r, and γ is the discount factor. $\mathcal{T}$ denotes the dynamics. For a system with dynamics

$$\dot{x}_t = f(x_t, u_t), \tag{4.1}$$

by fixing the control signal to u_{t_k} between consecutive times t_k and t_{k+1}, we have:

$$x_{t_{k+1}} = x_{t_k} + \int_{t_k}^{t_{k+1}} f(x_\chi, u_{t_k}) d\chi, \tag{4.2}$$

following (4.1).

(ii) Safety

Here we first define the following subspaces [120].

1. $\mathcal{X}_{\text{vio}}$: Set of states where safety violation occurs.
2. $\mathcal{X}_{\text{irrec}}$: Set of states $x \notin \mathcal{X}_{\text{vio}}$, but under any controller, the trajectory starting from $x_{t_0} = x$ will enter $\mathcal{X}_{\text{vio}}$ at least once with a positive probability, i.e., $\forall \pi \in \Pi$, $\exists k > 0$, such that $Pr(x_{t_k} \in \mathcal{X}_{\text{vio}}) > 0$, where Π denotes the set of all policies.
3. $\mathcal{X}_{\text{unsafe}} = \mathcal{X}_{\text{vio}} \cup \mathcal{X}_{\text{irrec}}$ is the unsafe set, and $\mathcal{X}_{\text{safe}} = \mathcal{X} \backslash \mathcal{X}_{\text{unsafe}}$ is safe set.

Remark 4.1. *For any $x_{t_0} = x \in \mathcal{X}_{safe}$, there exists controller π, such that $\forall k \geq 0$, $Pr(x_{t_k} \in \mathcal{X}_{vio}) = 0$, which means any trajectory starting from $x \in \mathcal{X}_{safe}$ will enter $\mathcal{X}_{vio}$ and cause safety violations with probability 0, and then it is said that safety is maintained for state x under such control policy π. Furthermore, a policy that maintains safety for all $x \in \mathcal{X}_{safe}$ is denoted as $\pi^s \in \Pi^s$, where Π^s is the set of controllers under which $\forall k \geq 0$, $Pr(x_{t_k} \in \mathcal{X}_{vio}) = 0$ for any trajectory starting from any $x_{t_0} \in \mathcal{X}_{safe}$.*

With m distinct state-wise constraints, safety is characterized by the predefined functions $h_i, i \in [1, \cdots, m]$ where the condition

$$h_i(x) \geq 0$$

indicates that the i-th safety requirement is satisfied. Consequently, the corresponding safe set is defined as the super-level set

$$\mathcal{C}_{i,0} = \{x_{t_k} \in \mathcal{X} | h_i(x_{t_k}) \geq 0\}, \tag{4.3}$$

which should satisfy that $\bigcap_{i=1}^{m} \mathcal{C}_{i,0} \subset \mathcal{X}_{\text{safe}}$. The system is then safe if $\bigcap_{i=1}^{m} \mathcal{C}_{i,0}$ is forward invariant. For some systems, however, the control input does not directly influence the one-step evolution of $h_i(x)$. In particular, when the safety function has relative degree $r_i > 1$, the control input first appears in the r_i-th difference of the safety function. In such cases, auxiliary barrier functions $\Phi_{i,j}$ are introduced recursively to propagate the safety condition through the system dynamics until the control input explicitly appears. This leads to the following higher-order discrete-time CBF construction [121] since RL generates the control signal discretely:

$$\begin{aligned}
\Phi_{i,0}(x_{t_k}) &:= h_i(x_{t_k}), \\
\Phi_{i,1}(x_{t_k}) &:= \Delta\Phi_{i,0}(x_{t_k}, u_{t_k}) + \kappa_{i,1}(\Phi_{i,0}(x_{t_k})), \\
&\vdots \\
\Phi_{i,r_i}(x_{t_k}) &:= \Delta\Phi_{i,r_i-1}(x_{t_k}, u_{t_k}) + \kappa_{i,r_i}(\Phi_{i,r_i-1}(x_{t_k})),
\end{aligned} \tag{4.4}$$

where r_i is the relative degree [122, 123], $\Delta\Phi_{i,j}(x_{t_k}, u_{t_k}) := \Phi_{i,j}(x_{t_{k+1}}) - \Phi_{i,j}(x_{t_k})$, $j = 0, 1, \ldots, r_i - 1$, and $\kappa_{i,j}(\cdot)$ are class $\mathcal{K}$ functions satisfying $\kappa_{i,j}(x) < x$, where $j = 1, \ldots, r_i$, for the i-th safety constraints.

Definition 4.1 (Discrete-time Control Barrier Function [121]). *For the system (4.2), the predefined function $h_i : \mathbb{R}^n \to \mathbb{R}$ is called a discrete-time CBF of relative degree r_i if there exists $\Phi_{i,j}(x_{t_k})$, $j = 0, 1, \ldots, r_i$ defined by (4.4) and $\mathcal{C}_{i,j}$ which are super-level sets of $\Phi_{i,j}(x_{t_k})$ defined*

similarly to (4.3), $j = 0, 1, \ldots, r_i - 1$ *such that for all* $x_{t_k} \in \bigcap_{j=0}^{r_i-1} \mathcal{C}_{i,j}$,

$$\Phi_{i,r_i}(x_{t_k}) \geq 0. \tag{4.5}$$

Given the existence of a CBF, a controller satisfying (4.5) will make the set $\bigcap_{j=0}^{r-1} \mathcal{C}_{i,j}$ forward invariant. Consequently, safety is maintained if there exists a controller such that $\forall i \in [1, \ldots, m]$, (4.5) holds for all $x_{t_k} \in \bigcap_{i=1}^{m} \bigcap_{j=0}^{r_i-1} \mathcal{C}_{i,j}$.

(iii) Stability

For stabilization tasks, the system state is required to eventually reach a goal state. As in the previous Chapter 3, define an additional cost signal as $c(x_{t_k}, u_{t_k}) = \left\| x_{t_{k+1}} - x_{\text{desired}} \right\|$ where $x_{t_{k+1}}$ is the next state following (4.2), and x_{desired} is the desired state (goal). Furthermore, define the cost function under the controller π as $c_\pi(x_{t_k}) = \mathbb{E}_{u_{t_k} \sim \pi} c(x_{t_k}, u_{t_k}) = \mathbb{E}_{u_{t_k} \sim \pi}[\left\| x_{t_{k+1}} - x_{\text{desired}} \right\|]$. Following Definition 3.2 provided in Section 3.2, define $L^\pi(x_{t_k}) = \mathbb{E}_{\tau \sim \pi}\left[\sum_{i=0}^{\infty} \gamma_c^i c_\pi(x_{t_{k+i}}) \right]$ as the value function of the cost at x_{t_k}, where γ_c is the discount factor, and $\tau = \{x_{t_k}, x_{t_{k+1}}, x_{t_{k+2}}, \ldots\}$ is a trajectory under controller π starting from x_{t_k}. As shown in Chapter 3, the system is stable in mean cost if there exist a positive constant β and a controller π, such that

$$\mathbb{E}_{x_{t_k} \sim \mu_\pi}\left[L^\pi(x_{t_{k+1}}) - L^\pi(x_{t_k}) \right] \leq -\beta \mathbb{E}_{x_{t_k} \sim \mu_\pi}\left[L^\pi(x_{t_k}) \right]. \tag{4.6}$$

Here $\mu_\pi(x)$ is the sampling distribution. Such L^π is also called an exponentially stabilizing Lyapunov certificate.

(iv) Neural Ordinary Differential Equations (NODEs) for System Modeling

In many cases, the precise system dynamics or even a necessary nominal model are unknown. Neural networks are important tools for estimating dynamics, and there has been a rise in considering neural network hidden layers as states. It is possible to expand on this notion by introducing an ODE to model continuous dynamics $\dot{\hat{x}}_{t_k} = \mathcal{F}_\psi\left(t_k, \hat{x}_{t_k}, u_{t_k}\right)$ where $\hat{x}_{t_0} = x_{t_0}$ and $\mathcal{F}_\psi$ is a neural network with parameter ψ. NODEs allow simple incorporation of prior information (such as control-affine property, where $\mathcal{F}_\psi = f_{\psi_1}(t_k, \hat{x}_{t_k}) + g_{\psi_2}(t_k, \hat{x}_{t_k}) u_{t_k}$). By

assuming a constant control signal, i.e., $u_\chi = u_{t_k}, \chi \in [t_k, t_{k+1})$, we have:

$$\hat{x}_{t_{k+1}} = \hat{x}_{t_k} + \int_{t_k}^{t_{k+1}} \mathcal{F}_\psi \left(\chi, \hat{x}_\chi, u_{t_k}\right) d\chi, \tag{4.7}$$

which can be used to estimate (4.2).

4.2.2 Definition of the Safe and Stable Control Problem

Building upon the previous subsection, here we define:

Problem 4.1. *For a system described by (4.2), given a desired goal state $x_{desired} \in \mathcal{X}_{safe}$, a set $\mathcal{X}_0$ which is the set of x_{t_0} denoting initial states such that $\mathcal{X}_0 \subset \mathcal{X}_{safe}$, find a controller π producing sequence $\{u_{t_k}\}_{k\geq 0}$ such that the state sequence $\{x_{t_k}\}_{k\geq 0}$ following (4.2) satisfies:*

1. ***Safety**: $\forall k \geq 0,\ Pr(x_{t_k} \in \mathcal{X}_{vio}) = 0$.*
2. ***Stability:** $\lim_{t_k \to \infty} \mathbb{E}_{x_{t_k} \sim \upsilon}\left[c_\pi(x_{t_k})\right] = 0$.*

Remark 4.2. *In this chapter, we only consider cases where $\mathcal{X}_{safe}$ is a connected set and where there exists at least one control policy that enables the system to reach $x_{desired}$ from x_{t_0} without passing through $\mathcal{X}_{vio}$. We exclude tasks where no such path exists, particularly when x_{t_0} and $x_{desired}$ belong to different parts of $\mathcal{X}_{safe}$ that are completely separated by $\mathcal{X}_{vio}$.*

4.3 Training with Predefined Control Barrier Functions

4.3.1 Controller Update via ALM

When predefined CBFs are available, to achieve state-wise safety and stability (approaching the desired goal state at every timestep), a more stringent state-wise version of CBF constraints (4.5) and the Lyapunov-based constraint (4.6) is proposed:

$$\begin{aligned} &-\Phi_{i,r_i}(x_{t_k}) \leq 0, && \forall x_{t_k} \in \mathcal{X}, \forall i \in [1, \ldots, m], \\ &L^{\pi_p}(x_{t_{k+1}}) - L^{\pi_p}(x_{t_k}) + \beta L^{\pi_p}(x_{t_k}) \leq 0, && \forall x_{t_k} \in \mathcal{X}, \end{aligned} \tag{4.8}$$

where $\{x_{t_{k+1}}, x_{t_{k+2}}, \ldots, x_{t_{k+r_i}}\}$ are future states following a primary controller π_p according to (4.2), and thus (4.8) are constraints that π_p should satisfy. It is noteworthy that the CBF constraints are also imposed when $x_{t_k} \in \mathcal{X}_{\text{unsafe}}$, aiming to obtain a control policy that is able to

drive the agent from $\mathcal{X}_{\text{unsafe}}$ to $\mathcal{X}_{\text{safe}}$ and then maintain safety. Although the dual ascent update has been widely used in previous studies, this method is sensitive to hyperparameters and can lead to lengthy and possibly oscillating training processes with many constraint violations. To quickly find feasible solutions under state-wise constraints, here we first convert inequality constraints (4.8) to equality ones as

$$\begin{aligned} &ReLU\big(-\Phi_{i,r_i}(x_{t_k})\big) = 0, && \forall x_{t_k} \in \mathcal{X}, \forall i \in [1, \dots, m], \\ &ReLU\big(L^{\pi_p}(x_{t_{k+1}}) - L^{\pi_p}(x_{t_k}) + \beta L^{\pi_p}(x_{t_k})\big) = 0, && \forall x_{t_k} \in \mathcal{X}. \end{aligned} \tag{4.9}$$

Therefore, we obtain the constrained optimization problem:

$$\begin{aligned} &\min_{\pi_p} \ \mathbb{E}_{x_{t_k}}[-V^{\pi_p}(x_{t_k})] \\ &s.t. \sum_{x_{t_k} \in \mathcal{X}} ReLU\big(-\Phi_{i,r_i}(x_{t_k})\big) = 0 \quad \forall i \in [1, \dots, m], \\ &\qquad \sum_{x_{t_k} \in \mathcal{X}} ReLU\big(L^{\pi_p}(x_{t_{k+1}}) - L^{\pi_p}(x_{t_k}) + \beta L^{\pi_p}(x_{t_k})\big) = 0, \end{aligned} \tag{4.10}$$

where $V^{\pi_p}(x_{t_k}) = \mathbb{E}_{\tau \sim \pi_p}\big[\sum_{i=0}^{\infty} \gamma^i r(x_{t_{k+i}})\big]$ is the value function under the primary policy π_p. The commonly used dual ascent update relies on the application of a Lagrangian function, however, here, similar to Chapter 3, we enhance the Lagrangian function by incorporating an additional squared term, resulting in an augmented Lagrangian function. This squared term helps accelerate convergence and reduce sensitivity to hyperparameters, mitigating issues such as oscillation during training. Based on this augmented Lagrangian function, the augmented Lagrangian method can be effectively utilized to handle complex constraints, leading to improved performance in constrained RL tasks. To elaborate, denote the Lagrangian multipliers for CBF and the Lyapunov-based constraints in Problem (4.10) as λ_i and ζ, respectively. Additionally, let $c > 0$ represent a penalty parameter that increases monotonically with the number of updates and approaches infinity as the updates progress. Using these parameters, the augmented

Lagrangian function is formulated as follows:

$$\mathcal{L}_c(\pi_p, \lambda_i, \zeta) = \mathbb{E}_{x_{t_k}}\left[-V^{\pi_p}(x_{t_k})\right] + \sum_{i=1}^{m} \lambda_i \left(\sum_{x_{t_k} \in \mathcal{X}} ReLU\big(- \Phi_{i,r_i}(x_{t_k})\big) \right)$$
$$+ \sum_{i=1}^{m} \frac{c}{2} \left(\sum_{x_{t_k} \in \mathcal{X}} ReLU\big(- \Phi_{i,r_i}(x_{t_k})\big) \right)^2 + \zeta \left(\sum_{x_{t_k} \in \mathcal{X}} ReLU\big(L^{\pi_p}(x_{t_{k+1}}) - L^{\pi_p}(x_{t_k}) + \beta L^{\pi_p}(x_{t_k})\big) \right)$$
$$+ \frac{c}{2} \left(\sum_{x_{t_k} \in \mathcal{X}} ReLU\big(L^{\pi_p}(x_{t_{k+1}}) - L^{\pi_p}(x_{t_k}) + \beta L^{\pi_p}(x_{t_k})\big) \right)^2 . \tag{4.11}$$

In this equation, the first term on the right-hand side corresponds to the objective function of Problem (4.10). The second term enforces the safety constraints based on CBFs, as described in Problem (4.10), with the third term being its corresponding squared penalty term with the penalty parameter c. Also, m represents the number of CBF constraints. Similarly, the fourth term imposes the stability constraints derived from the Lyapunov-based constraint, and the fifth term is its corresponding squared penalty term. With function (4.11), the augmented Lagrangian method can be applied to solve the state-wise constrained RL problem in non-CMDP settings, as shown in Algorithm 4.1. Furthermore, considering practical needs, such as approximating necessary functions and reducing the computational burden, a practical implementation with more detailed descriptions is provided in Section 4.5.

Algorithm 4.1 State-Wise Stable and Safe RL via ALM

Input: Initialized policy π_p^0, Lagrangian multipliers λ_i^0 and ζ^0, penalty parameter c^0, factor $\rho_c \in (1, \infty)$, maximum number of iterations J.

1: **for** $j = 0, 1, \ldots, J$ **do**
2: $\quad \pi_p^j \leftarrow \arg\min_{\pi_p} \mathcal{L}_{c^j}(\pi_p, \lambda_i^j, \zeta^j)$
3: $\quad \lambda_i^{j+1} \leftarrow \lambda_i^j + c^j \left(\sum_{x_{t_k} \in \mathcal{X}} ReLU\big(- \Phi_{i,r_i}(x_{t_k})\big) \right)$
4: $\quad \zeta^{j+1} \leftarrow \zeta^j + c^j \left(\sum_{x_{t_k} \in \mathcal{X}} ReLU\big(L^{\pi_p^j}(x_{t_{k+1}}) - L^{\pi_p^j}(x_{t_k}) + \beta L^{\pi_p^j}(x_{t_k})\big) \right)$
5: $\quad c^{j+1} \leftarrow \rho_c c^j$
6: **end for**

Output: π^J

Remark 4.3. *Different from [116], which solves problems in a special subclass of CMDPs, the method proposed here can directly handle state-wise constraints in non-CMDP settings via ALM, which allows for constraints that are not solely based on the instantaneous cost of the state-action-*

transition tuple. Also, by updating the Lagrangian multipliers following Lines 3 and 4, this method eliminates the need to tune a learning rate for updating the Lagrangian multipliers, as required in the previous chapter.

4.3.2 Backup Controller Design with Predefined Control Barrier Functions

When state-wise Lyapunov-based and CBF constraints are considered simultaneously, there may exist some states where the controlled system's approach towards the goal x_{desired}, driven by the Lyapunov-based constraint, will cause immediate safety violations. Therefore, the following definition is given:

Definition 4.2. *$\mathcal{X}_{infeasible}$ is the set of states where there exists no controller such that the agent can further approach towards the goal $x_{desired}$ according to the Lyapunov-based constraint without causing immediate safety violations.*

At $x_{t_k} \in \mathcal{X}_{\text{infeasible}}$, once the controller satisfies the Lyapunov-based constraint and drives the controlled agent to approach towards x_{desired}, safety violations will happen, which means the predefined CBF constraint is violated. At such states, priority should be given to safety constraints. Different from Chapter 3 where a QP-based or other manually specified controller is used, here we use an additional RL-based backup controller. This approach can handle nonconvex constraints, and for a primary RL-based controller π_p trained with CBF and Lyapunov-based constraints, the corresponding RL-based backup controller π_b can be designed to be the solution of the following constrained optimization problem:

$$\begin{aligned} &\min_{\pi_b} \ \mathbb{E}_{x_{t_k}, u_{t_k} \sim \pi_b}[-V^{\pi_p}(x_{t_{k+1}})] \\ &s.t. \ \sum_{x_{t_k} \in \mathcal{X}} ReLU\big(-\Phi^b_{i,r_i}(x_{t_k})\big) = 0 \ \forall i \in [1, \dots, m]. \end{aligned} \tag{4.12}$$

The definition of Φ^b_{i,r_i} is the same as that of Φ_{i,r_i}, except that the states $\{x_{t_{k+1}}, x_{t_{k+2}}, \dots, x_{t_{k+r_i}}\}$ used in the objective function and constraints are future states following the backup controller π_b. The objective function in Problem (4.12) reflects the intention of only using the backup controller π_b for one step and then switching back to the primary controller π_p while maximizing

the cumulative discounted reward. The augmented Lagrangian function for updating π_b is

$$\begin{aligned}\mathcal{L}_{c_b}(\pi_b, \lambda_{b,i}) = \mathbb{E}_{x_{t_k}, u_{t_k} \sim \pi_b}\left[-V^{\pi_p}(x_{t_{k+1}})\right] + \sum_{i=1}^{m} \lambda_{b,i} \left(\sum_{x_{t_k} \in \mathcal{X}} ReLU\big(-\Phi^b_{i,r_i}(x_{t_k})\big) \right) \\ + \sum_{i=1}^{m} \frac{c_b}{2} \left(\sum_{x_{t_k} \in \mathcal{X}} ReLU\big(-\Phi^b_{i,r_i}(x_{t_k})\big) \right)^2,\end{aligned} \tag{4.13}$$

where $\lambda_{b,i}$ and c_b are Lagrangian multipliers for CBF constraints and the penalty parameter to update π_b.

When $x_{t_k} \in \mathcal{X}_{\text{infeasible}}$, the control signals generated may impede the system's swift approaching towards x_{desired} when CBF constraint is satisfied instead. For instance, the agent is forbidden to cross an obstacle by the CBF constraints while the Lyapunov-based constraint requires it to do so to reach x_{desired}. The agent therefore gets stuck close to the obstacle (see Fig. 4.2), which is easy to detect. Based on the aforementioned analysis, the policy π_b is activated when the agent is detected to be trapped for some time. The primary controller is reinstated when the agent moves a certain distance away from the trapped position, or the time threshold for using the backup controller is exceeded. These conditions can be encoded in the implementation easily. Moreover, when integrated with SAC, an additional advantage of using the proposed RL-based backup controller is that SAC can encourage exploration by simply tuning a hyperparameter that sets the required lower bound of entropy, denoted as $\mathcal{H}_b$. The exploration and exploitation inherent in SAC with CBFs as constraints allow the system to escape trapped positions through diverse actions, without significantly reducing the rewards obtained. The detailed implementation is provided in Section 4.5.

4.4 Training with Learned Barrier Certificate

It is common for useful predefined CBFs to be unavailable before training. In this section, we present a convenient way to extend the proposed method to tasks without predefined CBFs by using neural barrier certificates, and provide corresponding theoretical analysis.

4.4.1 Neural Barrier Certificates

A previous study [124] showed that the value function of a fixed policy that tries to maintain safety greedily can be modified to be a CBF to separate $\mathcal{X}_{\text{safe}}$ and $\mathcal{X}_{\text{unsafe}}$. However, here we need to update the policy jointly with a learned barrier certificate to satisfy both state-wise stability and safety constraints in an online manner. Thus, firstly, we provide the following definition:

Definition 4.3. *Three sets dependent on the control policy π are defined:*

1. $\mathcal{X}^{\pi}_{irrec}$*: Set of states $x \notin \mathcal{X}_{vio}$, but under the controller π, the trajectory starting from $x_{t_0} = x$ will enter $\mathcal{X}_{vio}$ at least once with a positive probability, i.e., $\exists k \geq 0$, such that $Pr(x_{t_k} \in \mathcal{X}_{vio}) > 0$ under the controller π.*

2. $\mathcal{X}^{\pi}_{unsafe} = \mathcal{X}_{vio} \cup \mathcal{X}^{\pi}_{irrec}$ *is the unsafe set under π, and $\mathcal{X}^{\pi}_{safe} = \mathcal{X} \backslash \mathcal{X}^{\pi}_{unsafe}$ is the safe set under π, i.e., for every $x_{t_0} = x \in \mathcal{X}^{\pi}_{safe}$, under the controller π, for all $k \geq 0$, $Pr(x_{t_k} \in \mathcal{X}_{vio}) = 0$, namely the safety is maintained under π.*

Later, $\mathcal{X}^{\pi}_{\text{safe}}$ will be considered instead since π is changing during learning. For further analysis, the following assumptions are provided:

Assumption 4.1. *There exists an upper bound N such that all trajectories starting from $x_{t_0} \in \mathcal{X}^{\pi}_{irrec}$ will enter the set $\mathcal{X}_{vio}$ at least once within N steps under the controller π.*

Assumption 4.1 requires that a safety violation will happen reasonably soon after entering $\mathcal{X}^{\pi}_{\text{irrec}}$. Further, the definition of a barrier certificate is listed below:

Definition 4.4. *H^{π} is a barrier certificate of the policy π if following properties hold:*

1. $\forall x_{t_k} \in \mathcal{X}^{\pi}_{unsafe}$, $H^{\pi}(x_{t_k}) < 0$;

2. $\forall x_{t_k} \in \mathcal{X}^{\pi}_{safe}$, $H^{\pi}(x_{t_k}) \geq 0$;

3. $\forall x_{t_k} \in \mathcal{X}^{\pi}_{safe}$, $H^{\pi}(x_{t_{k+1}}) - H^{\pi}(x_{t_k}) \geq -\epsilon H^{\pi}(x_{t_k})$ *where $\epsilon \in (0, 1)$ and $x_{t_{k+1}}$ is the next state following the controller π.*

Then, based on Assumption 4.1 and Definition 4.4, we provide the following lemma:

Lemma 4.1. *Introduce an additional barrier signal as*

$$b(x_{t_k}) = \begin{cases} d & \text{if } x \notin \mathcal{X}_{vio}, \\ D & \text{if } x \in \mathcal{X}_{vio}, \end{cases} \tag{4.14}$$

where $d = 0$ *and* $D < 0$ *are constants. Under Assumption 4.1, the function* $H^\pi(x_{t_k}) = \mathbb{E}_{\tau\sim\pi}\big[\sum_{i=0}^{\infty}\gamma^i b(x_{t_{k+i}})\big]$*, where* $\gamma \in (0,1)$ *is the discount factor and* $\tau = \{x_{t_k}, x_{t_{k+1}}, x_{t_{k+2}}, \dots\}$ *is a trajectory under controller* π *starting from* x_{t_k}*, is a barrier certificate of policy* π*.*

Proof. When $x_{t_k} \in \mathcal{X}^\pi_{\text{safe}}$, by Definition 4.3, under the controller π, for all $i \geq 0$, $Pr(x_{t_{k+i}} \in \mathcal{X}_{\text{vio}}) = 0$. Therefore, $H^\pi(x_{t_k}) = H^\pi(x_{t_{k+1}}) = \mathbb{E}_{\tau\sim\pi}\big[\sum_{i=0}^{\infty}\gamma^i d\big] = 0$ given $d = 0$, and hence the second property in Definition 4.4 is satisfied. When $x_{t_k} \in \mathcal{X}^\pi_{\text{unsafe}}$, according to Definition 4.3, there exists $n \geq 0$ such that $Pr(x_{t_{k+n}} \in \mathcal{X}_{\text{vio}}) > 0$ under the controller π, and therefore $\mathbb{E}_{x_{t_{k+n}}}\big[b(x_{t_{k+n}})\big] < 0$. Furthermore, under Assumption 4.1, we have $n \leq N$. Thus, we have:

$$H^\pi(x_{t_k}) = \mathbb{E}_{\tau\sim\pi}\big[\sum_{i=0}^{\infty}\gamma^i b(x_{t_{k+i}})\big] \leq \gamma^n \mathbb{E}_{x_{t_{k+n}}}\big[b(x_{t_{k+n}})\big] \leq \gamma^N \mathbb{E}_{x_{t_{k+n}}}\big[b(x_{t_{k+n}})\big] < 0,$$

since $\gamma \in (0,1)$. Thus, the first property in Definition 4.4 is satisfied. Also, when $x_{t_k} \in \mathcal{X}^\pi_{\text{safe}}$, by Definition 4.3, $x_{t_{k+1}} \in \mathcal{X}^\pi_{\text{safe}}$, and thus $H^\pi(x_{t_{k+1}}) - H^\pi(x_{t_k}) = 0 - 0 = -\epsilon H^\pi(x_{t_k})$. Therefore, the third property in Definition 4.4 is satisfied. In sum, all three properties for being a barrier certificate are satisfied, and thus, the function $H^\pi(x_{t_k})$ is a barrier certificate of policy π. □

Remark 4.4. *It is noteworthy that due to the setting* $d = 0$*, which indicates* $\forall x_{t_k} \in \mathcal{X}^\pi_{safe}$*,* $H^\pi(x_{t_k}) = 0$*, the range of* ϵ *in the third condition of Definition 4.4 can also be changed to* $\epsilon \geq 1$*.*

Common methods in previous studies using barrier functions require the forward invariance of a predefined safe set, which can be inaccurate and conservative in practice, or rely on an extra safe policy and system dynamics to train a barrier certificate by imposing additional conditions on derivatives. Different from these prior works, this method uses a barrier certificate that, by design, has a more negative value when more severe safety violations occur, and 0 when safety is maintained at every subsequent timestep under the current policy π. Also, this certificate can be directly trained, as a value function, following standard RL algorithms in a model-free,

convenient, and principled way.

4.4.2 Theoretical Analysis

A lemma showing the convergence of the barrier certificate of a fixed policy π is presented:

Lemma 4.2. *For a fixed control policy π, consider a mapping $H^0 : \mathcal{X} \to \mathcal{R}$ and the Bellman operator:*

$$\mathcal{B}^\pi[H(x_{t_k})] = b(x_{t_k}) + \gamma \sum\nolimits_{u_{t_k}} \pi(u_{t_k}|x_{t_k}) H(x_{t_{k+1}}), \tag{4.15}$$

where $x_{t_{k+1}}$ is the next state following x_{t_k} and u_{t_k} according to the deterministic system dynamics, by defining q as the number of updates and $H^{q+1} = \mathcal{B}^\pi[H^q]$, the sequence H^q will converge to the barrier certificate H^π when $q \to \infty$.

Proof. For any two mappings H and H', define the maximum norm as: $||H - H'||_\infty = \max_{x_{t_k} \in \mathcal{X}} |H(x_{t_k}) - H'(x_{t_k})|$. Then,

$$\begin{aligned}
||\mathcal{B}^\pi[H] - \mathcal{B}^\pi[H']||_\infty &= \max_{x_{t_k} \in \mathcal{X}} |\mathcal{B}^\pi[H(x_{t_k})] - \mathcal{B}^\pi[H'(x_{t_k})]| \\
&= \max_{x_{t_k} \in \mathcal{X}} |\gamma \sum\nolimits_{u_{t_k}} \pi(u_{t_k}|x_{t_k})[H(x_{t_{k+1}}) - H'(x_{t_{k+1}})]| \\
&\leq \gamma \max_{x_{t_k} \in \mathcal{X}} \sum\nolimits_{u_{t_k}} \pi(u_{t_k}|x_{t_k}) ||H - H'||_\infty.
\end{aligned}$$

Therefore, $||\mathcal{B}^\pi[H] - \mathcal{B}^\pi[H']||_\infty \leq \gamma ||H - H'||_\infty$. Since $\gamma \in (0, 1)$, the Bellman operator $\mathcal{B}^\pi$ is a γ contraction mapping. According to [125, Proposition A.26], it can be concluded that $H^q(x_{t_k})$ will converge to $H^\pi(x_{t_k})$ for $x_{t_k} \in \mathcal{X}$ when $q \to \infty$. □

Denote π_n to be the control policy obtained at n^{th} time, then we need to solve the following constrained optimization problem, for all $x_{t_k} \in \mathcal{X}$, to obtain the policy π_{n+1}:

$$\pi_{n+1} = \arg\min_{\pi \in \Pi} \ -\mathbb{E}_{u_{t_k} \sim \pi} V^{\pi_n}(x_{t_{k+1}}) \tag{4.16a}$$

$$s.t. - H^{\pi_n}(x_{t_{k+1}}) + H^{\pi_n}(x_{t_k}) - \epsilon H^{\pi_n}(x_{t_k}) \leq 0 \ \ \forall x_{t_k} \in \mathcal{X}, \tag{4.16b}$$

where $x_{t_{k+1}}$ is the next state following x_{t_k} and u_{t_k} according to the system dynamics. Different

from previous studies [126] applying region-wise policy improvement that firstly separate the whole state space $\mathcal{X}$ into different regions according to certain functions, e.g., constraint decay function (CDF), and then solve different optimization problems on different regions with the safety being enhanced by minimizing safety-related objective functions, in our proposed method, the safety is required and achieved by applying the barrier certificate as a constraint of a constrained optimization problem defined and should be solved for each $x_{t_k} \in \mathcal{X}$. This proposed method avoids dividing $\mathcal{X}$ into different regions based on values output by a certain neural network trained during the learning process to approximate functions like the CDF, thereby circumventing additional errors, such as misclassifications of states, that occur when the neural network's approximation is inaccurate.

Then, we prove that by solving Problem (4.16) iteratively, under certain assumptions, the set $\mathcal{X}_{\text{safe}}^{\pi_n}$ will converge to $\mathcal{X}_{\text{safe}}$, and V^{π_n} will converge to an optimal solution on $\mathcal{X}_{\text{safe}}$. The additional assumption required is given as follows:

Assumption 4.2. *For any state $x_{t_k} \in \mathcal{X}$ where there exists at least one policy that satisfies the constraint (4.16b), the solution π_{n+1} obtained by solving Problem (4.16) via constrained optimization methods will satisfy the constraint (4.16b) at x_{t_k}. For states x_{t_k} where no policy that satisfies the constraint (4.16b) exists, the policy π_{n+1} satisfies $\pi_{n+1}(x_{t_k}) = \arg\max_{\pi \in \Pi} H^{\pi_n}(x_{t_{k+1}})$ where $x_{t_{k+1}}$ is the next state following π.*

Assumption 4.2 means that once at least one control policy exists to satisfy the constraint (4.16b) at x_{t_k}, such policy (policies) will be searched for, and the policy π_{n+1} returned by solving Problem (4.16) will be one of them. $\forall x_{t_k} \in \mathcal{X}_{\text{safe}}^{\pi_n}$, π_n satisfies (4.16b) and at least π_{n+1} can be set equal to π_n. For $\mathcal{X}_{\text{unsafe}}^{\pi_n}$, there may exist some states $x_{t_k} \in \mathcal{X}_{\text{unsafe}}^{\pi_n}$ such that no policy satisfying (4.16b) exists. At such states, π_{n+1} is the policy under which the next state $x_{t_{k+1}}$ has the maximum value $H^{\pi_n}(x_{t_{k+1}})$. Then, we provide the following theorem:

Theorem 4.1. *Suppose that initial policy π_0 satisfies $\mathcal{X}_{safe}^{\pi_0} \neq \emptyset$, and denote the control policy obtained by iteratively solving Problem 4.16 as $\{\pi_n\}_{n>0}$. Under Assumption 4.2, when $\epsilon \geq 1$, the safe set under the obtained policy monotonically expands, i.e., $\mathcal{X}_{safe}^{\pi_n} \subseteq \mathcal{X}_{safe}^{\pi_{n+1}}$, and will converge to $\mathcal{X}_{safe}$, i.e., $\lim_{n\to\infty} \mathcal{X}_{safe}^{\pi_n} = \mathcal{X}_{safe}$. Additionally, for the objective function V^{π_n}, $V^{\pi_n}(x_{t_k}) \leq V^{\pi_{n+1}}(x_{t_k})$ holds for all $x_{t_k} \in \mathcal{X}_{safe}^{\pi_n}$.*

Proof. Firstly we show that the set $\mathcal{X}_{\text{safe}}^{\pi_n}$ monotonically expands.

For all $x_{t_k} \in \mathcal{X}_{\text{safe}}^{\pi_n}$, the condition in Problem (4.16) requires $H^{\pi_n}(x_{t_{k+1}}) \geq (1-\epsilon)H^{\pi_n}(x_{t_k}) = 0$ for $x_{t_{k+1}}$ following the newly obtained controller π_{n+1}. Therefore, $x_{t_{k+1}} \in \mathcal{X}_{\text{safe}}^{\pi_n}$, and similarly, $\forall x_{t_0} \in \mathcal{X}_{\text{safe}}^{\pi_n}$, the trajectory $\{x_{t_0}, x_{t_1}, x_{t_2}, \dots\}$ under the policy π_{n+1} remains in $\mathcal{X}_{\text{safe}}^{\pi_n}$ without entering $\mathcal{X}_{\text{vio}}$ with a positive probability. Thus, $x_{t_0} \in \mathcal{X}_{\text{safe}}^{\pi_{n+1}}$ since $H^{\pi_{n+1}}(x_{t_0}) = \mathbb{E}_{\tau\sim\pi_{n+1}}\big[\sum_{i=0}^{\infty}\gamma^i b(x_{t_{k+i}})\big] = \mathbb{E}_{\tau\sim\pi_{n+1}}\big[\sum_{i=0}^{\infty}\gamma^i d\big] = 0$.

When $\epsilon \geq 1$, if $\exists x_{t_k} \in \mathcal{X}_{\text{irrec}}^{\pi_n}$, such that under the new policy π_{n+1}, $H^{\pi_n}(x_{t_{k+1}}) = \max_{\pi\in\Pi} H^{\pi_n}(x_{t_{k+1}}^{\pi}) = 0$, where $x_{t_{k+1}}$ follows the policy π_{n+1} and $x_{t_{k+1}}^{\pi}$ follows any other policy π, then, according to the previous analysis, the following states $\{x_{t_{k+2}}, x_{t_{k+3}}, x_{t_{k+4}}, \dots\}$ will stay within $\mathcal{X}_{\text{safe}}^{\pi_n}$ under π_{n+1}, and therefore $x_{t_k} \in \mathcal{X}_{\text{safe}}^{\pi_{n+1}}$ since $H^{\pi_{n+1}}(x_{t_k}) = 0$ given $b(x_{t_k}) = 0$.

Based on the aforementioned analysis, given $\mathcal{X}_{\text{safe}}^{\pi_0} \neq \emptyset$, it can be concluded that the set $\mathcal{X}_{\text{safe}}^{\pi_n}$ monotonically expands, i.e., $\mathcal{X}_{\text{safe}}^{\pi_n} \subseteq \mathcal{X}_{\text{safe}}^{\pi_{n+1}}$.

Secondly, we show $\lim_{n\to\infty} \mathcal{X}_{\text{safe}}^{\pi_n} = \mathcal{X}_{\text{safe}}$[1]. Since $\{\mathcal{X}_{\text{safe}}^{\pi_n}\}_{n\geq 0}$ is an increasing sequence of sets, i.e., $\mathcal{X}_{\text{safe}}^{\pi_n} \subseteq \mathcal{X}_{\text{safe}}^{\pi_{n+1}}$ for each n, then $\lim_{n\to\infty} \mathcal{X}_{\text{safe}}^{\pi_n} = \cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n}$. Because $\forall x_{t_k} \in \cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n}$, there exists a number $Z \in \mathbb{Z}^+$ specific to x_{t_k} such that $\forall i \geq Z$, $x_{t_k} \in \mathcal{X}_{\text{safe}}^{\pi_i}$. Hence, there exist control policies maintaining safety for x_{t_k}, which indicates $x_{t_k} \in \mathcal{X}_{\text{safe}}$, and therefore $\cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n} \subseteq \mathcal{X}_{\text{safe}}$. Also, $\forall x_{t_k} \notin \cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n}$, assume there exists at least a control policy under which safety can be maintained for x_{t_k}, according to Assumption 4.2 which implies that the search is performed on the whole policy space, there exists a positive number $M \in \mathbb{Z}^+$ such that under the new policy π_{M+1}, $H^{\pi_M}(x_{t_{k+1}}) = \max_{\pi\in\Pi} H^{\pi_M}(x_{t_{k+1}}^{\pi}) = 0$, where $x_{t_{k+1}}$ follows the policy π_{M+1} and $x_{t_{k+1}}^{\pi}$ follows any other policy π, then, according to the previous analysis, $H^{\pi_{M+1}}(x_{t_k}) = 0$ and $x_{t_k} \in \mathcal{X}_{\text{safe}}^{\pi_{M+1}}$, which contradicts with the condition that $x_{t_k} \notin \cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n}$. Hence, there exists no control policy π to achieve $H^{\pi}(x_{t_k}) = 0$, which indicates either $x_{t_k} \in \mathcal{X}_{\text{vio}}$, or $\forall\pi$, the trajectory starting from x_{t_k} will enter $\mathcal{X}_{\text{vio}}$ with a positive probability. Thus, by definition, $x_{t_k} \in \mathcal{X}_{\text{vio}} \cup \mathcal{X}_{\text{irrec}} = \mathcal{X}_{\text{unsafe}}$. Since $\mathcal{X}_{\text{safe}} = \mathcal{X}\backslash\mathcal{X}_{\text{unsafe}}$, we have $x_{t_k} \notin \mathcal{X}_{\text{safe}}$. Because $\forall x_{t_k} \notin \cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n}$, we have $x_{t_k} \notin \mathcal{X}_{\text{safe}}$, equivalently we have $\forall x_{t_k} \in \mathcal{X}_{\text{safe}}$, $x_{t_k} \in \cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n}$, namely $\mathcal{X}_{\text{safe}} \subseteq \cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n}$. Given $\cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n} \subseteq \mathcal{X}_{\text{safe}}$ and $\mathcal{X}_{\text{safe}} \subseteq \cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n}$, we have $\mathcal{X}_{\text{safe}} = \cup_{n=1}^{\infty}\mathcal{X}_{\text{safe}}^{\pi_n}$,

[1]Here, we consider cases where each update results in a nonzero-measure expansion of the identified safe set if new safe states are found.

and thus $\lim_{n\to\infty} \mathcal{X}_{\text{safe}}^{\pi_n} = \mathcal{X}_{\text{safe}}$.

Furthermore, for any policy π_n, $\forall x_{t_k} \in \mathcal{X}_{\text{safe}}^{\pi_n}$, according to the objective function in Problem 4.16, we have:

$$\mathbb{E}_{u_{t_k}\sim\pi_{n+1}} V^{\pi_n}(x_{t_{k+1}}) \geq \mathbb{E}_{u_{t_k}\sim\pi_n} V^{\pi_n}(x'_{t_{k+1}}),$$

where $x_{t_{k+1}}$ is the next state following π_{n+1} and $x'_{t_{k+1}}$ is the next state following π_n. Such π_{n+1} must exist because at least π_{n+1} can be chosen to be equal to π_n. Therefore,

$$\begin{aligned}
V^{\pi_n}(x_{t_k}) &= r(x_{t_k}) + \gamma \mathbb{E}_{u_{t_k}\sim\pi_n} V^{\pi_n}(x'_{t_{k+1}}) \\
&\leq r(x_{t_k}) + \gamma \mathbb{E}_{u_{t_k}\sim\pi_{n+1}} V^{\pi_n}(x_{t_{k+1}}) \\
&\leq r(x_{t_k}) + \mathbb{E}_{u_{t_k}\sim\pi_{n+1}}[\gamma r(x_{t_{k+1}}) + \gamma^2 \mathbb{E}_{u_{t_{k+1}}\sim\pi_{n+1}} V^{\pi_n}(x_{t_{k+2}})] \\
&\leq r(x_{t_k}) + \mathbb{E}_{\pi_{n+1}}[\gamma r(x_{t_{k+1}}) + \gamma^2 r(x_{t_{k+2}}) + \dots] \\
&= V^{\pi_{n+1}}(x_{t_k}).
\end{aligned}$$

Thus, $\forall x_{t_k} \in \mathcal{X}_{\text{safe}}^{\pi_n}$, $V^{\pi_n}(x_{t_k}) \leq V^{\pi_{n+1}}(x_{t_k})$. □

Based on Theorem 4.1, a proposition can be given as follows:

Proposition 4.1. *Under Assumption 3.1, if there exists $Z \in \mathbb{Z}^+$ such that $\forall i \geq Z$, $\mathcal{X}_{safe}^{\pi_i} = \mathcal{X}_{safe}^{\pi_Z}$, then, the objective function V^{π_n} will converge, i.e., $\lim_{n\to\infty} V_{safe}^{\pi_n} = V^*$, where $V^*(x_{t_k}) \geq V^{\pi}(x_{t_k})$, $\forall \pi \in \Pi^s$ and $\forall x_{t_k} \in \mathcal{X}_{safe}$.*

Proof. According to Theorem 4.1, $\forall i \geq Z$, $\mathcal{X}_{\text{safe}}^{\pi_i} = \lim_{n\to\infty} \mathcal{X}_{\text{safe}}^{\pi_n} = \mathcal{X}_{\text{safe}}$, and thus when $n \to \infty$, V^{π_n} is optimized on the set $\mathcal{X}_{\text{safe}}$. By Theorem 4.1, $V^{\pi_n}(x_{t_k}) \leq V^{\pi_{n+1}}(x_{t_k})$ for all $x_{t_k} \in \mathcal{X}_{\text{safe}}$. Given the reward signal is upper bounded under Assumption 3.1, i.e., $r(x_{t_k}) \leq r_{max}$ for all $x_{t_k} \in \mathcal{X}$, $V^{\pi_n}(x_{t_k}) \leq \frac{r_{max}}{1-\gamma}$ is also upper bounded, and thus V^{π_n} will converge to a V^* with the corresponding policy π^*. At convergence, it follows that for all $\pi \in \Pi^s$ and $x_{t_k} \in \mathcal{X}_{\text{safe}}$,

$$\mathbb{E}_{u_{t_k}\sim\pi^*} V^*(x^*_{t_{k+1}}) \geq \mathbb{E}_{u_{t_k}\sim\pi} V^*(x^{\pi}_{t_{k+1}}),$$

where $x^*_{t_{k+1}}$ is the next state following π^* and $x^\pi_{t_{k+1}}$ is the next state following π. Hence,

$$\begin{aligned} V^*(x_{t_k}) &= r(x_{t_k}) + \gamma \mathbb{E}_{u_{t_k}\sim\pi^*} V^*(x^*_{t_{k+1}}) \\ &\geq r(x_{t_k}) + \gamma \mathbb{E}_{u_{t_k}\sim\pi} V^*(x^\pi_{t_{k+1}}) \\ &\geq r(x_{t_k}) + \mathbb{E}_{u_{t_k}\sim\pi}[\gamma r(x^\pi_{t_{k+1}}) + \gamma^2 \mathbb{E}_{u_{t_{k+1}}\sim\pi} V^*(x^\pi_{t_{k+2}})] \\ &\geq r(x_{t_k}) + \mathbb{E}_{\pi}[\gamma r(x^\pi_{t_{k+1}}) + \gamma^2 r(x^\pi_{t_{k+2}}) + \dots] \\ &= V^\pi(x_{t_k}), \end{aligned}$$

for all $x_{t_k} \in \mathcal{X}_{\text{safe}}$, which means the converged V^* has a larger or equal value compared to the value function of any other policy $\pi \in \Pi^s$, i.e., $V^*(x_{t_k}) \geq V^\pi(x_{t_k})$, $\forall \pi \in \Pi^s$ and $\forall x_{t_k} \in \mathcal{X}_{\text{safe}}$. □

Remark 4.5. *In practical implementation, it is usually difficult to find the optimizer of Problem (4.16) on the whole policy set at each iteration, and therefore gradient-based methods are commonly applied, which means π_{n+1} is usually found in a neighborhood of π_n. Thus, a locally optimal solution, whose safe set is a subset of $\mathcal{X}_{safe}$ and value function is only locally optimal, is usually obtained in real implementation which is described in Section 4.5.*

4.4.3 Combination with the Lyapunov-based Constraint

Similar to the previous section, the Lyapunov-based constraint can be added to the Problem (4.16) to have a new constrained optimization problem, which has equality constraints and is listed below, to additionally maintain stability:

$$\begin{aligned} \pi_{n+1} = \arg\min_{\pi} \quad & -\mathbb{E}_{u_{t_k}\sim\pi} V^{\pi_n}(x_{t_{k+1}}) \\ s.t. \quad & \sum_{x_{t_k}\in\mathcal{X}} ReLU\big(-H^{\pi_n}(x_{t_{k+1}}) + H^{\pi_n}(x_{t_k}) - \epsilon H^{\pi_n}(x_{t_k})\big) = 0, \\ & \sum_{x_{t_k}\in\mathcal{X}} ReLU\big(L^{\pi_n}(x_{t_{k+1}}) - L^{\pi_n}(x_{t_k}) + \beta L^{\pi_n}(x_{t_k})\big) = 0, \end{aligned} \tag{4.17}$$

where $x_{t_{k+1}}$ is the next state following x_{t_k} and u_{t_k} according to the system dynamics. Moreover, adding the Lyapunov-based constraint offers an extra benefit that, as the controller is updated to drive the agent towards the desired state x_{desired}, certain states near x_{desired} may have their

safety easily maintained by the updated controller that directly drives the agent to reach x_{desired}. Then, the safe set will monotonically expand, as stated in Theorem 4.1.

Remark 4.6. *The above formulation represents an idealized constrained optimization problem defined over the full state space. In practical implementations, however, the expectations and constraints involved in the optimization are approximated using sampled trajectories collected through interactions with the environment, together with replay-buffer-based learning and neural function approximation. Consequently, the implemented algorithm should be viewed as a sampled approximation of the idealized optimization problem rather than an exact realization of it. The practical algorithm seeks to approximate the idealized constrained optimization problem for corresponding desired properties through data-driven learning.*

Problem (4.17) will still be solved by using the augmented Lagrangian method. However, still, there may exist $x_{t_k} \in \mathcal{X}_{\text{infeasible}}$ that we hope to avoid. Therefore, we introduce the following assumption:

Assumption 4.3. $\forall n \geq 0$ *and* $\forall x_{t_k} \in \mathcal{X}_{infeasible}$*, the probability that the policy* π_{n+1} *obtained by solving Problem (4.17) via the augmented Lagrangian method satisfies the Lyapunov-based constraint is positive.*

This assumption assumes that when the agent is within $\mathcal{X}_{\text{infeasible}}$ where no feasible solution exists to simultaneously maintain safety and drive the agent closer to the goal according to the Lyapunov-based constraint, for each iteration, there is a positive probability that the augmented Lagrangian method outputs a control policy that satisfies the Lyapunov-based constraint, thereby driving the agent closer to the goal and causing immediate safety violations. Then, a proposition is given:

Proposition 4.2. *Suppose* $\exists Z \in \mathbb{Z}^+$ *such that* $x_{t_0} \in \mathcal{X}_{safe}^{\pi_Z}$*, then, under Assumption 4.3, when* $n \to \infty$*, for any trajectory that starts from* x_{t_0} *and is driven by* π_n *obtained by solving Problem (4.17) iteratively,* $Pr(x_{t_k} \in \mathcal{X}_{infeasible}) = 0$*,* $\forall k \geq 0$*.*

Proof. Since $x_{t_0} \in \mathcal{X}_{\text{safe}}^{\pi_i}$, $\forall i \geq Z$, when $n \to \infty$, $x_{t_0} \in \mathcal{X}_{\text{safe}}^{\pi_n}$ and $H^{\pi_n}(x_{t_0}) = 0$. Because $\forall n \geq 0$ and $\forall x_{t'_k} \in \mathcal{X}_{\text{infeasible}}$, $Pr(x_{t'_{k+1}} \in \mathcal{X}_{\text{vio}}) > 0$ according to Assumption 4.3 and the definition of

$\mathcal{X}_{\text{infeasible}}$, it can be concluded that

$$H^{\pi_{n+1}}(x_{t'_k}) = b(x_{t'_k}) + \gamma \sum\nolimits_{u_{t_k}} \pi_{n+1}(u_{t_k}|x_{t'_k}) H^{\pi_{n+1}}(x_{t'_{k+1}}) < 0$$

since $Pr(H^{\pi_{n+1}}(x_{t'_{k+1}}) < 0) > 0$ given $Pr(x_{t'_{k+1}} \in \mathcal{X}_{\text{vio}}) > 0$. Because $\forall n \geq Z$, $H^{\pi_n}(x_{t_0}) = 0 > H^{\pi_n}(x_{t'_k})$ for any $x_{t'_k} \in \mathcal{X}_{\text{infeasible}}$, any trajectory starts from x_{t_0} will not enter $\mathcal{X}_{\text{infeasible}}$ with a positive probability under π_n when $n \to \infty$ due to the neural barrier certificate. □

According to Proposition 4.2, the controlled system will be prevented from visiting $\mathcal{X}_{\text{infeasible}}$ due to the neural barrier certificate. Therefore, under such circumstances, the backup controller proposed in the previous section where predefined CBFs are used can be not necessary.

4.5 Practical Implementation with SAC and NODEs

In this section, we provide a practical implementation, which is called Neural ordinary differential equations-based Lyapunov-Barrier Actor-Critic (NLBAC) by combining the proposed method with Soft Actor-Critic (SAC) and applying neural ordinary differential equations (NODEs) for system modeling. Specifically, depending on whether predefined CBFs are available, two versions of the algorithm NLBAC are given separately with individual pseudocodes.

If useful predefined CBFs are given prior to the training process, then, according to (4.4), future states $\{x_{t_{k+1}}, x_{t_{k+2}}, \ldots, x_{t_{k+r_i}}\}$ under the controller π_p, where r_i is the relative degree, are required. When a learned barrier certificate is used, $x_{t_{k+1}}$ is also required to construct Problem (4.17). Therefore, predictions must be made in a real implementation. Without nominal models, instead of conventional neural networks, we choose NODEs to directly approximate the dynamics of the system, since NODEs offer adaptability to discretization steps due to their inherent capability to learn the continuous form of system dynamics which are usually differential equations. To train the NODE model, we use state sequences $X = \{x_{t_k}, x_{t_{k+1}}, \ldots, x_{t_{k+h}}\}$ and corresponding control sequences $U = \{u_{t_k}, u_{t_{k+1}}, \ldots, u_{t_{k+h-1}}\}$ whose elements are collected during the system's interaction with the environment and then stored in the replay buffer which saves all data, including states x_{t_k}, action signals u_{t_k}, reward signals r_{t_k}, cost signals c_{t_k} and barrier signals b_{t_k} (if used). The model then computes the approximated state sequence

$\{\hat{x}_{t_{k+1}}, \hat{x}_{t_{k+2}}, \ldots, \hat{x}_{t_{k+h}}\}$ based on (4.7). The loss function used to train the NODE model is:

$$\ell = \frac{1}{h}\sum_{i=1}^{h}|x_{t_{k+i}} - \hat{x}_{t_{k+i}}|. \tag{4.18}$$

Comparisons showing NODEs have better performance in modeling compared to conventional neural networks are provided on our GitHub page at the link shown in Section 4.6.

Instead of using the learned system dynamics to generate imaginary rollouts to help train value functions, we utilize the output of the learned NODEs to construct the Lyapunov-based constraint and CBFs (or the neural barrier certificate-based constraint) for clearly guiding the controller to satisfy stability and safety constraints quickly. Therefore, the prediction horizon equals the relative degree r_i (or 1), which alleviates the problem of large error propagation since r_i is usually not large.

In the implementation, we construct constrained optimization problems (4.10) and (4.17) based on the batch $\mathcal{D}$ sampled from the replay buffer, which is common practice in RL. Here we first consider the case where predefined CBFs are available. Denote the parameters of the primal RL-based control policy π_p and action-value networks $Q_i^{\pi_p}, i = 1, 2$ by θ_p and ϕ_i, and use L_ν, referred to as the Lyapunov network, to approximate L^{π_p} with parameters ν. By applying the Soft Actor-Critic (SAC) and the sampled data, the objective function of Problem (4.10) is rewritten as:

$$-R^{\theta_p} = -\,\mathbb{E}_{x_{t_k}\sim\mathcal{D},\xi\sim\mathcal{N}}\big[\min_{j=1,2} Q_{\phi_j}(x_{t_k}, \tilde{u}_{\theta_p}(x_{t_k}, \xi)) - \alpha_p \log \pi_{\theta_p}(\tilde{u}_{\theta_p}(x_{t_k}, \xi)|x_{t_k})\big], \tag{4.19}$$

where $\xi \sim \mathcal{N}(0, I)$ and $\tilde{u}_{\theta_p}$ denotes the action generated by the controller π_{θ_p}. The loss functions $J_{Q^{\pi_p}}(Q_{\phi_i})$, $J_{\alpha_p}(\alpha_p)$ and $J_{L^{\pi_p}}(L_\nu)$ used to update $Q_{\phi_i}, i = 1, 2$, coefficient α_p, and Lyapunov network L_ν are given as:

$$\begin{aligned} J_{Q^{\pi_p}}(Q_{\phi_i}) = \mathbb{E}_{(x_{t_k}, u_{t_k}, r_{t_k}, x_{t_{k+1}})\sim\mathcal{D},\xi\sim\mathcal{N}}\Big[\big[r_{t_k} + \gamma\big(\min_{j=1,2} Q_{targ,\phi_j}(x_{t_{k+1}}, \tilde{u}_{\theta_p}(x_{t_{k+1}}, \xi)) \\ - \alpha_p \log \pi_{\theta_p}(\tilde{u}_{\theta_p}(x_{t_{k+1}}, \xi)|x_{t_{k+1}})\big) - Q_{\phi_i}(x_{t_k}, u_{t_k})\big]^2\Big], \end{aligned} \tag{4.20}$$

$$J_{\alpha_p}(\alpha_p) = -\alpha_p \times \mathbb{E}_{x_{t_k}\sim\mathcal{D},\xi\sim\mathcal{N}}\big[\log \pi_{\theta_p}(\tilde{u}_{\theta_p}(x_{t_k}, \xi)|x_{t_k}) + \mathcal{H}\big], \tag{4.21}$$

$$J_{L^{\pi_p}}(L_\nu) = \mathbb{E}_{(x_{t_k}, c_{t_k}, x_{t_{k+1}})\sim\mathcal{D}}\Big[\big[c_{t_k} + \gamma_c L_{targ,\nu}(x_{t_{k+1}}) - L_\nu(x_{t_k})\big]^2\Big]. \tag{4.22}$$

Here L_ν can also be designed as an action-value network in the implementation. Furthermore, a surrogate augmented Lagrangian function based on neural networks and sampled data is given:

$$\begin{aligned}\mathcal{L}_c^{surr}(\theta_p, \lambda_i, \zeta) = &-R^{\theta_p} + \sum_{i=1}^{m}\lambda_i\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(-\Phi_{i,r_i}(x_{t_k})\big)\right)\\ &+\sum_{i=1}^{m}\frac{c}{2}\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(-\Phi_{i,r_i}(x_{t_k})\big)\right)^2 + \zeta\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(L_\nu(\hat{x}_{t_{k+1}}) - L_\nu(x_{t_k}) + \beta L_\nu(x_{t_k})\big)\right)\\ &+\frac{c}{2}\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(L_\nu(\hat{x}_{t_{k+1}}) - L_\nu(x_{t_k}) + \beta L_\nu(x_{t_k})\big)\right)^2.\end{aligned} \tag{4.23}$$

where $\{\hat{x}_{t_{k+1}}, \hat{x}_{t_{k+2}}, \ldots, \hat{x}_{t_{k+r_i}}\}$ are predicted future states, output by the NODE model, following the controller π_{θ_p}. To reduce the computational burden of the typical augmented Lagrangian method, which involves repeated minimization of $\mathcal{L}_c^{surr}$ over θ_p by RL with λ_i, ζ, and c that are also updated, we use a different timescale method to update action-value networks (including the Lyapunov network) and the policy network with different learning rates. Delayed update [19] is also applied, which is shown in Algorithm 4.2.

With respect to the backup controller π_b which is parameterized by θ_b, by applying SAC, the objective function of Problem (4.12) is rewritten as: $-R^{\theta_b} = -\mathbb{E}_{x_{t_k}\sim\mathcal{D},\xi\sim\mathcal{N}}\big[\min_{j=1,2} Q_{\phi_j}(x_{t_k}, \tilde{u}_{\theta_b}(x_{t_k}, \xi)) - \alpha_b \log\pi_{\theta_b}(\tilde{u}_{\theta_b}(x_{t_k}, \xi)|x_{t_k})\big]$. The loss function to update the coefficient α_b is: $J_{\alpha_b}(\alpha_b) = -\alpha_b \times \mathbb{E}_{x_{t_k}\sim\mathcal{D},\xi\sim\mathcal{N}}\big[\log\pi_{\theta_b}(\tilde{u}_{\theta_b}(x_{t_k}, \xi)|x_{t_k}) + \mathcal{H}_b\big]$, where $\mathcal{H}_b$ can be tuned to encourage exploration. Similarly, a surrogate augmented Lagrangian function for updating θ_b can be given as follows:

$$\begin{aligned}\mathcal{L}_{c_b}^{surr}(\theta_b, \lambda_{b,i}) = &-R^{\theta_b} + \sum_{i=1}^{m}\lambda_{b,i}\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(-\Phi_{i,r_i}^b(x_{t_k})\big)\right)\\ &+\sum_{i=1}^{m}\frac{c_b}{2}\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(-\Phi_{i,r_i}^b(x_{t_k})\big)\right)^2.\end{aligned} \tag{4.24}$$

With these elements, the complete algorithm NLBAC, when predefined CBFs are available, is provided in Algorithm 4.2.

Then we consider the case where the learned barrier certificate is used. With a slight abuse of notation, denote the parameter of the RL-based controller by θ, and define the parameters of the action-value networks and the Lyapunov network similarly to the previous part where predefined CBFs are used. The barrier certificate H^π is parameterized by ω as H_ω, and in this implementation, H_ω is designed as an action-value function and trained by minimizing the TD error in SAC for better sample efficiency. Thus, the constraint based on the barrier certificate in Problem (4.17) can be rewritten as $\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(-H_\omega(\hat{x}_{t_{k+1}},\hat{u}_{t_{k+1}})+H_\omega(x_{t_k},u_{t_k})-\epsilon H_\omega(x_{t_k},u_{t_k})\big)=0$, where $\hat{x}_{t_{k+1}}$ is the next state predicted by the NODE given x_{t_k} and $u_{t_k}=\pi_\theta(x_{t_k})$, and $\hat{u}_{t_{k+1}}=\pi_\theta(\hat{x}_{t_{k+1}})$. Here, the loss function $J_{H^\pi}(H_\omega)$ to update H_ω is:

$$J_{H^\pi}(H_\omega)=\mathbb{E}_{(x_{t_k},u_{t_k},b_{t_k},x_{t_{k+1}})\sim\mathcal{D}}\Big[\big[b_{t_k}+\gamma H_{targ,\omega}(x_{t_{k+1}},u_{t_{k+1}})-H_\omega(x_{t_k},u_{t_k})\big]^2\Big], \tag{4.25}$$

where $u_{t_{k+1}}=\pi_\theta(x_{t_{k+1}})$. The objective function of Problem (4.17) is given as:

$$-R^\theta_{\text{certificate}}=-\mathbb{E}_{x_{t_k}\sim\mathcal{D},\xi\sim\mathcal{N}}\big[\min_{j=1,2} Q_{\phi_j}(x_{t_k},\tilde{u}_\theta(x_{t_k},\xi))-\alpha\log\pi_\theta(\tilde{u}_\theta(x_{t_k},\xi)|x_{t_k})\big], \tag{4.26}$$

and a surrogate augmented Lagrangian function is further provided as follows:

$$\begin{aligned}
&\mathcal{L}^{surr}_{c,\text{certificate}}(\theta,\lambda_i,\zeta)=-R^\theta_{\text{certificate}}\\
&+\sum_{i=1}^m\lambda_i\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(-H_\omega(\hat{x}_{t_{k+1}},\hat{u}_{t_{k+1}})+H_\omega(x_{t_k},u_{t_k})-\epsilon H_\omega(x_{t_k},u_{t_k})\big)\right)\\
&+\sum_{i=1}^m\frac{c}{2}\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(-H_\omega(\hat{x}_{t_{k+1}},\hat{u}_{t_{k+1}})+H_\omega(x_{t_k},u_{t_k})-\epsilon H_\omega(x_{t_k},u_{t_k})\big)]\right)^2\\
&+\zeta\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(L_\nu(\hat{x}_{t_{k+1}})-L_\nu(x_{t_k})+\beta L_\nu(x_{t_k})\big)\right)\\
&+\frac{c}{2}\left(\sum_{x_{t_k}\in\mathcal{D}} ReLU\big(L_\nu(\hat{x}_{t_{k+1}})-L_\nu(x_{t_k})+\beta L_\nu(x_{t_k})\big)\right)^2.
\end{aligned} \tag{4.27}$$

Then, the algorithm NLBAC, when a learned barrier certificate is used, is shown in Algorithm

Algorithm 4.2 Neural Ordinary Differential Equations-based Lyapunov-Barrier Actor-Critic (NLBAC) with predefined CBFs

Input: Maximum number of iterations N, initialized NODE parameterized by ψ, action-value networks Q_{ϕ_i}, Lyapunov network L_ν, primary controller network π_{θ_p}, α_p, Lagrangian multipliers λ_i and ζ, backup controller network π_{θ_b}, α_b, Lagrangian multipliers $\lambda_{b,i}$, penalty parameter c and c_b, factor $\rho_c \in (1, \infty)$, learning rates $\eta_1 > \eta_2$, η_3, delay steps n_m, n_L, and n_b.

for $n = 0, 1, \ldots, N$ **do**
  if $n \bmod n_m = 0$ **then**
    $\psi \leftarrow \psi - \eta_3 \nabla_\psi \ell$
  end if
  $\phi_i \leftarrow \phi_i - \eta_1 \nabla_{\phi_i} J_{Q^{\pi_p}}(Q_{\phi_i})$ for $i \in \{1, 2\}$;
  $\nu \leftarrow \nu - \eta_1 \nabla_\nu J_{L^{\pi_p}}(L_\nu)$
  $\theta_p \leftarrow \theta_p - \eta_2 \nabla_{\theta_p} \mathcal{L}_c^{surr}(\theta_p, \lambda_i, \zeta)$;
  $\alpha_p \leftarrow \alpha_p - \eta_2 \nabla_{\alpha_p} J_{\alpha_p}(\alpha_p)$
  if $n \bmod n_L = 0$ **then**
    $\lambda_i \leftarrow \lambda_i + c \left(\sum_{x_{t_k} \in \mathcal{D}} ReLU\big(- \Phi_{i,r_i}(x_{t_k}) \big) \right)$;
    $\zeta \leftarrow \zeta + c \left(\sum_{x_{t_k} \in \mathcal{D}} ReLU\big(L_\nu(\hat{x}_{t_{k+1}}) - L_\nu(x_{t_k}) + \beta L_\nu(x_{t_k}) \big) \right)$
  end if
  $c \leftarrow \rho_c \times c$
  if $n \bmod n_b = 0$ **then**
    $\theta_b \leftarrow \theta_b - \eta_2 \nabla_{\theta_b} \mathcal{L}_{c_b}^{surr}(\theta_b, \lambda_{b,i})$;
    $\alpha_b \leftarrow \alpha_b - \eta_2 \nabla_{\alpha_b} J_{\alpha_b}(\alpha_b)$
    if $n \bmod (n_b \times n_L) = 0$ **then**
      $\lambda_{b,i} \leftarrow \lambda_{b,i} + c_b \left(\sum_{x_{t_k} \in \mathcal{D}} ReLU\big(- \Phi^b_{i,r_i}(x_{t_k}) \big) \right)$
    end if
    $c_b \leftarrow \rho_c \times c_b$
  end if
  if Backup controller should be used according to the conditions specific to the task **then**
    $u_{t_k} \sim \pi_{\theta_b}(u_{t_k} | x_{t_k})$ and apply u_{t_k}
  else
    $u_{t_k} \sim \pi_\theta(u_{t_k} | x_{t_k})$ and apply u_{t_k}, and store the pair $(x_{t_k}, u_{t_k}, r_{t_k}, c_{t_k}, x_{t_{k+1}})$ in the replay buffer
  end if
end for

Output: π_{θ_p}, π_{θ_b}, $Q_{\phi_i}, i = 1, 2$, and L_ν

4.3.

Algorithm 4.3 Neural Ordinary Differential Equations-based Lyapunov-Barrier Actor-Critic (NLBAC) with Learned Barrier Certificate

Input: Maximum number of iterations N, initialized NODE parameterized by ψ, action-value networks Q_{ϕ_i}, barrier certificate H_ω, Lyapunov network L_ν, primary controller network π_θ, α, Lagrangian multipliers λ_i and ζ, penalty parameter c, factor $\rho_c \in (1, \infty)$, learning rates $\eta_1 > \eta_2$, η_3, delay steps n_m, n_L.

1: **for** $n = 0, 1, \ldots, N$ **do**
2: **if** $n \bmod n_m = 0$ **then**
3: $\psi \leftarrow \psi - \eta_3 \nabla_\psi \ell$
4: **end if**
5: $\phi_i \leftarrow \phi_i - \eta_1 \nabla_{\phi_i} J_{Q^\pi}(Q_{\phi_i})$ for $i \in \{1, 2\}$;
$\nu \leftarrow \nu - \eta_1 \nabla_\nu J_{L^\pi}(L_\nu)$;
$\omega \leftarrow \omega - \eta_1 \nabla_\omega J_{H^\pi}(H_\omega)$
6: $\theta \leftarrow \theta - \eta_2 \nabla_\theta \mathcal{L}^{surr}_{c,\text{certificate}}(\theta, \lambda_i, \zeta)$;
$\alpha \leftarrow \alpha - \eta_2 \nabla_\alpha J_\alpha(\alpha)$
7: **if** $n \bmod n_L = 0$ **then**
8: $\lambda_i \leftarrow \lambda_i + c \Big(\sum_{x_{t_k} \in \mathcal{D}} ReLU\big(- H_\omega(\hat{x}_{t_{k+1}}, \hat{u}_{t_{k+1}}) + H_\omega(x_{t_k}, u_{t_k}) - \epsilon H_\omega(x_{t_k}, u_{t_k}) \big) \Big)$;
9: $\zeta \leftarrow \zeta + c \Big(\sum_{x_{t_k} \in \mathcal{D}} ReLU\big(L_\nu(\hat{x}_{t_{k+1}}) - L_\nu(x_{t_k}) + \beta L_\nu(x_{t_k}) \big) \Big)$
10: **end if**
11: $c \leftarrow \rho_c \times c$
12: $u_{t_k} \sim \pi_\theta(u_{t_k} | x_{t_k})$ and apply u_{t_k}, and store the pair $(x_{t_k}, u_{t_k}, r_{t_k}, c_{t_k}, b_{t_k}, x_{t_{k+1}})$ in the replay buffer
13: **end for**

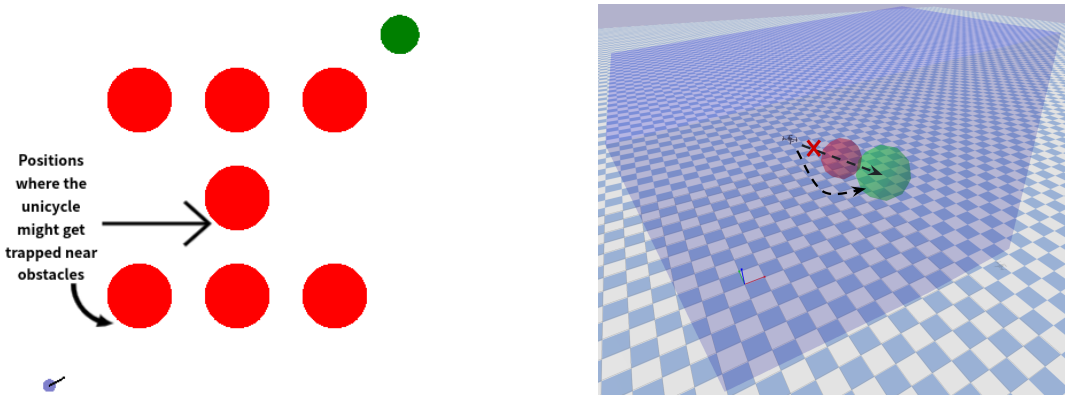
Output: π_θ, $Q_{\phi_i}, i = 1, 2$, and L_ν

4.6 Experiments

In this section, we evaluate the proposed NLBAC algorithm (with either predefined CBFs or neural barrier certificates) on four tasks to answer the following research questions:

1. Does the NLBAC algorithm result in a decreasing number of safety violations during training and achieve zero or near-zero safety violations by the end of training?
2. Does the NLBAC algorithm effectively drive the agent toward the desired state?
3. Does the NLBAC algorithm exhibit better sample efficiency compared to other baselines?

Baselines. Baselines compared here are CMDP-based or developed to solve state-wise constraints, and range from model-free to model-based approaches. CMDP baselines include MBPPO-Lagrangian [127], CPO [13], PPO-Lagrangian and TRPO-Lagrangian [117]. SAC-



(a) The unicycle environment. The blue point is the controlled unicycle, the green circle is the destination (desired state), and the red circles are obstacles to avoid. Some positions where the unicycle can get trapped due to the infeasibility are shown.

(b) The quadrotor environment. The green ball represents the destination that the controlled quadrotor should reach, while the red ball is an obstacle that needs to be avoided. Additionally, the quadrotor is required to remain within the blue rectangular area for safety.

Figure 4.2: "Unicycle" and "Quadrotor" tasks.

RCBF [80] is the baseline for hard safety constraints with CBFs. LAC [22] is used as the model-free baseline combining the Lyapunov-based constraint for stability. These baselines encompass diverse approaches, and are representative works in the field of constrained RL.

Environments. We conduct experiments on four widely-used tasks [2] for safe model-based RL studies [36, 80, 82, 128, 129]. These tasks are carefully selected and ordered to progressively demonstrate the capabilities of the proposed NLBAC algorithm. The progression moves from lower-dimensional systems (e.g., Unicycle) to higher-dimensional systems (e.g., Car Following), from static obstacle environments (e.g., Unicycle) to dynamic obstacle environments (e.g., Car Following and PVTOL), and from systems with lower relative degrees (e.g., Unicycle and Car Following) to those with higher relative degrees (e.g., PVTOL). Additionally, the tasks transition from simpler gym-based environments to more sophisticated physics-engine-based scenarios (e.g., Quadrotor), providing a comprehensive evaluation of the algorithm's adaptability and effectiveness. Details of these four tasks are provided below:

Unicycle: This task is designed to evaluate the proposed algorithm in a simpler environment with a low-dimensional state space and predefined CBFs of relative degree one. Both versions of the algorithm (with predefined CBFs and with neural barrier certificates) are tested with

[2]Codes are on our GitHub page: `https://github.com/LiqunZhao/Neural-ordinary-differential-equations-based-Lyapunov-Barrier-Actor-Critic-NLBAC`.

static obstacles. Same as the description in Chapter 3, a unicycle is tasked with reaching the designated location, i.e., the destination, while avoiding collisions with obstacles, as shown in Fig. 4.2a. The real dynamics of the system is:

$$\dot{x}_{t_k} = \begin{bmatrix} \cos(\theta_{t_k}) & 0 \\ \sin(\theta_{t_k}) & 0 \\ 0 & 1.0 \end{bmatrix} (u_{t_k} + u_{d,t_k}).$$

Here $x_{t_k} = [x_{1t_k}, x_{2t_k}, \theta_{t_k}]^T$ is the state where x_{1t_k} and x_{2t_k} represent the X-coordinate and Y-coordinate of the unicycle and θ_{t_k} is the orientation at the timestep t_k. The control signal $u_{t_k} = [v_{t_k}, \omega_{t_k}]^T$ comprises linear velocity v_{t_k} and angular velocity ω_{t_k}. The time interval used in this experiment is 0.02s. $u_{d,t_k} = -0.1[\cos(\theta_{t_k}), 0]^T$. As to NODE-based models, the input of network $\hat{f}$ is the state with the dimension of 3, and the output dimension is 3; the input of network $\hat{g}$ is the state with the dimension of 3, and the output dimension is 6 (which is then be resized as $[3, 2]$). Then, a point at a distance l_p ahead of the unicycle is considered to establish safety constraints for collision-free navigation, and the function $p : \mathbb{R}^3 \to \mathbb{R}^2$ is defined as:

$$p(x_{t_k}) = \begin{bmatrix} x_{1t_k} \\ x_{2t_k} \end{bmatrix} + l_p \begin{bmatrix} \cos(\theta_{t_k}) \\ \sin(\theta_{t_k}) \end{bmatrix}.$$

The reward signal is formulated as $-K_1(v_t - v_s)^2 + K_2\Delta d$, where v_s represents the predefined velocity, Δd denotes the reduction in the distance between the unicycle and the destination in two consecutive timesteps, and K_1 and K_2 are coefficients set to 0.1 and 30, respectively. An additional reward of 500 will be given if the unicycle reaches a small neighborhood of the destination. The cost signal is given by $\left\|p(x_{t_{k+1}}) - p(x_{\text{desired}})\right\|$ where $p(x_{\text{desired}}) = [x_{1\text{desired}}, x_{2\text{desired}}]^T$ denotes the position of the desired location. predefined CBFs, if provided, are $h_i(x_{t_k}) = \frac{1}{2}\big((p(x_{t_k}) - p_{\text{obs}_i})^2 - \delta^2\big)$ where p_{obs_i} represents the position of the ith obstacle, and δ is the minimum required distance between the unicycle and obstacles. Hence, the relative degree is 1 and the planning horizon for NODEs predictions is 1. When no predefined CBFs are available, the neural barrier certificate will be learned jointly with the controller by using additional barrier signals with $D = -20$.

When predefined CBFs are used, the inability to satisfy safety and stability constraints simultaneously at state $x_{t_k} \in \mathcal{X}_{\text{infeasible}}$ can cause the unicycle to become trapped near obstacles. In such cases, the backup controller takes over from the primary controller. The primary controller is reinstated when the unicycle moves a long distance away from the trapped position, or when the predefined time threshold for using the backup controller is exceeded.

Car Following: This task is designed to evaluate the proposed algorithm in an environment with a higher-dimensional state space and predefined CBFs of relative degree two. The version of the algorithm with predefined CBFs is tested with dynamic obstacles. This environment simulates a chain of five cars following each other on a straight road. The objective is to control the acceleration of the 4th car to maintain a desired distance from the 3rd car while avoiding collisions with other cars. The real dynamics of all cars except the 4th one is given by:

$$\dot{x}_{t_k,i} = \begin{bmatrix} v_{t_k,i} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 + d_i \end{bmatrix} a_{t_k,i}, \qquad \forall i \in \{1, 2, 3, 5\}.$$

Each state of the system is denoted as $x_{t_k,i} = [p_{t_k,i}, v_{t_k,i}]^T$, indicating the position $p_{t_k,i}$ and velocity $v_{t_k,i}$ of the ith car at the timestep t_k, $d_i = 0.1$. The time interval used in this experiment is 0.02s. The predefined velocity of the 1st car is $v_s - 4\sin(t_k)$, where $v_s = 3.0$. Its acceleration is given as $a_{t_k,1} = k_v(v_s - 4\sin(t_k) - v_{t_k,1})$ where $k_v = 4.0$. Accelerations of Car 2 and 3 are given by:

$$a_{t_k,i} = \begin{cases} k_v(v_s - v_{t_k,i}) - k_b(p_{t_k,i-1} - p_{t_k,i}), & |p_{t_k,i-1} - p_{t_k,i}| < 6.5, \\ k_v(v_s - v_{t_k,i}), & \text{otherwise}, \end{cases}$$

where $k_b = 20.0$ and $i = 2, 3$. The acceleration of the 5th car is:

$$a_{t_k,5} = \begin{cases} k_v(v_s - v_{t_k,5}) - k_b(p_{t_k,3} - p_{t_k,5}), & |p_{t_k,3} - p_{t_k,5}| < 13.0, \\ k_v(v_s - v_{t_k,5}), & \text{otherwise}. \end{cases}$$

The model of the 4th car is as follows:

$$\dot{x}_{t_k,4} = \begin{bmatrix} v_{t_k,4} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1.0 \end{bmatrix} u_{t_k},$$

where u_{t_k} is the acceleration of the 4th car, and also the control signal generated by the controller at the timestep t_k.

The reward signal is defined to minimize the overall control effort, and an additional reward of 2.0 is granted during timesteps when $d_{t_k} = p_{t_k,3} - p_{t_k,4}$, which is the distance between the 3rd and 4th car, falls within $[9.0, 10.0]$. This range is defined as the desired region for d_{t_k}. Thus, the cost signal is determined as $\left\| d_{t_{k+1}} - d_{\text{desired}} \right\|$, where $d_{\text{desired}} = 9.5$. CBFs are defined as $h_1(x_{t_k}) = p_{t_k,3} - p_{t_k,4} - \delta$ and $h_2(x_{t_k}) = p_{t_k,4} - p_{t_k,5} - \delta$, with δ being the minimum required distance between the cars. Hence, the relative degree is 2 and the planning horizon for making predictions using NODE is 2.

When we use NODEs to model this system, we assume that we do not have the priori information that this system is control-affine. The input of network $\mathcal{F}$ is $(t_k, \hat{x}_{t_k}, u_{t_k})$ with the dimension of 12, and the output dimension is 10. When a safety constraint is violated if two types of constraints cannot be satisfied simultaneously, the 4th car might be in close proximity to the 5th car in order to make d_{t_k} be within $[9.0, 10.0]$. In such cases, the backup controller is activated. The primary controller is reactivated when the 4th car moves beyond the dangerous area, namely out of the vicinity of the 5th car, or when the predetermined time threshold for utilizing the backup controller is exceeded.

Planar Vertical Take-Off and Landing (PVTOL): This task evaluates the proposed algorithm in an environment with both static and dynamic obstacles, more complex system dynamics, and predefined CBFs of relative degree three. Both versions of the algorithm (with predefined CBFs and with neural barrier certificates) are tested in scenarios with static obstacles. In this experiment, a quadcopter is required to reach a destination while avoiding obstacles and keeping within a specified range along the y-axis and a specific distance from a safety pilot along the

x-axis. The real dynamics of the system is:

$$\dot{x}_{t_k} = \begin{bmatrix} v_{1t_k} \\ v_{2t_k} \\ 0 \\ -\sin(\theta_{t_k}) \times f_{t_k} \\ \cos(\theta_{t_k}) \times f_{t_k} - 1.0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} u_{t_k}.$$

Here $x_{t_k} = [x_{1t_k}, x_{2t_k}, \theta_{t_k}, v_{1t_k}, v_{2t_k}, f_{t_k}]^T$ is the state where x_{1t_k} and x_{2t_k} represent the X-coordinate and Y-coordinate of the quadcopter and θ_{t_k} is the orientation at the timestep t_k. v_{1t_k}, v_{2t_k} are velocities along X and Y axis, and f_{t_k} is the thrust. The control signal $u_{t_k} = [u_{ft_k}, \omega_{t_k}]^T$ includes the derivative of the thrust and angular velocity. The time interval used in this experiment is 0.02s.

The reward signal is defined to minimize the distance from the destination, and an additional reward of 1500 will be given if the quadcopter reaches a small neighborhood of the destination. The cost signal is the current distance from the destination, and the safety operator tracks the quadcopter via a proportional controller along the x-axis. When predefined CBFs are used, collision avoidance and confinement within specific ranges along the x-axis and y-axis are ensured by those fixed CBFs, following a similar approach to the previous two environments. The relative degree, and therefore the planning horizon for NODEs predictions, is 3. When no predefined CBFs are available, the neural barrier certificate will be learned jointly with the controller by using additional barrier signals with $D = -0.1$.

For NODE-based models, the input of network $\hat{f}$ is the state of dimension 6, and the output dimension is 6; the input of network $\hat{g}$ is the state with the dimension of 6, and the output dimension is 12 (which is then be resized as $[6, 2]$). The conditions for activating the backup controller resemble those in the previous two tasks.

Quadrotor: This task evaluates the proposed algorithm in a sophisticated physics-engine-based environment. The version of the algorithm using neural barrier certificates is tested. This

is a stabilization task which is modified from safety-control-gym [129]. In this task, the 2D quadrotor needs to approach towards and finally arrive at a desired destination, while keeping both the vertical and horizontal positions within a predefined range, and avoiding the collision with an obstacle shown in Fig. 4.2b. A more detailed description, including the state, action, and system dynamics, can be found in [129]. The reward signal is designed to minimize the distance between the current position and the goal position, which is also defined to be the cost signal. An additional reward of 250 is given when the quadrotor reaches the goal. In this task, no predefined CBFs are provided, and therefore, a neural barrier certificate will be learned jointly with the control policy. We define the barrier signal as $D_1 = -1.0$ when the quadrotor's position is not within the predefined range, and $D_2 = -10.0$ when the quadrotor collides with the obstacle.

For the NODE used to model the system dynamics, we also assume that we do not have any prior information. Different from previous tasks, here states and actions are normalized before being used as the input of the NODE, and outputs of the NODE are denormalized before being used as predictions. The input dimension of the NODE is 8, and the output dimension is 6.

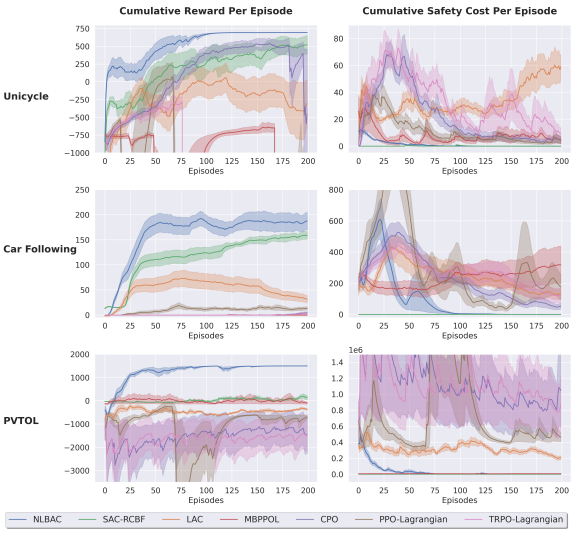


Figure 4.3: The cumulative reward and cumulative safety cost of each episode in "Unicycle", "Car Following" and "PVTOL" tasks are compared between the proposed NLBAC (with predefined CBFs and in blue) and baselines. Each curve illustrates the average across ten experiments employing different random seeds, with the shaded area denoting the standard error.

Compute Resources. Our code runs on Intel Platinum 8628 (Cascade Lake), 2.90 GHz

CPUs, and NVIDIA Tesla V100 GPUs.

Experimental results and comparisons. Simulations were conducted for the tasks where predefined CBFs are available ("Unicycle", "Car Following", and "PVTOL"); the results are depicted in Fig. 4.3. Since in these tasks, high rewards are granted when the system approaches and achieves the desired state (destination), we use the cumulative reward as a measure, where a higher cumulative reward can indicate better convergence to the destination. In the experiments, our framework demonstrates superior performance, yielding consistently higher cumulative rewards compared to baselines, which suggests that the controlled agents, under our framework, can more efficiently approach and reach their desired states (destinations) in much fewer episodes. Moreover, the fluctuations observed in the cumulative rewards are notably smaller than those exhibited by other baselines. This indicates that the systems can quickly recover to their previous high reward levels after experiencing deterioration. This is attributed to the Lyapunov-based constraint, which explicitly makes the algorithm goal-oriented. Also, compared to the model-free LAC methods with Lyapunov certificates, our algorithm utilizes NODEs for effective modeling, and utilizes state-wise Lyapunov-based constraints instead of expectation-based constraints, resulting in better performance.

Concerning safety, our framework exhibits much fewer violations compared to other baselines except for SAC-RCBF. By using predefined CBFs to enforce state-wise safety constraints, our methods result in substantially fewer safety violations compared to CMDP-based model-free baselines. The model-based baseline MBPPO-Lagrangian did not perform well in our experiments. This could be due to its reliance on CMDP-based constraints and extensive use of imaginary rollouts to update parameters, which may introduce error aggregation. When compared to SAC-RCBF, our NLBAC framework achieves higher cumulative rewards with the assistance of the Lyapunov-based constraint, ultimately reaching zero or near-zero safety violations without requiring a good nominal model. Additionally, our framework avoids the need to solve a QP, making it directly applicable to nonconvex settings, and enabling the integration of techniques like neural barrier certificates.

Moreover, our algorithm requires the fewest interactions with the environment to achieve a good performance with regard to reward and safety, which means it has higher sample efficiency.

This suggests its potential applicability in scenarios where data availability may be limited.

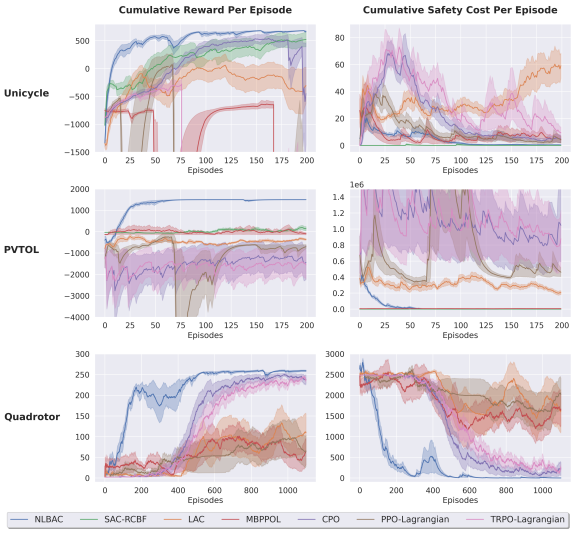


Figure 4.4: The cumulative reward and cumulative safety cost of each episode in "Unicycle", "PVTOL" and "Quadrotor" tasks are compared between the proposed NLBAC (with neural barrier certificate and in blue) and baselines. For "Unicycle" and "PVTOL", each curve illustrates the average across ten experiments employing different random seeds, with the shaded area denoting the standard error. For "Quadrotor", each curve illustrates the average across five experiments employing different random seeds, and the baseline SAC-RCBF is not performed due to the lack of a nominal model.

We also conduct evaluations of NLBAC using neural barrier certificates in the "Unicycle", "PVTOL" and "Quadrotor" tasks, with the results presented in Fig. 4.4. Similarly, even without good nominal models or predefined CBFs, NLBAC with neural barrier certificate achieves satisfactory performance compared to other baselines. However, since the neural barrier certificate is trained jointly with the control policy from scratch, while the fixed predefined CBFs may be well-defined, more safety violations may be observed at the beginning of the training process if a neural barrier certificate is used.

To better show the comparisons between different algorithms, here we present some statistics in Table. 4.1. These statistics are obtained by calculating the average value of the cumulative rewards and cumulative safety costs of the last 50 episodes run by each algorithm, and the result shows that our method achieves the highest average cumulative rewards, and zero or near-zero average cumulative safety costs.

To provide a clearer understanding of the computational complexity of the proposed algorithm, we calculated the training times of NLBAC and the baseline algorithms across all four tasks, ensuring that each algorithm was trained for the same maximum number of episodes. For

Table 4.1: Average cumulative reward and average cumulative safety cost on tasks.

Environments	NLBAC (ours)		SAC-RCBF		LAC		MBPPOL		CPO		PPO-Lagrangian		TRPO-Lagrangian	
	ACR	ACSC	ACR	ACSC	ACR	ACSC	ACR	ACSC	ACR	ACSC	ACR	ACSC	ACR	ACSC
Unicycle	**692.59** (predefined CBFs) **663.71** (learned barrier certificate)	**0.00** 0.51	489.80	**0.00**	-261.06	53.99	-198574.01	6.61	221.08	4.99	-17658.89	3.48	-48140.71	15.72
Car Following	**185.31** (predefined CBFs)	**0.00**	155.26	**0.00**	38.21	137.52	-0.22	295.46	1.76	63.21	13.50	233.39	1.39	140.53
PVTOL	**1498.47** (predefined CBFs) **1498.60** (learned barrier certificate)	**0.66** **0.28**	82.31	4.54	-403.53	259310.51	-27.09	6309.82	-1286.91	921072.58	-800.51	508446.27	-1529.78	939415.86
Quadrotor	**259.34** (learned barrier certificate)	**0.09**	N.A.	N.A.	109.29	1624.35	90.08	1835.45	239.28	207.69	60.74	2083.70	239.70	204.88

*ACR and ACSC mean Average Cumulative Reward and Average Cumulative Safety Cost, respectively. The best results have been marked in bold.

example, in Unicycle, Car Following, and PVTOL tasks, every algorithm was trained for 200 episodes, while in the Quadrotor task, each algorithm was trained for 1100 episodes. These statistics are summarized in Table 4.2. As can be seen, NLBAC consistently demonstrates a

Table 4.2: Running times of different algorithms on tasks.

Environments	NLBAC	SAC-RCBF	LAC	MBPPOL	CPO	PPO-Lagrangian	TRPO-Lagrangian
Unicycle (200 episodes)	2h3min (predefined CBFs) 2h58min (learned barrier certificate)	9h43min	1h45min	3h1min	17min49s	20min50s	16min56s
Car Following (200 episodes)	2h43min (predefined CBFs)	5h30min	49min	1h13min	6min32s	5min50s	6min49s
PVTOL (200 episodes)	2h52min (predefined CBFs) 2h16min (learned barrier certificate)	26h53min	2h35min	1h54min	20min17s	19min34s	21min58s
Quadrotor (1100 episodes)	40min8s (learned barrier certificate)	N.A.	20min20s	2h18min	6min11s	10min11s	6min31s

competitive balance between computational efficiency and performance. Specifically, compared to SAC-RCBF, NLBAC achieves significantly shorter training times across all tasks, reflecting its superior efficiency. This is especially notable in tasks like PVTOL, where SAC-RCBF required over 26 hours, while NLBAC completed training in less than 3 hours. This is possibly due to the computational overhead of its differentiable QP layer for safety. While model-free baselines (e.g., PPO-Lagrangian and CPO) require less training time with the same maximum number of episodes, they generally underperform in cumulative rewards and safety costs. NLBAC also achieves better safety and performance metrics than MBPPOL and LAC, despite requiring comparable or slightly longer training times, which justifies its computational cost.

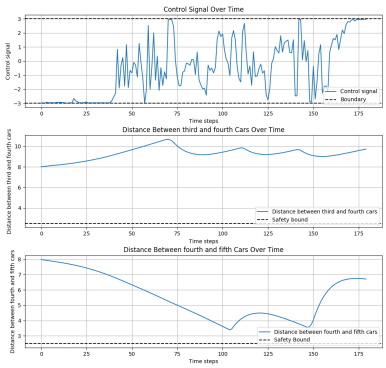


Figure 4.5: Control signals generated during deployment for the "Car Following" task.

Next, the control signals generated by the controller trained with NLBAC are plotted to

demonstrate its effectiveness. For this purpose, we chose the Car Following task. As shown in Fig. 4.5, the applied control signal is always within the allowed boundary of -3 to 3. The distance between the third and fourth cars is roughly maintained within 9 to 10, which is the desired region, while the distances between the third and fourth cars and the fourth and fifth cars remain greater than the minimum safety threshold of 2.5. These results demonstrate the algorithm's capacity to effectively maintain both the desired inter-vehicle distances and safety constraints during deployment.

Sensitivity Analysis. Robustness is key for the successful application of RL algorithms to uncertain safety-critical tasks. Here, we evaluate the robustness of control policies after being trained with NLBAC in the presence of external disturbance and parameter uncertainties. The sensitivity analysis was conducted on two tasks: Unicycle and PVTOL. The following modifications were made to the original settings introduced earlier:

1. Unicycle: a disturbance term $u_{d,t_k} = -\alpha[\cos(\theta_{t_k}), 0]^T$ is applied. The disturbance coefficient α is systematically varied from 0 to 0.9 in increments of 0.1.

2. PVTOL: Horizontal wind disturbances are introduced into the system dynamics in the form of a force term F_x, with increments from 0 to 0.9 in steps of 0.1.

For parameter uncertainties:

1. Unicycle: a scaling factor β was applied to the linear velocity terms in the dynamics, modifying the equations to:

$$\dot{x}_{t_k} = \begin{bmatrix} \beta\cos(\theta_{t_k}) & 0 \\ \beta\sin(\theta_{t_k}) & 0 \\ 0 & 1.0 \end{bmatrix} (u_{t_k} + u_{d,t_k}),$$

   where $u_{dt_k} = -0.1[\cos(\theta_{t_k}), 0]^T$. The scaling factor β is varied from 0.6 to 1.5 in increments of 0.1.

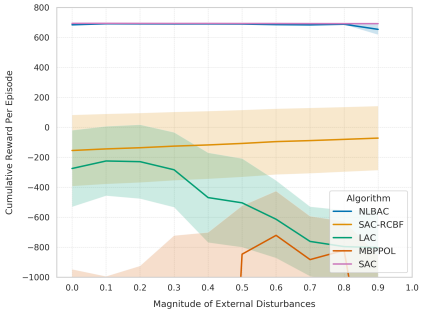
2. PVTOL: Parameter uncertainty is introduced by applying a scaling factor δ to the angular velocity term in the dynamics. The modified control input matrix for angular velocity

was scaled by δ:

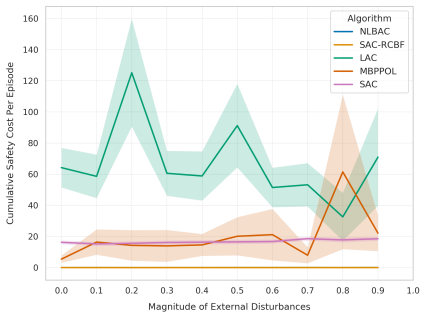
$$\dot{x}_{t_k} = \begin{bmatrix} v_{1t_k} \\ v_{2t_k} \\ 0 \\ -\sin(\theta_{t_k}) \times f_{t_k} \\ \cos(\theta_{t_k}) \times f_{t_k} - 1.0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & \delta \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} u_{t_k}.$$

δ is incremented from 0.6 to 1.5 in steps of 0.1, and the system's performance was assessed accordingly.

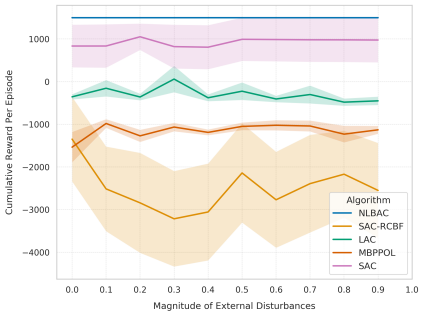
The performance of the proposed NLBAC algorithm under external disturbances and parameter uncertainties during deployment after training is presented in Figures 4.6 and 4.7, respectively. We include SAC as an additional baseline for comparison.



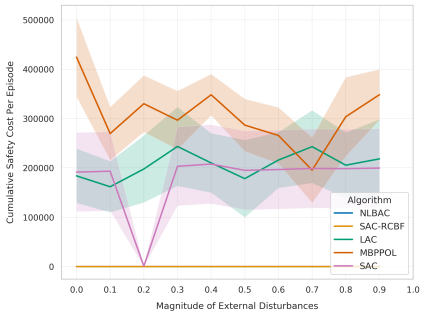
(a) Unicycle: Cumulative rewards



(b) Unicycle: Cumulative safety costs



(c) PVTOL: Cumulative rewards

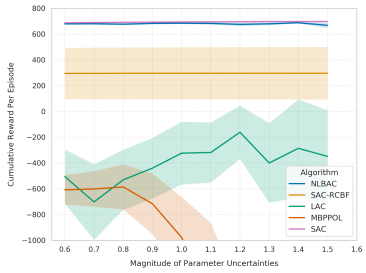


(d) PVTOL: Cumulative safety costs

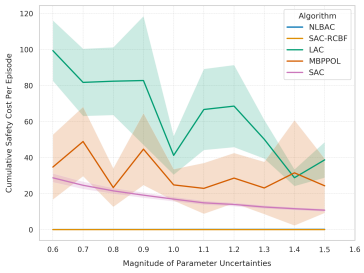
Figure 4.6: The effects of external disturbances during deployment on the cumulative reward and cumulative safety cost in the "Unicycle" and "PVTOL" tasks. In the cumulative safety cost plots (right column), the curve representing NLBAC overlaps with that of SAC-RCBF, making only the SAC-RCBF curve visible. In subfigure (d), an upper limit has been applied to standard errors to enhance clarity by reducing excessively large shaded areas.

As shown in Fig. 4.6, during deployment after the training process is complete, the proposed algorithm NLBAC demonstrates robust performance under external disturbances of varying magnitudes. Both cumulative rewards and safety costs of the NLBAC algorithm remain at superior levels compared to other baselines on both tasks, achieving high cumulative rewards while maintaining zero or near-zero cumulative safety violations (the curve for cumulative safety costs of NLBAC overlaps with the curve of SAC-RCBF). Importantly, NLBAC achieves these results without significant fluctuations or performance deterioration.

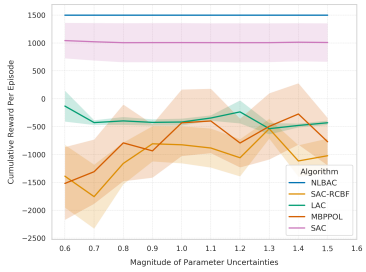
SAC achieves good cumulative rewards but fails to maintain low cumulative safety costs, rendering it less suitable for safety-critical applications. On the other hand, SAC-RCBF demonstrates robust safety performance during deployment under various external disturbances, maintaining near-zero safety violations. However, it cannot achieve high cumulative rewards, even with prior knowledge of the system dynamics.



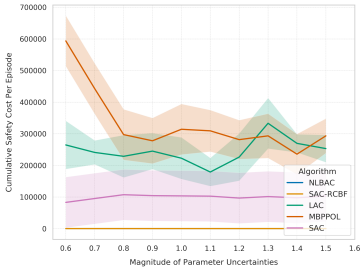
(a) Unicycle: Cumulative rewards



(b) Unicycle: Cumulative safety costs



(c) PVTOL: Cumulative rewards



(d) PVTOL: Cumulative safety costs

Figure 4.7: The effects of parameter uncertainties during deployment on the cumulative reward and cumulative safety cost in the "Unicycle" and "PVTOL" tasks. In the cumulative safety cost plots (right column), the curve representing NLBAC overlaps with that of SAC-RCBF, making only the SAC-RCBF curve visible. In subfigure (d), an upper limit has been applied to standard errors to enhance clarity by reducing excessively large shaded areas.

Similar to the analysis of external disturbances, as shown in Fig. 4.7, during deployment,

when parameter updates are finalized, NLBAC demonstrates robust performance under parameter uncertainties of varying magnitude. Both the cumulative rewards and safety costs of the NLBAC algorithm remain at favourable levels compared to other baselines. Notably, in the Unicycle task, the cumulative rewards obtained by controllers trained with NLBAC show some noticeable degradation. However, overall, NLBAC achieves strong performance without significant fluctuations, demonstrating robustness to parameter uncertainties during deployment.

While SAC achieves high cumulative rewards, it fails to maintain low cumulative safety costs. Furthermore, MBPPOL exhibits significant performance variation under different system parameters, showing its sensitivity to parameter uncertainties.

These results demonstrate the ability of NLBAC to handle external disturbances and parameter uncertainties during deployment while achieving reliable performance in terms of both cumulative rewards and cumulative safety costs.

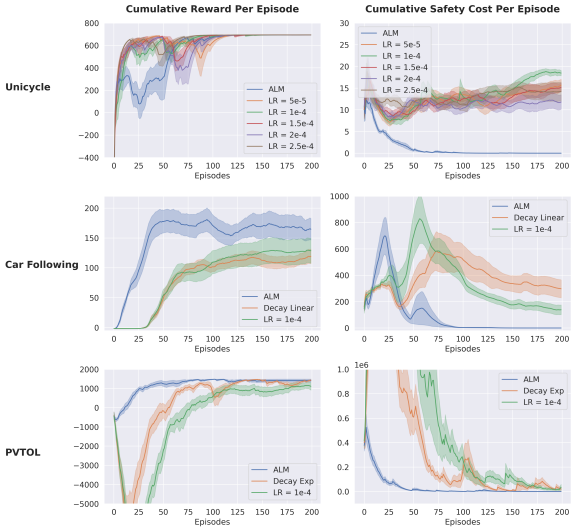


Figure 4.8: Results of the ablation study. Each curve shows the average across fifteen experiments using different random seeds.

Ablation Studies We also compare results obtained by our NLBAC algorithm with those obtained by using the commonly-used dual ascent update. All other parameters remain consistent for comparison, and the results are depicted in Fig. 4.8. In the "unicycle" environment, five different fixed learning rates are used. In the "Car Following" and "PVTOL" environments, linear decay and exponential decay are used respectively to dynamically adjust the learning rates for more extensive comparisons. The results indicate that the dual ascent update results in

much more safety violations compared to ALM, which eventually achieves high rewards, while introducing the least safety violations throughout most of the training process. This suggests that more sophisticated parameter tuning for the learning rate, including the selection of initial values and decay types, and perhaps other hyperparameters, may be necessary when utilizing the dual ascent update, whereas ALM demonstrates superior performance.

4.7 Conclusions

This chapter presents the Neural ordinary differential equations–based Lyapunov–Barrier Actor–Critic (NLBAC) algorithm, which considers gray-box settings by leveraging the augmented Lagrangian method and Neural ODEs. The proposed algorithm supports tasks with either predefined control barrier functions or learned barrier certificates, and addresses the issue of infeasibility arising from the simultaneous enforcement of multiple state-wise constraints. Experimental results demonstrate that NLBAC achieves higher cumulative rewards, fewer safety violations, and improved sample efficiency compared to baseline approaches.

Chapters 3 and 4 focus on model-based settings, where a system model, either a nominal model known prior to learning and refined using data, or a model learned entirely from data, is required. Moreover, both chapters concentrate on single-agent cases without considering multi-agent systems. Developing state-wise safe and stable reinforcement learning for general multi-agent systems without utilizing system models is challenging, due to fundamental issues inherent in multi-agent reinforcement learning, such as non-stationarity, partial observability, and scalability. These challenges motivate the developments in the next chapter, where we investigate state-wise safe and stable reinforcement learning for general multi-agent systems in a fully model-free setting.

CHAPTER 5

State-Wise Safe and Stable Reinforcement Learning for General Multi-agent Settings in a Totally Model-free Manner

In the previous chapters, state-wise safety and stability were studied in single-agent white-box and gray-box settings, where system models were either known or learned from data. In this chapter, we take a further step and investigate state-wise safe and stable reinforcement learning for general multi-agent systems in a fully model-free setting, where neither system dynamics nor analytical certificates are required. This introduces new challenges inherent to multi-agent reinforcement learning, including non-stationarity and partial observability, which necessitate different algorithmic and theoretical analysis.

5.1 Introduction

Multi-agent reinforcement learning (MARL) has attracted significant attention due to its many successes in solving decision-making tasks for multi-agent systems (MAS), and researchers have started to focus on its application to real-world tasks such as autonomous vehicles [64]. Among previous studies, cooperative tasks that require agents to be controlled to accomplish complex tasks collaboratively while obeying local and/or system-level constraints have received great attention [17, 66]. To efficiently train policies to achieve the team goal under constraints, a single joint reward is typically required for each agent in the team, and many studies impose global state observability [66, 90] to enable centralized training (CT) [26, 130]. Thanks to recent studies by [67], [131] and [91], research on decentralized constrained cooperative MARL has advanced; however, less attention has been paid to general MARL tasks [70], where each agent can have its own reward instead of sharing a single joint team reward, and access to global

information may no longer hold due to the possible competition among agents, even if many real-world domains are naturally aligned with this general setting.

Among all constraints, safety has received the greatest focus in recent years [46]. However, drawing on single-agent safe RL methods, many safe MARL algorithms so far formulate the problem based on the constrained Markov decision process (CMDP) [17, 66–69] that only requires the expected cumulative discounted safety costs along trajectories to be smaller than a predefined threshold. Since real-world multi-agent safety-critical tasks usually require efficient training for zero safety violations at every step to avoid harmful consequences, more advanced methods remain in urgent demand.

Maintaining stability is also crucial in many tasks, and the Lyapunov framework has been applied [93, 94]. A small-gain theorem-based learning method has been proposed to develop stable controllers for networked systems [20]. However, this approach requires knowledge of system dynamics for gradient computation or one-step simulation, which may limit the applicability of the algorithm to tasks with sophisticated dynamics [86]. Further, when using safety and stability conditions separately, the simultaneous enforcement of both raises the challenging issue of infeasibility [111] also for multi-agent cases.

Contributions: Here we address these challenges by proposing the first framework that helps to obtain a joint controller with post-convergence state-wise safety and stability guarantees under mild assumptions in general MARL tasks with unknown dynamics. The main contributions are:

1. We develop a method to train neural barrier certificates for MAS, which is then integrated into general model-free MARL tasks with state-wise safety requirements to obtain safe and locally optimal controllers via a modified dual ascent update, whose convergence to a safe local Nash equilibrium is theoretically proved under certain assumptions.

2. Based on the concept of input-to-state stability (ISS) [132, 133], we establish the local condition for each agent to achieve stable control of general networked multi-agent systems, supported by theoretical proofs. In particular, we show that ISS can be enforced using only each agent's own state and a few scalars from others, providing a theoretical

justification for decentralized training with minimal communication. We further study the issue of infeasibility under state-wise safety and stability constraints, showing that this issue does not prevent the framework from finding safe and stable controllers for multi-agent systems.

3. We propose **M**ulti-**A**gent **S**afe and **S**table **S**oft **A**ctor-**C**ritic (**MAS^{3}AC**), a unified framework to solve model-free multi-agent safe and stable control problems under centralized or decentralized training. We also provide environments for benchmarking safe and stable MARL algorithms. Empirical evaluations demonstrate that MAS^{3}AC consistently and efficiently achieves a strong balance between reward maximization and constraint satisfaction, outperforming baselines in both cooperative and non-cooperative settings.

5.2 Problem Formulation and Preliminaries

Multi-Agent Reinforcement Learning: In this study, we consider the MARL problem represented by a tuple $\mathcal{M} = (\mathcal{N}, \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{U}}, f, \boldsymbol{R}, \boldsymbol{\mathcal{O}}, \gamma)$. Here, $\mathcal{N} = \{1, \dots, N\}$ denotes the set of $N > 1$ interacting agents, $\boldsymbol{\mathcal{X}} = (\mathcal{X}^1, \mathcal{X}^2, \dots, \mathcal{X}^N)$ and $\boldsymbol{\mathcal{U}} = (\mathcal{U}^1, \mathcal{U}^2, \dots, \mathcal{U}^N)$ denote the joint state and action spaces, with $\boldsymbol{\mathcal{O}} = (\mathcal{O}^1, \mathcal{O}^2, \dots, \mathcal{O}^N)$ where $\mathcal{O}^i$ is the local observation space of agent i. The system dynamics $f : \boldsymbol{\mathcal{X}} \times \boldsymbol{\mathcal{U}} \to \boldsymbol{\mathcal{X}}$ is assumed to be unknown. $\boldsymbol{r} = (r^1, r^2, \dots, r^N)$ represents the reward functions, with r^i being the individual reward of agent i. In this work, we consider general settings without imposing specific requirements on $r^i, \forall i \in \mathcal{N}$. $\gamma \in (0, 1)$ is the discount factor. At each timestep t, every agent i selects an action u_t^i according to its policy π^i, and the joint action $\boldsymbol{u}_t = (u_t^1, u_t^2, \dots, u_t^N)$ is generated under the joint policy $\boldsymbol{\pi} = \prod_{i=1}^N \pi^i \in \boldsymbol{\Pi}$, where $\boldsymbol{\Pi}$ denotes the set of all joint policies. $\boldsymbol{\pi}^{-i}$ denotes the joint policy of all agents except i. When global information is available, the joint state $\boldsymbol{x}_t = (x_t^1, x_t^2, \dots, x_t^N)$ serves as the input to every π^i. For tasks where only local observations are available, we consider cases in which interconnections among agents are represented by a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where each node corresponds to an agent. A directed edge $(j, i) \in \mathcal{E}$ indicates that agent j directly influences agent i through a class-$\mathcal{K}$ gain function $\delta_{ij} \in \mathcal{K}$; otherwise, no direct influence exists from agent j to agent i, and $\delta_{ij} = 0$. In this setting, the local observation of agent i at timestep t, denoted o_t^i, is used as the input to its individual policy π^i.

Safety and Stability: For each agent $i \in \mathcal{N}$, when global information is available [1], the state-wise safety requirement is $b^i(\boldsymbol{x}_t) \geq 0, \forall t \geq 0$, where $b^i(\boldsymbol{x}_t)$ is a barrier signal measuring the safety of agent i at the state $\boldsymbol{x}_t$. Based on this notation, the following subspaces are defined.

Definition 5.1. *The overall state space $\boldsymbol{\mathcal{X}}$ can be partitioned into: (1). $\boldsymbol{\mathcal{X}}_{vio}$: The set of states $\boldsymbol{x}_t$ where $\exists i \in \mathcal{N}$ such that $b^i(\boldsymbol{x}_t) < 0$. (2). $\boldsymbol{\mathcal{X}}_{irrec}$: The set of states $\boldsymbol{x}_0 \notin \boldsymbol{\mathcal{X}}_{vio}$, but under any joint controller $\boldsymbol{\pi} \in \boldsymbol{\Pi}$, the trajectory starting from $\boldsymbol{x}_0$ will enter $\boldsymbol{\mathcal{X}}_{vio}$ at least once with a positive probability, i.e., $\forall \boldsymbol{\pi} \in \boldsymbol{\Pi}$, $\exists t > 0$ such that $Pr(\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}_{vio}) = Pr(\exists i \in \mathcal{N}, b^i(\boldsymbol{x}_t) < 0) > 0$. (3). $\boldsymbol{\mathcal{X}}_{unsafe} = \boldsymbol{\mathcal{X}}_{vio} \cup \boldsymbol{\mathcal{X}}_{irrec}$ is the unsafe set, and $\boldsymbol{\mathcal{X}}_{safe} = \boldsymbol{\mathcal{X}} \setminus \boldsymbol{\mathcal{X}}_{unsafe}$ is the safe set.*

Here, we consider tasks where there exist control policies under which trajectories starting from any initial state do not enter $\boldsymbol{\mathcal{X}}_{\text{vio}}$ with a positive probability. Otherwise, no valid controller can maintain safety for the MAS. In addition to safety, ensuring stability for networked MAS is also critical, as it indicates convergence of the controlled MAS to the desired equilibrium $\boldsymbol{x}_{\text{desired}}$, thereby enabling successful completion of tasks like reach-and-avoid. Without loss of generality, we set $\boldsymbol{x}_{\text{desired}} = \mathbf{0}$, the zero vector with the same dimension as $\boldsymbol{x}$, since a simple translation can be applied whenever an equilibrium is specified for a given task. Then, for stabilization tasks, the condition $\lim_{t\to\infty} ||\boldsymbol{x}_t|| = 0$ should hold to ensure stability of the networked MAS. Based on all these definitions, here we define:

Problem 5.1 (Stable and Safe Control Problem for MAS)**.** *For a MAS represented by $\mathcal{M}$, given a desired equilibrium $\boldsymbol{x}_{desired} \in \boldsymbol{\mathcal{X}}_{safe}$ and an initial set $\boldsymbol{\mathcal{X}}_0 \subseteq \boldsymbol{\mathcal{X}}_{safe}$, find a joint policy $\boldsymbol{\pi} \in \boldsymbol{\Pi}$ producing a sequence $\{\boldsymbol{u}_t\}_{t\geq 0}$ such that the state sequence $\{\boldsymbol{x}_t\}_{t\geq 0}$ satisfies: (1). **Safety**: $\forall t \geq 0$, $Pr(\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}_{vio}) = Pr(\exists i \in \mathcal{N}, b^i(\boldsymbol{x}_t) < 0) = 0$. (2). **Stability**: $\lim_{t\to\infty} ||\boldsymbol{x}_t - \boldsymbol{x}_{desired}|| = 0$. Here, $\boldsymbol{\mathcal{X}}_{vio}$ and $\boldsymbol{\mathcal{X}}_{safe}$ are as defined in Definition 5.1, and $b^i(\boldsymbol{x}_t)$ is the barrier signal of agent i at $\boldsymbol{x}_t$.*

[1] For convenience and clarity, we present the algorithm design and analysis for safety under global information. In practice, the algorithm can also be applied to tasks with only local observations, such as inter-agent collision avoidance, where observing neighbors suffices to maintain safety. Related analyses can be found in [67, 91, 131]. For stability, local observations are used, and theoretical analysis and empirical results under this setting are reported in Section 5.3.2 and Section 5.4, respectively.

5.3 Methodology

5.3.1 State-Wise Safety via Neural Barrier Certificates for General MARL

In this section, we investigate the state-wise safety for MAS. Since it is natural to update each individual policy π^i to satisfy the state-wise safety requirement given by b^i, inspired by CBFs, we aim to construct a learned barrier certificate for each agent i, and update agent i's policy jointly with this learned barrier certificate. We begin with the following definitions and lemma:

Definition 5.2. *For the update of agent i, similar to Definition 5.1, the overall state space can be divided into the following subspaces: (1). $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{vio}$: The set of states $\boldsymbol{x}_t$ where $b^i(\boldsymbol{x}_t) < 0$. (2). $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{irrec}$: The set of states $\boldsymbol{x}_0$ where $b^i(\boldsymbol{x}_0) \geq 0$, but from which the trajectory will reach a state with $b^i(\boldsymbol{x}_t) < 0$ at least once with a positive probability under the current joint policy $\boldsymbol{\pi}$, i.e., $\exists t > 0$ such that $Pr(b^i(\boldsymbol{x}_t) < 0) > 0$ under the controller π^i when all other agents follow $\boldsymbol{\pi}^{-i}$. (3). $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{unsafe} = \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{vio} \cup \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{irrec}$ is the unsafe set, and $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{safe} = \boldsymbol{\mathcal{X}} \backslash \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{unsafe}$ is the safe set for agent i. i.e., for every $\boldsymbol{x}_0 \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{safe}$, under the joint policy $\boldsymbol{\pi}$, $Pr(b^i(\boldsymbol{x}_t) < 0) = 0$ for all $t \geq 0$, meaning the safety of agent i is maintained under π^i when other agents follow $\boldsymbol{\pi}^{-i}$.*

Definition 5.3. *$H^i_{\boldsymbol{\pi}}$ is a barrier certificate for the agent i if: (1). $\forall \boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{unsafe}$, $H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t) < 0$. (2). $\forall \boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{safe}$, $H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t) \geq 0$. (3). $\forall \boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{safe}$, the certificate value satisfies $H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_{t+1}) - H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t) \geq -\epsilon H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t)$ where $\epsilon \in (0, 1)$ and $\boldsymbol{x}_{t+1}$ is the next state induced by $\boldsymbol{\pi}$.*

Informal Lemma 5.1. *Define the barrier signal b^i as:*

$$b^i(\boldsymbol{x}_t) = \begin{cases} d & \text{if } \boldsymbol{x}_t \notin \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{vio}, \\ D & \text{if } \boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{vio}, \end{cases} \tag{5.1}$$

where $d = 0$ and $D < 0$ are constants. Under certain assumption, the function $H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t) = \mathbb{E}_{\tau \sim \boldsymbol{\pi}}\big[\sum_{k=0}^{\infty} \gamma^k b^i(\boldsymbol{x}_{t+k})\big]$, where $\gamma \in (0, 1)$ is the discount factor and $\tau = \{\boldsymbol{x}_t, \boldsymbol{x}_{t+1}, \boldsymbol{x}_{t+2}, \dots\}$ is a trajectory under policy $\boldsymbol{\pi}$ starting from $\boldsymbol{x}_t$, is a barrier certificate for the agent i, given that all other agents follow $\boldsymbol{\pi}^{-i}$.

The formal version and proof are provided in Appendix 5.6.3 (i). This result implies that the value function of b^i is a barrier certificate for agent i. Consider the update process where, without loss of generality, agents are updated sequentially from 1 to N, then, with its action-

value function $Q^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t, \boldsymbol{u}_t) = \mathbb{E}_{\tau\sim\boldsymbol{\pi}}\big[\sum_{k=0}^{\infty}\gamma^k r^i(\boldsymbol{x}_{t+k}, \boldsymbol{u}_{t+k})\big]$, and barrier certificate $H^i_{\boldsymbol{\pi}}$, the optimization for agent i for all $\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}$ is formulated as:

$$\pi^i_{e+1} = \arg\min_{\pi^i} \quad -Q^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, (\boldsymbol{\pi}^{1:i-1}_{e+1}(\boldsymbol{x}_t), \pi^i(\boldsymbol{x}_t), \boldsymbol{\pi}^{i+1:N}_{e}(\boldsymbol{x}_t))) \tag{5.2a}$$

$$\text{s.t.} \quad -H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_{t+1}) + H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t) - \epsilon H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t) \leq 0 \tag{5.2b}$$

where $\hat{\boldsymbol{\pi}}_i = (\boldsymbol{\pi}^{1:i-1}_{e+1}, \boldsymbol{\pi}^{i:N}_{e})$, $Q^i_{\hat{\boldsymbol{\pi}}_i}$ and $H^i_{\hat{\boldsymbol{\pi}}_i}$ are the action-value function and barrier certificate of agent i under the joint policy $\hat{\boldsymbol{\pi}}_i$, $\boldsymbol{x}_{t+1}$ is the next state following $(\boldsymbol{\pi}^{1:i}_{e+1}, \boldsymbol{\pi}^{i+1:N}_{e})$, and e is the number of updates. We now present a proposition formalizing a key desirable property of the update:

Proposition 5.1. *For each $i \in \mathcal{N}$, suppose that all other agents follow the fixed policies $\boldsymbol{\pi}^{1:i-1}_{e+1}$ and $\boldsymbol{\pi}^{i+1:N}_{e}$, then $\mathcal{X}^{(\boldsymbol{\pi}^{1:i-1}_{e+1}, \boldsymbol{\pi}^{i:N}_{e}), i}_{safe} \subseteq \mathcal{X}^{(\boldsymbol{\pi}^{1:i}_{e+1}, \boldsymbol{\pi}^{i+1:N}_{e}), i}_{safe}$, namely, the safe set for agent i will not shrink when π^i_{e+1} is obtained as the solution to Problem (5.2).*

The proof is provided in Appendix 5.6.3 (ii). This proposition characterizes an idealized theoretical property under the conditions that all other agents follow their certain policies and that agent i's action-value function and barrier certificate are exact. Under these conditions, by solving Problem (5.2), the region in which safety is maintained for agent i expands or at least remains unchanged, which is desirable for safe control tasks. This good property serves as a theoretical insight that motivates the more practical algorithm design detailed as follows:

Proposition 5.1, although desirable, requires **ideal** conditions that:

1. Agent i's action-value function and barrier certificate are perfect and precise when being used for updating the policy π^i.

2. Other agents' policies, $\boldsymbol{\pi}^{-i}$, are fixed when π^i is being updated.

However, under model-free actor-critic RL settings, action-value functions and barrier certificates are approximated by neural networks rather than being perfect all the time, and requiring other agents to follow certain joint policies $\boldsymbol{\pi}^{-i}$ when updating π^i naturally indicates a "multi-loop" update manner, where the next agent's update starts after the previous agent' s policy converges. This multi-loop update method would be a practical bottleneck that limits the ap-

plicability due to its practical nested design, in which certain agents cannot update at all even if the environment evolves. The necessity of dealing with these two practical difficulties to bridge the gaps and make the algorithm less restricted by ideal conditions, suggested by Proposition 5.1, while still able to lead to joint policy with good safety performance, motivates the multi-timescale approach (see Assumption 5.2 in Appendix 5.6.2 for details) to design an iterative process which can be theoretically shown to converge to a feasible local Nash equilibrium.

Definition 5.4. *With $Q^i_{\boldsymbol{\pi}}$ denoting the action-value function of agent i, and $b^i(\boldsymbol{x}_t)$ denoting the state-wise safety for agent i, $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$ is a local Nash equilibrium [134] if for each $i \in \mathcal{N}$, there exists a neighborhood $\mathcal{W}^i \subseteq \Pi^i$ of π_i^* such that $Q^i_{(\pi_i^*, \boldsymbol{\pi}^*_{-i})}(\boldsymbol{x}, \boldsymbol{u}) \geq Q^i_{(\pi^i, \boldsymbol{\pi}^*_{-i})}(\boldsymbol{x}, \boldsymbol{u}), \quad \forall \pi^i \in \mathcal{W}^i$. If, in addition, for all $i \in \mathcal{N}$, the joint policy $\boldsymbol{\pi}^*$ satisfies the safety constraint $Pr(b^i(\boldsymbol{x}_t) < 0) = 0, \ \forall t \geq 0$, then it is called a **feasible local Nash equilibrium**.*

Since the actor-critic framework is usually applied for model-free RL training, the barrier certificate can thus be approximated by an action-value function $H_{\omega_i}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ in practice. Accordingly, we define two loss functions, J_{Q^i} and J_{H^i}, for updating the action-value function and barrier certificate of agent i, respectively, with their explicit forms provided in Appendix 5.6.1. In a general case, for each $i \in \mathcal{N}$, we augment the Lagrangian function with additional squared terms, and sample data and $\tilde{\boldsymbol{u}}_{t+1}$ according to a joint policy $\bar{\boldsymbol{\pi}}_i = (\pi^i, \boldsymbol{\pi}^{-i})$ where π^i is the policy of agent i to be updated and $\boldsymbol{\pi}^{-i}$ is a certain joint policy for all other agents. If π^i is parameterized by θ_i and $\boldsymbol{\pi}^{-i}$ by $\boldsymbol{\theta}_{-i}$, then, the modified Lagrangian function for agent i is:

$$\mathcal{L}^i_{\boldsymbol{\theta}^{-i}}(\theta_i, \lambda_i) = \mathbb{E}_{\bar{\boldsymbol{\pi}}_i}\big[- Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) + \lambda_i \sum_{t=0}^{\infty} \gamma^t \max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) + H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t)\} + \frac{\lambda_i}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) + H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t)\}]^2\big], \tag{5.3}$$

where λ_i is the Lagrangian multiplier for agent i. Compared to the standard Lagrangian commonly used for traditional dual ascent updates, here we add extra squared terms to more strongly penalize safety violations, thereby improving safe policy learning. Moreover, the constraint is formulated using the difference between consecutive barrier certificates, rather than relying on

a single critic function to impose the safety constraint as in the CMDP setting. Finally, we denote J_{θ_i} and J_{λ_i} as loss functions to update θ_i and λ_i, respectively, with their detailed forms in Appendix 5.6.1. With these loss functions, we provide Algorithm 5.1, together with the following theorem:

Algorithm 5.1 State-Wise Safe MARL via Neural Barrier Certificates and Multi-Timescales

Input: Learning rates $\eta_1(e)$, $\eta_{2,i}(e)$, and $\eta_{3,i}(e)$. Initialized parameters $\{\phi_i^0\}_{i\in\mathcal{N}}$, $\{\omega_i^0\}_{i\in\mathcal{N}}$, $\{\theta_i^0\}_{i\in\mathcal{N}}$, $\{\lambda_i^0\}_{i\in\mathcal{N}}$ for corresponding neural networks and parameters.

1: **for** $e = 0, 1, \ldots$ **do**
2: **for** $i = 1, 2, \ldots, N$ **do**
3: $\phi_i^{e+1} \leftarrow \phi_i^e - \eta_1(e)\nabla_{\phi_i} J_{Q^i}(Q^i)$, $\omega_i^{e+1} \leftarrow \omega_i^e - \eta_1(e)\nabla_{\omega_i} J_{H^i}(H^i)$.
4: $\theta_i^{e+1} \leftarrow \theta_i^e - \eta_{2,i}(e)\nabla_{\theta_i} J_{\theta_i}(\theta_i)$.
5: $\lambda_i^{e+1} \leftarrow [\lambda_i^e + \eta_{3,i}(e)\nabla_{\lambda_i} J_{\lambda_i}(\lambda_i)]_+$.
6: **end for**
7: **end for**

Output: $\{\theta_i\}_{i\in\mathcal{N}}$

Informal Theorem 5.1. *Under certain assumptions, the joint policy updated in Algorithm 5.1 will almost surely converge to a feasible local Nash equilibrium $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$, where $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) \geq 0$ for all $(\boldsymbol{x}_t, \boldsymbol{u}_t)$ along the trajectory under $\boldsymbol{\pi}^*$, for every $i \in \mathcal{N}$ and $t \geq 0$. Further, define $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) := \min_{i\in\mathcal{N}} H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$, then $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ is a global barrier certificate for the MAS.*

Remark 5.1. *The multi-timescale setting is applied to diminish the approximation error in the learned value functions and certificates asymptotically as the number of iterations increases. Therefore, Theorem 5.1 provides a post-convergence guarantee applying to the limiting joint policy obtained after convergence of the learning process through training in an asymptotic manner, rather than to transient policies obtained during training (especially policies encountered at the beginning of training) and thus different from Proposition 5.1. This theorem indicates that the converged joint policy satisfies the constraints, and in practical implementation, training is run for a large number of (at least millions of) updates until the reward and safety violation metrics reach steady and better values compared to baselines, as detailed in Section 5.4. This practical procedure serves as a finite-training approximation to the asymptotic convergence regime considered in the theorem.*

The formal version and proof are provided in Appendix 5.6.3 (iii). This theorem applies

to general MARL settings beyond purely cooperative tasks, and while the joint policy only converges to a feasible *local* Nash equilibrium, different initializations may lead to different local solutions; in practice, one can select solutions that yield better task performance. Hence, the proposed framework offers a principled way to obtain safe controllers.

5.3.2 State-Wise Stability via ISS and Infeasibility under Multiple Constraints

In the previous section, we investigated the problem of ensuring state-wise safety for MAS. We now turn to stability, as shown in Problem 5.1, using local observations, since, unlike safe MARL with local observations, which has been well studied previously [67, 91, 131], stability in MARL under local observations remains relatively underexplored. Moreover, for large-scale control problems, decentralized training with local information is generally more favorable due to the curse of dimensionality and the inability to obtain global information. Based on the graph structure $\mathcal{G}$, the MDP is accordingly partitioned into N subsystems, each corresponding to one agent. For each agent i, a cost signal is defined as $c^i(x_t^i) = ||x_t^i - x_{\text{desired}}^i||$, where, still without loss of generality, we set the desired equilibrium x_{desired}^i to be $\mathbf{0}$ for the subsequent analysis. The value function associated with this cost is given by $V_{\boldsymbol{\pi}}^i(x_t^i) = \mathbb{E}_{\tau^i \sim \boldsymbol{\pi}}\big[\sum_{k=0}^{\infty} \gamma^k c^i(x_{t+k}^i)\big]$, where $\tau^i = \{x_t^i, x_{t+1}^i, x_{t+2}^i \ldots\}$ is the trajectory of agent i under $\boldsymbol{\pi}$ starting from x_t^i [2]. With $\delta_{ij} \in \mathcal{K} \cup \{0\}, \forall i, j \in \mathcal{N}$ denoting the gain functions representing inter-agent couplings, and $\kappa_i(||\boldsymbol{d}_t||)$, where $\kappa_i \in \mathcal{K} \cup \{0\}$, denote the influence of exogenous signal $\boldsymbol{d}_t$ on agent i, we propose the following theorem by using the concept of input-to-state stability (ISS) [132]:

Theorem 5.2. *Suppose for a set of $\mathcal{K}_\infty$ functions $\{\beta_i\}_{i\in\mathcal{N}}$, there exists a set of $\mathcal{K}_\infty$ functions $\{\alpha_i\}_{i\in\mathcal{N}}$ and $\{\mu_i\}_{i\in\mathcal{N}}$ satisfying $\sum_{j\neq i}\delta_{ij}(V^j(x_t^j)) + \beta_i(\sum_{j\neq i}\delta_{ij}(V^j(x_t^j))) \leq \alpha_i(V^i(x_t^i))$, and $V^i(x_t^i) \leq \mu_i(||x_t^i||)$, $\forall i \in \mathcal{N}$, for each state sampled under certain joint policy $\boldsymbol{\pi}$. Then, if*

$$\dot{V}^i(x_t^i) \leq -\alpha_i(V^i(x_t^i)) + \sum_{j\neq i}\delta_{ij}(V^j(x_t^j)) + \kappa_i(||\boldsymbol{d}_t||), \forall i \in \mathcal{N}, \tag{5.4}$$

also holds for each state sampled under $\boldsymbol{\pi}$, we have $||\boldsymbol{x}_t|| \leq \xi(||\boldsymbol{x}_0||, t) + \rho(||\boldsymbol{d}||_\infty)$. Here $\xi \in \mathcal{KL}$ is a function such that $\xi(\cdot, t) \in \mathcal{K}$ for all $t \geq 0$, and $\xi(r, \cdot) \in \mathcal{L}$ for all $r > 0$, where $\mathcal{L}$ is the set of strictly decreasing functions with $\lim_{t\to\infty}\xi(||\boldsymbol{x}_0||, t) = 0$. $\rho \in \mathcal{K}$ and $||\boldsymbol{d}||_\infty = sup\{||\boldsymbol{d}_t||, t \geq 0\}$.

[2] For simplicity, we later use V^i to denote $V_{\boldsymbol{\pi}}^i$ when the dependence on $\boldsymbol{\pi}$ is clear from context.

The proof is provided in Appendix 5.6.4 (i). The above theorem indicates that if $\{\alpha_i\}_{i\in\mathcal{N}}$ are chosen sufficiently large, then under other mild conditions, the networked multi-agent system is input-to-state stable when (5.4) is satisfied. For simplicity and ease of implementation, here we assume a fixed list of appropriate gain functions δ_{ij} describing the coupling between agent i and j, based on which there exist $\{\alpha_i\}_{i\in\mathcal{N}}$ that are large enough such that $\sum_{j\neq i}\delta_{ij}(V^j(x_t^j))+\beta_i(\sum_{j\neq i}\delta_{ij}(V^j(x_t^j)))\leq\alpha_i(V^i(x_t^i))$ under joint policy $\boldsymbol{\pi}\in\boldsymbol{\Pi}$. Suppose there exists another set of functions $\{\alpha_i'\}_{i\in\mathcal{N}}$ satisfying $(\alpha_i-\alpha_i')(V^i(x_t^i))\leq\sum_{j\neq i}\delta_{ij}(V^j(x_t^j))\leq\sum_{j\neq i}\delta_{ij}(V^j(x_t^j))+\beta_i(\sum_{j\neq i}\delta_{ij}(V^j(x_t^j)))\leq\alpha_i(V^i(x_t^i))$, for all $i\in\mathcal{N}$ and $\boldsymbol{x}_t$ sampled under $\boldsymbol{\pi}$, then, the condition (5.4) can be further strengthened to $\dot{V}^i(x_t^i)\leq-\alpha_i'(V^i(x_t^i))+\kappa_i(||\boldsymbol{d}_t||),\forall i\in\mathcal{N}$, since $-\alpha_i'(V^i(x_t^i))\leq-\alpha_i(V^i(x_t^i))+\sum_{j\neq i}\delta_{ij}(V^j(x_t^j))\leq-\beta_i(\sum_{j\neq i}\delta_{ij}(V^j(x_t^j)))$. In practice, $\{\alpha_i'\}_{i\in\mathcal{N}}$ are treated as tunable hyperparameters, and Theorem 5.2 provides guidance for this tuning in a MAS: for agents that exhibit strong couplings with other agents via physical properties or communication designs, the corresponding α_i' should be chosen large to counteract the influence from other agents and realize dissipation. Notably, it is possible to have the further extension where class-$\mathcal{K}$ functions are jointly trained together with the policy $\boldsymbol{\pi}$, rather than being predefined and fixed. In this way, some conditions in Theorem 5.2 need not be assumed to hold, but can instead be enforced as constraints during training. This extension is investigated in the next chapter.

Remark 5.2. *Theorem 5.2 shows that each agent can help enforce ISS of the MAS by using only its own state and a few exchanged scalar quantities, without requiring complete states of other agents. This provides a theoretical foundation and justification for designing decentralized training schemes with minimal communication, reducing both communication and computation burdens. More broadly, stability can serve as an implicit coordination mechanism: when rewards are coupled with convergence toward the desired equilibrium, stability directly improves returns by preventing significant divergence, therefore mitigating severe performance deterioration during learning and alleviating the conservatism introduced by safety constraints, if present. This insight opens new possibilities for scalable MARL algorithms that rely on limited information exchange [135].*

Since the control signal is generated by certain joint policy $\boldsymbol{\pi}$ rather than by any exogenous

generator during training, and the control policy takes the state as input, the networked system can be regarded as autonomous, and the term $\kappa_i(||\boldsymbol{d}_t||)$ can be discarded when other external disturbances are negligible. By replacing the derivative with the difference between consecutive steps, and extending Problem 5.2 to include stability constraints, we obtain the optimization problem for agent i:

$$\pi^i_{e+1} = \arg\min_{\pi^i} \quad -Q^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, (\boldsymbol{\pi}^{1:i-1}_{e+1}(\boldsymbol{x}_t), \pi^i(\boldsymbol{x}_t), \boldsymbol{\pi}^{i+1:N}_{e}(\boldsymbol{x}_t))) \tag{5.5a}$$

$$s.t. - H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) + H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) - \epsilon H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) \leq 0 \tag{5.5b}$$

$$V^i_{\hat{\boldsymbol{\pi}}_i}(x^i_{t+1}) - V^i_{\hat{\boldsymbol{\pi}}_i}(x^i_t) + \alpha'_i V^i_{\hat{\boldsymbol{\pi}}_i}(x^i_t) \leq 0 \tag{5.5c}$$

where $\hat{\boldsymbol{\pi}}_i = (\boldsymbol{\pi}^{1:i-1}_{e+1}, \boldsymbol{\pi}^{i:N}_{e})$, $\boldsymbol{u}_{t+1}$ is output by the updated joint policy $(\boldsymbol{\pi}^{1:i}_{e+1}, \boldsymbol{\pi}^{i+1:N}_{e})$, and data is sampled under $\hat{\boldsymbol{\pi}}_i$. It is worth noting that $V^i_{\hat{\boldsymbol{\pi}}_i}$ can be designed as an action-value network in practice.

Remark 5.3. *The above formulation represents an idealized constrained optimization problem defined over the full state space. In practical implementations, however, the expectations and constraints involved in the optimization are approximated using sampled trajectories collected through interactions with the environment, together with replay-buffer-based learning and neural function approximation. Consequently, the implemented algorithm should be viewed as a sampled approximation of the idealized optimization problem rather than an exact realization of it. The practical algorithm seeks to approximate the idealized constrained optimization problem for corresponding desired properties through data-driven learning.*

Although it is natural to incorporate stability constraints into the training of each agent alongside safety constraints, as in Problem 5.5, one well-known challenge in control theory is that enforcing both state-wise safety and stability constraints simultaneously can lead to the issue of infeasibility [23]. To formalize this, we introduce the following definition:

Definition 5.5. $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{infeasible}$ *is the set of states for which no controller π^i exists such that agent i can further approach towards its desired equilibrium $x^i_{desired}$ (stability constraint) without causing immediate safety violations, when other agents follow $\boldsymbol{\pi}^{-i}$.*

An illustration is shown in Figure 5.6 in the Appendix. To address this issue, we provide

Proposition 5.2.

Informal Proposition 5.2. *Under certain assumptions, suppose for each $i \in \mathcal{N}$, at its K_i-th update the initial state $\boldsymbol{x}_0$ is included within the safe set of agent i, then, when the number of updates for agent i goes to infinity, for any trajectory that starts from $\boldsymbol{x}_0$ under the controller obtained by solving Problem 5.5, $Pr(\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{infeasible}) = 0$, $\forall t \geq 0$.*

The formal version and proof are provided in Appendix 5.6.4 (ii). This proposition shows that, as the number of updates grows and goes to infinity, the system is prevented from visiting $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{infeasible}}$ for all $i \in \mathcal{N}$, under the current joint policy $\boldsymbol{\pi}$, due to the neural barrier certificate. This naturally fits the multi-timescale technique in Theorem 5.1, which enables that each individual policy receives sufficient updates in theory. Consequently, the issue of infeasibility becomes negligible when integrated with the multi-timescale method, and adding the extra stability constraint into Algorithm 5.1 does not alter the overall reasoning or structure of the relevant proof. By following Algorithm 5.1 with the added stability constraint for each agent, a locally optimal solution is finally obtained, and:

Corollary 5.1. *Every agent i will converge to its desired equilibrium (goal), $x^i_{desired}$, while satisfying safety constraints, under the policy $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$ obtained when Algorithm 5.1 with extra stability constraint (5.5c) converges, in the absence of exogenous signal $\boldsymbol{d}_t$.*

The proof is in Appendix 5.6.4 (iii). The complete framework, termed **M**ulti-**A**gent **S**afe and **S**table **S**oft **A**ctor-**C**ritic (**MAS^{3}AC**), is shown in Figure 5.4, with pseudocode (Algorithm 5.2) in Section 5.6.5.

5.4 Experiments

In this section, we answer **Q1**: In cooperative MARL tasks with global information, where high rewards are achieved when the MAS converges to the desired equilibrium (e.g., navigation), can MAS^{3}AC outperform baselines in reward maximization and safety constraint satisfaction? **Q2**: In non-cooperative MARL tasks with only local observations, can MAS^{3}AC consistently achieve high rewards while maintaining few safety violations compared to baselines, for each individual agent? **Q3**: How does the size of an agent's local observation affect MAS^{3}AC's performance? Can MAS^{3}AC with a smaller field of view outperform baselines having more

informative observations?

Baselines: We compare MAS^{3}AC against standard MARL baselines: HASAC, HAPPO, MAPPO, MASAC, and IPPO, and safe MARL baselines: MACPO, MAPPO-Lagrangian, abbreviated as MAPPOL, and MAFOCOPS. Further details of these baselines are provided in Appendix 5.6.6.

Setup: The safe and stable general MARL benchmarks that we introduce consist of two parts: 1. We modified three environments based on the widely used Safe Multi-Agent MuJoCo benchmark [17] to construct safe and stable cooperative MARL benchmarks. In addition, we design new decentralized non-cooperative versions of the Coupled HalfCheetah and ant environments with local observations only. 2. We introduce a multi-agent electrical bidding task whose system dynamics is fundamentally different from robot control tasks as another decentralized non-cooperative benchmark. Further details of these benchmarks are provided in Appendix 5.6.7.

Main result: To answer **Q1**, we first evaluate MAS^{3}AC on the three cooperative benchmarks (Ant, HalfCheetah, and Coupled HalfCheetah), where global information is available and centralized training is applied. Across all tasks, MAS^{3}AC consistently achieves the best trade-off between maximizing reward and minimizing safety costs. Since rewards are tightly coupled with approaching toward and maintaining the desired equilibrium, the strong reward performance of MAS^{3}AC indicates its ability to help to achieve stability for the systems. Unlike baselines that either achieve high reward at the expense of frequent safety violations (e.g., HASAC, HAPPO) or enforce safety while greatly sacrificing task performance (e.g., MAFOCOPS), MAS^{3}AC combines both advantages. As shown in Figure 5.1 and Table 5.1, MAS^{3}AC demonstrates rapid learning progress from the start, stable convergence to high rewards, and sustained low safety costs without unrecoverable degradation. These results confirm that MAS^{3}AC can effectively coordinate agents in cooperative settings to achieve safe and stable control.

To answer **Q2**, we first evaluate MAS^{3}AC on the decentralized non-cooperative Coupled HalfCheetah benchmark, where each agent has access only to local observations within its neighborhood. The observation size is treated as a tunable hyperparameter (see the ablation study and

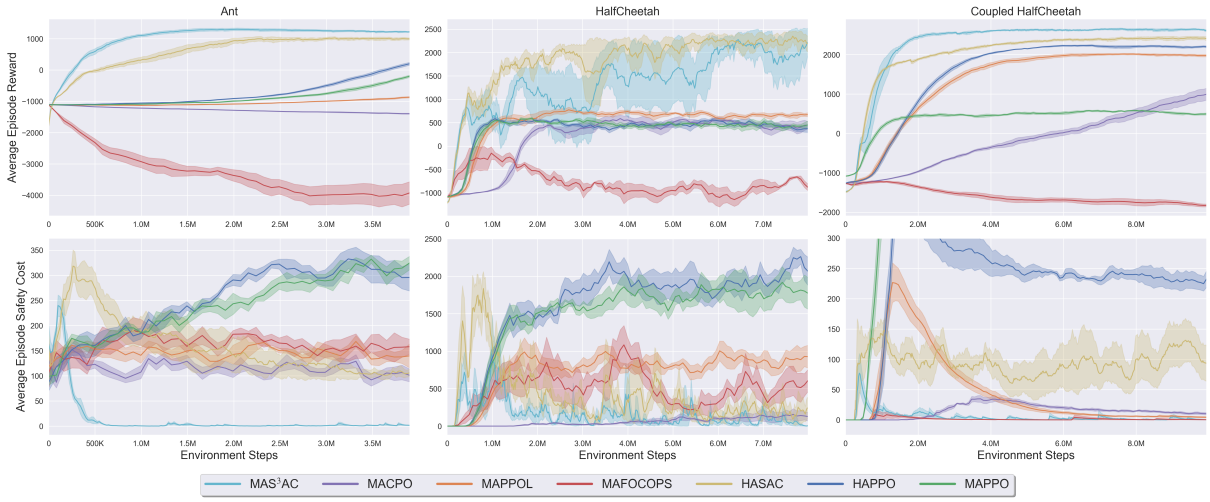


Figure 5.1: Average episode reward and average episode safety cost in the three cooperative tasks, comparing MAS^{3}AC with baselines. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

Table 5.1: Reward and safety cost comparison across cooperative benchmarks.

	Ant		HalfCheetah		Coupled HalfCheetah	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
HAPPO	41.29 ± 215.89	311.96 ± 67.96	373.64 ± 200.58	2131.21 ± 483.13	2197.22 ± 112.77	230.28 ± 67.11
HASAC	993.56 ± 87.49	107.49 ± 22.64	$\mathbf{2229.41 \pm 335.17}$	211.07 ± 300.58	2410.71 ± 106.28	84.77 ± 65.91
MACPO	-1381.66 ± 37.86	105.24 ± 50.66	443.50 ± 265.72	134.58 ± 142.09	1104.05 ± 356.15	9.32 ± 8.53
MAFOCOPS	-3965.01 ± 720.78	156.47 ± 77.84	-781.55 ± 302.59	581.41 ± 494.46	-1865.72 ± 117.10	0.14 ± 0.97
MAPPO	-354.58 ± 203.39	322.31 ± 64.85	418.49 ± 194.12	1834.08 ± 513.17	480.86 ± 195.30	1764.08 ± 431.53
MAPPOL	-903.98 ± 67.67	142.52 ± 58.00	672.25 ± 143.48	898.40 ± 379.02	1968.86 ± 118.37	3.67 ± 3.39
MAS^{3}AC	$\mathbf{1221.79 \pm 50.45}$	$\mathbf{2.10 \pm 6.92}$	2177.90 ± 714.48	$\mathbf{6.22 \pm 25.28}$	$\mathbf{2596.56 \pm 105.95}$	$\mathbf{0.03 \pm 0.17}$

Appendix 5.6.8 (ii) for details). Each agent is trained with its own reward, safety, and stability cost signals, and rewards remain closely tied to the multi-agent system's ability to converge to the desired equilibrium. This setting is different from the cooperative benchmarks since, instead of sharing a joint objective and global state, agents must make decisions independently, which increases the complexity due to partial observability and potential conflicts among individual goals. Despite these challenges, MAS^{3}AC consistently achieves high final rewards and low safety violations for *each* agent, outperforming the baselines. As shown in Figure 5.2 and Table 5.2, while some baselines may show strong performance for certain agents in either reward or safety costs, they either fail to maintain this across all agents, or often trade off stability for safety or vice versa. In contrast, MAS^{3}AC exhibits superior learning process, ensuring that every agent benefits from both the high final reward and strong constraint satisfaction. The evaluation of MAS^{3}AC on the decentralized electrical bidding task is in Appendix 5.6.8 (i). These results highlight the ability of MAS^{3}AC to generalize beyond cooperative tasks and effectively handle decentralized, non-cooperative MARL settings.

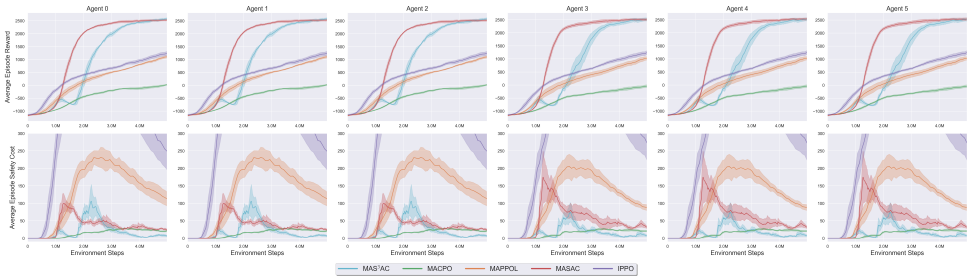


Figure 5.2: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{3}AC against baselines. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

Table 5.2: Reward and safety cost comparison for the non-cooperative Coupled HalfCheetah task across agents.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	902.29 ± 386.41	451.38 ± 305.37	904.99 ± 389.86	451.38 ± 305.37	905.91 ± 388.62	451.38 ± 305.37
MACPO	-140.28 ± 276.72	21.48 ± 25.26	-141.91 ± 276.68	21.48 ± 25.26	-142.75 ± 276.65	21.48 ± 25.26
MAPPOL	735.92 ± 416.97	174.96 ± 143.68	735.77 ± 417.75	174.96 ± 143.68	732.35 ± 417.00	174.96 ± 143.68
MASAC	2499.75 ± 78.45	25.25 ± 20.76	2503.90 ± 84.95	25.25 ± 20.76	2489.93 ± 81.61	25.25 ± 20.76
MAS^{3}AC	**2531.32 ± 136.89**	**6.35 ± 14.12**	**2533.62 ± 141.96**	**6.35 ± 14.12**	**2529.90 ± 142.89**	**6.35 ± 14.12**
	Agent 3		**Agent 4**		**Agent 5**	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	898.48 ± 395.57	432.92 ± 273.22	900.26 ± 397.68	432.92 ± 273.22	903.87 ± 397.26	432.92 ± 273.22
MACPO	-215.51 ± 277.43	21.14 ± 27.75	-216.04 ± 277.56	21.14 ± 27.75	-215.42 ± 277.29	21.14 ± 27.75
MAPPOL	646.70 ± 464.02	150.76 ± 134.02	648.57 ± 465.17	150.76 ± 134.02	648.17 ± 466.71	150.76 ± 134.02
MASAC	2479.16 ± 102.04	35.51 ± 26.71	**2502.30 ± 96.44**	35.51 ± 26.71	**2501.68 ± 108.95**	35.51 ± 26.71
MAS^{3}AC	**2486.43 ± 172.32**	**7.48 ± 20.11**	2488.44 ± 171.39	**7.48 ± 20.11**	2489.25 ± 175.59	**7.48 ± 20.11**

Ablation studies: To answer **Q3**, we conduct an ablation study on the decentralized Coupled HalfCheetah benchmark to examine how the **neighbor size**, i.e., the radius of local observations, affects performance. Figure 5.3 and Table 5.6 (deferred to the Appendix) compare MAS^{3}AC with neighbor size 0 (self-state only) and 2, against baselines that use neighbor size 3. The results show a clear trend: with more information available to all critics, certificates, and policies, MAS^{3}AC achieves stronger performance. In particular, MAS^{3}AC with neighbor size 2 matches the reward level of MASAC using neighbor size 3, which purely maximizes rewards, while maintaining substantially lower safety costs, outperforming all baselines in balancing safety and performance. By contrast, MAS^{3}AC with neighbor size 0 suffers from higher safety violations, highlighting that barrier certificates for safety must incorporate information from neighbors to effectively enforce inter-agent safety requirements. Nevertheless, even with neighbor size 0, MAS^{3}AC remains competitive with some baselines, demonstrating the robustness of the framework even under severely limited information. Additional results with intermedi-

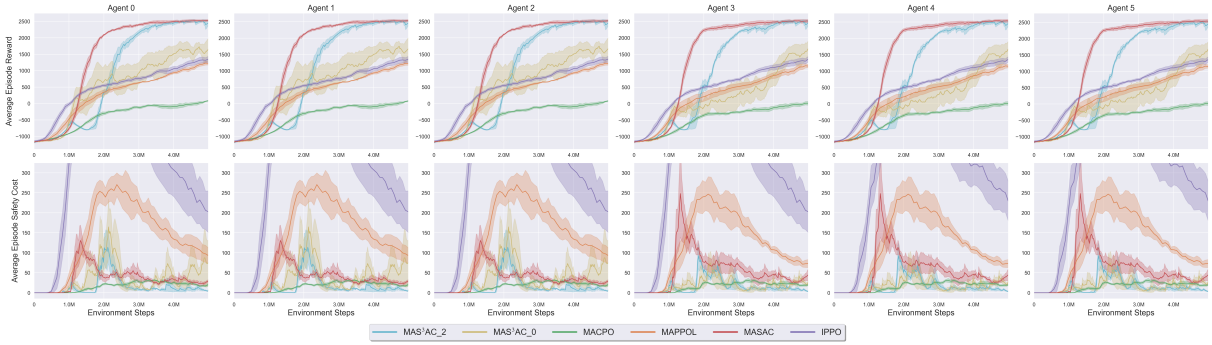


Figure 5.3: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{3}AC with different neighbor sizes against baselines.

ate neighbor sizes $(0, 1, 3)$ are provided in Appendix 5.6.8 (ii), which confirm the consistent advantage of MAS^{3}AC across different observation ranges.

We next examine the **effect of Lyapunov function inputs** on MAS^{3}AC by using the decentralized Coupled HalfCheetah benchmark. Detailed results and analysis are provided in Appendix 5.6.8 (iii), where we show how different Lyapunov certificate designs influence the overall performance of MAS^{3}AC, in alignment with our theoretical ISS analysis.

More ablations are provided in Appendix 5.6.9, including but not limited to robustness against external disturbances, scaling to larger MAS with partial observability, comparison with the additional baseline (modified Scal-MAPPO-L), new experiments on "Decentralized Ant" task, barrier versus reachability certificates for safety, the impact of input-to-state stability conditions on MAS^{3}AC, necessity of the squared term, and others, further supporting the claims.

5.5 Conclusions

In this work, we propose MAS^{3}AC, a framework that integrates state-wise constraints with model-free RL for safe and stable control of general multi-agent tasks. Theoretical proof is provided to show the convergence of the algorithm to a feasible locally optimal solution, and a specific investigation is conducted on the infeasibility issue. Empirical results across all proposed benchmarks demonstrate that our algorithm yields consistently high rewards with very few safety violations compared to baselines on both cooperative and non-cooperative MARL tasks with either centralized or decentralized training. We hope this MAS^{3}AC algorithm, together with new benchmarks, can facilitate future research and serve as a baseline for safe and stable MARL

algorithm development.

Despite the promising results obtained by the MAS^{3}AC algorithm proposed in this study, several limitations remain. First, from a theoretical perspective, the multi-timescale training scheme may result in high sample complexity. Second, for non-cooperative tasks, it would be valuable to establish an upper bound on the error between policy updates computed using global state information and local observations to provide a stronger theoretical justification for decentralized training. Third, a detailed theoretical analysis for extending the algorithm to incorporate adaptive class-$\mathcal{K}$ functions in stability constraints remains an important open direction. These issues are investigated in the next chapter.

5.6 Appendix

5.6.1 A. Loss Functions for Updates

In this part, we provide the explicit forms of loss functions used in the update. The loss functions to update the action-value function Q^i, parameterized by ϕ_i, and the barrier certificate H_{ω_i} for agent i are:

$$J_{Q^i}(Q^i) = \mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{u}_t, r_t^i, \boldsymbol{x}_{t+1}) \sim \mathcal{D}} \big[r_t^i + \gamma Q_{\phi_i, targ}(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) - Q_{\phi_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) \big]^2, \tag{5.6}$$

$$J_{H^i}(H^i) = \mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{u}_t, b_t^i, \boldsymbol{x}_{t+1}) \sim \mathcal{D}} \big[b_t^i + \gamma H_{\omega_i, targ}(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) - H_{\omega_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) \big]^2, \tag{5.7}$$

where $\tilde{\boldsymbol{u}}_{t+1}$ is generated by the current joint policy $\hat{\boldsymbol{\pi}}_i = (\boldsymbol{\pi}_{e+1}^{1:i-1}, \boldsymbol{\pi}_e^{i:N})$, and $\mathcal{D}$ denotes the replay buffer. Target networks $Q_{\phi_i, targ}$ and $H_{\omega_i, targ}$ are used in practical training. Since both the action-value function and the barrier certificate are represented by neural networks, the loss function to train the parameter θ_i for policy π^i, and its corresponding Lagrangian multiplier λ_i, derived from (5.3), is:

$$\begin{aligned} J_{\theta_i}(\theta_i) = J_{\lambda_i}(\lambda_i) = \mathbb{E}_{\mathcal{D}} \big[& - Q^i(\boldsymbol{x}_t, \hat{\boldsymbol{u}}_t; \phi_i) + \lambda_i \sum_{t=0}^{\infty} \gamma^t \max\{0, -H^i(\boldsymbol{x}_{t+1}, \hat{\boldsymbol{u}}_{t+1}; \omega_i) + H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i) \\ & - \epsilon H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i)\} + \frac{\lambda_i}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, -H^i(\boldsymbol{x}_{t+1}, \hat{\boldsymbol{u}}_{t+1}; \omega_i) + H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i) - \epsilon H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i)\}]^2 \big], \end{aligned} \tag{5.8}$$

where $\hat{\boldsymbol{u}}_t$ and $\hat{\boldsymbol{u}}_{t+1}$ are actions output by the policy network with gradient information for update θ_i. When stability constraints are also incorporated, we first train the function V^i, parameterized by χ_i for agent i, using the following loss function:

$$J_{V^i}(V^i) = \mathbb{E}_{(x_t^i, u_t^i, c_t^i, x_{t+1}^i)\sim\mathcal{D}}\big[c_t^i + \gamma V_{\chi_i, targ}(x_{t+1}^i, \tilde{u}_{t+1}^i) - V_{\chi_i}(x_t^i, u_t^i)\big]^2, \tag{5.9}$$

when V^i is implemented as an action-value network, and target function $V_{\chi_i, targ}$ is applied. Subsequently, the loss function used to train the policy π^i, and its corresponding Lagrangian multipliers, is given as

$$\begin{aligned}
J_{\theta_i}^{\text{full}}(\theta_i) = J_{\lambda_i}^{\text{full}}(\lambda_i) = \mathbb{E}_{\mathcal{D}}\big[& - Q^i(\boldsymbol{x}_t, \hat{\boldsymbol{u}}_t; \phi_i) + \lambda_{i,1} \sum_{t=0}^{\infty} \gamma^t \text{max}\{0, -H^i(\boldsymbol{x}_{t+1}, \hat{\boldsymbol{u}}_{t+1}; \omega_i) \\
& + H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i) - \epsilon H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i)\} + \frac{\lambda_{i,1}}{2} \sum_{t=0}^{\infty} \gamma^t [\text{max}\{0, -H^i(\boldsymbol{x}_{t+1}, \hat{\boldsymbol{u}}_{t+1}; \omega_i) \\
& + H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i) - \epsilon H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i)\}]^2 + \lambda_{i,2} \sum_{t=0}^{\infty} \gamma^t \text{max}\{0, V^i(x_{t+1}^i, \hat{u}_{t+1}^i, \chi_i) \\
& - V^i(x_t^i, u_t^i, \chi_i) + \alpha_i' V^i(x_t^i, u_t^i, \chi_i)\} + \frac{\lambda_{i,2}}{2} \sum_{t=0}^{\infty} \gamma^t [\text{max}\{0, V^i(x_{t+1}^i, \hat{u}_{t+1}^i, \chi_i) \\
& - V^i(x_t^i, u_t^i, \chi_i) + \alpha_i' V^i(x_t^i, u_t^i, \chi_i)\}]^2\big].
\end{aligned} \tag{5.10}$$

Here $\hat{\boldsymbol{u}}_t$, $\hat{\boldsymbol{u}}_{t+1}$ and $\hat{u}_{t+1}^i$ are actions output by the policy network with gradient information for update θ_i. $\lambda_{i,1}$ and $\lambda_{i,2}$ are Lagrangian multipliers for safety and stability constraints, respectively. When global information is not available, all inputs are replaced by local observations.

5.6.2 B. Assumptions

This section summarizes the assumptions used in the theoretical analysis. Some notations are formally introduced later in the proof sections, but are still included here for completeness.

Assumption 5.1. *There exists an upper bound T such that $\forall i \in \mathcal{N}$, all trajectories starting from $\boldsymbol{x}_0 \in \boldsymbol{\mathcal{X}}_{\text{irrec}}^{\boldsymbol{\pi},i}$ will result in $\Pr(b^i(\boldsymbol{x}_t) < 0) > 0$ at least once within T steps under the controller π^i when all other agents follow the fixed policy $\boldsymbol{\pi}^{-i}$.*

This assumes that states within the irrecoverable unsafe set $\boldsymbol{\mathcal{X}}_{irrec}^{\boldsymbol{\pi},i}$ lead to a safety violation

with non-zero probability within a bounded number of steps. This is a mild assumption since it does not require deterministic violations, but only requires that safety violations occur with positive probability under the given policy.

Assumption 5.2. *Define $\eta_1(e)$ as the learning rate for updating the action-value functions and barrier certificates of all agents. For each agent $i \in \mathcal{N}$, let $\eta_{2,i}(e)$ and $\eta_{3,i}(e)$ denote the learning rates for updating the policy and the Lagrangian multiplier, respectively. The following conditions hold:*

1. $\sum_e \eta_1(e) = \sum_e \eta_{2,i}(e) = \sum_e \eta_{3,i}(e) = \infty, \quad \forall i \in \mathcal{N}$
2. $\sum_e \eta_1(e)^2 < \infty, \quad \sum_e \eta_{2,i}(e)^2 < \infty, \quad \sum_e \eta_{3,i}(e)^2 < \infty, \quad \forall i \in \mathcal{N}$
3. $\eta_{2,1}(e) = o(\eta_1(e)); \quad \eta_{3,i}(e) = o(\eta_{2,i}(e)), \quad \forall i \in \mathcal{N}; \ \eta_{2,i}(e) = o(\eta_{3,i-1}(e)), \ \forall i \in \{2, \ldots, N\}$.

This assumption adopts a multi-timescale learning framework widely used in prior constrained reinforcement learning studies. It ensures that the critic and barrier functions for all agent are updated on the fastest timescale, while for each agent, the policy is updated on a faster timescale than its associated Lagrangian multiplier; and the update of the Lagrangian multiplier of agent i is faster than the update of the policy of the following agent $i+1$. This hierarchy enables effective tracking of inter-agent constraints, as in our case, and addresses non-stationarity, thereby improving learning stability in multi-agent systems.

Theoretically, step-size schedules such as $\eta_1(e) = \frac{1}{e}$, $\eta_{2,i}(e) = \frac{1}{e^{2i}}, \eta_{3,i}(e) = \frac{1}{e^{2i+1}}$ can be used to satisfy multi-timescale requirements. However, in practice, since the number of update steps is limited, it is difficult to satisfy the first point in Assumption 5.2 with any schedule. In implementation, we apply linearly decaying learning rates with different initial values to ensure $\eta_1(e) > \eta_{2,i}(e) > \eta_{3,i}(e), \forall e, \forall i \in \mathcal{N}$ and $\eta_{3,i-1}(e) > \eta_{2,i}(e), \forall e, \forall i \in \{2, \ldots, N\}$ to approximate the required relationships among the learning rates, which has been empirically proved to be sufficient to support stable and efficient training across all benchmarks compared to all baselines.

To mitigate potential slow convergence due to timescale separation, we incorporate additional squared penalty terms into the constraint loss, which encourage and accelerate the update

toward feasible solutions. Based on the experimental results, our algorithm has a better sample efficiency compared to baselines in quickly achieving higher rewards while maintaining very few or near-zero safety violations with fewer samples, allowing early stopping to reduce computational burden. Moreover, for homogeneous agents, the parameter sharing method can be applied to reduce the number of trainable parameters and distinct learning rates, which further improves the training efficiency. However, further investigation under the multi-timescale framework, including more techniques for enhancing sample efficiency, remains an important direction for future work.

Assumption 5.3. *For each agent i, the defined signals $\hat{r}_t^i$ and $\hat{r}_t^{i,*}$, the function $Q_H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_i)$ and the space $\hat{\Omega}_i$ are bounded. Besides, $\nabla_{\hat{\omega}_i} Q_H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_i)$ is a Lipschitz function in $\hat{\omega}_i$ for all $\boldsymbol{x}_t$ and $\boldsymbol{u}_t$.*

This assumption guarantees the boundedness of signals and critic functions for agents, which is a standard requirement in actor-critic methods to ensure stable training and convergence. The Lipschitz condition, which is also commonly used in previous RL studies, on the gradient promotes well-behaved updates. For neural networks, approximate Lipschitz continuity can be induced through techniques such as weight decay and clipping, making this assumption realistic in practice.

Note: The terms $\hat{r}_t^i$, $\hat{r}_t^{i,*}$, Q_H^i, $\hat{\omega}_i$, and $\hat{\Omega}_i$ are formally introduced in Section 5.6.3 (iii). They are referenced here to provide a complete overview of all theoretical assumptions, even though they are not defined in the main body of the submission.

Assumption 5.4. *For each agent i, there always exists a policy π^i such that*

$$H_{\boldsymbol{\pi}}^i(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) - H_{\boldsymbol{\pi}}^i(\boldsymbol{x}_t, \boldsymbol{u}_t) \geq -\epsilon H_{\boldsymbol{\pi}}^i(\boldsymbol{x}_t, \boldsymbol{u}_t)$$

for all consecutive pairs $(\boldsymbol{x}_t, \boldsymbol{u}_t)$ and $(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1})$, $\forall t \geq 0$, along with trajectories generated under any joint policy $\boldsymbol{\pi}^{-i}$. Further, all initial states are within the safe set $\boldsymbol{\mathcal{X}}_{safe}$.

This assumption rules out scenarios where the environment, including other agents' fixed policies, renders safety of agent i impossible, namely situations where no admissible action can help to satisfy safety requirements for agent i. Since the other agents' policies $\boldsymbol{\pi}^{-i}$ are

treated as part of the environment in this formulation, this feasibility condition is effectively an assumption for single-agent tasks. It ensures the problem is well-posed and that a safe policy is learnable and achievable under the given dynamics. The requirement that all initial states lie within the safe set is also standard in constrained control setups, as starting from unsafe states would inevitably lead to safety constraint violations.

Assumption 5.5. *The state space and action space for each agent are bounded. Besides, the parameter space Θ_i and Lagrangian multiplier λ_i for each agent i are also bounded. Further, $\nabla_{u_t^i} Q^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \phi_i)$ and $\nabla_{u_t^i} Q_H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_i)$ are Lipschitz functions in u_t^i for all $\boldsymbol{x}_t$, ϕ_i and $\hat{\omega}_i$, and are also Lipschitz functions in ϕ_i and $\hat{\omega}_i$ for all $\boldsymbol{x}_t$ and $\boldsymbol{u}_t$. $\nabla_{\theta_i} \pi^i(\boldsymbol{x}_t; \theta_i)$ is a Lipschitz function in θ_i.*

This assumption ensures the boundedness of the relevant spaces and the smoothness of gradients used in the policy update. Bounded state and action spaces are standard in RL and control theory to avoid unbounded optimization landscapes. Bounding the policy parameter and Lagrange multiplier spaces helps the dual updates and prevents divergence. The Lipschitz continuity of the critic gradients and policy gradient indicates well-behaved gradient-based updates during policy training. In practice, neural networks with smooth activations and regularization techniques can effectively lead to approximate Lipschitz properties, which makes the assumption mild and practical.

Assumption 5.6. *For each agent i, the function $Q_H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_i)$ is a Lipschitz function in u_t^i for all $\boldsymbol{x}_t$ and $\hat{\omega}_i$. Moreover, the gradients $\nabla_{u_t^i} Q_{\bar{\boldsymbol{\pi}}_i}^i(\boldsymbol{x}_t, \boldsymbol{u}_t)$ and $\nabla_{u_t^i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\boldsymbol{x}_t, \boldsymbol{u}_t)$ are Lipschitz functions in the joint policy $\bar{\boldsymbol{\pi}}_i$.*

This assumption ensures local smoothness of the value function and its gradients with respect to the agent's action. It is essential for bounding the approximation error during policy updates, and reflects a standard smoothness assumption in theoretical RL, capturing the idea that small changes in the policy lead to proportionally bounded changes in value estimates. In practice, the approximate Lipschitz property of value networks can also be induced through methods like regularization techniques, as stated in previous explanations.

Note: Related terms and gradients are formally introduced in Section 5.6.3 (iii). They are included here for completeness, as this appendix summarizes all theoretical assumptions supporting

the theoretical analysis.

Assumption 5.7. *During the update of agent i's policy while other agents follow certain $\boldsymbol{\pi}^{-i}$, denote k_i as the number of updates for π^i, then, for any $k_i \geq 0$ and any $\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{infeasible}$, the control signal u^i_t generated by the new policy $\pi^i_{k_i+1}$, obtained by solving*

$$\pi^i_{k_i+1} = \arg\min_{\pi^i} \quad -Q^i_{\boldsymbol{\pi}_{i,k_i}}(\boldsymbol{x}_t, (\pi^i(\boldsymbol{x}_t), \boldsymbol{\pi}^{-i}(\boldsymbol{x}_t))) \tag{5.11a}$$

$$s.t. - H^i_{\boldsymbol{\pi}_{i,k_i}}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) + H^i_{\boldsymbol{\pi}_{i,k_i}}(\boldsymbol{x}_t, \boldsymbol{u}_t) - \epsilon H^i_{\boldsymbol{\pi}_{i,k_i}}(\boldsymbol{x}_t, \boldsymbol{u}_t) \leq 0 \tag{5.11b}$$

$$V^i_{\boldsymbol{\pi}_{i,k_i}}(x^i_{t+1}) - V^i_{\boldsymbol{\pi}_{i,k_i}}(x^i_t) + \alpha'_i V^i_{\boldsymbol{\pi}_{i,k_i}}(x^i_t) \leq 0 \tag{5.11c}$$

where $\boldsymbol{\pi}_{i,k_i} = (\pi^i_{k_i}, \boldsymbol{\pi}^{-i})$ denotes the joint policy in which agent i's policy is being updated for the k_i-th time while others follow $\boldsymbol{\pi}^{-i}$, satisfies the stability constraint with a positive probability.

This assumption states that when the state lies within $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{infeasible}}$, where no controller exists to simultaneously maintain safety and drive the agent closer toward its goal according to the stability constraint, for each iteration, there is a positive probability that the control signal, output by the updated $\pi^i_{k_i+1}$, satisfies the stability constraint, thereby driving the agent closer to the goal and causing immediate safety violations.

5.6.3 C. Appendix for State-Wise Safety via Neural Barrier Certificates

(i) Formal Version and Proof for Lemma 5.1

Lemma 5.2. *Define the barrier signal b^i as:*

$$b^i(\boldsymbol{x}_t) = \begin{cases} d & \text{if } \boldsymbol{x}_t \notin \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{vio}, \\ D & \text{if } \boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{vio}, \end{cases} \tag{5.12}$$

where $d = 0$ and $D < 0$ are constants. Under Assumption 5.1, the function $H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t) = \mathbb{E}_{\tau\sim\boldsymbol{\pi}}\big[\sum_{k=0}^{\infty} \gamma^k b^i(\boldsymbol{x}_{t+k})\big]$, where $\gamma \in (0,1)$ is the discount factor and $\tau = \{\boldsymbol{x}_t, \boldsymbol{x}_{t+1}, \boldsymbol{x}_{t+2}, \dots\}$ is a trajectory under controller $\boldsymbol{\pi}$ starting from $\boldsymbol{x}_t$, is a barrier certificate for the agent i, given that all other agents following $\boldsymbol{\pi}^{-i}$.

Proof. When $\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{safe}}$, by Definition 5.2, under the controller $\boldsymbol{\pi} = (\pi^i, \boldsymbol{\pi}^{-i})$, for all $k \geq 0$,

$Pr(b^i(\boldsymbol{x}_{t+k}))=0$, and therefore, $H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t)=H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_{t+1})=\mathbb{E}_{\tau\sim\boldsymbol{\pi}}\big[\sum_{k=0}^{\infty}\gamma^k b^i(\boldsymbol{x}_{t+k})\big]=0$ given $d=0$. Hence, the second property in Definition 5.3 is satisfied. When $\boldsymbol{x}_t\in\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{unsafe}}$, according to Definition 5.2, there exists $n\geq 0$ such that $Pr(b^i(\boldsymbol{x}_{t+n})<0)>0$ under the joint controller $\boldsymbol{\pi}$, and therefore $\mathbb{E}_{\boldsymbol{x}_{t+n}}\big[b^i(\boldsymbol{x}_{t+n})\big]<0$. Furthermore, under Assumption 5.1, we have $n\leq T$. Thus, we have:

$$H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t)=\mathbb{E}_{\tau\sim\boldsymbol{\pi}}\big[\sum_{k=0}^{\infty}\gamma^k b^i(\boldsymbol{x}_{t+k})\big]\leq\gamma^n\mathbb{E}_{\boldsymbol{x}_{t+n}}\big[b^i(\boldsymbol{x}_{t+n})\big]\leq\gamma^T\mathbb{E}_{\boldsymbol{x}_{t+n}}\big[b^i(\boldsymbol{x}_{t+n})\big]<0,$$

since $\gamma\in(0,1)$. Therefore, the first property in Definition 5.3 is satisfied. Also, when $\boldsymbol{x}_t\in\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{safe}}$, by Definition 5.2, $\boldsymbol{x}_{t+1}\in\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{safe}}$, and thus $H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_{t+1})-H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t)=0-0=-\epsilon H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t)$. Therefore, the third property in Definition 5.3 is satisfied. In sum, all three properties for being a barrier certificate are satisfied, and thus, the function $H^i_{\boldsymbol{\pi}}(\boldsymbol{x}_t)$ is a barrier certificate of the agent i with all other agents following $\boldsymbol{\pi}^{-i}$. □

(ii) Proof for Proposition 5.1

Proof. For all $\boldsymbol{x}_t\in\mathcal{X}^{(\boldsymbol{\pi}^{1:i-1}_{e+1},\boldsymbol{\pi}^{i:N}_e),i}_{\text{safe}}$, the condition $H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_{t+1})\geq(1-\epsilon)H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t)=0$ in Problem (5.2) for $\boldsymbol{x}_{t+1}$ following the new joint policy $(\boldsymbol{\pi}^{1:i}_{e+1},\boldsymbol{\pi}^{i+1:N}_e)$. Therefore, $\boldsymbol{x}_{t+1}\in\mathcal{X}^{(\boldsymbol{\pi}^{1:i-1}_{e+1},\boldsymbol{\pi}^{i:N}_e),i}_{\text{safe}}$ and similarly, $\forall\boldsymbol{x}_t\in\mathcal{X}^{(\boldsymbol{\pi}^{1:i-1}_{e+1},\boldsymbol{\pi}^{i:N}_e),i}_{\text{safe}}$. the trajectory $\{\boldsymbol{x}_t,\boldsymbol{x}_{t+1},\dots\}$ under the policy $(\boldsymbol{\pi}^{1:i}_{e+1},\boldsymbol{\pi}^{i+1:N}_e)$ remains in $\mathcal{X}^{(\boldsymbol{\pi}^{1:i-1}_{e+1},\boldsymbol{\pi}^{i:N}_e),i}_{\text{safe}}$, without causing $b^i(\boldsymbol{x}_{t+k})<0$ at any $k\geq 0$ with a positive probability. Thus, $\boldsymbol{x}_t\in\mathcal{X}^{(\boldsymbol{\pi}^{1:i}_{e+1},\boldsymbol{\pi}^{i+1:N}_e),i}_{\text{safe}}$ since $\mathbb{E}_{(\boldsymbol{\pi}^{1:i}_{e+1},\boldsymbol{\pi}^{i+1:N}_e)}\big[\sum_{k=0}^{\infty}\gamma^k b^i(\boldsymbol{x}_{t+k})\big]=0$. Since $\epsilon\in(0,1)$, if $\exists\boldsymbol{x}_t\in\boldsymbol{\mathcal{X}}^{(\boldsymbol{\pi}^{1:i-1}_{e+1},\boldsymbol{\pi}^{i:N}_e),i}_{\text{irrec}}$, such that under the policy $(\boldsymbol{\pi}^{1:i}_{e+1},\boldsymbol{\pi}^{i+1:N}_e)$, $H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_{t+1})=0>(1-\epsilon)H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t)$, where $\boldsymbol{x}_{t+1}$ follows the new joint policy $(\boldsymbol{\pi}^{1:i}_{e+1},\boldsymbol{\pi}^{i+1:N}_e)$, then, according to the previous analysis, the following states $\{\boldsymbol{x}_{t+2},\boldsymbol{x}_{t+3},\dots\}$ will stay within $\mathcal{X}^{(\boldsymbol{\pi}^{1:i-1}_{e+1},\boldsymbol{\pi}^{i:N}_e),i}_{\text{safe}}$ under $(\boldsymbol{\pi}^{1:i}_{e+1},\boldsymbol{\pi}^{i+1:N}_e)$, and thus $\boldsymbol{x}_t\in\mathcal{X}^{(\boldsymbol{\pi}^{1:i}_{e+1},\boldsymbol{\pi}^{i+1:N}_e),i}_{\text{safe}}$ given $b^i(\boldsymbol{x}_t)=0$.
Based on the aforementioned analysis, it can be concluded that $\mathcal{X}^{(\boldsymbol{\pi}^{1:i-1}_{e+1},\boldsymbol{\pi}^{i:N}_e),i}_{\text{safe}}\subseteq\mathcal{X}^{(\boldsymbol{\pi}^{1:i}_{e+1},\boldsymbol{\pi}^{i+1:N}_e),i}_{\text{safe}}$, namely the safe set for agent i will not shrink when other agents follow the fixed policy $\boldsymbol{\pi}^{1:i-1}_{e+1}$ and $\boldsymbol{\pi}^{i+1:N}_e$. □

(iii) Formal Version and Proof for Theorem 5.1

Theorem 5.3. *Under Assumptions 5.1-5.6, the joint policy updated in Algorithm 5.1 will almost surely converge to a feasible local Nash equilibrium* $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$*, where* $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) \geq 0$ *for all* $(\boldsymbol{x}_t, \boldsymbol{u}_t)$ *along the trajectory induced by* $\boldsymbol{\pi}^*$*, for every* $i \in \mathcal{N}$ *and* $t \geq 0$*. Furthermore, define* $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) := \min_{i\in\mathcal{N}} H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$*, then* $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ *is a global barrier certificate for the multi-agent system.*

Proof. This proof is extended from [136, 137], but considers a different problem with additional squared terms to better achieve and maintain the satisfaction of the constraints, and include multiple agents, which necessitates additional analysis when updating the policy of agent i to account for the inaccuracy caused by the updates of previous agents (namely from 1 to $i-1$). The structure of the proof is as follows:

1. For the first updated agent, we first demonstrate that its critic and barrier certificate will converge to the fixed solution corresponding to the policy parameterized by $\{\theta_i^k\}_{i\in\mathcal{N}}$.
2. We show that the policy θ_1 and the corresponding Lagrangian multiplier λ_1 will converge to a local saddle point with respect to $(\{\theta_i^k\}_{i\in\{2,\dots,N\}}, \{\lambda_i^k\}_{i\in\{2,\dots,N\}})$.
3. For the second updated agent, we similarly prove that the policy θ_2 and λ_2 will converge to a local saddle point with respect to $(\{\theta_i^k\}_{i\in\{3,\dots,N\}}, \{\lambda_i^k\}_{i\in\{3,\dots,N\}})$. By induction, finally, the algorithm converge to a feasible local Nash equilibrium asymptotically.

We first consider the update of the first agent: When updating the first agent, since all other agents are updated on slower timescales, here we treat $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ (namely $\boldsymbol{\theta}_{-1}$) and $\{\lambda_i^k\}_{i\in\{2,\dots,N\}}$ as certain fixed values, and thus let $\bar{\boldsymbol{\pi}}_1 = (\theta_1^k, \theta_2^k, \dots, \theta_N^k)$ where θ_1^k is being updated and $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ are fixed constants. Then, we have the following different timescales:

Convergence of ϕ_1 and ω_1: The Assumption 5.2 indicates that the updates of ϕ_1^k and ω_1^k converge on a faster timescale than policies $\{\theta_i^k\}_{i\in\mathcal{N}}$ and $\{\lambda^k\}_{i\in\mathcal{N}}$. According to [138], we can deem $\{\theta_i^k\}_{i\in\mathcal{N}}$ and $\{\lambda^k\}_{i\in\mathcal{N}}$ as arbitrary fixed values when updating ϕ_1^k and ω_1^k, and therefore,

with the standard convergence results in [30], we can easily know that for policy $\bar{\boldsymbol{\pi}}_1$,

$$\begin{aligned}\mathcal{B}^{\bar{\boldsymbol{\pi}}_1}[Q^1](\boldsymbol{x}_t,\boldsymbol{u}_t) &= r^1 + \gamma \mathbb{E}_{\boldsymbol{u}_{t+1}\sim\bar{\boldsymbol{\pi}}_1}\big[Q^1(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1})\big]\\ \mathcal{B}^{\bar{\boldsymbol{\pi}}_1}[H^1](\boldsymbol{x}_t,\boldsymbol{u}_t) &= b^1 + \gamma \mathbb{E}_{\boldsymbol{u}_{t+1}\sim\bar{\boldsymbol{\pi}}_1}\big[H^1(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1})\big]\end{aligned} \tag{5.13}$$

are $\gamma-$contraction mappings, and therefore when $k\to\infty$, we have $\phi_1^k \to \phi_1^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k)\to Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^*) = Q^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_t,\boldsymbol{u}_t)$, and $\omega_1^k\to\omega_1^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\omega_1^k)\to H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\omega_1^*) = H^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_t,\boldsymbol{u}_t)$. Therefore, the convergence of this timescale is proved. Furthermore, define

$$\begin{aligned}\hat{r}_t^1(\boldsymbol{x}_t,\boldsymbol{u}_t) = \mathbb{E}_{\tilde{\boldsymbol{u}}_{t+1}\sim\bar{\boldsymbol{\pi}}_1}\big[&\max\{0,-H^1(\boldsymbol{x}_{t+1},\tilde{\boldsymbol{u}}_{t+1};\omega_1^k)+H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\omega_1^k)-\epsilon H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\omega_1^k)\}\\ &+\frac{1}{2}[\max\{0,-H^1(\boldsymbol{x}_{t+1},\tilde{\boldsymbol{u}}_{t+1};\omega_1^k)+H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\omega_1^k)-\epsilon H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\omega_1^k)\}]^2\big]\end{aligned} \tag{5.14}$$

and

$$\begin{aligned}\hat{r}_t^{1,*}(\boldsymbol{x}_t,\boldsymbol{u}_t) = \mathbb{E}_{\tilde{\boldsymbol{u}}_{t+1}\sim\bar{\boldsymbol{\pi}}_1}\big[&\max\{0,-H^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_{t+1},\tilde{\boldsymbol{u}}_{t+1})+H^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_t,\boldsymbol{u}_t)-\epsilon H^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_t,\boldsymbol{u}_t)\}\\ &+\frac{1}{2}[\max\{0,-H^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_{t+1},\tilde{\boldsymbol{u}}_{t+1})+H^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_t,\boldsymbol{u}_t)-\epsilon H^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_t,\boldsymbol{u}_t)\}]^2\big],\end{aligned} \tag{5.15}$$

with

$$Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u}) = \mathbb{E}_{\tau\sim\bar{\boldsymbol{\pi}}_1}\big[\sum_{t=0}^{\infty}\gamma^t\hat{r}_t^{1,*}(\boldsymbol{x}_t,\boldsymbol{u}_t)|\boldsymbol{x}_0=\boldsymbol{x},\boldsymbol{u}_0=\boldsymbol{u}\big], \tag{5.16}$$

then, given a step size $\eta_{Q_H}(e)$ such that $\sum_e \eta_{Q_H}(e)=\infty$, $\sum_e \eta_{Q_H}(e)^2<\infty$, $\eta_{Q_H}(e) = o(\eta_1(e)), \eta_{2,1}(e)=o(\eta_{Q_H}(e))$, if $Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}$ is parameterized by $\hat{\omega}_1$, we obtain the following update using sampled transitions:

$$\hat{\omega}_1^{k+1} = \Gamma_{\hat{\Omega}_1}\big[\hat{\omega}_1^k - \eta_{Q_H}\nabla_{\hat{\omega}_1}(\hat{r}_t^1+\gamma Q_H^1(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1};\hat{\omega}_1)-Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k}\big]. \tag{5.17}$$

Let us further have an ideal update

$$\hat{\omega}_1^{k+1} = \Gamma_{\hat{\Omega}_1}\big[\hat{\omega}_1^k - \eta_{Q_H}\nabla_{\hat{\omega}_1}(\hat{r}_t^{1,*}+\gamma Q_H^1(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1};\hat{\omega}_1)-Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k}\big]. \tag{5.18}$$

The difference between (5.17) and (5.18) is the use of signal ($\hat{r}_t^1$ or $\hat{r}_t^{1,*}$), which reveals the inaccuracy caused in the update of ω_1 in the previous timescale. Then, we can rewrite the real update when the sampled batch D follows the distribution caused by $\bar{\boldsymbol{\pi}}_1$ to be:

$$\begin{aligned}\hat{\omega}_1^{k+1} = \Gamma_{\hat{\Omega}_1}\big[\hat{\omega}_1^k - \eta_{Q_H}\nabla_{\hat{\omega}_1}\big((\hat{r}_t^{1,*} + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k} \\ + \delta\hat{\omega}_1^{k+1} + \delta\hat{\omega}_{1,\varepsilon}\big)\big].\end{aligned} \tag{5.19}$$

where

$$\begin{aligned}\delta\hat{\omega}_1^{k+1} = -\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k}\big] \\ + \nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k}\end{aligned} \tag{5.20}$$

and

$$\begin{aligned}\delta\hat{\omega}_{1,\varepsilon} = \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k} \\ - \nabla_{\hat{\omega}_1}(\hat{r}_t^{1,*} + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k}\big]\end{aligned} \tag{5.21}$$

are two terms representing the difference between the real update and the ideal update.

Under Assumption 5.3:

1. We show that $\delta\hat{\omega}_1^{k+1}$ is square integrable first. To be specific,

$$\begin{aligned}\mathbb{E}\big[||\delta\hat{\omega}_1^{k+1}||^2|\mathcal{F}_{\hat{\omega}_1,k}\big] \leq C_1||\nabla_{\hat{\omega}_1}Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k)||_\infty^2 \times (||\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1^k)||_\infty^2 \\ + ||Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k)||_\infty^2)\end{aligned} \tag{5.22}$$

where $\mathcal{F}_{\hat{\omega}_1,k} = \sigma(\hat{\omega}_1^m, \delta\hat{\omega}_1^m, m \leq k)$ is the filtration of $\hat{\omega}_1^k$ generated by different independent trajectories [137]. Furthermore, Assumption 5.3 requires that $||\nabla_{\hat{\omega}_1}Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1)||_\infty^2 \leq K_1(1 + ||\hat{\omega}_1||_\infty^2)$ and other terms are bounded, therefore, $\mathbb{E}\big[||\delta\hat{\omega}_1^{k+1}||^2|\mathcal{F}_{\hat{\omega}_1,k}\big] < \infty$, and $\delta\hat{\omega}_1^{k+1}$ is square integrable.

2. We then show that $\delta\hat{\omega}_{1,\varepsilon} \to 0$:

$$||\delta\hat{\omega}_{1,\varepsilon}|| \leq C_2||\nabla_{\hat{\omega}_1} Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1)||_\infty||\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[2(\hat{r}_t^{1,*} + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1^k) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k)) - 2(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1^k) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k))\big]||$$

(5.23)

Since $||\nabla_{\hat{\omega}_1} Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1)||$ is upper bounded, and we have $\hat{r}_t^1 \to \hat{r}_t^{1,*}$ given $H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_1^k) \to H_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)$ according to the analysis in the previous timescale, we have $||\delta\hat{\omega}_{1,\varepsilon}|| \to 0$, and therefore $\delta\hat{\omega}_{1,\varepsilon} \to 0$.

3. Because the $\nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k}$ is a sample of $\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\omega}_1}(\hat{r}_t^1+\gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1)-Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k}\big]$, we can conclude that $\mathbb{E}\big[\delta\hat{\omega}_1^{k+1}|\mathcal{F}_{\hat{\omega}_1,k}\big] = 0$.

When $\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\omega}_1}(\hat{r}_t^1+\gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1)-Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1=\hat{\omega}_1^k}\big]$ is a Lipschitz function in $\hat{\omega}_1$, following a similar way which will be detailed in the update of policy θ_1 in the next step, we conclude that $\hat{\omega}_1^k$ converges to a minimizer $\hat{\omega}_1^*$. Following the analysis of policy evaluation in previous studies, assume that a sufficiently expressive function approximator is used such that there is only one local minimum, within the space $\hat{\Omega}_1$, which minimizes the TD error to 0, then we have shown that $\hat{\omega}_1^k \to \hat{\omega}_1^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k) \to Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^*) = Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)$.

Convergence of θ_1: Since $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{1,\dots,N\}}$ are updated on slower timescales than θ_1, again, according to [138],we treat $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{1,\dots,N\}}$ as fixed values when update θ_1, and additionally, according to aforementioned analysis we have $||Q^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \phi_1^k) - Q_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)|| \to 0$ and $||Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k) - Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)|| \to 0$ almost surely. With the function $Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)$, by using data following $\bar{\boldsymbol{\pi}}_1$ and extending the constraint from being based on $Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_0, \boldsymbol{u}_0)$ to being based on general $Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)$, the ideal Lagrangian function can be written as:

$$\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1, \lambda_1) = \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[- Q_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_t, \boldsymbol{u}_t) + \lambda_1 Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)\big]. \tag{5.24}$$

It is noteworthy that, although these constraints may appear similar to those commonly used in CMDP settings, the application of the max function ensures that only states where the constraints are violated are taken into consideration. Therefore, safety constraints are imposed on each state. Besides, because of the use of barrier certificate $H^1_{\bar{\boldsymbol{\pi}}_1}$, one violation of safety constraints happening at timestep t can be recorded and reflected in all $\hat{r}^{1,*}_k, k \leq t$ through the calculation of difference (5.15), and therefore this violation can be sufficiently suppressed during the update. Later, by using the sampled data to update θ_1, we have:

$$\begin{aligned}\theta_1^{k+1} =&\Gamma_{\Theta_1}\big[\theta_1^k - \eta_{2,1}\nabla_{u_t^1}(-Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k) + \lambda_1^k Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k))|_{u_t^1=\pi^1(\boldsymbol{x}_t;\theta_1^k)}\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k}\big]\\ =&\Gamma_{\Theta_1}\big[\theta_1^k - \eta_{2,1}[\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^k} + \delta\theta_1^{k+1} + \delta\theta_{1,\varepsilon}]\big]\end{aligned} \tag{5.25}$$

where

$$\begin{aligned}\delta\theta_1^{k+1} =& -\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{u_t^1}(-Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k) + \lambda_1^k Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k))|_{u_t^1=\pi^1(\boldsymbol{x}_t;\theta_1^k)}\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k}\big]\\ &+\nabla_{u_t^1}(-Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k) + \lambda_1^k Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k))|_{u_t^1=\pi^1(\boldsymbol{x}_t;\theta_1^k)}\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k}\end{aligned} \tag{5.26}$$

and

$$\begin{aligned}\delta\theta_{1,\varepsilon} =& \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{u_t^1}(-Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k) + \lambda_1^k Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k))|_{u_t^1=\pi^1(\boldsymbol{x}_t;\theta_1^k)}\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k}\\ &-\nabla_{u_t^1}(-Q^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_t,\boldsymbol{u}_t) + \lambda_1^k Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}_t,\boldsymbol{u}_t))|_{u_t^1=\pi^1(\boldsymbol{x}_t;\theta_1^k)}\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k}\big]\end{aligned} \tag{5.27}$$

Under Assumption 5.5:

1. We show that $\delta\theta_1^{k+1}$ is square integrable first. To be specific,

$$\begin{aligned}\mathbb{E}\big[||\delta\theta_1^{k+1}||^2|\mathcal{F}_{\theta_1,k}\big] \leq C_3||\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k}||^2_\infty \times (||\nabla_{u_t^1}Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k)||^2_\infty\\ + ||\lambda_1||^2_\infty||\nabla_{u_t^1}Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k)||^2_\infty)\end{aligned} \tag{5.28}$$

where $\mathcal{F}_{\theta_1,k} = \sigma(\theta_1^m, \delta\theta_1^m, m \leq k)$ is the filtration of θ_1^k generated by different independent

trajectories. Based on Assumption 5.5, we know:

$$
\begin{aligned}
||\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k}||_\infty^2 &\leq K_2(1+||\theta_i^k||_\infty^2)\\
||\nabla_{u_t^1}Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k)||_\infty^2 &\leq K_3(1+||u_t^1||_\infty^2)\\
||\nabla_{u_t^1}Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k)||_\infty^2 &\leq K_4(1+||u_t^1||_\infty^2)
\end{aligned}
\tag{5.29}
$$

where K_2, K_3, K_4 are constants and $||\lambda_1||_\infty^2$ is bounded. Therefore, $\mathbb{E}\big[||\delta\theta_1^{k+1}||^2|\mathcal{F}_{\theta_1,k}\big] \leq C_3K_2(1+||\theta_i^k||_\infty^2)[K_3(1+||u_t^1||_\infty^2)+||\lambda_1||_\infty^2K_4(1+||u_t^1||_\infty^2)] < \infty$, and $\delta\theta_1^{k+1}$ is square integrable.

2. Later, we have:

$$
\begin{aligned}
\delta\theta_{1,\varepsilon} &= \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{u_t^1}(Q^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_t,\boldsymbol{u}_t)-Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k)+\lambda_1^kQ_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k)-\lambda_1^kQ^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}_t,\boldsymbol{u}_t))|_{u_t^1=\pi^1(\boldsymbol{x}_t;\theta_1^k)}\\
\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k} &= \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{u_t^1}(Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^*(\{\theta_i^k\}_{i\in\mathcal{N}}))-Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k)\\
&+\lambda_1^kQ_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k)-\lambda_1^kQ_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^*(\{\theta_i^k\}_{i\in\mathcal{N}})))|_{u_t^1=\pi^1(\boldsymbol{x}_t;\theta_1^k)}\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k}\\
&\leq \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\theta_1}\pi^1(\boldsymbol{x}_t;\theta_1)|_{\theta_1=\theta_1^k}\big](K_5||\phi_1^k-\phi_1^*||_\infty+||\lambda_1||_\infty^2K_6||\hat{\omega}_1^k-\hat{\omega}_1^*||_\infty)\to 0
\end{aligned}
\tag{5.30}
$$

since in the previous timescale, we showed $\phi_1^k \to \phi_1^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ and $\hat{\omega}_1^k \to \hat{\omega}_1^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ when $k\to\infty$.

3. Since $\nabla_{\theta_1}(-Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k)+\lambda_1^kQ_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k))$ is a sample of $\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^k}$, we have that $\mathbb{E}\big[\delta\theta_1^{k+1}|\mathcal{F}_{\theta_1,k}\big]=0$

Due to these three facts, the policy θ_1 update is a stochastic approximation of a continuous system $\theta_i(t)$ [138] represented by:

$$
\dot{\theta}_1(t) = \Upsilon_{\Theta_1}\big[-\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)\big] \tag{5.31}
$$

where

$$
\Upsilon_{\Theta_1}\big[F(\theta_1)\big] \triangleq \lim_{\eta\to 0^+}\frac{\Gamma_{\Theta_1}(\theta_1+\eta F(\theta_1))-\Gamma_{\Theta_1}(\theta_1)}{\eta} \tag{5.32}
$$

is the left directional derivative of the function $\Gamma_{\Theta_1}(\theta_1)$ in the direction of $F(\theta_1)$. According to [137], we have:

$$\frac{d\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)}{dt} = \nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)^T\Upsilon_{\Theta_1}\big[-\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)\big] \leq 0 \tag{5.33}$$

which is non-zero if $||\Upsilon_{\Theta_1}\big[-\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)\big]|| \neq 0$. Therefore, for given fixed $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\mathcal{N}}$, define a Lyapunov function:

$$L_{\{\theta_i^k\}_{i\in\{2,\dots,N\}},\{\lambda_i^k\}_{i\in\mathcal{N}}}(\theta_1) = \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1^k) - \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*,\lambda_1^k) \tag{5.34}$$

where θ_1^* is a local minimum point. Then, there exists a ball centered at θ_1^* with a radius r_{θ_1} such that $\forall\theta_1 \in B_{r_{\theta_1}}(\theta_1^*) = \{\theta_1 \mid ||\theta_1-\theta_1^*|| \leq r_{\theta_1}\}$, $L_{\{\theta_i^k\}_{i\in\{2,\dots,N\}},\{\lambda_i^k\}_{i\in\mathcal{N}}}(\theta_1) \geq 0$. Given $d\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)/dt \leq 0$, according to the Lyapunov theory for stable systems [139], for any initial $\theta_1(0) \in B_{r_{\theta_1}}(\theta_1^*)$, the trajectory $\theta_1(t)$ converges to θ_1^*, and thus $\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*,\lambda_1) \leq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1(t),\lambda_1) \leq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1(0),\lambda_1)$, $\forall t \geq 0$. Based on all aforementioned analysis and [137, Proposition 17], as well as the fact that θ_1^k is bounded in set Θ_1, according to [138, Theorem 2, Chapter 6], the sequence $\{\theta_1^k\}$ converges almost surely to the locally minimum point θ_1^*.

Convergence of λ_1: Since λ_1 is updated at a slower timescale than ϕ_1^k, $\hat{\omega}_1^k$ and θ_1^k, we have that, given fixed $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{2,\dots,N\}}$, $||\theta_1^k-\theta_1^*(\lambda_1^k)|| = 0$, $||Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k) - Q^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x}_t,\boldsymbol{u}_t)|| = 0$ and $||Q^1_H(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k) - Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}_t,\boldsymbol{u}_t)|| = 0$. With the continuity of $\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)$, we have $||\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^k,\lambda_1=\lambda_1^k} - \nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1^k),\lambda_1=\lambda_1^k}|| = 0$ almost surely. Thus, the update of λ_1 based on the sampled data is:

$$\begin{aligned}\lambda_1^{k+1} =& \Gamma_{\Lambda_1}\big[\lambda_1^k + \eta_{3,1}Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k)\big]\\ =& \Gamma_{\Lambda_1}\big[\lambda_1^k + \eta_{3,1}[\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1^k),\lambda_1=\lambda_1^k} + \delta\lambda_1^{k+1}]\big]\end{aligned} \tag{5.35}$$

where

$$\begin{aligned}\delta\lambda_1^{k+1} &= -\mathbb{E}\big[Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x}_t,\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}_t))\big] + Q_H^1(\boldsymbol{x}_t,\bar{\boldsymbol{\pi}}_1(\boldsymbol{x}_t);\hat{\omega}_1^k)\\ &= -\mathbb{E}\big[Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x}_t,\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}_t))\big] + Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}_t,\bar{\boldsymbol{\pi}}_1(\boldsymbol{x}_t)) - Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}_t,\bar{\boldsymbol{\pi}}_1(\boldsymbol{x}_t)) + Q_H^1(\boldsymbol{x}_t,\bar{\boldsymbol{\pi}}_1(\boldsymbol{x}_t);\hat{\omega}_1^k)\end{aligned} \tag{5.36}$$

Here, $\bar{\boldsymbol{\pi}}_1^*$ is the joint policy parameterized by $\{\theta_1^*(\lambda_1^k), \theta_2^k, \ldots, \theta_N^k\}$ where $\theta_1^*(\lambda_1^k)$ is the locally minimum solution given a fixed λ_1^k, and $\{\theta_i^k\}_{i\in\{2,\ldots,N\}}$ are certain policies for other agents. Similarly, here we have:

$$\mathbb{E}\big[||\delta\lambda_1^{k+1}||^2|\mathcal{F}_{\lambda_1,k}\big] \leq C_4 \max_{\boldsymbol{x}_t,\boldsymbol{u}_t,\hat{\omega}_1^k} ||Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k)||_\infty^2 < \infty \tag{5.37}$$

according to Assumption 5.3, where $\mathcal{F}_{\lambda_1,k} = \sigma(\lambda_1^m, \delta\lambda_1^m, m \leq k)$ is the filtration of λ_1^k generated by different independent trajectories, and since $||Q_H^1(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}_t);\hat{\omega}_1^k) - Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}_t))|| = 0$, and $Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}_t))$ is a sample of $\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1^k),\lambda_1=\lambda_1^k}$, we have $\mathbb{E}\big[\delta\lambda_1^{k+1}|\mathcal{F}_{\lambda_1,k}\big] = 0$ almost surely. Thus, similar to previous analysis, the λ_1's update is a stochastic approximation of the following system:

$$\dot{\lambda}_1(t) = \Upsilon_{\Lambda_1}\big[\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1)}\big]. \tag{5.38}$$

Similarly, we have:

$$\frac{d\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1),\lambda_1)}{dt} = \nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1)}{}^T\Upsilon_{\Lambda_1}\big[\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1)}\big] \geq 0 \tag{5.39}$$

For a local maximum point λ_1^*, define the following Lyapunov function:

$$L_{\{\theta_i^k\}_{i\in\{2,\ldots,N\}},\{\lambda_i^k\}_{i\in\{2,\ldots,N\}}}(\lambda_1) = \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1^*),\lambda_1^*) - \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1),\lambda_1), \tag{5.40}$$

there exists a ball centered at λ_1^* with a radius r_{λ_1} such that $\forall\lambda_1 \in B_{r_{\lambda_1}}(\lambda_1^*) = \{\lambda_1 \mid ||\lambda_1 - \lambda_1^*|| \leq r_{\lambda_1}\}$, $L_{\{\theta_i^k\}_{i\in\{2,\ldots,N\}},\{\lambda_i^k\}_{i\in\{2,\ldots,N\}}}(\lambda_1) \geq 0$. Furthermore,

$$dL_{\{\theta_i^k\}_{i\in\{2,\ldots,N\}},\{\lambda_i^k\}_{i\in\{2,\ldots,N\}}}(\lambda_1)/dt = -d\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1),\lambda_1)/dt \leq 0. \tag{5.41}$$

Therefore, starting from $\lambda_1(0) \in B_{r_{\lambda_1}}(\lambda_1^*)$, the trajectory $\lambda_1(t)$ converges to λ_1^*, which is a locally maximum point. Following the similar process in the update of θ_1, we conclude that $\{\lambda_1^k\}$ converges almost surely to λ_1^*, with $\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1^*),\lambda_1^*) \geq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1),\lambda_1)$, $\forall t \geq 0$.

From the aforementioned analysis, we have $\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*,\lambda_1) \leq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*,\lambda_1^*) \leq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1^*)$.

Therefore, $(\theta_1^*, \lambda_1^*)$ is a locally saddle point for the certain fixed $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{2,\dots,N\}}$, and the update of θ_1 and λ_1 will converge to the feasible locally optimal policy and multiplier with respect to its value function for the fixed set of policies $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and multipliers $\{\lambda_i^k\}_{i\in\{2,\dots,N\}}$ of other agents that have not been updated.

Later we analyse the update of the second agent. For the agent 2, according to the previous analysis, it is already obtained that $\lambda_1^k \to \lambda_1^*(\{\theta_i^k\}_{i\in\{2,\dots,N\}}, \{\lambda_i^k\}_{i\in\{2,\dots,N\}})$ and $\theta_1^k \to \theta_1^*(\{\theta_i^k\}_{i\in\{2,\dots,N\}}, \{\lambda_i^k\}_{i\in\{2,\dots,N\}})$ when we update θ_2^k and λ_2^k. Since agent 2 is updated at a quicker timescale than agent 3 to agent N, here we still have $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$ as fixed sets. Let $\bar{\boldsymbol{\pi}}_2 = (\theta_1^k, \theta_2^k, \dots, \theta_N^k)$ where $\theta_1^k \to \theta_1^*(\theta_2^k, \lambda_2^k)$ and $\lambda_1^k \to \lambda_1^*(\theta_2^k, \lambda_2^k)$, then, similar to the previous proof, we have $\phi_2^k \to \phi_2^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $Q^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \phi_2^k) \to Q^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \phi_2^*) = Q^2_{\bar{\boldsymbol{\pi}}_2}(\boldsymbol{x}_t, \boldsymbol{u}_t)$, and $\omega_2^k \to \omega_2^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $H^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_2^k) \to H^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_2^*) = H^2_{\bar{\boldsymbol{\pi}}_2}(\boldsymbol{x}_t, \boldsymbol{u}_t)$. Further, define $\hat{r}_t^2$ and $Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}$ similarly to $\hat{r}_t^1$ and $Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}$, and we use $\hat{\omega}_2$ to parameterize $Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}$, under Assumption 5.3, we can conclude that $\hat{\omega}_2^k \to \hat{\omega}_2^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $Q^2_H(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_2^k) \to Q^2_H(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_2^*) = Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \boldsymbol{u}_t)$. Then, we can analyze the convergence of θ_2 and λ_2.

Convergence of θ_2: For the convergence of θ_2, define $\bar{\boldsymbol{\pi}}_2^* = (\theta_1^*, \{\theta_i^k\}_{i\in\{2,\dots,N\}})$ which differs from $\bar{\boldsymbol{\pi}}_2 = (\theta_1^k, \theta_2^k, \dots, \theta_N^k)$ by θ_1^* and θ_1^k. Based on previous analysis, here we have $\theta_1^k \to \theta_1^*$ and therefore $\bar{\boldsymbol{\pi}}_2 \to \bar{\boldsymbol{\pi}}_2^*$ for θ_2^k which is updated in this timescale under certain fixed $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$. The ideal Lagrangian function to use now is:

$$\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2, \lambda_2) = \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[- Q^2_{\bar{\boldsymbol{\pi}}_2^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) + \lambda_2 Q^2_{H_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x}_t, \boldsymbol{u}_t)\big]. \tag{5.42}$$

where $\boldsymbol{\theta}^{-2}$ means θ_2 is updated, and during its update, $\theta_1^k = \theta_1^*(\{\theta_i^k\}_{i\in\{2,\dots,N\}}, \{\lambda_i^k\}_{i\in\{2,\dots,N\}})$ with $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$ being certain fixed values. Then, by sampling data from the distribution to update θ_2, we have:

$$\begin{aligned}\theta_2^{k+1} =& \Gamma_{\Theta_2}\big[\theta_2^k - \eta_{2,2}\nabla_{u_i^2}(-Q^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \phi_2^k) + \lambda_2^k Q^2_H(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_2^k))|_{u_i^2=\pi^2(\boldsymbol{x}_t;\theta_2^k)}\nabla_{\theta_2}\pi^2(\boldsymbol{x}_t;\theta_2)|_{\theta_2=\theta_2^k}\big] \\ =& \Gamma_{\Theta_2}\big[\theta_2^k - \eta_{2,2}[\nabla_{\theta_2}\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2, \lambda_2)|_{\theta_2=\theta_2^k} + \delta\theta_2^{k+1} + \delta\theta_{2,\varepsilon}]\big]\end{aligned} \tag{5.43}$$

where

$$\delta\theta_2^{k+1} = -\mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{u_t^2}(-Q^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_2^k) + \lambda_2^k Q_H^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_2^k))|_{u_t^2=\pi^2(\boldsymbol{x}_t;\theta_2^k)}\nabla_{\theta_2}\pi^2(\boldsymbol{x}_t;\theta_2)|_{\theta_2=\theta_2^k}\big]$$
$$+ \nabla_{u_t^2}(-Q^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_2^k) + \lambda_2^k Q_H^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_2^k))|_{u_t^2=\pi^2(\boldsymbol{x}_t;\theta_2^k)}\nabla_{\theta_2}\pi^2(\boldsymbol{x}_t;\theta_2)|_{\theta_2=\theta_2^k} \tag{5.44}$$

and

$$\delta\theta_{2,\varepsilon} = \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{u_t^2}(-Q^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_2^k) + \lambda_2^k Q_H^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_2^k))|_{u_t^2=\pi^2(\boldsymbol{x}_t;\theta_2^k)}\nabla_{\theta_2}\pi^2(\boldsymbol{x}_t;\theta_2)|_{\theta_2=\theta_2^k}$$
$$- \nabla_{u_t^2}(-Q^2_{\bar{\boldsymbol{\pi}}_2^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) + \lambda_2^k Q^2_{H_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x}_t,\boldsymbol{u}_t))|_{u_t^2=\pi^2(\boldsymbol{x}_t;\theta_2^k)}\nabla_{\theta_2}\pi^2(\boldsymbol{x}_t;\theta_2)|_{\theta_2=\theta_2^k}\big]. \tag{5.45}$$

1. We have the following:

$$\mathbb{E}\big[||\delta\theta_2^{k+1}||^2|\mathcal{F}_{\theta_2,k}\big] \le C_5||\nabla_{\theta_2}\pi^2(\boldsymbol{x}_t;\theta_2)|_{\theta_2=\theta_2^k}||_\infty^2 \times (||\nabla_{u_t^2}Q^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_2^k)||_\infty^2$$
$$+ ||\lambda_2||_\infty^2||\nabla_{u_t^2}Q_H^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_2^k)||_\infty^2) < \infty, \tag{5.46}$$

where $\mathcal{F}_{\theta_2,k} = \sigma(\theta_2^m, \delta\theta_2^m, m \le k)$ is the filtration of θ_2^k generated by different independent trajectories, and thus $\delta\theta_2^{k+1}$ is square integrable.

2. Under Assumption 5.6, we have:

$$\delta\theta_{2,\varepsilon} = \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{u_t^2}(Q^2_{\bar{\boldsymbol{\pi}}_2^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) - Q^2_{\bar{\boldsymbol{\pi}}_2}(\boldsymbol{x}_t,\boldsymbol{u}_t) + Q^2_{\bar{\boldsymbol{\pi}}_2}(\boldsymbol{x}_t,\boldsymbol{u}_t) - Q^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_2^k) + \lambda_2^k(Q_H^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_2^k)$$
$$- Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t,\boldsymbol{u}_t) + Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t,\boldsymbol{u}_t) - Q^2_{H_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x}_t,\boldsymbol{u}_t)))|_{u_t^2=\pi^2(\boldsymbol{x}_t;\theta_2^k)}\nabla_{\theta_2}\pi^2(\boldsymbol{x}_t;\theta_1)|_{\theta_2=\theta_2^k}$$
$$\le \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{\theta_2}\pi^2(\boldsymbol{x}_t;\theta_2)|_{\theta_2=\theta_2^k}\big](K_7||\bar{\boldsymbol{\pi}}_2^* - \bar{\boldsymbol{\pi}}_2||_\infty + K_8||\phi_2^k - \phi_2^*||_\infty + K_9||\hat{\omega}_2^k - \hat{\omega}_2^*||_\infty) \to 0 \tag{5.47}$$

3. Since $\nabla_{\theta_2}(-Q^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_2^k) + \lambda_2^k Q_H^2(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_2^k))$ is a sample of $\nabla_{\theta_2}\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2,\lambda_2)|_{\theta_2=\theta_2^k}$ given $\bar{\boldsymbol{\pi}}_2 \to \bar{\boldsymbol{\pi}}_2^*$, one can obtain that $\mathbb{E}\big[\delta\theta_2^{k+1}|\mathcal{F}_{\theta_2,k}\big] = 0$ almost surely.

Then, similar to the previous section using Lyapunov function, we conclude that the sequence $\{\theta_2^k\}$ converges to a locally minimum point θ_2^* such that $\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2^*,\lambda_2) \le \mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2(t),\lambda_2) \le \mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2(0),\lambda_2)$, $\forall t \ge 0$.

Convergence of λ_2: Similar to the previous section on λ_1, we can treat $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$ as fixed when update λ_2, and $||\theta_2^k - \theta_2^*|| = 0$. Then, the update of λ_2 can be written as:

$$\begin{aligned}\lambda_2^{k+1} =&\Gamma_{\Lambda_2}\big[\lambda_2^k + \eta_{3,2} Q_H^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_2^k)\big] \\ =&\Gamma_{\Lambda_2}\big[\lambda_2^k + \eta_{3,2}[\nabla_{\lambda_2}\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2, \lambda_2)|_{\theta_2=\theta_2^*(\lambda_2^k),\lambda_2=\lambda_2^k} + \delta\lambda_2^{k+1}]\big]\end{aligned} \tag{5.48}$$

where

$$\begin{aligned}\delta\lambda_2^{k+1} =& -\mathbb{E}\big[Q^2_{H_{\bar{\boldsymbol{\pi}}^*_{2,*}}}(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}^*_{2,*}(\boldsymbol{x}_t))\big] + Q_H^2(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t); \hat{\omega}_2^k) \\ =& -\mathbb{E}\big[Q^2_{H_{\bar{\boldsymbol{\pi}}^*_{2,*}}}(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}^*_{2,*}(\boldsymbol{x}_t))\big] + Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t)) - Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t)) + Q_H^2(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t); \hat{\omega}_2^k)\end{aligned} \tag{5.49}$$

where $\bar{\boldsymbol{\pi}}^*_{2,*} = (\theta_1^*, \theta_2^*(\lambda_2^k), \{\theta_i^k\}_{i\in\{3,\dots,N\}})$, $\bar{\boldsymbol{\pi}}_2 = (\theta_1^k, \theta_2^k, \{\theta_i^k\}_{i\in\{3,\dots,N\}})$ where $\theta_1^k \to \theta_1^*$ and $\theta_2^k \to \theta_2^*(\lambda_2^k)$ according to previous analysis. Therefore,

$$\mathbb{E}\big[||\delta\lambda_2^{k+1}||^2|\mathcal{F}_{\lambda_2,k}\big] \le C_6 \max_{\boldsymbol{x}_t,\boldsymbol{u}_t,\hat{\omega}_2^k} ||Q_H^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_2^k)||_\infty^2 < \infty, \tag{5.50}$$

according to Assumption 5.3. Since based on aforementioned analysis, $||Q_H^2(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t); \hat{\omega}_2^k) - Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t))|| \to 0$, and $Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t))$ is a sample of $\nabla_{\lambda_2}\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2, \lambda_2)|_{\theta_2=\theta_2^*(\lambda_2^k),\lambda_2=\lambda_2^k}$, we have $\mathbb{E}\big[\delta\lambda_2^{k+1}|\mathcal{F}_{\lambda_2,k}\big] = 0$ almost surely.

Then, similar to the previous section, we have that λ_2^k converges to λ_2^* which is a local maximum point, with $\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2^*(\lambda_2^*), \lambda_2^*) \ge \mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2^*(\lambda_2), \lambda_2)$, $\forall t \ge 0$. Therefore, $(\theta_2^*, \lambda_2^*)$ is a locally saddle point for the certain fixed $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$ of other agents that have not been updated, while θ_1 and λ_1 have already converged to the local optimal policy and multiplier for the agent 1 with respect to $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{2,\dots,N\}}$. Namely, we finally obtain (θ_1^*, θ_2^*) and $(\lambda_1^*, \lambda_2^*)$ with respect to $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$. This process will iterate until the agent N finishes its update, and we have a feasible joint policy $\{\theta_i^*\}_{i\in\mathcal{N}}$, where for each i, θ_i^* is a feasible policy. Furthermore, considering this obtained joint policy $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$ parameterized by $\{\theta_i^*\}_{i\in\mathcal{N}}$, denote $\boldsymbol{\theta}^{-i,*}$ as the obtained joint policy except agent i, then, consider another joint policy $\boldsymbol{\pi}^*_{-i}$ parameterized by $\{\theta_i, \boldsymbol{\theta}^{-i,*}\}$ where $\theta_i \in B_{r_{\theta_i}}(\theta_i^*)\backslash\theta_i^*$

is a feasible policy, we have :

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\pi}^*}\big[-Q^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t)\big] = \mathbb{E}_{\boldsymbol{\pi}^*}\big[-Q^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t)+\lambda_i^* Q^i_{H_{\boldsymbol{\pi}^*}}(\boldsymbol{x}_t,\boldsymbol{u}_t)\big] = \mathcal{L}_{\boldsymbol{\theta}^{-i,*}}(\theta_i^*,\lambda_i^*)\\
&\leq \mathcal{L}_{\boldsymbol{\theta}^{-i,*}}(\theta_i,\lambda_i^*) = \mathbb{E}_{\boldsymbol{\pi}^*_{-i}}\big[-Q^i_{\boldsymbol{\pi}^*_{-i}}(\boldsymbol{x}_t,\boldsymbol{u}_t)+\lambda_i^* Q^i_{H_{\boldsymbol{\pi}^*_{-i}}}(\boldsymbol{x}_t,\boldsymbol{u}_t)\big] = \mathbb{E}_{\boldsymbol{\pi}^*_{-i}}\big[-Q^i_{\boldsymbol{\pi}^*_{-i}}(\boldsymbol{x}_t,\boldsymbol{u}_t)\big],
\end{aligned} \tag{5.51}$$

since $\theta_i^* = \underset{\theta_i\in B_{r_{\theta_i}}(\theta_i^*)}{\arg\min}\ \mathcal{L}_{\boldsymbol{\theta}^{-i,*}}(\theta_i,\lambda_i^*)$. Thus, $\mathbb{E}_{\boldsymbol{\pi}^*}\big[Q^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t)\big] \geq \mathbb{E}_{\boldsymbol{\pi}^*_{-i}}\big[Q^i_{\boldsymbol{\pi}^*_{-i}}(\boldsymbol{x}_t,\boldsymbol{u}_t)\big]$ for all feasible policy $\theta_i \in B_{r_{\theta_i}}(\theta_i^*)\backslash\theta_i^*$, $\forall i \in \mathcal{N}$. Therefore, the obtained joint policy $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$ is a feasible local Nash equilibrium.

Now we show that $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) \geq 0$ for $(\boldsymbol{x}_t,\boldsymbol{u}_t)$ along the trajectory under $\boldsymbol{\pi}^*$, $\forall t \geq 0$ and $\forall i \in \mathcal{N}$: According to the definition, $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0) \leq 0$, $\forall i \in \mathcal{N}$. Consider any $i \in \mathcal{N}$ and let $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0) = p \leq 0$ with $\boldsymbol{u}_0$ being any control signal generated by the joint policy at $\boldsymbol{x}_0$, according to the barrier certificate, we have $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_1,\boldsymbol{u}_1) \geq (1-\epsilon)H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0)$ for all following $(\boldsymbol{x}_1,\boldsymbol{u}_1)$. Furthermore, according to the definition of barrier certificate, $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0) = b_0^i + \gamma\mathbb{E}_{\boldsymbol{x}_1,\boldsymbol{u}_1}[H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_1,\boldsymbol{u}_1)] \geq (1-\epsilon)\gamma p$ given $b_0^i = 0$. Thus, we have $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0) = p \geq (1-\epsilon)\gamma p$ at convergence. Since $p \leq 0$ and $1-\epsilon \in (0,1)$, $\gamma \in (0,1)$, we have $p = 0$, for all $i \in \mathcal{N}$ and all possible generated control signal $\boldsymbol{u}_0$. The similar process applies for all $t \geq 0$, and we have $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) \geq 0$, $\forall t \geq 0$ and $\forall i \in \mathcal{N}$.

Furthermore, for a state $\boldsymbol{x}_t$ along the trajectory under the control policy $\boldsymbol{\pi}^*$, let $\boldsymbol{x}_{t+1}$ represent the next state following $\boldsymbol{u}_t$ generated by $\boldsymbol{\pi}^*$, and let i_A and i_B denote the index of the maximum $H_{\boldsymbol{\pi}^*}$ at $(\boldsymbol{x}_t,\boldsymbol{u}_t)$ and $(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1})$, respectively, namely:

$$H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) = \min_{i\in\mathcal{N}} H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) = H^{i_A}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) \tag{5.52}$$

$$H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1}) = \min_{i\in\mathcal{N}} H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1}) = H^{i_B}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1}) \tag{5.53}$$

Then, given $1-\epsilon \in \mathcal{K}_\infty$, we have:

$$\begin{aligned}
H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1}) &= H^{i_B}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1}) \geq (1-\epsilon)H^{i_B}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t)\\
&\geq (1-\epsilon)H^{i_A}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) = (1-\epsilon)H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t),
\end{aligned} \tag{5.54}$$

since $\forall i \in \mathcal{N}$ and $\forall t \geq 0$, $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) \geq (1-\epsilon) H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ for all $\boldsymbol{u}_t$ generated by $\boldsymbol{\pi}^*$. Besides, it is easily known that $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) \geq 0$ implies that $\forall i \in \mathcal{N}$, $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) \geq 0$, which means safety is maintained for all agents. Additionally, $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) < 0$ if $\exists i \in \mathcal{N}$ such that $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) < 0$, which means at least one agent will experience safety violation with a positive probability. Therefore, $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ is a global barrier certificate for the whole multi-agent system, and the corresponding safe set is given as:

$$\mathcal{X}^g := \{\boldsymbol{x} \mid H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}, \boldsymbol{\pi}^*(\boldsymbol{x})) \geq 0\} = \cap_{i \in \mathcal{N}} \{\boldsymbol{x} \mid H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}, \boldsymbol{\pi}^*(\boldsymbol{x})) \geq 0\}. \tag{5.55}$$

□

5.6.4 D. Appendix for State-Wise Stability via ISS and Infeasibility under Multiple Constraints

(i) Proof for Theorem 5.2

Proof. This proof for Theorem 5.2 is developed based on previous papers on input-to-state stability [132, 140]. Define operators $A(\boldsymbol{V}(\boldsymbol{x}_t)) = [\alpha_1 V^1(x_t^1), \alpha_2 V^2(x_t^2), \ldots, \alpha_N V^1(x_t^N)]^T$ where $\boldsymbol{V}(\boldsymbol{x}_t) = [V^1(x_t^1), V^2(x_t^2), \ldots, V^N(x_t^N)]^T$ and $\dot{\boldsymbol{V}}(\boldsymbol{x}_t) = [\dot{V}^1(x_t^1), \dot{V}^2(x_t^2), \ldots, \dot{V}^N(x_t^N)]^T$, and $\{\alpha_i\}_{i \in \mathcal{N}}$ is the set of $\mathcal{K}_\infty$ functions.

Further, define the operator

$$\Delta(\boldsymbol{V}(\boldsymbol{x}_t)) = [\sum_{j \neq 1} \delta_{1j}(V^j(x_t^j)), \sum_{j \neq 2} \delta_{2j}(V^j(x_t^j)), \ldots, \sum_{j \neq N} \delta_{Nj}(V^j(x_t^j))]^T,$$

and the operator $K(||\boldsymbol{d}_t||) = [\kappa_1(||\boldsymbol{d}_t||), \kappa_2(||\boldsymbol{d}_t)||), \ldots, \kappa_N(||\boldsymbol{d}_t||)]^T$ to represent the gain terms describing the influence from agent $j \neq i$ to the agent i, $\forall i \in \mathcal{N}$, and the terms describing the influence from the external disturbance $\boldsymbol{d}_t$. Then, the constraint (5.4) can be written as:

$$\dot{\boldsymbol{V}}(\boldsymbol{x}_t) \leq (-A + \Delta)(\boldsymbol{V}(\boldsymbol{x}_t)) + K(||\boldsymbol{d}_t||) \tag{5.56}$$

for all $\boldsymbol{x}_t$ sampled under the joint policy $\boldsymbol{\pi}$. Based on [132], we need to first find a function V_{MAS} for the whole multi-agent system satisfying:

$$\dot{V}_{\text{MAS}}(\boldsymbol{x}_t) \leq -\alpha_{\text{MAS}}(V_{\text{MAS}}(\boldsymbol{x}_t)) + K_{\text{MAS}}(||\boldsymbol{d}_t||). \tag{5.57}$$

Thus, define $Y^i(x_t^i) = C \times V^i(x_t^i),\ i = 1, 2, \ldots, n$, where $C > 0$ is a constant. Let $V_{\text{MAS}}(\boldsymbol{x}_t) = \max_{i \in \mathcal{N}} Y^i(x_t^i)$, then the following properties are satisfied:

1. $Y^i(x_t^i)$ is strictly increasing, which means when Lyapunov function $V^i(x_t^i)$ grows, its contribution to the V_{MAS} will not shrink.
2. Since $Y^i(x_t^i) = C \times V^i(x_t^i) = \int_0^{V^i(x_t^i)} C(y)dy$ where $C(y) = C > 0$, we have $\int_1^\infty C(y)dy = \infty$ because $C(y)$ does not approach zero asymptotically as $V^i(x_t^i)$ goes to infinity, which means at large states, each $V^i(x_t^i)$ can make non-negligible contributions to $V_{\text{MAS}}(\boldsymbol{x}_t)$ if it is selected, and help to analyse the stability at states with large norms.

Define $\boldsymbol{Y}(\boldsymbol{x}_t) = [Y^1(x_t^1), Y^2(x_t^2) \ldots, Y^N(x_t^N)]^T$, and $\dot{\boldsymbol{Y}}(\boldsymbol{x}_t) = [\dot{Y}^1(x_t^1), \dot{Y}^2(x_t^2), \ldots, \dot{Y}^N(x_t^N)]^T$, with a set of $\mathcal{K}_\infty$ functions $\{\beta_i\}_{i \in \mathcal{N}}$, define

$$D(\boldsymbol{V}(\boldsymbol{x}_t)) = [\sum_{j \neq 1} \delta_{1j}(V^j(x_t^j)) + \beta_1(\sum_{j \neq 1} \delta_{1j}(V^j(x_t^j))), \ldots, \sum_{j \neq N} \delta_{Nj}(V^j(x_t^j)) + \beta_N(\sum_{j \neq N} \delta_{Nj}(V^j(x_t^j)))]^T \tag{5.58}$$

and let ι_i be defined according to $\text{Id} - \iota_i = (\text{Id} + \beta_i)^{-1}$ where Id denotes the identity map, then $\iota_i \in \mathcal{K}_\infty$. Pick $\zeta_i \in \mathcal{K}_\infty$ satisfying $\iota_i - \zeta_i \in \mathcal{K}_\infty$, $\zeta_i \in \mathcal{K}_\infty$ and $\text{Id} - \zeta_i \in \mathcal{K}_\infty$, and construct the operator F to be $F(\boldsymbol{s}) = [\zeta_1(s_1), \ldots, \zeta_N(s_N)]^T$ where $\boldsymbol{s} = [s_1, s_2, \ldots, s_N]$. Then, we have:

$$\begin{aligned}
\dot{\boldsymbol{Y}}(\boldsymbol{x}_t) &\leq C[(-A + \Delta)(\boldsymbol{V}(\boldsymbol{x}_t)) + K(||\boldsymbol{d}_t||)] \\
&= C[\{-(\text{Id} - F + F) \circ A + \Delta\}(\boldsymbol{V}(\boldsymbol{x}_t)) + K(||\boldsymbol{d}_t||)] \\
&= C\{-(\text{Id} - F) \circ A + \Delta\}(\boldsymbol{V}(\boldsymbol{x}_t)) + C[-F \circ A(\boldsymbol{V}(\boldsymbol{x}_t)) + K(||\boldsymbol{d}_t||)] \\
&\leq C\{-(\text{Id} - F) \circ A + \Delta\}(\boldsymbol{V}(\boldsymbol{x}_t)) + CK(||\boldsymbol{d}_t||)
\end{aligned} \tag{5.59}$$

Suppose at the state $\boldsymbol{x}_t$, the maximization of $Y^i(x_t^i)$ is attained at $i = p \in \mathcal{N}$, then, $V_{\text{MAS}}(\boldsymbol{x}_t) = Y^p(x_t^p) = C \times V^p(x_t^p)$ and the above inequality yields:

$$\begin{aligned}\dot{V}_{\text{MAS}}(\boldsymbol{x}_t) &= C \times \dot{V}^p(x_t^p) \\ &\leq C\{\big(-(\text{Id} - \zeta_p) + (\text{Id} + \beta_p)^{-1} - (\text{Id} + \beta_p)^{-1}\big) \circ \alpha_p\}(V^p(x_t^p)) + C[\Delta(\boldsymbol{V}(\boldsymbol{x}_t))]_p + C[K(||\boldsymbol{d}_t||)]_p \\ &= C\{-(\iota_p - \zeta_p) \circ \alpha_p\}(V^p(x_t^p)) + C\{-(\text{Id} + \beta_p)^{-1} \circ \alpha_p\}(V^p(x_t^p)) + C[\Delta(\boldsymbol{V}(\boldsymbol{x}_t))]_p + C[K(||\boldsymbol{d}_t||)]_p.\end{aligned} \tag{5.60}$$

Given

$$(\text{Id} + \beta_p)([\Delta(\boldsymbol{V}(\boldsymbol{x}_t))]_p) = \sum_{j \neq p} \delta_{pj}(V^j(x_t^j)) + \beta_p(\sum_{j \neq p} \delta_{pj}(V^j(x_t^j))) \leq \alpha_p(V^p(x_t^p)) \tag{5.61}$$

according to the assumption in Theorem 5.2, we have $(\text{Id} + \beta_p)^{-1}\alpha_p(V^p(x_t^p)) \geq [\Delta(\boldsymbol{V}(\boldsymbol{x}_t))]_p$, and therefore

$$\dot{V}_{\text{MAS}}(\boldsymbol{x}_t) \leq -C(\iota_p - \zeta_p) \circ \alpha_p(V^p(x_t^p)) + C\kappa_p(||\boldsymbol{d}_t||), \tag{5.62}$$

where $C(\iota_p - \zeta_p) \circ \alpha_p$ and $C\kappa_p$ are $\mathcal{K}_\infty$ functions. Therefore, there exists $\hat{\alpha}_p \in \mathcal{K}_\infty$ and $\hat{\kappa}_p \in \mathcal{K}_\infty$ such that $\dot{V}_{\text{MAS}}(\boldsymbol{x}_t) \leq -\hat{\alpha}_p(V^p(x_t^p)) + \hat{\kappa}_p(||\boldsymbol{d}_t||)$. Furthermore, repeating the above process for $p \in \mathcal{N}$, and since $V_{\text{MAS}}(\boldsymbol{x}_t) = C \times V^p(x_t^p)$, we have $\dot{V}_{\text{MAS}}(\boldsymbol{x}_t) \leq -\alpha_{\text{MAS}}(V_{\text{MAS}}(\boldsymbol{x}_t)) + K_{\text{MAS}}(||\boldsymbol{d}_t||)$ where α_{MAS} and K_{MAS} are defined by:

$$\begin{aligned}\alpha_{\text{MAS}}(s) &= \min_i \hat{\alpha}_i(\frac{s}{C}), \\ K_{\text{MAS}}(s) &= \max_i \hat{\kappa}_i(s).\end{aligned} \tag{5.63}$$

Later, we prove that $V_{\text{MAS}}(\boldsymbol{x}_t)$ can be lower-bounded and upper-bounded by two $\mathcal{K}_\infty$ functions. For each $V^i(x_t^i)$, it is natural that $||x_t^i|| \leq \mathbb{E}_{\tau^i \sim \pi^i}\big[\sum_{k=0}^{\infty} \gamma^k c^i(x_{t+k}^i)\big] = V^i(x_t^i)$ given τ^i starts from x_t^i and $c^i(x_t^i) = ||x_t^i||$ when the desired goal is, without loss of generality, at the origin. Then, under the assumption in Theorem 5.2 we have that $V^i(x_t^i) \leq \mu_i(||x_t^i||)$, for $\mu_i \in \mathcal{K}_\infty$ under the policy. This is not unreasonable since given the state space is bounded, $V^i(x_t^i) = \mathbb{E}_{\tau^i \sim \pi^i}\big[\sum_{k=0}^{\infty} \gamma^k c^i(x_{t+k}^i)\big] \leq \frac{||x^i||_\infty}{1-\gamma}$ where $||x^i||_\infty$ is the maximum norm of x_t^i in the bounded

state space, and therefore $V^i(x_t^i)$ is upper bounded. Given $||x_t^i|| \leq V^i(x_t^i) \leq \mu_i(||x_t^i||)$, we have the following properties:

1. Since $V_{\text{MAS}}(\boldsymbol{x}_t) = \max_{i\in\mathcal{N}} C \times V^i(x_t^i)$, suppose $p = \arg\max_i C \times V^i(x_t^i)$, we have $V_{\text{MAS}}(\boldsymbol{x}_t) = C \times V^p(x_t^p) \leq C \times \mu_p(||x_t^p||) \leq C \times \mu_p(||\boldsymbol{x}_t||) \leq C \times (\sum_{k=1}^N \mu_i)(||\boldsymbol{x}_t||) = \hat{\mu}_{\text{MAX}}(||\boldsymbol{x}_t||)$.

2. $V_{\text{MAS}}(\boldsymbol{x}_t) = C \times V^p(x_t^p) \geq C \times V^i(x_t^i) = C \times \sum_{k=0}^{\infty} \gamma_c^k ||x_{t+k}^i|| \geq C \times ||x_t^i||$, $\forall i \in \mathcal{N}$. Select $p' = \max_i ||x_t^i||$, we thus have $V_{\text{MAS}}(\boldsymbol{x}_t) \geq C \times ||x_t^{p'}|| \geq C\frac{||\boldsymbol{x}_t||}{\sqrt{N}} \geq \hat{\mu}_{\text{MIN}}(||\boldsymbol{x}_t||)$ since $||\boldsymbol{x}_t|| = \sqrt{\sum_{i=1}^N ||x_t^i||^2} \leq \sqrt{N \times ||x_t^{p'}||^2} = \sqrt{N}||x_t^{p'}||$.

In sum, we have

$$\hat{\mu}_{\text{MIN}}(||\boldsymbol{x}_t||) \leq V_{\text{MAS}}(\boldsymbol{x}_t) \leq \hat{\mu}_{\text{MAX}}(||\boldsymbol{x}_t||) \tag{5.64}$$

and $\dot{V}_{\text{MAS}}(\boldsymbol{x}_t) \leq -\alpha_{\text{MAS}}(V_{\text{MAS}}(\boldsymbol{x}_t)) + K_{\text{MAS}}(||\boldsymbol{d}_t||) \leq -\hat{\alpha}_{\text{MAS}}(||\boldsymbol{x}_t||) + K_{\text{MAS}}(||\boldsymbol{d}_t||)$ where $\hat{\alpha}_{\text{MAS}}(s) = \alpha_{\text{MAS}} \circ \hat{\mu}_{\text{MIN}}(s)$. Based on [132, Remark 2.4], we have $||\boldsymbol{x}_t|| \leq \xi(||\boldsymbol{x}_0||, t) + \rho(||\boldsymbol{d}||_\infty)$.

□

(ii) Formal Version and Proof for Proposition 5.2

Proposition 5.3. *Under the multi-timescale framework detailed in Theorem 5.1 and Assumption 5.7, suppose for each $i \in \mathcal{N}$, at its K_i-th update the initial state $\boldsymbol{x}_0$ is included within the safe set of agent i, then, when the number of updates for agent i goes to infinity, for any trajectory that starts from $\boldsymbol{x}_0$ under the controller obtained by solving Problem (5.5), $Pr(\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}_{infeasible}^{\boldsymbol{\pi},i}) = 0$, $\forall t \geq 0$.*

Proof. For simplicity, the barrier certificate taking the state only as input is used in the proof, and here we use induction to conduct the proof.

For the first agent, when update π^1, under Assumption 5.2, all other agents are following fixed policies $\{\pi^m\}_{m\in\{2,\ldots,N\}}$, and the update can be represented by Problem (5.11) by letting $i = 1$. Then, according to Proposition 5.1, we have the safe set $\boldsymbol{\mathcal{X}}_{\text{safe}}^{\boldsymbol{\pi}_{1,k_1},1}$ for the agent 1 is non-shrinking. Since $\boldsymbol{x}_0 \in \boldsymbol{\mathcal{X}}_{\text{safe}}^{\boldsymbol{\pi}_{1,K_1},1}$, we have $\boldsymbol{x}_0 \in \boldsymbol{\mathcal{X}}_{\text{safe}}^{\boldsymbol{\pi}_{1,k_1},1}$ for $k_1 \geq K_1$. Therefore,

when $k_1 \to \infty$, according to Lemma 5.1, we have $H^1_{\boldsymbol{\pi}_{1,k_1}}(\boldsymbol{x}_0) = 0$. Because $\forall k_1 \geq 0$ and $\forall \boldsymbol{x}'_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},1}_{\text{infeasible}}$, $Pr(\boldsymbol{x}'_{t+1} \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},1}_{\text{vio}}) > 0$ according to Assumption 5.7 and the definition of $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},1}_{\text{infeasible}}$, it can be concluded that

$$H^1_{\boldsymbol{\pi}_{1,k_1+1}}(\boldsymbol{x}'_t) = \mathbb{E}_{\tau \sim \boldsymbol{\pi}_{1,k_1+1}}\big[\sum_{j=0}^{\infty} \gamma^k b^1(\boldsymbol{x}'_{t+j})\big] < 0 \tag{5.65}$$

since $Pr(b^1(\boldsymbol{x}'_{t+1}) < 0) > 0$. Because $\forall k_1 \geq K_1$, $H^1_{\boldsymbol{\pi}_{1,k_1}}(\boldsymbol{x}_0) = 0 > H^1_{\boldsymbol{\pi}_{1,k_1}}(\boldsymbol{x}'_t)$ for any $\boldsymbol{x}'_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},1}_{\text{infeasible}}$, any trajectory starting from $\boldsymbol{x}_0$ will not enter $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},1}_{\text{infeasible}}$ with a positive probability under $\boldsymbol{\pi}_{1,k_1}$ when $k_1 \to \infty$ due to the application of the neural barrier certificate.

Then, assume for any $i-1 \in \{1, \ldots, N-1\}$, $\forall m \in \{1, \ldots, i-1\}$, $Pr(\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},m}_{\text{infeasible}}) = 0$, $\forall t \geq 0$, then, same as the proof of Theorem 5.1, under Assumption 5.2, when update the control policy π^i for agent i, $\{\pi^m\}_{m\in\{1,\ldots,i-1\}}$ for agent 1 to $i-1$ have converged to their locally optimal policies without entering their infeasible sets at a positive probability, and will not result in the shrink of the safe set of agent i since they avoid introducing safety violations between agent i and agent m, $\forall m \in \{1, \ldots, i-1\}$, almost surely when the policy for the agent i is updated from $\pi^i_{k_i}$ to $\pi^i_{k_i+1}$. Additionally, since $\{\pi^m\}_{m\in\{i+1,\ldots,N\}}$ are fixed policies under Assumption 5.2, the update for agent i can be represented by Problem (5.11), and thus based on analysis similar to Proposition 5.1, we have the safe set $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi}_{i,k_i},i}_{\text{safe}}$ for the agent i is non-shrinking. Since $\boldsymbol{x}_0 \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi}_{i,K_i},i}_{\text{safe}}$, $\boldsymbol{x}_0 \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi}_{i,k_i},i}_{\text{safe}}$ for $k_i \geq K_i$, and $H^i_{\boldsymbol{\pi}_{i,k_i}}(\boldsymbol{x}_0) = 0$. Furthermore, under Assumption 5.7, the set $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{infeasible}}$ depends on agents from $i+1$ to N and the environment, given $\{\pi^m\}_{m\in\{1,\ldots,i-1\}}$ have converged to feasible solutions leading to safety violations with agent i with probability 0. Thus, this infeasible set is fixed during the update for agent i when $\{\pi^m\}_{m\in\{i+1,\ldots,N\}}$ are fixed. Similar to the previous analysis for agent 1, given $\forall k_i \geq 0$ and $\forall \boldsymbol{x}'_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{infeasible}}$, $Pr(\boldsymbol{x}'_{t+1} \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{vio}}) > 0$, we have

$$H^i_{\boldsymbol{\pi}_{i,k_i+1}}(\boldsymbol{x}'_t) = \mathbb{E}_{\tau \sim \boldsymbol{\pi}_{i,k_i+1}}\big[\sum_{j=0}^{\infty} \gamma^j b^i(\boldsymbol{x}'_{t+j})\big] < 0. \tag{5.66}$$

Since $\forall k_i \geq K_i$, $H^i_{\boldsymbol{\pi}_{i,k_i}}(\boldsymbol{x}_0) = 0 > H^i_{\boldsymbol{\pi}_{i,k_i}}(\boldsymbol{x}'_t)$ for any $\boldsymbol{x}'_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{infeasible}}$, any trajectory from $\boldsymbol{x}_0$ will not enter $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{infeasible}}$ with a positive probability under $\boldsymbol{\pi}_{i,k_i}$ when $k_i \to \infty$ because of the neural barrier certificate, and thus $Pr(\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{infeasible}}) = 0$, $\forall t \geq 0$.

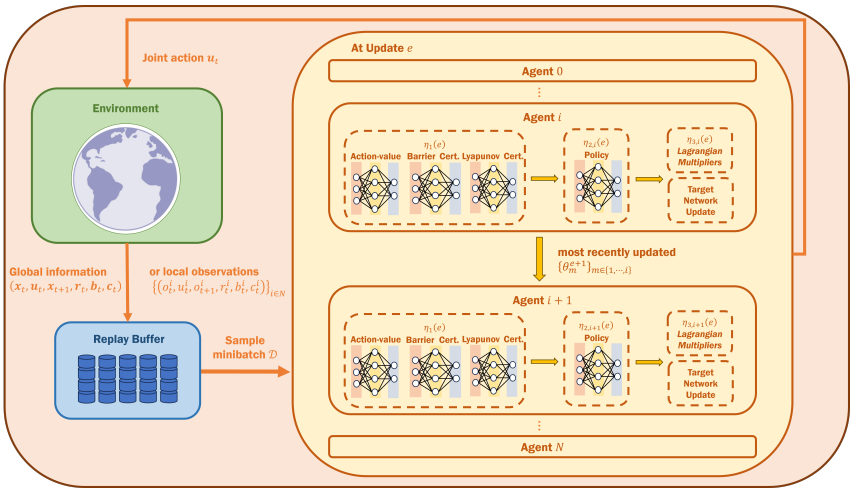


Figure 5.4: Overview of the **M**ulti-**A**gent **S**afe and **S**table **S**oft **A**ctor-**C**ritic (**MAS^{3}AC**) framework. Depending on the task setting, either global information or local observations are collected. Agents are updated sequentially, as detailed in Algorithm 5.2.

Given the aforementioned analysis, it can be obtained that $\forall i \in \mathcal{N}$, $Pr(\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{infeasible}}) = 0$, $\forall t \geq 0$ when $\{\pi^m\}_{m\in\{i+1,\ldots,N\}}$ are fixed, and $k_i \to \infty$. Since the multi-timescale framework is applied, the joint policy updated will converge to $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$, and thus when the number of updates for agent i goes to infinity, $Pr(\boldsymbol{x}_t \in \boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{infeasible}}) = 0$, $\forall t \geq 0$. □

(iii) Proof for Corollary 5.1

Proof. For each agent i, the policy converges to a feasible policy π_i^* that satisfies both the safety and stability constraints. Hence, the joint control signal $\boldsymbol{u}_t$, generated by $\boldsymbol{\pi}^*$, is a function of states, and therefore the whole multi-agent system becomes an autonomous system without external control input. In the absence of exogenous signal $\boldsymbol{d}_t$, we have $||\boldsymbol{x}_t|| \leq \xi(||\boldsymbol{x}_0||, t)$, which further implies that $\lim_{t\to\infty} ||\boldsymbol{x}_t|| = 0$. Therefore, the whole multi-agent system converges to the desired equilibrium (goal), and so does each subsystem (agent). □

5.6.5 E. Pseudocode for MAS^{3}AC

Here we provide the detailed Algorithm 5.2, which essentially requires that, within each update e, the most recently updated $\{\theta_m^{e+1}\}_{m\in\{1,\ldots,i-1\}}$, instead of $\{\theta_m^e\}_{m\in\{1,\ldots,i-1\}}$, are used when updating θ_i for agent i. Other techniques, such as double-critic architectures, can also be incorporated in the real implementation for achieving better performance.

Algorithm 5.2 **M**ulti-**A**gent **S**afe and **S**table **S**oft **A**ctor-**C**ritic (**MAS^{3}AC**)

Input: $\eta_1(e)$ for updating the action-value networks, barrier certificates, and Lyapunov certificates; $\eta_{2,i}(e)$ for updating the individual control policy π^i; and $\eta_{3,i}(e)$ for updating Lagrangian multipliers for agent i. Initialized parameters $\{\phi_i^0\}_{i\in\mathcal{N}}$, $\{\omega_i^0\}_{i\in\mathcal{N}}$, $\{\chi_i^0\}_{i\in\mathcal{N}}$, $\{\theta_i^0\}_{i\in\mathcal{N}}$, $\{\lambda_i^0\}_{i\in\mathcal{N}}$ for corresponding neural networks; maximum iterations E; policy update delay step n_p; Polyak coefficient τ, and target neural networks $\{\phi_{i,targ}^0\}_{i\in\mathcal{N}} \leftarrow \{\phi_i^0\}_{i\in\mathcal{N}}$, $\{\omega_{i,targ}^0\}_{i\in\mathcal{N}} \leftarrow \{\omega_i^0\}_{i\in\mathcal{N}}$, $\{\chi_{i,targ}^0\}_{i\in\mathcal{N}} \leftarrow \{\chi_i^0\}_{i\in\mathcal{N}}$. $\eta_1(e) > \eta_{2,1}(e), \eta_{2,i}(e) > \eta_{3,i}(e), \forall i \in \mathcal{N}$, and $\eta_{3,i-1}(e) > \eta_{2,i}(e), \forall i \in \{2,\dots,N\}$.

for $e = 0, 1, \dots, E$ **do**
  Collect transitions $(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{x}_{t+1}, \boldsymbol{r}_t, \boldsymbol{b}_t, \boldsymbol{c}_t)$ when global information is available, or $\{(o_t^i, u_t^i, o_{t+1}^i, r_t^i, b_t^i, c_t^i)\}_{i\in\mathcal{N}}$ when only local observations are available, by using the joint policy $\boldsymbol{\pi}$ parameterized by $\{\theta_i^e\}_{i\in\mathcal{N}}$.
  Store transitions into the replay buffer.
  Sample a random minibatch $\mathcal{D}$ of transitions from the replay buffer.
  for $i = 1, 2, \dots, N$ **do**
    Update agent i's action-value network, barrier certificate, and Lyapunov certificate with the learning rate $\eta_1(e)$ and loss functions detailed in Section 5.6.1:
    $\phi_i^{e+1} \leftarrow \phi_i^e - \eta_1(e)\nabla_{\phi_i} J_{Q^i}(Q^i)$;
    $\omega_i^{e+1} \leftarrow \omega_i^e - \eta_1(e)\nabla_{\omega_i} J_{H^i}(H^i)$;
    $\chi_i^{e+1} \leftarrow \chi_i^e - \eta_1(e)\nabla_{\chi_i} J_{V^i}(V^i)$.
    if $e \bmod n_p = 0$ **then**
      Update agent i's control policy with the learning rate $\eta_{2,i}(e)$ and the loss function detailed in Section 5.6.1:
      $\theta_i^{e+1} \leftarrow \theta_i^e - \eta_{2,i}(e)\nabla_{\theta_i} J_{\theta_i}^{\text{full}}(\theta_i)$.
      Update agent i's Lagrangian multipliers with the learning rate $\eta_{3,i}(e)$ and the loss function detailed in Section 5.6.1:
      $\lambda_{i,1}^{e+1} \leftarrow \left[\lambda_{i,1}^e + \eta_{3,i}(e)\nabla_{\lambda_{i,1}} J_{\lambda_i}^{\text{full}}(\lambda_i)\right]_+$;
      $\lambda_{i,2}^{e+1} \leftarrow \left[\lambda_{i,2}^e + \eta_{3,i}(e)\nabla_{\lambda_{i,2}} J_{\lambda_i}^{\text{full}}(\lambda_i)\right]_+$.
      Update agent i's target networks:
      $\phi_{i,targ}^{e+1} \leftarrow (1-\tau)\phi_{i,targ}^e + \tau\phi_i^{e+1}$;
      $\omega_{i,targ}^{e+1} \leftarrow (1-\tau)\omega_{i,targ}^e + \tau\omega_i^{e+1}$;
      $\chi_{i,targ}^{e+1} \leftarrow (1-\tau)\chi_{i,targ}^e + \tau\chi_i^{e+1}$.
    end if
  end for
  Apply linear decay to all learning rates.
end for

Output: $\{\theta_i^E\}_{i\in\mathcal{N}}$

5.6.6 F. Baselines

We compare MAS^{3}AC against both safe MARL algorithms and standard model-free MARL baselines: 1. **Cooperative tasks (global information)** As safe MARL baselines, we include MACPO, MAPPO-Lagrangian [17], which is abbreviated as MAPPOL in figures and tables, and MAFOCOPS [66]. HASAC [141], HAPPO [142], and MAPPO [143] are used as standard MARL baselines. 2. **Non-cooperative tasks (local observations)** To ensure fair comparisons, we construct decentralized versions of MACPO and MAPPO-Lagrangian, restricting them to local information without coordination, as safe MARL baselines. For standard MARL baselines, we employ MASAC (the decentralized version of HASAC) and IPPO [144]. These baselines are widely adopted in prior studies and cover a broad spectrum of approaches, ranging from trust region-based methods to traditional dual ascent update, and from centralized to decentralized training.

5.6.7 G. Benchmark Environments

To evaluate our proposed framework, we propose a suite of multi-agent benchmarks that capture diverse aspects of constrained decision-making. These environments are built upon MuJoCo-based robotic control tasks and an electrical bidding task with explicit state-wise safety and stability requirements, in addition to different reward structures. Each benchmark specifies: (i) the reward design, (ii) the safety constraints that agents need to satisfy, and (iii) desired equilibrium states associated with stability requirements. Each tasks involves multiple agents, with either global information or local observations to support centralized or decentralized training. Together, these benchmarks provide a comprehensive testbed for studying safe and stable MARL, spanning cooperative and non-cooperative settings, physics-based robotic control, and optimization-driven domains.

(i) Cooperative Ant

We adopt the standard partitioning from the Safe Multi-Agent MuJoCo benchmark, where the Ant is divided into four agents, each controlling two joints (“4x2” configuration). The joint action is the concatenation of all agent actions, and all agents receive the full global state augmented with an agent identifier, and shared signals. The task is fully cooperative.

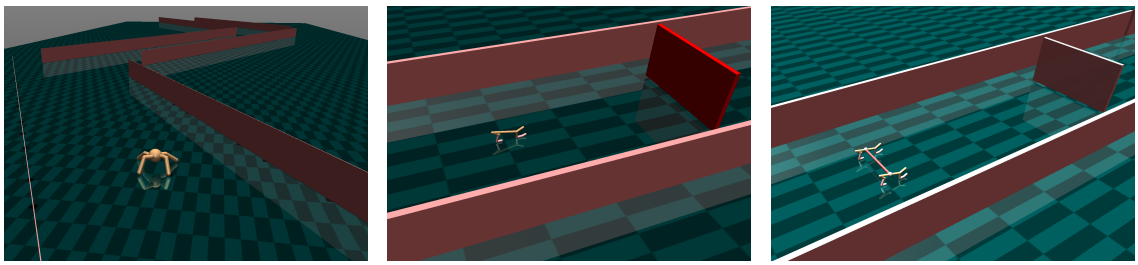
The reward encourages locomotion at a desired forward velocity $v^* = 0.7$ along the x-axis, while penalizing control effort. Formally, the reward at time t is given by $r_t = -|v_t - v^*| - 0.5\|a_t\|^2 + \mathbf{1}_{[0.5 \leq v_t \leq 0.9]} \times 2.0$, where v_t is the torso' s forward velocity along the x-axis and a_t is the joint action vector, and $\mathbf{1}_{[\cdot]}$ denotes the indicator function.

Safety constraints are imposed by the presence of slanted walls forming a narrowing corridor. A safety violation occurs if the torso' s lateral position y_t is within 1.8 units of either wall, in which case the instantaneous safety cost (barrier signal) $b_t = -1.0$ is triggered; otherwise, $b_t = 0$. To capture stability, a Lyapunov-inspired deviation cost signal can be defined as $c_t = |v_t - v^*|$, which penalizes deviation from the desired velocity. In practice, we further add an additional part to c_t to avoid excessive actuation. Notably, the issue of infeasibility arises when the stability constraint conflicts with the safety constraint: maintaining the desired forward velocity along the x-axis when Ant is close to the slanted walls can inevitably lead the ant into unsafe regions that are too close to the wall in subsequent states. An illustration can be found in Figure 5.6.

In sum, in this cooperative task, agents must coordinate their leg motions to maximize locomotion reward while simultaneously maintaining safety within the corridor and stability towards the equilibrium state.

(ii) Cooperative HalfCheetah

We adopt the standard Safe Multi-Agent MuJoCo partitioning, splitting the HalfCheetah into three agents, each controlling two joints ("3x2" configuration). The joint action is the concatenation of per-agent actions. All agents receive the full global state augmented with an agent identifier, and shared signals, yielding a fully cooperative setting.



(a) Ant **(b)** HalfCheetah **(c)** Coupled HalfCheetah

Figure 5.5: Three cooperative tasks with global information.

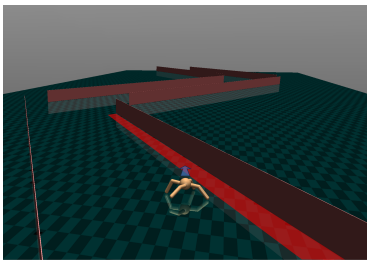


Figure 5.6: Illustration of the issue of infeasibility in the Ant task. Each agent is required to maintain a desired velocity along the x-axis (blue arrow). From certain states near the wall, satisfying this stability requirement inevitably leads to safety violations: the agent enters the danger region (red) that lies too close to the wall.

Let x_t denote the cheetah torso position along the x-axis and w_t the position of the movable wall along the x-axis. Define the distance $d_t := w_t - x_t$. The objective is to maintain a desired distance $d^* = 12.0$ while minimizing control effort. The per-timestep reward is $r_t = -0.1 \times |d_t - d^*| - 0.1 \|a_t\|^2 + 3.0 \times \mathbf{1}_{[7.0 \le d_t \le 14.0]}$, where a_t is the joint action vector and $\mathbf{1}_{[\cdot]}$ denotes the indicator function.

A safety violation occurs when the cheetah is too close to the moving wall: $d_t < d_{\text{min}}$ with $d_{\text{min}} = 9.0$. We expose a barrier signal $b_t = -\max\{0, d_{\text{min}} - d_t\}$, which is zero when the constraint is satisfied and strictly negative otherwise.

Stability cost can also be defined to penalize departures from the equilibrium distance: $c_t = 0.1 \times |d_t - d^*|$, and can practically be added an extra term to avoid excessive actuation. Similarly, this environment exhibits conflicting scenarios: an additional reward encouraging stability is granted for $7 \le d_t \le 14$, which partially overlaps with the unsafe region $d_t < 9$. In sum, agents must coordinate to track the movable boundary at the desired distance while ensuring state-wise constraints.

(iii) Cooperative Coupled HalfCheetah

We adopt the standard Safe Multi-Agent MuJoCo partitioning for the Coupled HalfCheetah, yielding two agents each controlling one HalfCheetah. Two HalfCheetahs are coupled by a tendon with a finite length range, enforcing coordination between the bodies. The joint action is the concatenation of all per-agent actions. All agents can access the full global state augmented with an agent identifier, and shared signals, resulting in a fully cooperative task.

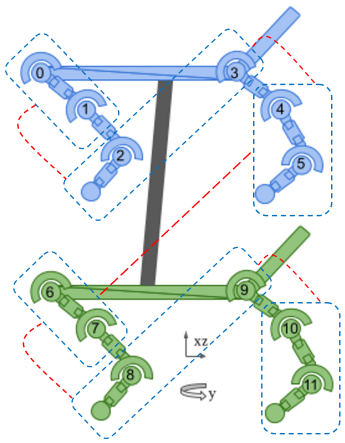


Figure 5.7: Communication in the decentralized non-cooperative Coupled HalfCheetah task. Each dashed rectangle represents one agent, which controls two joints. Information can be communicated through the red dashed line to neighbor agents. The number of red lines that the information can go through is dependent on the defined neighbor size.

Let $x_t^{(1)}$ and $x_t^{(2)}$ denote the positions of the two cheetah torsos along the x-axis, and w_t the position of the movable wall along the x-axis. Define the midpoint $m_t := \frac{1}{2}\left(x_t^{(1)} + x_t^{(2)}\right)$ and the distance $d_t := w_t - m_t$. The objective is to maintain a desired distance $d^* = 12.0$ while minimizing control effort. The per-timestep reward is $r_t = -0.1 \times |d_t - d^*| - 0.1 \|a_t\|^2 + 3.0 \times \mathbf{1}_{[10 \le d_t \le 14]}$, where a_t is the joint action vector and $\mathbf{1}_{[\cdot]}$ is the indicator function.

State-wise safety requires each cheetah to keep a minimum distance to the wall. With $d_t^{(i)} := w_t - x_t^{(i)}$ and $d_{\text{min}} = 11.0$, a violation occurs when $d_t^{(i)} < d_{\text{min}}$. We expose a barrier signal $b_t = -\sum_{i=1}^{2} \max\{0,\, d_{\text{min}} - d_t^{(i)}\}$, which is zero when both cheetahs are safe and strictly negative otherwise.

The cost penalizing departures from the equilibrium distance is: $c_t = 0.1 \times |d_t - d^*|$. An extra term can also be added to avoid excessive activation. This environment also exhibits conflicting cases: the stability bonus is granted for $10 \le d_t \le 14$, which can partially overlap with unsafe regions where either $d_t^{(1)}$ or $d_t^{(2)}$ falls below 11. In sum, the Coupled HalfCheetah benchmark challenges agents to coordinate locomotion across two physically coupled bodies, maintain a desired distance from a movable wall, and balance reward maximization with both state-wise safety and stability requirements.

(iv) Non-cooperative Coupled HalfCheetah

This environment comprises two tendon-coupled HalfCheetah and six agents; each agent controls two actuators: *(1) bthigh, bshin*, *(2) bfoot, fthigh*, *(3) fshin, ffoot*, *(4) bthigh2, bshin2*, *(5) bfoot2, fthigh2*, *(6) fshin2, ffoot2*. Agents observe only *local* information with optional k-neighbor augmentation (here, k=3 with zero-padding at the boundaries). For each agent i, the observation includes its own and k-neighbor's joint positions/velocities and a compact set of environment features (wall velocity, etc).

Let $x_t^{(1)}$ and $x_t^{(2)}$ be the x-positions of the two torsos and w_t the wall position. Define per-cheetah distances $d_t^{(1)}{:=}w_t{-}x_t^{(1)}$, $d_t^{(2)}{:=}w_t{-}x_t^{(2)}$, the desired standoff $d^*{=}12.0$, and the stability interval $[d_{\text{min}}^{\text{stab}}, d_{\text{max}}^{\text{stab}}]{=}[10, 14]$. For agents $i{\in}\{1, 2, 3\}$ (first cheetah),

$$r_t^{(i)} \;=\; -0.1 \times |d_t^{(1)} - d^*| \;-\; 0.1\,\|a_t^{(i)}\|^2 \;+\; 3.0 \times \mathbf{1}_{[\,10 \le d_t^{(1)} \le 14\,]} \;+\; \rho_t^{(1)},$$

and for agents $i{\in}\{4, 5, 6\}$ (second cheetah),

$$r_t^{(i)} \;=\; -0.1 \times |d_t^{(2)} - d^*| \;-\; 0.1\,\|a_t^{(i)}\|^2 \;+\; 3.0 \times \mathbf{1}_{[\,10 \le d_t^{(2)} \le 14\,]} \;+\; \rho_t^{(2)},$$

where $a_t^{(i)}$ is agent i's action vector, and $\rho_t^{(1)}, \rho_t^{(2)}$ are small competition terms that reward being ahead of the other HalfCheetah (zero if leading, -0.05 otherwise).

Each HalfCheetah must keep a minimum clearance $d_{\text{min}}{=}11.0$ from the wall. We expose a per-halfcheetah barrier signal and feed it to each agent on that body:

$$b_t^{(1)} \;=\; -\max\{0,\, d_{\text{min}} - d_t^{(1)}\}, \qquad b_t^{(2)} \;=\; -\max\{0,\, d_{\text{min}} - d_t^{(2)}\}.$$

A safe state yields $b_t^{(\cdot)}{=}0$; otherwise it is strictly negative.

Stability is encouraged by the per-cheetah cost that each local agent inherits, for distance tracking:

$$c_t^{(1)} \;=\; 0.1 \times |d_t^{(1)} - d^*|, \qquad c_t^{(2)} \;=\; 0.1 \times |d_t^{(2)} - d^*|.$$

Extra terms can also be added to these costs to avoid excessive activation. Conflicts may arise

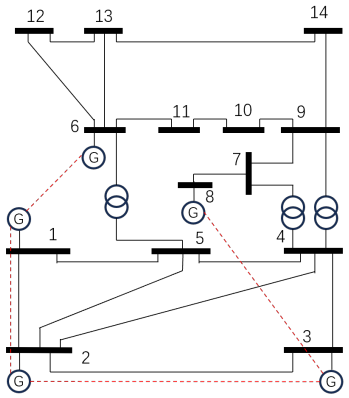


Figure 5.8: The network topology for the non-cooperative electrical bidding task. Each generator, marked with a 'G', represents an independent agent competing in the electricity market. The numbered nodes are buses interconnected by transmission lines. The red dashed lines indicate the communication lines between specific generator agents.

since the stability reward interval $[10, 14]$ partially overlaps the unsafe region $d_t^{(\cdot)}<11$. In sum, this non-cooperative benchmark evaluates decentralized control with local sensing and per-agent objectives across two physically coupled half cheetahs; each agent must balance its own performance against state-wise constraints under partial observability. The communication among agents in this multi-agent system is described in Figure 5.7.

(v) Non-cooperative Electrical Bidding Task

This environment models a competitive electricity market based on the IEEE 14-bus system, featuring five generator nodes that function as agents, indexed by $i \in \{1, \ldots, 5\}$. Each agent participates in the market by submitting a bid price, $k_{i,t}$, at each time step t.

Agents operate under partial observability and observe only *local* and neighboring information. For each agent i, the observation includes the load demand $P_{\text{load},j,t}$ from its neighboring nodes j, the previous power outputs $P_{\text{out},j,t-1}$ of neighboring generators, and its own previous market outcomes—the clearing price $\lambda_{i,t-1}$ and power output $P_{\text{out},i,t-1}$.

At each time step, a central system operator collects the bid prices $\mathbf{K_t}$ and solves a DC Optimal Power Flow (DC-OPF) problem to clear the market. This process determines the power output $\mathbf{P}_{\text{out,t}}$ and the market clearing price (locational marginal price) $\boldsymbol{\lambda_t}$ for each generator.

The optimization is defined as:

$$
\begin{aligned}
\min_{\mathbf{P_{out,t}}} \quad & \mathbf{K_t^\top P_{out,t}}, \\
\text{s.t.} \quad & \mathbf{P_n = B\theta} : \boldsymbol{\lambda_t}, \\
& -\mathbf{P_{branchlimit}} \leq \mathbf{P_{b,t}} \leq \mathbf{P_{branchlimit}}, \\
& \mathbf{P_{lb}} \leq \mathbf{P_{out,t}} \leq \mathbf{P_{ub}}, \\
& \mathbf{P_{out,t-1}} - \boldsymbol{\square}\mathbf{P_{out}} \leq \mathbf{P_{out,t}} \leq \mathbf{P_{out,t-1}} + \Delta P_{out}
\end{aligned}
$$

Here, the constraints represent the DC power flow balance, transmission line capacity limits, generator output limits, and ramp-rate limits, respectively.

The reward for each agent i is a composite signal designed to balance economic revenue with operational stability. It consists of the scaled revenue, a bonus for operating within a desired output range, and a penalty for deviating from a specific target output:

$$
\begin{aligned}
r_t^{(i)} = \alpha_r(\lambda_{i,t}P_{\text{out},i,t} - c_{g,i,t}) + \beta_s \cdot \mathbf{1}_{[\,P_{\text{des,low},i} \leq P_{\text{out},i,t} \leq P_{\text{des,up},i}\,]} - \gamma_d \cdot |P_{\text{out},i,t} - P_{\text{des},i}|, \\
\text{where} \quad c_{g,i,t} = a_i P_{\text{out},i,t}^2 + b_i P_{\text{out},i,t} + c_i.
\end{aligned}
$$

Here, $c_{g,i,t}$ is the quadratic generation cost, $\mathbf{1}_{[\cdot]}$ is the indicator function, $[P_{\text{des,low},i}, P_{\text{des,up},i}]$ defines the stable operating range, and $P_{\text{des},i}$ is the single ideal output point. The terms α_r, β_s, and γ_d are positive weighting coefficients for revenue, stability bonus, and deviation penalty, respectively.

To enforce environmental constraints, a safety barrier signal $b_t^{(i)}$ is provided to each agent i. This signal penalizes any generation that causes carbon emissions to exceed a predefined allowance, $Carbon_{\text{allow},i}$:

$$
b_t^{(i)} = -\max\{0, \phi_i P_{\text{out},i,t} - Carbon_{\text{allow},i}\},
$$

where ϕ_i is the carbon emission intensity of generator i. A safe state yields $b_t^{(i)} = 0$; otherwise, it is strictly negative.

Operational stability is encouraged via a cost signal $c_t^{(i)}$, which penalizes the absolute devi-

ation from a single desired power output point $P_{\text{des},i}$:

$$c_t^{(i)} = \gamma_d \cdot |P_{\text{out},i,t} - P_{\text{des},i}|.$$

In sum, this environment serves as a non-cooperative benchmark for evaluating decentralized bidding strategies under partial observability. Conflicts may arise since the stability reward interval partially overlaps the unsafe region. Each agent must learn to balance maximizing its own revenue against system-wide constraints, including carbon emission limits (safety) and operational stability.

(vi) Non-cooperative Ant

In addition to the decentralized version of Coupled HalfCheetah, here we propose another decentralized setting for robot control. This non-cooperative Ant environment comprises eight agents, with each agent controlling one actuator: *(1) hip1*, *(2) ankle1*, *(3) hip2*, *(4) ankle2*, *(5) hip3*, *(6) ankle3*, *(7) hip4*, *(8) ankle4*. Agents observe only *local* information with optional k-neighbor augmentation (here, k=3 with zero-padding at the boundaries). For each agent i, the observation includes its own and k-neighbor's joint positions/velocities and a compact set of environment features (lateral offset, etc).

Let v_t be the torso x-velocity, and v^*=0.7 be the desired forward velocity along the x-axis. Define the deviation δ_t:=$|v_t - v^*|$ and the stability interval $[v_{\text{min}}^{\text{stab}}, v_{\text{max}}^{\text{stab}}]$=$[0.5, 0.9]$. Each agent i controls a single actuator $a_t^{(i)}$ and belongs to one of four legs $\ell(i) \in \{1, 2, 3, 4\}$. The reward for agent i is given by

$$r_t^{(i)} \;=\; -0.5||a_t^{(i)}||^2 \;-\; |v_t - v^*| \;+\; 2.0 \times \mathbf{1}_{[0.5 \le v_t \le 0.9]} \;+\; \rho_t^{(\ell(i))},$$

where $\rho_t^{(\ell)}$ is a competition term based on the leg's forward rank (by x-position of the leg's center of mass): $\rho_t^{(\ell)} \in \{-0.15, -0.09, 0.09, 0.15\}$ from trailing to leading leg, and shared by the two actuators on that leg.

The Ant moves through a piecewise-slanted corridor bounded by two walls at lateral offsets ± 5 from a segmented centerline. For each leg ℓ with position $(x_t^{(\ell)}, y_t^{(\ell)})$, a per-agent barrier

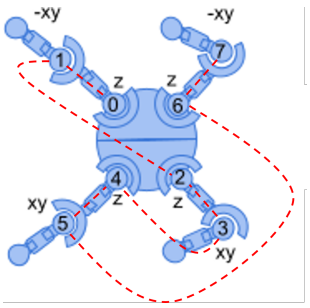


Figure 5.9: Communication in the decentralized Ant task. Information can be communicated through the red dashed line to neighbor agents. The number of red lines that the information can go through is dependent on the defined neighbor size.

signal is computed as

$$b_t^{(i)} \;=\; -\,\mathbf{1}\Big\{\min\big(|y_t^{(\ell(i))}-\text{wall}^{+}(x_t^{(\ell(i))})|,\,|y_t^{(\ell(i))}-\text{wall}^{-}(x_t^{(\ell(i))})|\big) < 1.0\Big\},$$

which equals 0 in safe states and -1 when the leg is within $1.0\,\text{m}$ of either wall.

To capture stability, a Lyapunov-inspired deviation cost signal for each agent i can be defined as

$$c_t^{(i)} \;=\; |v_t - v^*|,$$

which penalizes deviation from the desired velocity. In practice, we further add an additional part to $c_t^{(i)}$ to avoid excessive actuation. In summary, this non-cooperative benchmark evaluates decentralized control with local sensing and per-agent objectives across the four-legged Ant. Each agent must coordinate its own motion to maintain the target velocity, stay within the slanted corridor, and compete for forward advancement under partial observability. The communication among agents in this multi-agent system is described in Figure 5.9.

(vii) Hyperparameters Setting

We provide the setting for some hyperparameters for the cooperative ant task in Table 5.3. Notably, although linear decay is always applied to all learning rates, different initial values can be assigned. Besides the values provided in Table 5.3, we additionally try this set of initial values where differences are larger such that learning rates during training are more different from each other: Critics learning rate initial value 0.004, Policies learning rates initial values [0.002, 0.0005, 0.000125, 0.00003125], Lagrangian multipliers learning rates initial values [0.001, 0.00025,



Figure 5.10: Average episode reward and average episode safety cost in the non-cooperative electrical bidding task, comparing MAS^{3}AC against baselines. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

0.0000625, 0.000015625]. The corresponding results are still good, as shown in Figure 5.22.

Table 5.3: Parameters setting for MAS^{3}AC ("lr" denotes "learning rate").

Hyperparameter	Value
Discount factor	0.99
Optimizer	Adam
Target smoothing coefficient	0.005
Initial entropy coefficient	0.2
Entropy coefficient lr	3×10^{-4}
Policy update interval	2
Target update interval	2
Activation function	ReLU
Final activation function	Tanh
Initialization method	Orthogonal
Neural Network (MLP)	[256, 256]
Critics lr	Initial value 0.002, linear decay applied
Policies lr	Initially [0.001, 0.0008, 0.0006, 0.0004] , linear decay applied
Lagrangian multipliers lr	Initially [0.0009, 0.0007, 0.0005, 0.0003] , linear decay applied
Batch size	1000
Buffer size	1000000

5.6.8 H. Additional Experimental Results

(i) The Evaluation on the Electrical Bidding Task

We further evaluate MAS^{3}AC on the decentralized electrical bidding environment, where the system dynamics is governed by a constrained optimization problem rather than physical robot dynamics, providing a qualitatively different benchmark. In this setting, each generator agent

Table 5.4: Reward and safety cost comparison for the non-cooperative electrical bidding task across agents.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	-805.15 ± 406.63	$\mathbf{0.67 \pm 2.41}$	-141.19 ± 420.37	1.87 ± 3.29	58.84 ± 405.21	2.01 ± 3.26
MACPO	-315.37 ± 211.80	1.00 ± 1.27	-87.69 ± 179.42	1.02 ± 1.02	44.77 ± 110.30	1.81 ± 1.18
MAPPOL	-238.07 ± 85.68	1.51 ± 0.51	-83.65 ± 54.88	1.17 ± 0.40	78.57 ± 35.76	1.18 ± 0.43
MASAC	127.49 ± 35.71	0.80 ± 0.15	169.07 ± 41.26	$\mathbf{0.24 \pm 0.07}$	$\mathbf{348.67 \pm 48.67}$	$\mathbf{0.10 \pm 0.07}$
MAS^{3}AC	$\mathbf{131.42 \pm 38.09}$	0.75 ± 0.16	$\mathbf{194.20 \pm 31.61}$	$\mathbf{0.24 \pm 0.07}$	276.81 ± 70.11	0.15 ± 0.16

	Agent 3		Agent 4	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	38.08 ± 358.01	2.78 ± 2.73	27.36 ± 273.47	1.33 ± 1.52
MACPO	93.70 ± 73.92	1.67 ± 0.84	111.83 ± 63.72	1.11 ± 0.55
MAPPOL	106.33 ± 35.23	1.20 ± 0.39	116.92 ± 44.45	0.94 ± 0.25
MASAC	$\mathbf{270.74 \pm 59.21}$	0.32 ± 0.17	115.16 ± 77.97	$\mathbf{0.42 \pm 0.15}$
MAS^{3}AC	258.59 ± 67.66	$\mathbf{0.31 \pm 0.12}$	$\mathbf{231.22 \pm 75.31}$	0.46 ± 0.17

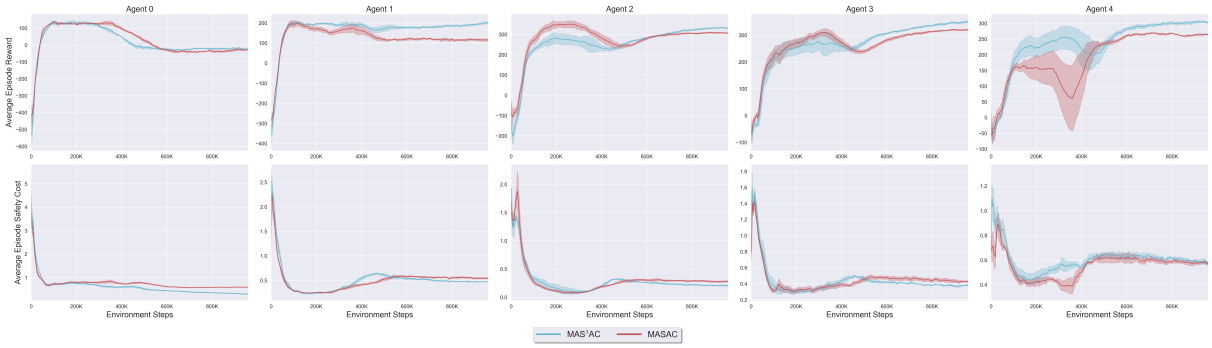


Figure 5.11: Average episode reward and average episode safety cost in the non-cooperative electrical bidding task, comparing MAS^{3}AC and MASAC over an extended training horizon.

optimizes its own objective under local safety and stability constraints. Results show that MAS^{3}AC still mostly outperforms baselines across rewards and safety costs, except for MASAC (see Figure 5.10 and Table 5.4). To further investigate this, we conducted extended training runs with a longer horizon. As shown in Figure 5.11 and Table 5.5, the difference between the two algorithms becomes obvious with more training steps, and MAS^{3}AC succeeds in obtaining better performance in most cases. This extended evaluation confirms that MAS^{3}AC maintains a long-term advantage in both reward maximization and safety constraint satisfaction, highlighting its robustness in complex, optimization-driven multi-agent environments.

(ii) Additional Results on Neighbor Size

We further provide extended results on the effect of neighbor size in the decentralized Coupled HalfCheetah benchmark, shown in Figure 5.12 and Table 5.7. Across all cases, MAS^{3}AC outperforms the baseline MAPPOL, confirming its robustness under varying information avail-

Table 5.5: Reward and safety cost comparison for the non-cooperative electrical bidding task with extended training steps.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
MASAC	-29.93 ± 24.78	0.59 ± 0.07	116.23 ± 20.19	0.54 ± 0.07	307.58 ± 11.91	0.28 ± 0.05
MAS^{3}AC	**-21.56 ± 23.98**	**0.31 ± 0.05**	**197.39 ± 18.30**	**0.47 ± 0.05**	**331.98 ± 13.67**	**0.21 ± 0.04**

	Agent 3		Agent 4	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**
MASAC	319.78 ± 11.31	0.43 ± 0.06	262.92 ± 15.75	**0.57 ± 0.07**
MAS^{3}AC	**347.71 ± 13.34**	**0.37 ± 0.05**	**303.22 ± 16.13**	0.58 ± 0.08

Table 5.6: Reward and safety cost comparison for the non-cooperative Coupled HalfCheetah task with different neighbor sizes.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	1379.02 ± 226.55	165.11 ± 107.22	1387.27 ± 225.99	165.11 ± 107.22	1387.49 ± 223.97	165.11 ± 107.22
MACPO	193.36 ± 152.44	25.36 ± 29.17	191.02 ± 152.34	25.36 ± 29.17	189.91 ± 152.60	25.36 ± 29.17
MAPPOL	1288.79 ± 230.43	80.14 ± 49.00	1289.21 ± 235.01	80.14 ± 49.00	1287.07 ± 230.98	80.14 ± 49.00
MASAC	2486.44 ± 65.09	35.17 ± 21.86	2485.82 ± 77.74	35.17 ± 21.86	2468.01 ± 62.02	35.17 ± 21.86
MAS^{3}AC_0	1743.53 ± 716.81	80.66 ± 107.23	1764.67 ± 733.83	80.66 ± 107.23	1769.68 ± 722.35	80.66 ± 107.23
MAS^{3}AC_2	**2502.49 ± 86.37**	**8.54 ± 17.46**	**2508.21 ± 95.13**	**8.54 ± 17.46**	**2503.10 ± 91.08**	**8.54 ± 17.46**

	Agent 3		Agent 4		Agent 5	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	1425.27 ± 286.82	165.11 ± 107.22	1429.73 ± 285.75	183.70 ± 98.07	1434.62 ± 287.63	183.70 ± 98.07
MACPO	80.03 ± 250.42	20.82 ± 15.67	80.26 ± 250.40	20.82 ± 15.67	80.64 ± 250.74	20.82 ± 15.67
MAPPOL	1205.69 ± 186.90	76.28 ± 42.80	1213.27 ± 187.39	76.28 ± 42.80	1211.63 ± 185.34	76.28 ± 42.80
MASAC	2423.02 ± 63.20	70.67 ± 49.07	2451.57 ± 64.97	70.67 ± 49.07	2447.62 ± 72.79	70.67 ± 49.07
MAS^{3}AC_0	1833.44 ± 403.83	43.64 ± 38.40	1830.81 ± 408.90	43.64 ± 38.40	1837.18 ± 402.57	43.64 ± 38.40
MAS^{3}AC_2	**2523.87 ± 71.62**	**2.21 ± 4.16**	**2521.11 ± 69.39**	**2.21 ± 4.16**	**2521.40 ± 71.64**	**2.21 ± 4.16**

Table 5.7: Reward and safety cost comparison for the non-cooperative Coupled HalfCheetah task with more different neighbor sizes (1 and 3).

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
MAPPOL	1229.41 ± 246.37	96.94 ± 68.86	1230.19 ± 247.36	96.94 ± 68.86	1226.87 ± 245.02	96.94 ± 68.86
MAS^{3}AC_0	1587.19 ± 777.46	85.27 ± 164.09	1610.15 ± 788.09	85.27 ± 164.09	1610.81 ± 778.27	85.27 ± 164.09
MAS^{3}AC_1	2491.39 ± 136.21	7.09 ± 14.27	2492.96 ± 130.55	7.09 ± 14.27	2489.45 ± 132.87	7.09 ± 14.27
MAS^{3}AC_3	**2576.03 ± 124.63**	**6.86 ± 13.78**	**2581.35 ± 125.65**	**6.86 ± 13.78**	**2579.96 ± 125.70**	**6.86 ± 13.78**

	Agent 3		Agent 4		Agent 5	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
MAPPOL	1147.33 ± 271.41	69.14 ± 39.50	1151.22 ± 273.06	69.14 ± 39.50	1152.16 ± 273.03	69.14 ± 39.50
MAS^{3}AC_0	1662.30 ± 504.71	21.93 ± 25.29	1658.89 ± 515.04	21.93 ± 25.29	1666.24 ± 503.59	21.93 ± 25.29
MAS^{3}AC_1	2449.61 ± 362.57	**4.02 ± 6.72**	2444.83 ± 362.42	**4.02 ± 6.72**	2438.13 ± 364.22	**4.02 ± 6.72**
MAS^{3}AC_3	**2522.42 ± 142.71**	5.70 ± 8.02	**2525.37 ± 138.01**	5.70 ± 8.02	**2528.33 ± 143.15**	5.70 ± 8.02

ability. Within MAS^{3}AC itself, we observe clear differences: with neighbor size 0, the algorithm performs poorly in terms of safety, as relying only on self-information fails to capture the inherently inter-agent nature of the safety constraints. Comparing neighbor size 1 and 3, larger neighborhoods do not always translate into uniformly better results. While size 3 tends to yield

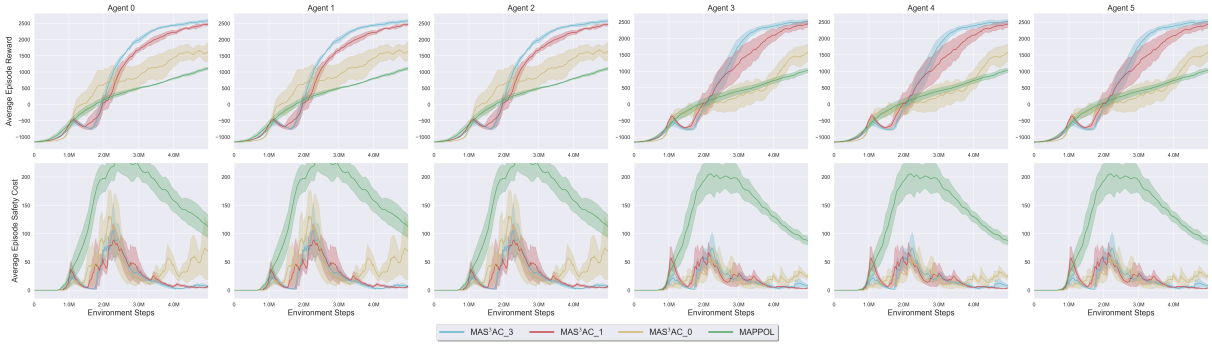


Figure 5.12: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{3}AC with more different neighbor sizes (1 and 3).

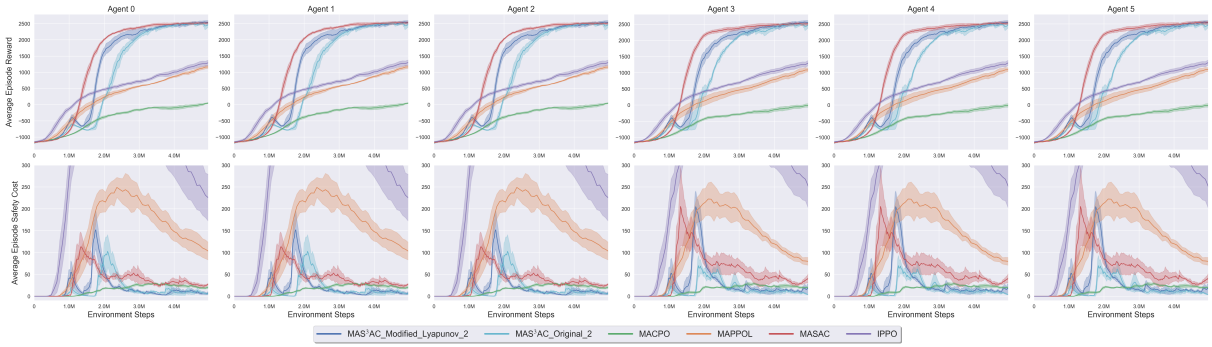


Figure 5.13: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{3}AC with different Lyapunov certificate designs. The modified one represents the case where the input to the Lyapunov certificate only includes the agent's own state and action, while the original one represents the case where the input also includes information from all neighbor agents.

higher rewards, size 1 achieves lower final safety violations for Agent 3, 4, and 5, suggesting that adding more information does not monotonically improve performance for all agents. A possible explanation is that larger neighborhoods introduce more complex inputs that require more sophisticated learning, or that each task may have an upper bound on useful neighbor size, beyond which additional information does not provide proportional benefits.

(iii) Additional Results on Lyapunov Function Input

We next examine the effect of Lyapunov function inputs in the decentralized Coupled HalfCheetah benchmark. Specifically, we compare two versions of MAS^{3}AC: (*i*) the *modified* version, where each agent's Lyapunov certificate only uses its own state and action as input; and (*ii*) the *original* version, where Lyapunov certificates also take the states and actions of neighboring agents as input. As shown in Figure 5.13 and Table 5.8, both versions of MAS^{3}AC maintain a strong balance between reward and safety, clearly outperforming all baselines by achieving

Table 5.8: Reward and safety cost comparison for the non-cooperative Coupled HalfCheetah task with different Lyapunov certificate designs.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	902.29 ± 386.41	451.38 ± 305.37	904.99 ± 389.86	451.38 ± 305.37	905.91 ± 388.62	451.38 ± 305.37
MACPO	-140.28 ± 276.72	21.48 ± 25.26	-141.91 ± 276.68	21.48 ± 25.26	-142.75 ± 276.65	21.48 ± 25.26
MAPPOL	735.92 ± 416.97	174.96 ± 143.68	735.77 ± 417.75	174.96 ± 143.68	732.35 ± 417.00	174.96 ± 143.68
MASAC	2501.57 ± 86.21	28.06 ± 28.15	2508.60 ± 90.78	28.06 ± 28.15	2496.33 ± 89.72	28.06 ± 28.15
MAS^{3}AC_2	2481.78 ± 217.30	$\mathbf{8.20 \pm 19.13}$	2485.79 ± 217.27	$\mathbf{8.20 \pm 19.13}$	2481.58 ± 224.22	$\mathbf{8.20 \pm 19.13}$
MAS^{3}AC	$\mathbf{2511.67 \pm 164.10}$	9.06 ± 16.43	$\mathbf{2510.17 \pm 161.60}$	9.06 ± 16.43	$\mathbf{2505.89 \pm 166.87}$	9.06 ± 16.43
	Agent 3		**Agent 4**		**Agent 5**	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	898.48 ± 395.57	432.92 ± 273.22	900.26 ± 397.68	432.92 ± 273.22	903.87 ± 397.26	432.92 ± 273.22
MACPO	-215.51 ± 277.43	21.14 ± 27.75	-216.04 ± 277.56	21.14 ± 27.75	-215.42 ± 277.29	21.14 ± 27.75
MAPPOL	646.70 ± 464.02	150.76 ± 134.02	648.57 ± 465.17	150.76 ± 134.02	648.17 ± 466.71	150.76 ± 134.02
MASAC	2485.39 ± 102.35	37.73 ± 35.27	$\mathbf{2502.09 \pm 97.16}$	37.73 ± 35.27	$\mathbf{2505.65 \pm 108.50}$	37.73 ± 35.27
MAS^{3}AC_2	2471.78 ± 226.85	$\mathbf{8.67 \pm 19.91}$	2466.98 ± 226.57	$\mathbf{8.67 \pm 19.91}$	2464.22 ± 227.26	$\mathbf{8.67 \pm 19.91}$
MAS^{3}AC	$\mathbf{2515.31 \pm 175.23}$	14.58 ± 24.06	2502.86 ± 179.74	14.58 ± 24.06	2509.51 ± 169.11	14.58 ± 24.06

significantly fewer safety violations while still obtaining high rewards.

Comparing the two versions, we find that the modified Lyapunov version is more sample efficient: its reward curve rises faster than that of the original version, while both lead to similar final performance. This observation aligns with our theoretical ISS analysis, which requires only an agent's own state (plus at most a few scalar signals from neighbors) rather than full neighbor states and actions. Including more information in the Lyapunov input may increase learning complexity without providing proportional benefits. Importantly, these results demonstrate that MAS^{3}AC can still perform well even with restricted Lyapunov inputs, offering a practical and theoretically justified approach to decentralized training with minimal inter-agent information exchange. We further note that in this setting, MAS^{3}AC uses neighbor size 2 for all other components (critics, barrier certificates, policies), which provides strictly less information than the baselines (neighbor size 3), yet it still outperforms them.

5.6.9 I. Further Extra Experiments and Ablation Studies

In this part, we conduct a comprehensive set of additional experiments and ablation studies to evaluate the robustness, scalability, and specific component contributions of the proposed MAS^{3}AC algorithm.

(i) Robustness against External Disturbances

We consider the scenario where an external disturbance $\boldsymbol{d}_t$ is applied to the system. To account for this, we incorporate the disturbance into the Lyapunov stability constraints by adding the term $\kappa_i(||\boldsymbol{d}_t||)$ to the stability requirement and re-running the experiments. We implement the experiment to investigate how the external disturbance affects the system's performance reflected by the reward. We use the cooperative Ant task, and apply external disturbances (random torques) on its 8 rotors. Give the control input's maximum magnitude is 1 for this task, four settings are tested, where the external disturbance's maximum magnitude $||\boldsymbol{d}||_\infty = \sup\{||\boldsymbol{d}_t||, t \geq 0\}$ are set to be 0 (no external disturbances), 0.1 (small external disturbances), 0.25 (medium external disturbances) and 0.4 (strong external disturbance). Terms $\kappa_i(||\boldsymbol{d}_t||)$ are also incorporated into the training process. Baselines under no external disturbances are also presented for comparison.

It can be found that as the disturbance becomes stronger, the algorithm's performance inevitably deteriorates in both reward maximization and safety satisfaction. However, even under the strong disturbances, the proposed algorithm still achieves better rewards and fewer safety violations compared to commonly-used baselines under no disturbance, which demonstrates the good performance of MAS^{3}AC.

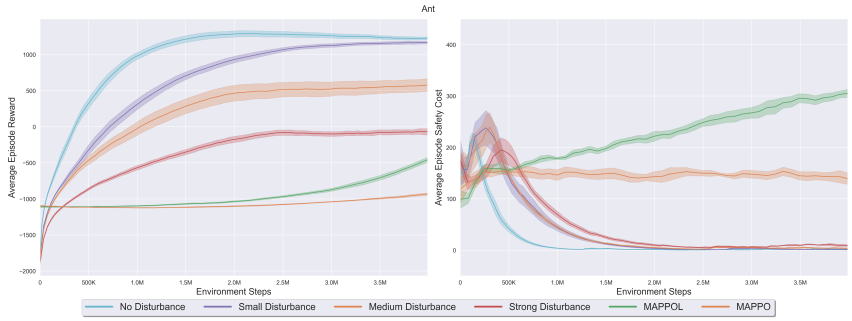


Figure 5.14: Performance influenced by external disturbances with varying magnitudes (No, Small, Medium, Strong) on the Ant task.

(ii) Scaling to Larger N with Partial Observability

To test scalability, we scaled the system to $N \geq 10$ agents using a modified Coupled HalfCheetah task, where the system is composed of 12 agents (12×1 setting) operating under partial observability. We ran the non-cooperative version of the experiment, ensuring all baselines were

Table 5.9: Reward and cost comparisons (external disturbances in Ant).

	Ant	
Condition	**Reward**	**Cost**
MAPPO	-834.56 ± 317.29	249.46 ± 87.03
MAPPOL	-1039.23 ± 95.85	145.60 ± 58.37
Strong Disturbance (MAS^{3}AC)	-73.35 ± 100.57	7.86 ± 12.35
Medium Disturbance (MAS^{3}AC)	550.07 ± 194.18	4.02 ± 8.27
Small Disturbance (MAS^{3}AC)	1152.68 ± 58.21	1.69 ± 5.90
No Disturbance (MAS^{3}AC)	1232.87 ± 57.45	2.02 ± 5.45

also rerun on this new 12-agent setting, demonstrating the effectiveness of MA^{3}SAC when scaling to larger multi-agent systems.

Figure 5.15 and Table 5.10 illustrate the results. Despite the existence of more agents, MAS^{3}AC successfully scales, providing stable reward improvement and best safety satisfaction.

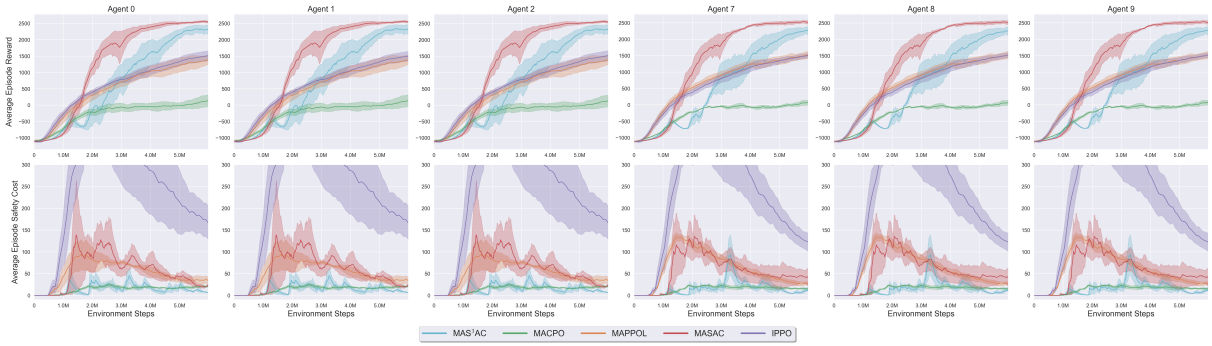


Figure 5.15: Results on the 12-agent non-cooperative decentralized Coupled HalfCheetah task. We report the training results of agent 0, 1, 2, 7, 8 and 9 for simplicity.

Table 5.10: Reward and cost per agent (12-agent non-cooperative decentralized Coupled HalfCheetah task, report the training results of agent 0, 1, 2, 7, 8 and 9 for simplicity).

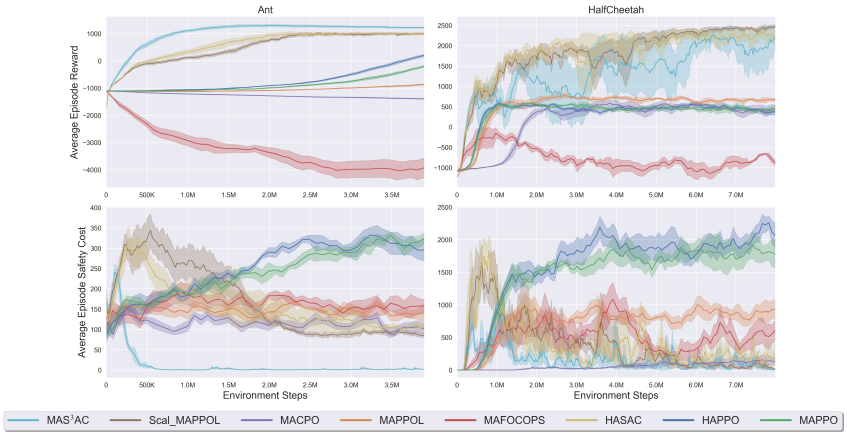
	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	1255.74 ± 372.60	242.76 ± 175.59	1246.24 ± 366.04	242.76 ± 175.59	1255.41 ± 374.03	242.76 ± 175.59
MACPO	31.32 ± 313.10	18.64 ± 22.76	31.77 ± 313.63	18.64 ± 22.76	27.10 ± 312.90	18.64 ± 22.76
MAPPOL	1154.25 ± 397.81	46.82 ± 32.72	1152.97 ± 398.61	46.82 ± 32.72	1153.45 ± 398.27	46.82 ± 32.72
MASAC	$\mathbf{2541.37 \pm 63.98}$	26.23 ± 21.01	$\mathbf{2542.25 \pm 63.80}$	26.23 ± 21.01	$\mathbf{2541.03 \pm 65.82}$	26.23 ± 21.01
MAS^{3}AC	2312.98 ± 227.15	$\mathbf{10.07 \pm 22.83}$	2318.55 ± 234.76	$\mathbf{10.07 \pm 22.83}$	2317.08 ± 233.75	$\mathbf{10.07 \pm 22.83}$

	Agent 7		Agent 8		Agent 9	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	1277.15 ± 332.06	218.90 ± 168.98	1290.05 ± 337.91	218.90 ± 168.98	1289.84 ± 335.09	218.90 ± 168.98
MACPO	0.75 ± 268.65	17.85 ± 21.58	-2.33 ± 268.52	17.85 ± 21.58	2.16 ± 268.55	17.85 ± 21.58
MAPPOL	1318.95 ± 346.96	43.45 ± 34.27	1321.36 ± 347.10	43.45 ± 34.27	1318.80 ± 346.23	43.45 ± 34.27
MASAC	$\mathbf{2513.88 \pm 88.55}$	43.84 ± 33.44	$\mathbf{2510.71 \pm 94.22}$	43.84 ± 33.44	$\mathbf{2515.29 \pm 89.45}$	43.84 ± 33.44
MAS^{3}AC	2219.22 ± 240.95	$\mathbf{8.63 \pm 19.38}$	2208.60 ± 238.10	$\mathbf{8.63 \pm 19.38}$	2220.00 ± 238.44	$\mathbf{8.63 \pm 19.38}$

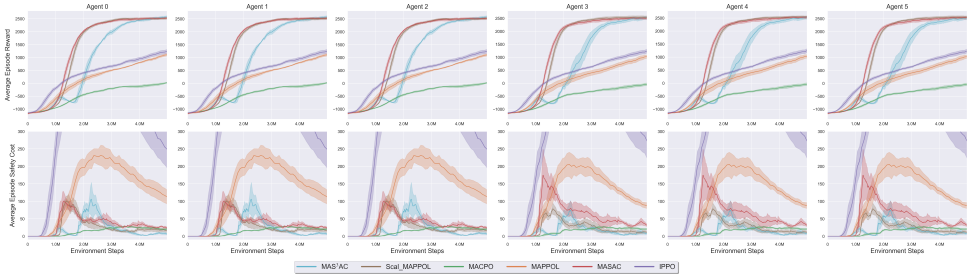
(iii) Comparison with the Additional Baseline (Modified Scal-MAPPO-L)

Here, we incorporated the modified Scal-MAPPO-L as an additional baseline to compare against the proposed algorithm. Scal-MAPPO-L [67] is a strong additional baseline, and: 1. it is a recent decentralized constrained MARL method accepted at NeurIPS 2024; 2. it has been tested on Safe MAMujoco tasks. Further, we made the following modifications to Scal-MAPPO-L: 1. We modified Scal-MAPPO-L to use max-over-time safety certificates (similar to the DCBF used in [91]) instead of CMDP constraints. 2. We implement an off-policy MASAC-based version instead of the original on-policy version because: (a) we already include MAPPO-Lagrangian as a baseline, and (b) this enables controlled comparison with our off-policy MAS^{3}AC.

Figure 5.16 and Tables 5.11 and 5.12 show the comparison on new baseline cooperative Ant, cooperative HalfCheetah, and non-cooperative Coupled HalfCheetah tasks. MAS^{3}AC consistently demonstrates superior performance, especially regarding safety.



(a) Comparison in cooperative Ant and HalfCheetah tasks.



(b) Comparison in the non-cooperative Coupled HalfCheetah task.

Figure 5.16: Comparison with the additional modified Scal-MAPPO-L baseline.

Table 5.11: Reward and cost comparison with the additional modified Scal-MAPPO-L baseline (cooperative tasks).

	Ant		HalfCheetah	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**
HAPPO	41.29 ± 215.89	311.96 ± 67.96	373.64 ± 200.58	2131.21 ± 483.13
HASAC	993.56 ± 87.49	107.49 ± 22.64	2229.41 ± 335.17	211.07 ± 300.58
MACPO	-1381.66 ± 37.86	105.24 ± 50.66	443.50 ± 265.72	134.58 ± 142.09
MAFOCOPS	-3965.01 ± 720.78	156.47 ± 77.84	-781.55 ± 302.59	581.41 ± 494.46
MAPPO	-354.58 ± 203.39	322.31 ± 64.85	418.49 ± 194.12	1834.08 ± 513.17
MAPPOL	-903.98 ± 67.67	142.52 ± 58.00	672.25 ± 143.48	898.40 ± 379.02
Scal_MAPPOL	997.24 ± 64.07	86.24 ± 12.39	**2460.87 ± 95.22**	22.44 ± 67.28
MAS^{3}AC	**1221.79 ± 50.45**	**2.10 ± 6.92**	2177.90 ± 714.48	**6.22 ± 25.28**

Table 5.12: Reward and cost comparison with the additional modified Scal-MAPPO-L baseline (non-cooperative Coupled HalfCheetah task).

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	902.29 ± 386.41	451.38 ± 305.37	904.99 ± 389.86	451.38 ± 305.37	905.91 ± 388.62	451.38 ± 305.37
MACPO	-140.28 ± 276.72	21.48 ± 25.26	-141.91 ± 276.68	21.48 ± 25.26	-142.75 ± 276.65	21.48 ± 25.26
MAPPOL	735.92 ± 416.97	174.96 ± 143.68	735.77 ± 417.75	174.96 ± 143.68	732.35 ± 417.00	174.96 ± 143.68
MASAC	2499.75 ± 78.45	25.25 ± 20.76	2503.90 ± 84.95	25.25 ± 20.76	2489.93 ± 81.61	25.25 ± 20.76
Scal_MAPPOL	2524.31 ± 84.52	18.51 ± 17.73	2516.04 ± 88.89	18.51 ± 17.73	2512.03 ± 83.29	18.51 ± 17.73
MAS^{3}AC	**2531.32 ± 136.89**	**6.35 ± 14.12**	**2533.62 ± 141.96**	**6.35 ± 14.12**	**2529.90 ± 142.89**	**6.35 ± 14.12**

	Agent 3		Agent 4		Agent 5	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	898.48 ± 395.57	432.92 ± 273.22	900.26 ± 397.68	432.92 ± 273.22	903.87 ± 397.26	432.92 ± 273.22
MACPO	-215.51 ± 277.43	21.14 ± 27.75	-216.04 ± 277.56	21.14 ± 27.75	-215.42 ± 277.29	21.14 ± 27.75
MAPPOL	646.70 ± 464.02	150.76 ± 134.02	648.57 ± 465.17	150.76 ± 134.02	648.17 ± 466.71	150.76 ± 134.02
MASAC	2479.16 ± 102.04	35.51 ± 26.71	2502.30 ± 96.44	35.51 ± 26.71	2501.68 ± 108.95	35.51 ± 26.71
Scal_MAPPOL	**2564.30 ± 89.97**	13.98 ± 19.73	**2554.35 ± 81.43**	13.98 ± 19.73	**2577.36 ± 87.96**	13.98 ± 19.73
MAS^{3}AC	2486.43 ± 172.32	**7.48 ± 20.11**	2488.44 ± 171.39	**7.48 ± 20.11**	2489.25 ± 175.59	**7.48 ± 20.11**

(iv) Performance on New Decentralized Ant Task

Here, we provide the results obtained from the **decentralized Ant** task. As depicted in Figure 5.17 and Table 5.13, MA^{3}SAC achieves the highest rewards and lowest final safety violations compared to baselines for almost all agents, which demonstrates the efficacy of the proposed algorithm on more decentralized non-cooperative task.

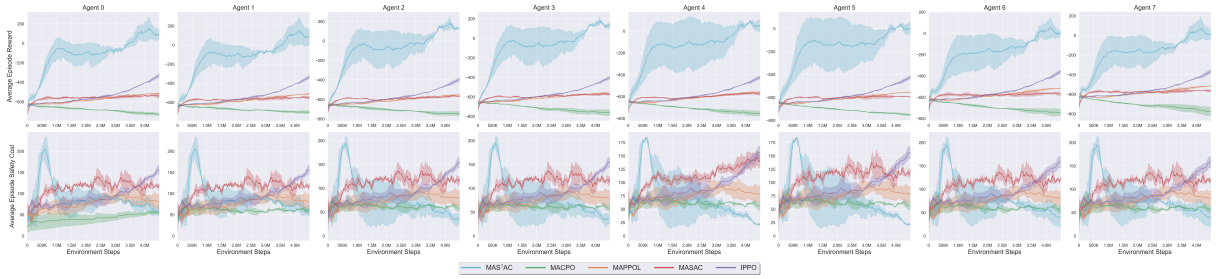


Figure 5.17: Performance on the non-cooperative Decentralized Ant task.

Table 5.13: Reward and cost per agent (non-cooperative Decentralized Ant task).

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	-473.73 ± 127.13	111.25 ± 56.83	-474.56 ± 124.97	111.25 ± 56.83	-514.63 ± 106.13	109.26 ± 56.65
MACPO	-704.12 ± 33.29	**47.89 ± 27.50**	-690.74 ± 26.51	**59.28 ± 32.32**	-729.84 ± 34.31	60.08 ± 32.47
MAPPOL	-552.92 ± 50.26	83.31 ± 40.09	-552.76 ± 51.65	83.31 ± 40.09	-587.23 ± 42.12	80.83 ± 38.76
MASAC	-539.92 ± 30.43	116.42 ± 50.14	-551.35 ± 29.67	116.42 ± 50.14	-573.06 ± 26.74	114.57 ± 49.26
MAS^{3}AC	**106.95 ±147.84**	65.33± 33.32	**102.38 ± 164.69**	65.33 ± 33.32	**143.49 ± 130.09**	**40.42 ± 26.09**

	Agent 3		Agent 4		Agent 5	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	-519.11 ± 102.94	109.26 ± 56.65	-531.28 ± 97.42	110.10 ± 57.07	-529.98 ± 95.81	110.10 ± 57.07
MACPO	-742.70 ± 42.06	60.08 ± 32.47	-733.88 ± 35.31	59.72 ± 34.65	-730.70 ± 29.61	59.72 ± 34.65
MAPPOL	-590.74 ± 41.15	80.83 ± 38.76	-590.14 ± 41.02	80.37 ± 37.84	-587.67 ± 39.76	80.37 ± 37.84
MASAC	-575.84 ± 26.83	114.57 ± 49.26	-573.30 ± 33.81	139.44 ± 57.73	-593.74 ± 23.85	115.38 ± 50.61
MAS^{3}AC	**139.48 ± 128.53**	**40.42 ± 26.09**	**56.88 ± 119.08**	**28.69± 22.59**	**4.71 ± 115.05**	**28.69 ± 22.59**

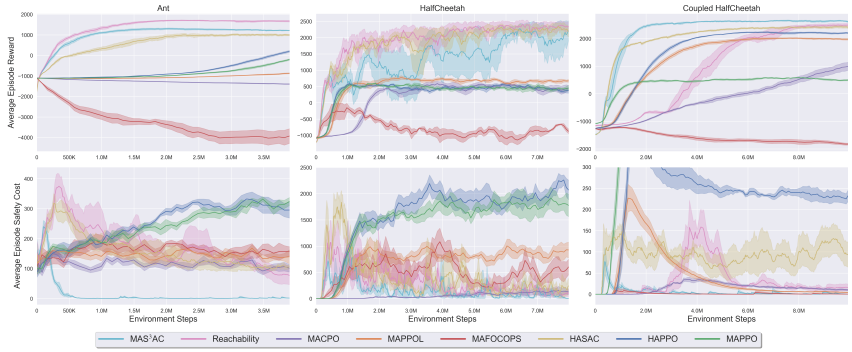
	Agent 6		Agent 7	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	-488.03 ± 116.87	112.29 ± 56.79	-493.59 ± 113.75	112.29 ± 56.79
MACPO	-721.77 ± 39.74	58.83 ± 34.20	-741.17 ± 51.10	58.83 ± 34.20
MAPPOL	-554.36 ± 48.57	83.16 ± 38.62	-558.00 ± 50.22	83.16 ± 38.62
MASAC	-565.41 ± 26.94	116.93 ± 50.39	-564.21 ± 25.39	116.93 ± 50.39
MAS^{3}AC	**16.19 ± 129.88**	**46.70 ± 32.85**	**30.22 ± 137.99**	**46.70 ± 32.85**

(v) Ablation: Max-over-time vs. Sum-over-time Certificates

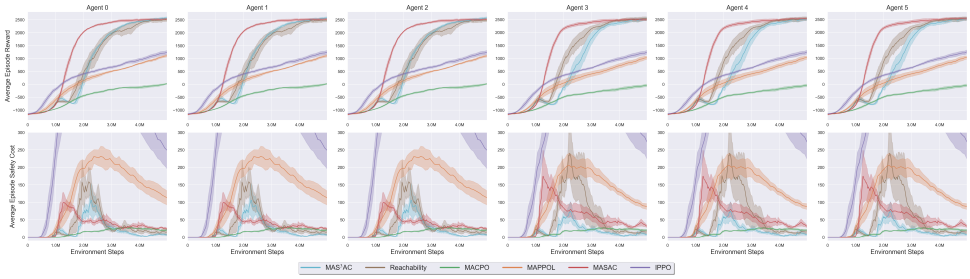
We replaced the safety constraints from the sum-over-time neural barrier-based certificates to the max-over-time neural reachability-based certificates [136], and compared the performance. Figure 5.18 and Tables 5.14 and 5.15 show the results. Based on the figures and statistics, it can be observed that for safety, using barrier certificates usually achieves better safety satisfaction than the counterpart with reachability-based certificates. One possible reason is the constraint to apply: barrier certificates enforce safety in an exponential manner (the barrier certificate value, if negative, is explicitly required to increase at an exponential rate to achieve safety), enabling stronger and faster safety enforcement, while the constraints using barrier certificates only require the value to be non-positive and updated via standard gradient descent; the additional exponential mechanism for rapid safety enforcement is missing.

(vi) Ablation: Impact of Input-to-State Stability (ISS) Part

To analyze how ISS helps to improve performance, we conducted an ablation study by completely removing the ISS part. Figure 5.19 and Table 5.16 show that the efficacy of the ISS part differs across tasks. In Ant, without ISS, the improvement in reward is slower than that of the



(a) Comparisons between different certificates on cooperative tasks.



(b) Comparisons between different certificates on non-cooperative Coupled HalfCheetah task.

Figure 5.18: Ablation study comparing the sum-over-time barrier-based certificates against the max-over-time reachability-based certificates.

Table 5.14: Reward and cost comparisons between different certificates on cooperative tasks.

	Ant		HalfCheetah		Coupled HalfCheetah	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
HAPPO	41.29 ± 215.89	311.96 ± 67.96	373.64 ± 200.58	2131.21 ± 483.13	2197.22 ± 112.77	230.28 ± 67.11
HASAC	993.56 ± 87.49	107.49 ± 22.64	$\mathbf{2229.41 \pm 335.17}$	211.07 ± 300.58	2410.71 ± 106.28	84.77 ± 65.91
MACPO	-1381.66 ± 37.86	105.24 ± 50.66	443.50 ± 265.72	134.58 ± 142.09	1104.05 ± 356.15	9.32 ± 8.53
MAFOCOPS	-3965.01 ± 720.78	156.47 ± 77.84	-781.55 ± 302.59	581.41 ± 494.46	-1865.72 ± 117.10	0.14 ± 0.97
MAPPO	-354.58 ± 203.39	322.31 ± 64.85	418.49 ± 194.12	1834.08 ± 513.17	480.86 ± 195.30	1764.08 ± 431.53
MAPPOL	-903.98 ± 67.67	142.52 ± 58.00	672.25 ± 143.48	898.40 ± 379.02	1968.86 ± 118.37	3.67 ± 3.39
Reachability	$\mathbf{1671.62 \pm 77.84}$	78.82 ± 63.39	1939.28 ± 138.34	272.27 ± 149.15	2513.78 ± 155.96	13.15 ± 17.28
MAS^{3}AC	1221.79 ± 50.45	$\mathbf{2.10 \pm 6.92}$	2177.90 ± 714.48	$\mathbf{6.22 \pm 25.28}$	$\mathbf{2596.56 \pm 105.95}$	$\mathbf{0.03 \pm 0.17}$

complete version where ISS is used, especially at the beginning of the training process. Moreover, regarding the final reward obtained, with ISS the final reward is higher than that achieved by the counterpart without ISS constraints. In the other two tasks, when ISS is removed, the reward cannot meaningfully increase and therefore remains at a low level. The reason is that with the barrier certificate for safety but without the ISS component to encourage convergence toward the desired state, the controlled agent becomes overly conservative during exploration. It is prevented from achieving high rewards by approaching the desired goal due to the strong repulsion imposed by the exponential safety constraint. In such settings, it is essential to incorporate an additional stimulus to encourage effective exploration while maintaining state-wise

Table 5.15: Reward and cost comparisons between different certificates on the non-cooperative Coupled HalfCheetah task.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	902.29 ± 386.41	451.38 ± 305.37	904.99 ± 389.86	451.38 ± 305.37	905.91 ± 388.62	451.38 ± 305.37
MACPO	-140.28 ± 276.72	21.48 ± 25.26	-141.91 ± 276.68	21.48 ± 25.26	-142.75 ± 276.65	21.48 ± 25.26
MAPPOL	735.92 ± 416.97	174.96 ± 143.68	735.77 ± 417.75	174.96 ± 143.68	732.35 ± 417.00	174.96 ± 143.68
MASAC	2499.75 ± 78.45	25.25 ± 20.76	2503.90 ± 84.95	25.25 ± 20.76	2489.93 ± 81.61	25.25 ± 20.76
Reachability	2479.88 ± 166.99	14.22 ± 26.00	2472.27 ± 161.99	14.22 ± 26.00	2471.29 ± 161.62	14.22 ± 26.00
MAS^{3}AC	$\mathbf{2531.32 \pm 136.89}$	$\mathbf{6.35 \pm 14.12}$	$\mathbf{2533.62 \pm 141.96}$	$\mathbf{6.35 \pm 14.12}$	$\mathbf{2529.90 \pm 142.89}$	$\mathbf{6.35 \pm 14.12}$

	Agent 3		Agent 4		Agent 5	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	898.48 ± 395.57	432.92 ± 273.22	900.26 ± 397.68	432.92 ± 273.22	903.87 ± 397.26	432.92 ± 273.22
MACPO	-215.51 ± 277.43	21.14 ± 27.75	-216.04 ± 277.56	21.14 ± 27.75	-215.42 ± 277.29	21.14 ± 27.75
MAPPOL	646.70 ± 464.02	150.76 ± 134.02	648.57 ± 465.17	150.76 ± 134.02	648.17 ± 466.71	150.76 ± 134.02
MASAC	2479.16 ± 102.04	35.51 ± 26.71	2502.30 ± 96.44	35.51 ± 26.71	2501.68 ± 108.95	35.51 ± 26.71
Reachability	$\mathbf{2506.60 \pm 194.16}$	14.05 ± 23.27	$\mathbf{2506.42 \pm 190.56}$	14.05 ± 23.27	$\mathbf{2504.92 \pm 187.60}$	14.05 ± 23.27
MAS^{3}AC	2486.43 ± 172.32	$\mathbf{7.48 \pm 20.11}$	2488.44 ± 171.39	$\mathbf{7.48 \pm 20.11}$	2489.25 ± 175.59	$\mathbf{7.48 \pm 20.11}$

safety, and this is exactly the role of the ISS part.

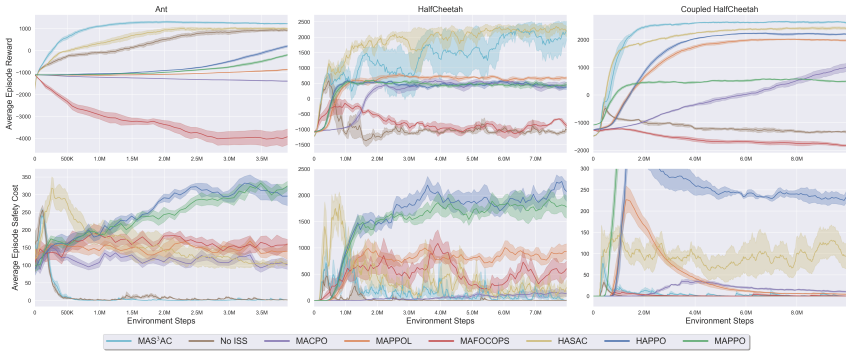


Figure 5.19: Ablation study on the ISS part.

Table 5.16: Reward and cost comparisons for ISS ablation study.

	Ant		HalfCheetah		Coupled	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
HAPPO	41.29 ± 215.89	311.96 ± 67.96	373.64 ± 200.58	2131.21 ± 483.13	2197.22 ± 112.77	230.28 ± 67.11
HASAC	993.56 ± 87.49	107.49 ± 22.64	$\mathbf{2229.41 \pm 335.17}$	211.07 ± 300.58	2410.71 ± 106.28	84.77 ± 65.91
MACPO	-1381.66 ± 37.86	105.24 ± 50.66	443.50 ± 265.72	134.58 ± 142.09	1104.05 ± 356.15	9.32 ± 8.53
MAFOCOPS	-3965.01 ± 720.78	156.47 ± 77.84	-781.55 ± 302.59	581.41 ± 494.46	-1865.72 ± 117.10	0.14 ± 0.97
MAPPO	-354.58 ± 203.39	322.31 ± 64.85	418.49 ± 194.12	1834.08 ± 513.17	480.86 ± 195.30	1764.08 ± 431.53
MAPPOL	-903.98 ± 67.67	142.52 ± 58.00	672.25 ± 143.48	898.40 ± 379.02	1968.86 ± 118.37	3.67 ± 3.39
No-ISS	921.66 ± 104.09	$\mathbf{1.53 \pm 1.53}$	-1043.67 ± 202.75	$\mathbf{0.00 \pm 0.00}$	-1317.85 ± 163.06	$\mathbf{0.00 \pm 0.00}$
MAS^{3}AC	$\mathbf{1221.79 \pm 50.45}$	2.10 ± 6.92	2177.90 ± 714.48	6.22 ± 25.28	$\mathbf{2596.56 \pm 105.95}$	0.03 ± 0.17

(vii) Ablation: No Reward Signal

Similarly, we considered completely removing the reward signal and only using the ISS stability term to guide the training. Based on these results, the reward signal also plays an important role in training: (1) It helps achieve a more efficient increase in reward, especially at the

beginning of training, as demonstrated in Ant. Reward signals provide useful direction for policy improvement. (2) In some tasks such as HalfCheetah, without the reward signal, the training may totally fail to achieve an increase in reward. The likely reason is that, on some tasks, without reward to provide a stable improvement direction, the applied constraint terms may compound and be large in size, which leads to numerical issues in the training and thus the learning fails.

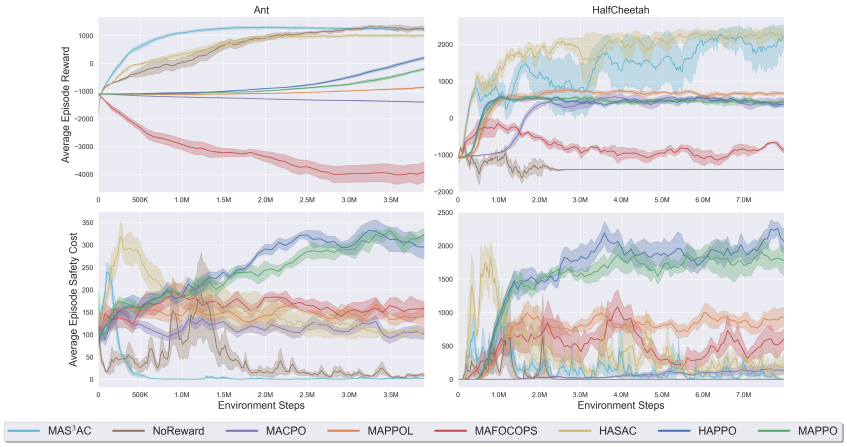


Figure 5.20: Ablation study on the reward signal.

Table 5.17: Reward and cost comparisons for ablation study on the reward signal.

	Ant		HalfCheetah	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**
HAPPO	41.29 ± 215.89	311.96 ± 67.96	373.64 ± 200.58	2131.21 ± 483.13
HASAC	993.56 ± 87.49	107.49 ± 22.64	$\mathbf{2229.41 \pm 335.17}$	211.07 ± 300.58
MACPO	-1381.66 ± 37.86	105.24 ± 50.66	443.50 ± 265.72	134.58 ± 142.09
MAFOCOPS	-3965.01 ± 720.78	156.47 ± 77.84	-781.55 ± 302.59	581.41 ± 494.46
MAPPO	-354.58 ± 203.39	322.31 ± 64.85	418.49 ± 194.12	1834.08 ± 513.17
MAPPOL	-903.98 ± 67.67	142.52 ± 58.00	672.25 ± 143.48	898.40 ± 379.02
No-Reward	$\mathbf{1258.19 \pm 167.19}$	8.78 ± 10.02	-1396.78 ± 5.49	$\mathbf{0.00 \pm 0.00}$
MAS^{3}AC	1221.79 ± 50.45	$\mathbf{2.10 \pm 6.92}$	2177.90 ± 714.48	6.22 ± 25.28

(viii) Ablation: Necessity of the Squared Term

We performed an experiment where the additional squared term was removed to observe the performance change. Based on the figures and statistics, it can be found that with the extra squared terms, the reward obtained will not experience noticeable deteriorations that are difficult to recover (for HalfCheetah, the reward curve finally falls down without efficient recovery when no extra squared terms are used. This squared term design also applies to stability constraints beyond safety constraints, and therefore, the reward curve also shows its efficacy). Regarding safety, the version with extra squared terms generally achieves better safety satisfaction during

the learning process on both tasks. This design is specifically proposed as an effort to minimize the safety violations, even during the learning time.

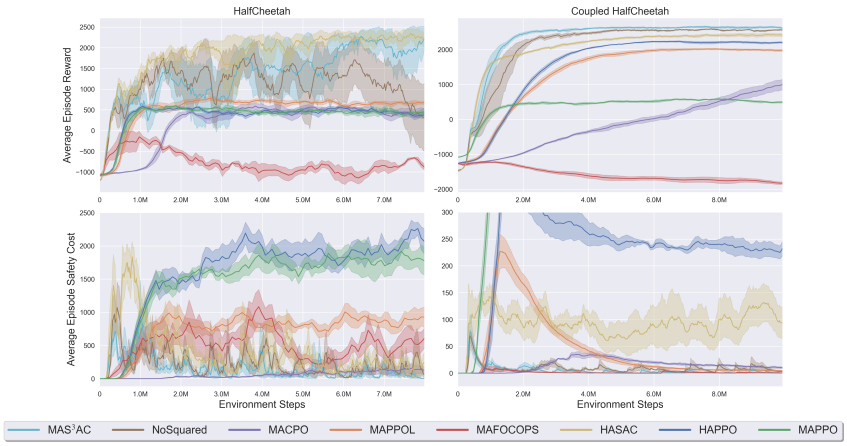


Figure 5.21: Ablation study on the additional squared term.

Table 5.18: Reward and cost comparisons for the ablation study on the additional squared term.

	HalfCheetah		Coupled HalfCheetah	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**
HAPPO	373.64 ± 200.58	2131.21 ± 483.13	2197.22 ± 112.77	230.28 ± 67.11
HASAC	$\mathbf{2229.41 \pm 335.17}$	211.07 ± 300.58	2410.71 ± 106.28	84.77 ± 65.91
MACPO	443.50 ± 265.72	134.58 ± 142.09	1104.05 ± 356.15	9.32 ± 8.53
MAFOCOPS	-781.55 ± 302.59	581.41 ± 494.46	-1865.72 ± 117.10	0.14 ± 0.97
MAPPO	418.49 ± 194.12	1834.08 ± 513.17	480.86 ± 195.30	1764.08 ± 431.53
MAPPOL	672.25 ± 143.48	898.40 ± 379.02	1968.86 ± 118.37	3.67 ± 3.39
No-Squared	342.48 ± 1599.78	161.24 ± 447.73	2564.67 ± 72.10	2.58 ± 6.56
MAS^{3}AC	2177.90 ± 714.48	$\mathbf{6.22 \pm 25.28}$	$\mathbf{2596.56 \pm 105.95}$	$\mathbf{0.03 \pm 0.17}$

(ix) Lagrangian Multipliers Evolution

Here, we draw the initial and end values of the Lagrangian multipliers for the cooperative ant task. Figure 5.22 illustrates the reward and cost for the task, and Figure 5.23 illustrates the adaptation of the Lagrangian multipliers corresponding to the safety and stability constraints over the course of training. The results show that, for all four agents, the Lagrangian multipliers associated with both the safety and stability constraints exhibit stable and gradual convergence.

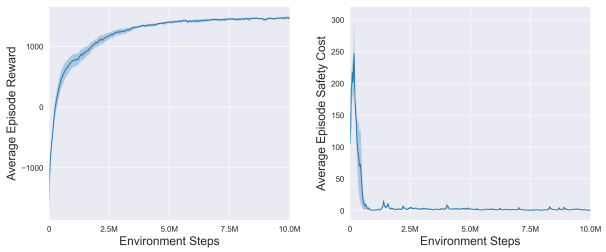


Figure 5.22: Reward and cost of the proposed MAS^{3}AC over the learning.

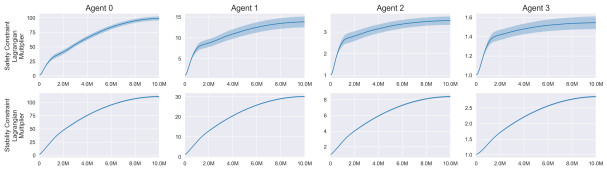
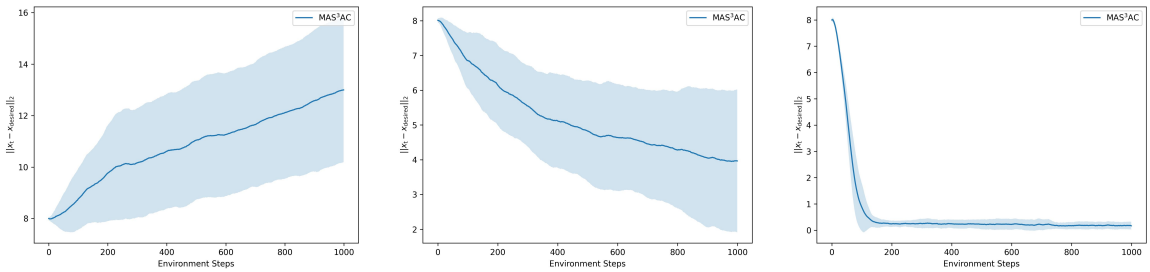


Figure 5.23: Evolution of the Lagrangian multipliers (corresponding to safety and stability constraints) of the proposed MAS^{3}AC over the learning.



(a) After 100K training steps **(b)** After 300K training steps **(c)** After 3M training steps

Figure 5.24: Differences between the desired state under the execution of the joint policy obtained via MAS^{3}AC after different training steps.

(x) Plots of Differences between the Desired State

In this part, we provide some plots on the differences between the controlled system's current state and the desired state to investigate whether the proposed MAS^{3}AC effectively helps to minimize such differences and encourages convergence towards the predefined desired goal. Figure 5.24 clearly shows that the proposed algorithm can effectively train the policy to drive the controlled multi-agent system towards the desired goal without noticeable divergence, and Figure 5.25 shows that MAS^{3}AC consistently achieves better minimization of the difference compared to standard and safe MARL baselines.

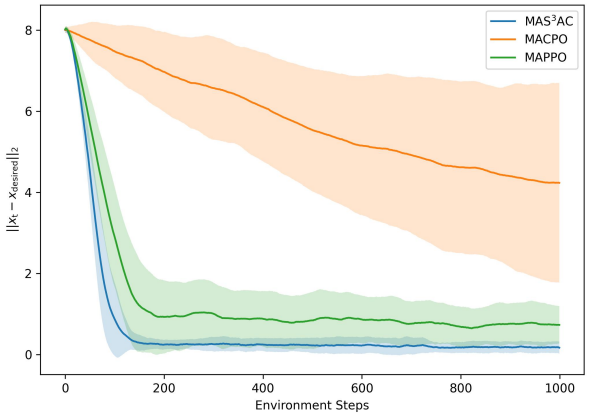


Figure 5.25: Comparisons among the differences between the desired state under the execution of joint policies obtained via MAS^{3}AC, MACPO, and MAPPO after 3M training steps.

CHAPTER 6

Improvements regarding Flexibility, Scalability, and Sample Efficiency

In the previous chapter, state-wise safe and stable RL was developed for general multi-agent systems in a fully model-free setting. In this chapter, we further extend this line of research by focusing on flexibility, scalability, and sample efficiency, which are critical for deploying safe and stable MARL algorithms in large-scale and data-limited environments. To this end, we refine the stability constraints, provide theoretical justification for decentralized training under limited information, and introduce NODE-based policy refinement to improve efficiency.

6.1 Introduction

Beyond safety and stability, the choice of training paradigm critically affects the scalability and practicality of MARL in MAS [26, 145]. Many existing algorithms adopt a centralized training (CT) paradigm, where agents update their policies using global information [70, 146, 147]. However, CT can become infeasible in some general tasks where global information is unavailable due to competition among agents. Moreover, the joint state and/or action spaces under CT scale exponentially with the number of agents, leading to severe communication and computation burdens, which is a manifestation of the "curse of dimensionality" [148]. These limitations restrict the applicability of CT to large-scale MAS. Recent advances in decentralized constrained cooperative MARL [67, 91, 131] have sought to mitigate these issues by enabling each agent to learn from local observations without global information. Nevertheless, decentralized constrained general MARL remains relatively underexplored, and the increased data demand of multi-agent training makes sample efficiency a critical bottleneck. Therefore, de-

veloping scalable and sample-efficient constrained MARL frameworks is essential for extending safe and stable RL to complex multi-agent environments.

Contributions: Building upon the above motivations, this chapter proposes a comprehensive framework that offers state-wise guarantees under mild assumptions for general multi-agent tasks with unknown dynamics, with scalability analysis and improved sample efficiency. The main contributions are:

1. We develop a unified framework that integrates neural barrier certificates for state-wise safety and the concept of input-to-state stability (ISS) [132, 133] for state-wise stability into general model-free MARL. The framework establishes the local condition for each agent to achieve stable control of networked MAS, equipped with theoretical proof and adaptive coefficients that enhance flexibility in stability enforcement. A safe, stable, and locally optimal joint controller is obtained via a modified dual ascent update, supported by a complete theoretical analysis proving the convergence of the joint policy update to a feasible local Nash equilibrium under both safety and stability constraints.

2. We derive an upper bound on the update error of a surrogate policy gradient estimator arising from the use of local observations instead of global information, under certain assumptions on transition dynamics and policies. This provides a theoretical justification for the decentralized training decentralized execution paradigm, where each agent updates its policy using only detected information under limited communication and computation resources. Furthermore, we design a Neural ordinary differential equation (NODE)-aided policy network, incorporating the heavy-ball method, to improve the sample efficiency of the training, such that it has a better applicability to large-scale MAS.

3. We propose **M**ulti-**A**gent **S**afe **S**table **S**calable and **S**ample-efficient **N**eural ordinary differential equation–based **A**ctor–**C**ritic (**MAS^{4}NAC**), a unified framework for model-free multi-agent safe and stable control under centralized or decentralized training paradigms. Empirical evaluations across cooperative and general-sum tasks demonstrate that MAS^{4}NAC consistently achieves a strong balance between reward maximization and constraint satisfaction. Moreover, ablation studies show that incorporating the NODE-based policy

refinement design effectively improves sample efficiency, highlighting the framework's superior applicability.

6.2 Methodology

The design and analysis presented in this chapter build directly upon the material introduced in Chapter 5, as summarised below:

1. The problem formulation and preliminaries, including the multi-agent reinforcement learning setup, the definitions of safety and stability, and the description of the *Stable and Safe Control Problem for MAS*, as presented in Section 5.2 of Chapter 5.
2. The definitions of state sets, such as $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{unsafe}}$ and $\boldsymbol{\mathcal{X}}^{\boldsymbol{\pi},i}_{\text{safe}}$ (Definition 5.2), together with the definitions and results related to barrier certificates, including Definition 5.3, Lemma 5.1, and Proposition 5.1, detailed in Section 5.3.1 of Chapter 5.

The material reviewed above establishes the state-wise safety constraints also considered in this chapter. We now proceed to introduce a variant of the stability constraints in which adaptive coefficients are incorporated to enhance flexibility in stability enforcement.

6.2.1 Restated State-Wise Stability Constraints and Convergence Analysis

In this subsection, we study stability via the Lyapunov certificates taking only local observations as inputs, motivated by the decentralized training decentralized execution paradigm. This paradigm is particularly desirable for large-scale multi-agent systems and is detailed in Subsection 6.2.2 with analysis.

Similar to the setting described at the beginning of Section 5.3.2, according to the graph structure $\mathcal{G}$, the MDP is partitioned into N subsystems, each corresponding to one agent. For each agent i, a cost signal is defined as $c^i(x^i_t) = ||x^i_t - x^i_{\text{desired}}||$, where, still without loss of generality, we set the agent i's desired equilibrium x^i_{desired} to be $\mathbf{0}$ for the subsequent analysis. The value function associated with this cost, termed Lyapunov certificates, is given by $V^i_{\boldsymbol{\pi}}(x^i_t) = \mathbb{E}_{\tau^i\sim\boldsymbol{\pi}}\big[\sum_{k=0}^{\infty}\gamma^k c^i(x^i_{t+k})\big]$, where $\tau^i = \{x^i_t, x^i_{t+1}, x^i_{t+2}\ldots\}$ is the trajectory of agent i under $\boldsymbol{\pi}$ starting from x^i_t. $\delta_{ij} \in \mathcal{K}\cup\{0\}, \forall i,j \in \mathcal{N}$ denote the gain functions representing inter-

agent couplings, and $\kappa_i(||\boldsymbol{d}_t||)$, where $\kappa_i \in \mathcal{K} \cup \{0\}$, represent the influence of exogenous signal $\boldsymbol{d}_t$ on agent i. We now propose a variant of stability constraints introduced in Section 5.3.2, in which adaptive coefficients are incorporated to enhance flexibility in stability enforcement. We begin with the following theorem, where certain conditions required in Theorem 5.2 are no longer assumed *a priori*, but are instead enforced explicitly as constraints during training.

Theorem 6.1. *Suppose there exists a set of $\mathcal{K}_\infty$ functions $\{\mu_i\}_{i\in\mathcal{N}}$ satisfying $V^i_{\boldsymbol{\pi}}(x^i_t) \leq \mu_i(||x^i_t||)$, $\forall i \in \mathcal{N}$, for each state sampled under certain joint policy $\boldsymbol{\pi}$. Then, if for a set of $\mathcal{K}_\infty$ functions $\{\beta_i\}_{i\in\mathcal{N}}$, there exists a set of $\mathcal{K}_\infty$ functions $\{\alpha_i\}_{i\in\mathcal{N}}$ such that:*

$$\sum_{j\neq i} \delta_{ij}(V^j_{\boldsymbol{\pi}}(x^j_t)) + \beta_i(\sum_{j\neq i} \delta_{ij}(V^j_{\boldsymbol{\pi}}(x^j_t))) \leq \alpha_i(V^i_{\boldsymbol{\pi}}(x^i_t)), \tag{6.1a}$$

$$\dot{V}^i_{\boldsymbol{\pi}}(x^i_t) \leq -\alpha_i(V^i_{\boldsymbol{\pi}}(x^i_t)) + \sum_{j\neq i} \delta_{ij}(V^j_{\boldsymbol{\pi}}(x^j_t)) + \kappa_i(||\boldsymbol{d}_t||), \tag{6.1b}$$

for all $i \in \mathcal{N}$ and state sampled under $\boldsymbol{\pi}$, we have $||\boldsymbol{x}_t|| \leq \xi(||\boldsymbol{x}_0||, t) + \rho(||\boldsymbol{d}||_\infty)$. Here $\xi \in \mathcal{KL}$ is a function such that $\xi(\cdot, t) \in \mathcal{K}$ for all $t \geq 0$, and $\xi(r, \cdot) \in \mathcal{L}$ for all $r > 0$, where $\mathcal{L}$ is the set of strictly decreasing functions with $\lim_{t\to\infty} \xi(||\boldsymbol{x}_0||, t) = 0$. $\rho \in \mathcal{K}$ and $||\boldsymbol{d}||_\infty = sup\{||\boldsymbol{d}_t||, t \geq 0\}$.

This theorem follows as a variant of Theorem 5.2, and its proof is therefore provided in Appendix 5.6.4 (i). The above theorem indicates that under some mild conditions, the networked multi-agent system is input-to-state stable when conditions in (6.1) using partial information (e.g., Lyapunov certificate values of other agents denoted as $j \neq i$) are satisfied. Still, since the joint control action is generated by policy $\boldsymbol{\pi}$ based on the system state, rather than by any exogenous generator during training, the networked system can be regarded as autonomous, and the term $\kappa_i(||\boldsymbol{d}_t||)$ can be discarded when other external disturbances are negligible. By replacing the derivative with the difference between consecutive steps, and treating all $\mathcal{K}_\infty$ functions as proportional functions, using the same symbols to denote their corresponding coefficients for simplicity, we can have the optimization problem for agent i:

$$\pi^i_{e+1} = \arg\min_{\pi^i} \ -Q^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, (\boldsymbol{\pi}^{1:i-1}_{e+1}(\boldsymbol{x}_t), \pi^i(\boldsymbol{x}_t), \boldsymbol{\pi}^{i+1:N}_e(\boldsymbol{x}_t))) \tag{6.2a}$$

$$s.t. - H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) + H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) - \epsilon H^i_{\hat{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) \leq 0 \tag{6.2b}$$

$$\sum_{j\neq i}\delta_{ij}V^j_{\hat{\boldsymbol{\pi}}_i}(x^j_t,u^j_t)+\beta_i(\sum_{j\neq i}\delta_{ij}V^j_{\hat{\boldsymbol{\pi}}_i}(x^j_t,u^j_t))-\alpha_i V^i_{\hat{\boldsymbol{\pi}}_i}(x^i_t,u^i_t)\leq 0 \tag{6.2c}$$

$$V^i_{\hat{\boldsymbol{\pi}}_i}(x^i_{t+1},u^i_{t+1})-V^i_{\hat{\boldsymbol{\pi}}_i}(x^i_t,u^i_t)+\alpha_i V^i_{\hat{\boldsymbol{\pi}}_i}(x^i_t,u^i_t)-\sum_{j\neq i}\delta_{ij}V^j_{\hat{\boldsymbol{\pi}}_i}(x^j_t,u^j_t)\leq 0, \tag{6.2d}$$

where $\hat{\boldsymbol{\pi}}_i=(\boldsymbol{\pi}_{e+1}^{1:i-1},\boldsymbol{\pi}_e^{i:N})$. The joint control action $\boldsymbol{u}_{t+1}$, which includes u^i_{t+1}, and stability parameter α_i are both outputs of the updated joint policy $(\boldsymbol{\pi}_{e+1}^{1:i},\boldsymbol{\pi}_e^{i+1:N})$, with data sampled under $\hat{\boldsymbol{\pi}}_i$. It is worth noting that here barrier certificate $H^i_{\hat{\boldsymbol{\pi}}_i}$ and Lyapunov certificates $V^i_{\hat{\boldsymbol{\pi}}_i},\forall i\in\mathcal{N}$ are implemented as action-value networks, following the common practice in actor-critic learning. Moreover, the stability parameter α_i is treated as a learnable variable by making it one of the outputs of the control policy π^i. This design choice is motivated by two key considerations. First, selecting an appropriate fixed value for α_i often requires prior knowledge of the controlled task, which may not always be available. Second, allowing α_i to be trainable and adaptive to input information enhances the flexibility of stability enforcement, enabling dynamic safety–stability trade-offs and mitigating potential infeasibility issues [23] arising from conflicting state-wise safety and stability constraints.

Remark 6.1. *The above formulation represents an idealized constrained optimization problem defined over the full state space. In practical implementations, however, the expectations and constraints involved in the optimization are approximated using sampled trajectories collected through interactions with the environment, together with replay-buffer-based learning and neural function approximation. Consequently, the implemented algorithm should be viewed as a sampled approximation of the idealized optimization problem rather than an exact realization of it. The practical algorithm seeks to approximate the idealized constrained optimization problem for corresponding desired properties through data-driven learning.*

Based on the above analysis, two requirements still should be satisfied: (1). When updating the policy, critic functions, e.g., $Q^i_{\boldsymbol{\pi}}$, $H^i_{\boldsymbol{\pi}}$, and $V^i_{\boldsymbol{\pi}}$, must be accurately approximated for the corresponding agent $i\in\mathcal{N}$. (2). When updating the individual policy π^i of agent i multiple times via gradient descent to expand its certified safe region, all other agents should keep their certain joint policy $\boldsymbol{\pi}^{-i}$. This is crucial for agent i to update toward an improved policy that is consistent with and dependent on the current policies of other agents during its update phase.

Thus, we adopt a multi-timescale approach to design an iterative update process that can be proved to converge to a feasible local Nash equilibrium (developed from the one in the previous chapter), with the two aforementioned requirements being theoretically satisfied.

Definition 6.1 (Feasible Local Nash Equilibrium). *With $Q^i_{\boldsymbol{\pi}}$ denoting the action-value function of agent i, and $b^i(\boldsymbol{x}_t)$ denoting the state-wise safety for agent i, $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$ is a local Nash equilibrium [134] if for each $i \in \mathcal{N}$, there exists a neighborhood $\mathcal{W}^i \subseteq \Pi^i$ of π_i^* such that $Q^i_{(\pi_i^*, \boldsymbol{\pi}^*_{-i})}(\boldsymbol{x}, \boldsymbol{u}) \geq Q^i_{(\pi^i, \boldsymbol{\pi}^*_{-i})}(\boldsymbol{x}, \boldsymbol{u}), \quad \forall \pi^i \in \mathcal{W}^i$. If, in addition, for all $i \in \mathcal{N}$, the joint policy $\boldsymbol{\pi}^*$ satisfies: 1. The safety constraint: $Pr(b^i(\boldsymbol{x}_t) < 0) = 0,\ \forall t \geq 0$. 2. The stability constraint: $\lim_{t\to\infty} ||\boldsymbol{x}_t - \boldsymbol{x}_{desired}|| = 0$. Then it is called a **feasible local Nash equilibrium**.*

For the RL training, we use ϕ_i, ω_i and χ_i to parameterize and approximate the action-value function $Q^i_{\boldsymbol{\pi}}$, barrier certificate $H^i_{\boldsymbol{\pi}}$, and Lyapunov certificate $V^i_{\boldsymbol{\pi}}$, respectively, with corresponding loss functions J_{Q^i}, J_{H^i}, and J_{V^i} for each agent i (explicit forms are provided in Appendix 6.5.1). In a general case, for each $i \in \mathcal{N}$, we augment the Lagrangian function with additional squared penalty terms. Data and $\tilde{\boldsymbol{u}}_{t+1}$ are sampled according to joint policy $\bar{\boldsymbol{\pi}}_i = (\pi^i, \boldsymbol{\pi}^{-i})$ where π^i is the policy of agent i to be updated and $\boldsymbol{\pi}^{-i}$ is a certain joint policy of all other agents. $\tilde{u}^i_{t+1}$, and α_i are output by π^i. If π^i is parameterized by θ_i and $\boldsymbol{\pi}^{-i}$ by $\boldsymbol{\theta}_{-i}$, then, for agent i, the modified Lagrangian function with squared terms is:

$$
\begin{aligned}
&\mathcal{L}^i_{\boldsymbol{\theta}^{-i}}(\theta_i, \lambda_i) = \mathbb{E}_{\bar{\boldsymbol{\pi}}_i}\big[- Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) + \lambda_{i,1} \sum_{t=0}^{\infty} \gamma^t \max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) + (1-\epsilon) H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t)\} \\
&+ \frac{\lambda_{i,1}}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) + (1-\epsilon) H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}_t, \boldsymbol{u}_t)\}]^2 + \lambda_{i,2} \sum_{t=0}^{\infty} \gamma^t \max\{0, \\
&(1+\beta_i)(\sum_{j\neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(x^j_t, u^j_t)) - \alpha_i(\boldsymbol{x}_t) V^i_{\bar{\boldsymbol{\pi}}_i}(x^i_t, u^i_t)\} + \frac{\lambda_{i,2}}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, (1+\beta_i)(\sum_{j\neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(x^j_t, u^j_t)) \\
&- \alpha_i(\boldsymbol{x}_t) V^i_{\bar{\boldsymbol{\pi}}_i}(x^i_t, u^i_t)\}]^2 + \lambda_{i,3} \sum_{t=0}^{\infty} \gamma^t \max\{0, V^i_{\bar{\boldsymbol{\pi}}_i}(x^i_{t+1}, \tilde{u}^i_{t+1}) - (1-\alpha_i(\boldsymbol{x}_t)) V^i_{\bar{\boldsymbol{\pi}}_i}(x^i_t, u^i_t) - \sum_{j\neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(x^j_t, u^j_t)\} \\
&+ \frac{\lambda_{i,3}}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, V^i_{\bar{\boldsymbol{\pi}}_i}(x^i_{t+1}, \tilde{u}^i_{t+1}) - (1-\alpha_i(\boldsymbol{x}_t)) V^i_{\bar{\boldsymbol{\pi}}_i}(x^i_t, u^i_t) - \sum_{j\neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(x^j_t, u^j_t)\}]^2\big].
\end{aligned}
\tag{6.3}
$$

Here $\lambda_{i,1}$, $\lambda_{i,2}$ and $\lambda_{i,3}$ (collectively denoted as λ_i) are Lagrangian multipliers associated with the corresponding constraints of agent i. Finally, we denote J_{θ_i} and J_{λ_i} as loss functions to

update θ_i and λ_i, respectively, with their detailed forms in Appendix 6.5.1. With these loss functions, we provide Algorithm 6.1, together with the following theorem:

Algorithm 6.1 State-Wise Safe and Stable MARL via Neural Barrier Certificates, ISS, and Multi-Timescales

Input: Learning rates $\eta_1(e)$, $\eta_{2,i}(e)$, and $\eta_{3,i}(e)$. Initialized parameters $\{\phi_i^0\}_{i\in\mathcal{N}}$, $\{\omega_i^0\}_{i\in\mathcal{N}}$, $\{\theta_i^0\}_{i\in\mathcal{N}}$, $\{\lambda_i^0\}_{i\in\mathcal{N}}$ for corresponding neural networks and parameters.

for $e = 0, 1, \ldots$ **do**
  for $i = 1, 2, \ldots, N$ **do**
    $\phi_i^{e+1} \leftarrow \phi_i^e - \eta_1(e)\nabla_{\phi_i} J_{Q^i}(Q^i)$.
    $\omega_i^{e+1} \leftarrow \omega_i^e - \eta_1(e)\nabla_{\omega_i} J_{H^i}(H^i)$.
    $\chi_i^{e+1} \leftarrow \chi_i^e - \eta_1(e)\nabla_{\chi_i} J_{V^i}(V^i)$.
    $\theta_i^{e+1} \leftarrow \theta_i^e - \eta_{2,i}(e)\nabla_{\theta_i} J_{\theta_i}(\theta_i)$.
    $\lambda_i^{e+1} \leftarrow [\lambda_i^e + \eta_{3,i}(e)\nabla_{\lambda_i} J_{\lambda_i}(\lambda_i)]_+$.
  end for
end for

Output: $\{\theta_i\}_{i\in\mathcal{N}}$

Informal Theorem 6.2. *Under certain assumptions, the joint policy updated in Algorithm 6.1 will almost surely converge to a feasible local Nash equilibrium* $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$*, where* $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) \geq 0$ *for all* $(\boldsymbol{x}_t, \boldsymbol{u}_t)$ *along the trajectory under* $\boldsymbol{\pi}^*$*, for every* $i \in \mathcal{N}$ *and* $t \geq 0$*. Define* $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) := \min_{i\in\mathcal{N}} H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$*, then* $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ *is a global barrier certificate for the MAS. Furthermore, in the absence of exogenous signal* $\boldsymbol{d}_t$*, every agent* i *will converge to its desired state (goal),* $x^i_{desired}$*, under the policy* $\boldsymbol{\pi}^*$*.*

The formal version and proof of this theorem are in Appendix 6.5.3, and the detailed description of the multi-timescale approach are given in Appendix 6.5.2 (Assumption 6.4).

6.2.2 Scalability Analysis for the Decentralized Training Decentralized Execution Paradigm

In the previous subsections, the centralized scheme was considered. For safety, the learned barrier certificates $H^i_{\boldsymbol{\pi}}$ take global state-action information as inputs, which corresponds to a centralized training paradigm. For stability, even if the Lyapunov certificates $V^i_{\boldsymbol{\pi}}$ take only local observations as inputs, each individual policy can still use the global state information as input, resulting in the centralized execution scheme. In this subsection, we investigate the decentralized training decentralized execution (DTDE) version of Algorithm 6.1 with reduced communication and computation requirements, supported by a theoretical scalability analysis.

For tasks where unconstrained and instantaneous information exchange among all agents is allowed and global information is available to each agent, regardless of communication or computational burden, the centralized training centralized execution (CTCE) scheme can be applied to train, where an ideal loss function to update the policy parameter θ_i for agent i is:

$$J_{\theta_i}^{\text{real}}(\theta_i) = \mathcal{L}_{\boldsymbol{\theta}^{-i}}^{i}(\theta_i, \lambda_i). \tag{6.4}$$

Namely, we use the $\mathcal{L}_{\boldsymbol{\theta}^{-i}}^{i}(\theta_i, \lambda_i)$ proposed in (6.3), where global information is used as input to both the policy π^i (to generate the action and the stability parameter α_i) and all critic functions (i.e., $Q_{\bar{\boldsymbol{\pi}}_i}^{i}$, $H_{\bar{\boldsymbol{\pi}}_i}^{i}$ and $V_{\bar{\boldsymbol{\pi}}_i}^{i}$), as the loss function to realize the CTCE scheme. Notably, the ideal loss function requires the precise $Q_{\bar{\boldsymbol{\pi}}_i}^{i}$, $H_{\bar{\boldsymbol{\pi}}_i}^{i}$, and $V_{\bar{\boldsymbol{\pi}}_i}^{i}$ instead of networks being parameterized. Under this ideal setting, the full CTCE scheme is achieved without invalidating the convergence analysis shown in Theorem 6.2. Nevertheless, even if $\nabla_{\theta_i}^{\text{real}} J_{\theta_i}(\theta_i)$ represents the precise and ideal gradient to update the policy of agent i, its computation relies on (1) accurately learned critic functions, (2) full availability of global information, and (3) rich computation resources. These requirements make the CTCE scheme impractical for many settings, including (1) non-cooperative tasks where global information is unavailable due to the competitive nature of tasks, and (2) large-scale tasks where global communication and required computation become prohibitively expensive due to the curse of dimensionality.

To address such challenges, the decentralized training decentralized execution (DTDE) version of Algorithm 6.1 should be investigated for solving large-scale tasks or tasks where global information is not available by task nature (see the last three tasks in Section 5.6.7 for examples). However, a notable concern with such DTDE paradigm is the update error induced by the loss of information outside the agent's local detection area [67, 131]. Interestingly, we found that such update error decreases exponentially with the radius of agent i' s local detection area, for neighboring agents' information, under certain conditions. First, define W^{ij} to be:

$$W^{ij} = \sup_{z^j, \breve{z}^j, \mathbf{z}^{-j}} \left\| P^i(\cdot | z^j, \mathbf{z}^{-j}) - P^i(\cdot | \breve{z}^j, \mathbf{z}^{-j}) \right\|_1, \tag{6.5}$$

where $z^j = (x^j, u^j)$ and $\breve{z}^j = (\breve{x}^j, \breve{u}^j)$ represent two different state-action pairs of agent j,

respectively, and $\mathbf{z}^{-j}$ denotes the state-action pair of all agents other than j. The value of W^{ij} reflects the extent to which the transition of agent i is influenced by agent j's state and action. Then, we give the following assumptions [67, 149]:

Assumption 6.1 (Spatial Decay of Correlation for the Dynamics)**.** *There exists $l > 0$ such that for any agents $i, j \in \mathcal{N}$, we have:*

$$\max_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} e^{ld(i,j)} W^{ij} \leq q, \tag{6.6}$$

where $d(i, j)$ represents the distance between agent i and j, and $q < \frac{2}{\gamma}$ is a non-negative constant where γ is the discount factor.

Next, let $\boldsymbol{X}_{N_{k,t}^i}$ denote the set of states of agent i's neighbors, within the allowed communication radius k at time t, and $\boldsymbol{X}_{N_{k,t}^{-i}}$ the set of states outside this neighborhood. We then introduce an additional assumption:

Assumption 6.2 (Spatial Decay of Correlation for the Policy)**.** *There exist $v, w \geq 0$ such that for any agent $i \in \mathcal{N}$, and state $\boldsymbol{x}_{N_{k,t}^i} \in \boldsymbol{X}_{N_{k,t}^i}$, $\boldsymbol{x}_{N_{k,t}^{-i}} \in \boldsymbol{X}_{N_{k,t}^{-i}}$ and $\breve{\boldsymbol{x}}_{N_{k,t}^{-i}} \in \boldsymbol{X}_{N_{k,t}^{-i}}$, we have*

$$\sup_{\boldsymbol{x}_{N_{k,t}^i}, \boldsymbol{x}_{N_{k,t}^{-i}}, \breve{\boldsymbol{x}}_{N_{k,t}^{-i}}} \left| \pi^i \left(\cdot | \boldsymbol{x}_{N_{k,t}^i}, \boldsymbol{x}_{N_{k,t}^{-i}} \right) - \pi^i \left(\cdot | \boldsymbol{x}_{N_{k,t}^i}, \breve{\boldsymbol{x}}_{N_{k,t}^{-i}} \right) \right| \leq v e^{-wk}. \tag{6.7}$$

For more explanations of these two assumptions and justification of their applicability, we refer the reader to [67, Appendix B]. Using the same notation for neighboring agents' state–action pairs, define $\boldsymbol{x}_t = (\boldsymbol{x}_{N_{k,t}^i}, \boldsymbol{x}_{N_{k,t}^{-i}})$, $\boldsymbol{u}_t = (\boldsymbol{u}_{N_{k,t}^i}, \boldsymbol{u}_{N_{k,t}^{-i}})$ as the true global state–action information, and $\breve{\boldsymbol{x}}_t = (\boldsymbol{x}_{N_{k,t}^i}, \breve{\boldsymbol{x}}_{N_{k,t}^{-i}})$, $\breve{\boldsymbol{u}}_t = (\boldsymbol{u}_{N_{k,t}^i}, \breve{\boldsymbol{u}}_{N_{k,t}^{-i}})$ as a different state-action pair in which only the observations within the allowed detection area $N_{k,t}^i$ correctly matches the real information, while $(\breve{\boldsymbol{x}}_{N_{k,t}^{-i}}, \breve{\boldsymbol{u}}_{N_{k,t}^{-i}})$ being any fixed state-action pair for agents outside $N_{k,t}^i$ used

as an arbitrary placeholder. Then, a surrogate loss function to update θ_i is:

$$\tilde{J}_{\theta_i}(\theta_i) = \mathbb{E}_{\bar{\boldsymbol{\pi}}_i}\big[- Q^i(\breve{\boldsymbol{x}}_t, \breve{\boldsymbol{u}}_t; \phi_i) + \lambda_{i,1} \sum_{t=0}^{\infty} \gamma^t \max\{0, -H^i(\breve{\boldsymbol{x}}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}; \omega_i) + (1-\epsilon) H^i(\breve{\boldsymbol{x}}_t, \breve{\boldsymbol{u}}_t; \omega_i)\}$$
$$+ \frac{\lambda_{i,1}}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, -H^i(\breve{\boldsymbol{x}}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}; \omega_i) + (1-\epsilon) H^i(\breve{\boldsymbol{x}}_t, \breve{\boldsymbol{u}}_t; \omega_i)\}]^2 + \lambda_{i,2} \sum_{t=0}^{\infty} \gamma^t \max\{0,$$
$$(1+\beta_i)(\sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)) - \alpha_i(\breve{\boldsymbol{x}}_t) V^i(x_t^i, u_t^i; \chi_i)\} + \frac{\lambda_{i,2}}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, (1+\beta_i)(\sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j))$$
$$- \alpha_i(\breve{\boldsymbol{x}}_t) V^i(x_t^i, u_t^i; \chi_i)\}]^2 + \lambda_{i,3} \sum_{t=0}^{\infty} \gamma^t \max\{0, V^i(x_{t+1}^i, \tilde{u}_{t+1}^i; \chi_i) - (1 - \alpha_i(\breve{\boldsymbol{x}}_t)) V^i(x_t^i, u_t^i; \chi_i) - \sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)\}$$
$$+ \frac{\lambda_{i,3}}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, V^i(x_{t+1}^i, \tilde{u}_{t+1}^i; \chi_i) - (1 - \alpha_i(\breve{\boldsymbol{x}}_t)) V^i(x_t^i, u_t^i; \chi_i) - \sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)\}]^2\big]. \tag{6.8}$$

Here, $\breve{\boldsymbol{x}}_t$, $\breve{\boldsymbol{u}}_t$ and $\breve{\boldsymbol{x}}_{t+1}$ are used as input to both critic networks, and policy to generate α_i and $\tilde{u}_{t+1}^i$, which is also a component of $\tilde{\boldsymbol{u}}_{t+1}$, to represent the case where only the observations from the allowed detection area $N_{k,t}^i$ is accurate, while the other part, denoted as $(\breve{\boldsymbol{x}}_{N_{k,t}^{-i}}, \breve{\boldsymbol{u}}_{N_{k,t}^{-i}})$, does not reflect the true state-action pair of further agents and thus the information outside the local detection area is lost. Notably, the resulting x_{t+1}^i here in (6.8) evolves from the inaccurate input $(\breve{\boldsymbol{x}}_t, \breve{\boldsymbol{u}}_t)$, and thus is different from the x_{t+1}^i in $J_{\theta_i}^{\text{real}}(\theta_i)$ that starts from the correct global information $(\boldsymbol{x}_t, \boldsymbol{u}_t)$ shown in (6.3). Parameterized critic networks Q^i, H^i, and V^i, rather than exact critic functions, are used to construct this surrogate loss function for update in practice. Based on the two loss functions, we provide the following theorem for off-policy RL training:

Informal Theorem 6.3. *Suppose that Assumption 6.1, 6.2 and certain other assumptions hold, then, under the multi-timescale update scheme used in Algorithm 6.1, if sizes of all signals, e.g., reward, barrier and cost signals, are upper bounded for each agent, we have:*

$$\|\nabla_{\theta_i} J_{\theta_i}^{real}(\theta_i) - \nabla_{\theta_i} \tilde{J}_{\theta_i}(\theta_i)\|_2 \leq E_{i,1} e^{-lk} + E_{i,2} e^{-wk} = \mathcal{O}(e^{-\min\{l,w\}k}), \tag{6.9}$$

where $E_{i,1}$ and $E_{i,2}$ are constants.

The formal version and its proof are provided in Appendix 6.5.4. This theorem indicates that the error between the true policy gradient and the surrogate gradient $\nabla_{\theta_i} \tilde{J}_{\theta_i}(\theta_i)$, used in

the DTDE paradigm for off-policy RL training where inputs to all critic and policy networks are not global information, decreases exponentially with k under certain assumptions. The key advantage of using the surrogate loss function $\tilde{J}_{\theta_i}(\theta_i)$ is that each agent i only needs to know partial information, which greatly reduces both communication and computational costs. This enhances the scalability and practicality of the framework, allowing it to handle large-scale tasks where the curse of dimensionality is severe or global information is unavailable, without incurring large update errors when k is properly selected.

Remark 6.2. *The constants $E_{i,1}$ and $E_{i,2}$ in Theorem 6.3 are directly dependent on several factors related to the MARL setting and the neural networks used. To efficiently reduce the upper bound provided in Theorem 6.3 such that the error of the surrogate policy gradient estimation is further mitigated, several practical measures can be adopted. Examples include: (1) Bounding the norms of reward, barrier, and cost signals through careful design, or standardizing (normalizing) them via their means and variances before using them for training. (2) Constraining the range of α_i if a reasonable bound for the investigated task is known in advance before training. (3). Applying techniques such as gradient clipping to limit the update magnitude of the networks, which prevents exploding gradients and implicitly controls Lipschitz constants.*

Remark 6.3. *Compared with MAS^{3}AC, the scalability contribution of this chapter primarily lies in the additional theoretical analysis rather than a fundamentally different algorithmic architecture. Specifically, Theorem 6.3 establishes an upper bound on the error between the true policy gradient and the surrogate policy gradient used under decentralized training and decentralized execution with local observations. This result provides a theoretical justification for replacing global information with partial local information, thereby reducing communication and computational requirements in large-scale systems. Furthermore, the resulting error bound and Remark 6.2 provide practical guidance regarding normalization, and gradient clipping to mitigate approximation errors in decentralized implementations. Consequently, the principal scalability contribution of this chapter is the theoretical support and practical guidance for scalable decentralized learning under limited information.*

6.2.3 Accelerating the Training with NODE-based Design

In addition to the scalability studied in the previous subsection, sample efficiency is also crucial for many tasks, especially in real-world scenarios. However, the application of the multi-timescale approach, detailed in subsection 6.2.1 and the corresponding assumption 6.4 in Appendix 6.5.2, implies a substantial demand for training steps and collected samples. This is because, theoretically, each agent requires infinite samples to train its individual policy, as suggested by the asymptotic convergence analysis (see the details and proof in Appendix 6.5.3). Such a demand is clearly undesired in many applications. Techniques like parameter sharing [150] can be applied to reduce the number of networks to train, but this approach is typically unsuitable for heterogeneous agents. Thus, it will be ideal if there is a method to improve the sample efficiency of each agent i's training, thereby enhancing the overall sample efficiency of the whole multi-agent system's training.

For each agent i, as shown in previous subsections and common actor-critic methods, its policy training is about minimizing a loss function constructed by the output of its policy network and critic functions via gradient descent. Thus, the policy network needs to be optimized to efficiently minimize the given loss function. In our setting, at each training step, the policy should aim to generate outputs that minimize the loss based on the Lagrangian function, which provides a meaningful and differentiable measure of how the generated output should be adjusted to achieve better reward maximization and fewer constraint violations at the present iteration. Hence, designing a specialized policy network architecture that directly facilitates the fast and efficient minimization of such Lagrangian loss, through its forward-generated outputs, will be helpful to accelerate the policy training and improve the sample efficiency by requiring fewer samples to achieve comparable performance.

We thus formulate this as an optimization problem, treating the policy output as the decision variable and seeking to refine it efficiently to reduce the associated Lagrangian loss. To this end, we introduce a heavy-ball–inspired Neural ODE that evolves the policy output over an artificial time dimension before execution. The heavy-ball method is an optimization technique that augments gradient descent with a momentum term, which smooths oscillations and accelerates convergence along flat, low-curvature directions. Momentum-based optimization

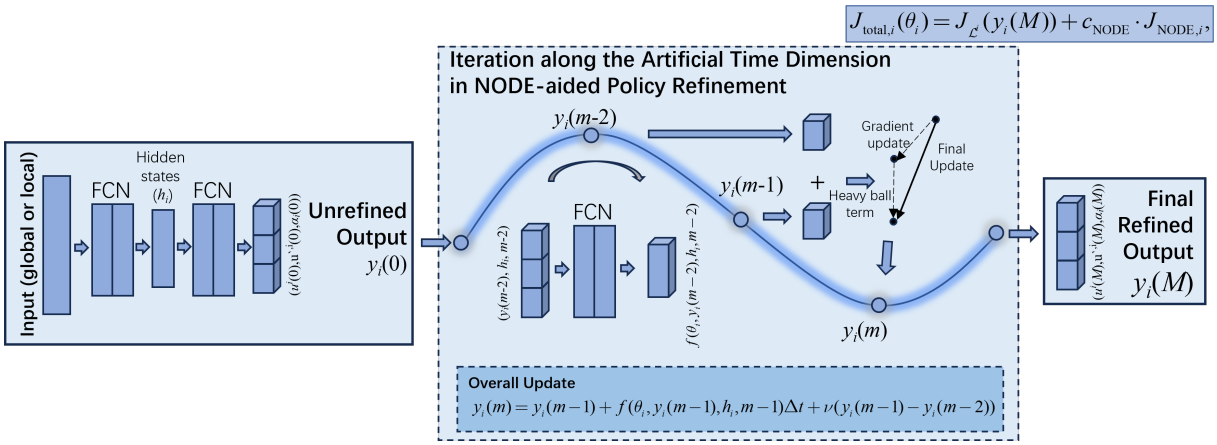


Figure 6.1: Framework for NODE-aided policy refinement. The diagram illustrates the process of refining an initial "Unrefined Output" ($y_i(0)$) to a "Final Refined Output" ($y_i(M)$). The core mechanism involves an iterative update along an artificial time dimension using a Neural ODE (NODE) approach, which incorporates a heavy ball momentum term. "FCN" here means fully connected networks.

has been widely adopted in machine learning (e.g., in Adam) and is known to improve empirical convergence. It is noteworthy that, in our case, the momentum mechanism is not used to update the parameter θ_i of the policy π^i, but is instead embedded in the refinement dynamics of the output, yielding a structurally novel policy architecture that enables constraint-aware corrections within each forward pass.

For agent i, assume that global information is available [1], we denote the output of a standard MLP-based block that is commonly used as the policy network in prior RL studies as $y_i(0) = (u^i(0), u'^{,i}(0), \alpha_i(0))$. Here, $u^i(0)$ is the action generated by the block with the input being $\boldsymbol{x}_t$, and $u'^{,i}(0)$ is the action when $\boldsymbol{x}_{t+1}$ is the input. $\alpha_i(0)$ is the stability parameter corresponding to the input $\boldsymbol{x}_t$. $\boldsymbol{x}_t$ and $\boldsymbol{x}_{t+1}$ are transitions sampled from the replay buffer during the implementation. Starting from the initial value $y_i(0)$, we define a sequence of refined actions-α_i tuples $y_i(m) = (u^i(m), u'^{,i}(m), \alpha_i(m))$, where $m \in \{1, \ldots, M\}$ denotes the number of refinements that the tuple has undergone along the artificial time dimension before producing the final real output at step M. Then, in practice, given that all critic networks are parameterized by corresponding parameters (e.g., ϕ_i, ω_i, and χ_i), we denote the loss function

[1]For cases where only partial observability is possible, in implementation, the inputs of corresponding critic and policy networks can simply be changed from global information, as used in the analysis here, to partial observations.

$J_{\mathcal{L}^i}(y_i(M))$ as follows:

$$
\begin{aligned}
J_{\mathcal{L}^i}(y_i(M)) = \mathbb{E}_{\mathcal{D}}\big[& -Q^i(\boldsymbol{x}_t, \hat{\boldsymbol{u}}_t; \phi_i) + \lambda_{i,1} \sum_{t=0}^{\infty} \gamma^t \text{max}\{0, -H^i(\boldsymbol{x}_{t+1}, \hat{\boldsymbol{u}}_{t+1}; \omega_i) + (1-\epsilon) H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i)\} \\
& + \frac{\lambda_{i,1}}{2} \sum_{t=0}^{\infty} \gamma^t [\text{max}\{0, -H^i(\boldsymbol{x}_{t+1}, \hat{\boldsymbol{u}}_{t+1}; \omega_i) + (1-\epsilon) H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i)\}]^2 + \lambda_{i,2} \sum_{t=0}^{\infty} \gamma^t \text{max}\{0, \\
& (1+\beta_i)(\sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)) - \alpha_i(M)(\boldsymbol{x}_t) V^i(x_t^i, u_t^i; \chi_i)\} + \frac{\lambda_{i,2}}{2} \sum_{t=0}^{\infty} \gamma^t [\text{max}\{0, (1+\beta_i)(\sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)) \\
& - \alpha_i(M)(\boldsymbol{x}_t) V^i(x_t^i, u_t^i; \chi_i)\}]^2 + \lambda_{i,3} \sum_{t=0}^{\infty} \gamma^t \text{max}\{0, V^i(x_{t+1}^i, u'^{,i}(M); \chi_i) - (1 - \alpha_i(M)(\boldsymbol{x}_t)) V^i(x_t^i, u_t^i; \chi_i) \\
& - \sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)\} + \frac{\lambda_{i,3}}{2} \sum_{t=0}^{\infty} \gamma^t [\text{max}\{0, V^i(x_{t+1}^i, u'^{,i}(M); \chi_i) - (1 - \alpha_i(M)(\boldsymbol{x}_t)) V^i(x_t^i, u_t^i; \chi_i) \\
& - \sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)\}]^2\big].
\end{aligned}
\tag{6.10}
$$

Here, after M steps of refinements, $y_i(M) = (u^i(M), u'^{,i}(M), \alpha_i(M))$ is policy π^i's final real output used for execution. It includes the action $u^i(M)$ used for control at time t corresponding to the input $\boldsymbol{x}_t$, $u'^{,i}(M)$ for time $t+1$ corresponding to $\boldsymbol{x}_{t+1}$, and stability parameter $\alpha_i(M)$ conditioned on $\boldsymbol{x}_t$. The loss function $J_{\mathcal{L}^i}(y_i(M))$ is defined based on the Lagrangian function (6.3). Specifically, with slight abuse of notation, in (6.10), $\hat{\boldsymbol{u}}_t = (u^i(M), \boldsymbol{u}^{-i}(M))$ denotes the joint action at time t, where $u^i(M)$ retains gradient information for updating π^i, while $\boldsymbol{u}^{-i}(M)$ being the actions of all other agents generated by their policies also after refinements, but without gradient information since the joint policy of other agents, namely $\boldsymbol{\pi}^{-i}$, is not updated when agent i is trained. Similarly, $\hat{\boldsymbol{u}}_{t+1} = (u'^{,i}(M), \boldsymbol{u}'^{,-i}(M))$ represents the joint action at $t+1$ under the same setting. All other variables, including $\boldsymbol{x}_t$, $\boldsymbol{u}_t$, $\boldsymbol{x}_{t+1}$, and their individual components, are all sampled from the replay buffer $\mathcal{D}$.

To efficiently minimize the Lagrangian-based loss function $J_{\mathcal{L}^i}(y_i(M))$, we draw inspiration from the heavy-ball method. Here, we interpret the sequence $y_i(m)$, $m \in \{1, \ldots, M\}$ as a trajectory of decision variables that evolves over the artificial time dimension and terminates at

$y_i(M)$. Its discrete-time update is given as follows:

$$y_i(m) = y_i(m-1) - \nabla_{y_i} J_{\mathcal{L}^i}(y_i(m-1))\Delta t + \nu(y_i(m-1) - y_i(m-2)), \tag{6.11}$$

where $\Delta t > 0$ denotes the step size along the artificial time dimension, and ν is the coefficient of the heavy-ball term $y_i(m-1) - y_i(m-2)$. For the first refinement step ($m = 1$), the momentum term is set to zero. Theoretical results [151, 152] have shown that heavy-ball dynamics can converge even in nonconvex settings.

To efficiently represent $\nabla_{y_i} J_{\mathcal{L}^i}(y_i(m-1))$, here we parameterize the negative gradient $-\nabla_{y_i} J_{\mathcal{L}^i}(y_i(m-1))$ with a fully connected layer $f(\theta_i, y_i(m-1), h_i, m-1)$, and the resulting update is:

$$y_i(m) = y_i(m-1) + f(\theta_i, y_i(m-1), h_i, m-1)\Delta t + \nu(y_i(m-1) - y_i(m-2)). \tag{6.12}$$

This formulation is motivated by Neural ODEs, which model the evolution of a system's state using an ordinary differential equation whose dynamics is governed by a neural network. Neural ODEs have been successfully applied to various supervised learning problems due to their ability to capture continuous-time dynamics. Here, the same principle is leveraged to refine the policy output iteratively within a single forward pass via f. We continue to use θ_i as the parameterization of f since this design of the refinement process is an integral component of agent i's policy network. Furthermore, at each refinement step, the hidden state h_i is reused as an external control variable to effectively embed the environmental information (i.e., $\boldsymbol{x}_t$ and $\boldsymbol{x}_{t+1}$). This design enables the update of $y_i(m)$ to be guided by relevant environmental features, since h_i is the extraction of the corresponding information used as input to the policy. Consequently, the final output $y_i(M)$ can more effectively minimize the Lagrangian loss constructed from these inputs. In addition to the Euler method shown in (6.12), other numerical solvers such as the Runge–Kutta (RK4) method can also be employed.

To make $f(\theta_i, y_i(m-1), h_i, m-1)$ effectively approximate $-\nabla_{y_i} J_{\mathcal{L}^i}(y_i(m-1))$ given the extracted input representation h_i, additional terms can be incorporated into the total loss to encourage the progressive minimization of $J_{\mathcal{L}^i}$ at each step of the refinement process. One

example is given as follows:

$$J_{\text{NODE},i} = \sum_{m=1}^{M} \max\{0, J_{\mathcal{L}^i}(y_i(m)) - J_{\mathcal{L}^i}(y_i(m-1))\}. \tag{6.13}$$

Here, $J_{\mathcal{L}^i}(y_i(m))$ has the same formulation as (6.10), but with different $\hat{\boldsymbol{u}}_t$, $\hat{\boldsymbol{u}}_{t+1}$, $u'^{,i}(m)$ and $\alpha_i(m)$ since now $y_i(m) = (u^i(m), u'^{,i}(m), \alpha_i(m))$ is the independent variable. This $J_{\text{NODE},i}$ enforces a monotonic decrease in the Lagrangian loss over the refinement steps $m \in \{1, \ldots, M\}$, thereby explicitly requiring the learned dynamics f to approximate $-\nabla_{y_i} J_{\mathcal{L}^i}$ and promote efficient loss reduction at each refinement step. Thus, the total loss function can be given as:

$$J_{\text{total},i}(\theta_i) = J_{\mathcal{L}^i}(y_i(M)) + c_{\text{NODE}} \cdot J_{\text{NODE},i}, \tag{6.14}$$

where c_{NODE} is a constant coefficient. Other designs of $J_{\text{NODE},i}$ can also be adopted to obtain the total loss $J_{\text{total},i}$. This $J_{\text{total},i}$ can further be used to update the Lagrangian multiplier λ_i (namely, $J_{\text{total},i}(\theta_i) = J_{\text{total},i}(\lambda_i) = J_{\mathcal{L}^i}(y_i(M)) + c_{\text{NODE}} \cdot J_{\text{NODE},i}$) since λ_i is also incorporated within this loss function [2]. Overall, the generation of the final real output $y_i(M)$ can be interpreted as a learned dynamic process, whose trajectory iteratively evolves toward minimizing the Lagrangian loss. Thus, beyond standard parameter update for policy π^i, this specifically designed NODE-based refinement mechanism enables the policy to generate outputs that effectively minimize loss functions to achieve higher rewards and fewer constraints violations with the same amount of data, leading to a notable improvement in sample efficiency for each agent i's training, and consequently, for the training of the whole MAS.

The network architecture of the policy incorporating the NODE-based refinement is illustrated in Figure 6.1, and with its experimental evaluation is provided in Section 6.3.

Remark 6.4. *Although the proposed refinement mechanism is developed in the context of multi-agent reinforcement learning with safety and stability constraints, the underlying design is not restricted to constrained settings. The NODE-based refinement operates by efficiently minimizing a task-specific loss with respect to the policy output, and therefore the same architecture can be*

[2] In practice, it is found that using $J_{\text{total},i}(\lambda_i) = J_{\mathcal{L}^i}(y_i(M))$ already works well and achieves better performance compared to baselines.

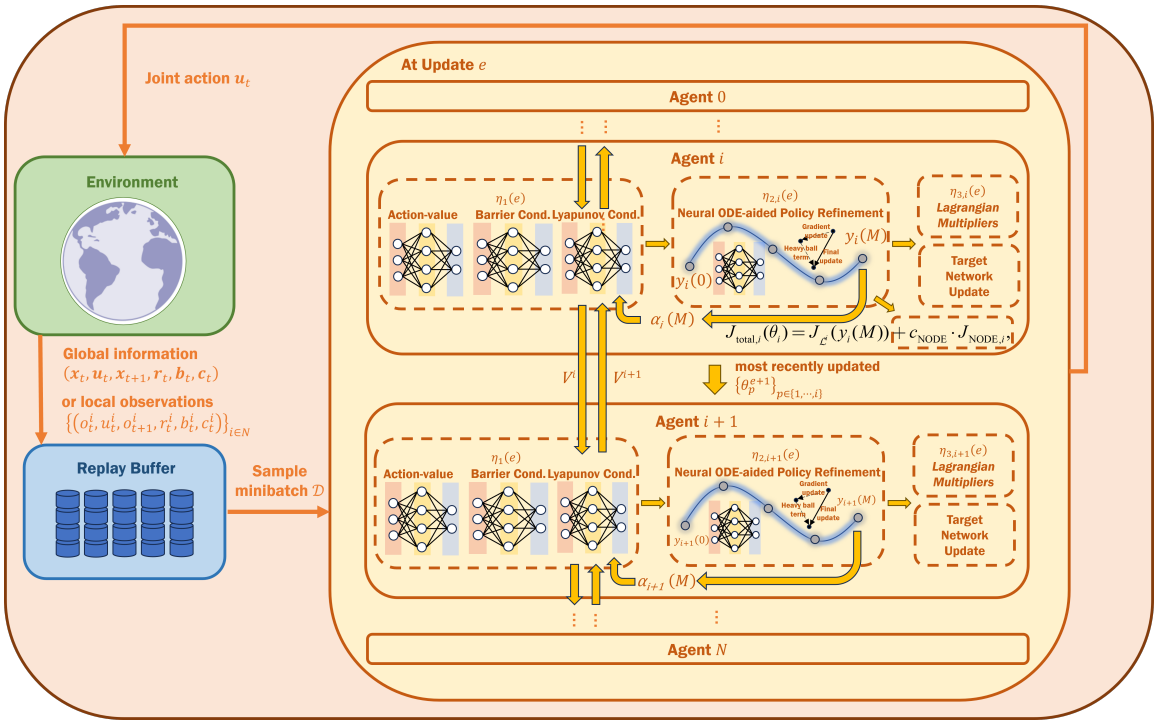


Figure 6.2: Overview of the **M**ulti-**A**gent **S**afe **S**table **S**calable and **S**ample-efficient **N**eural ordinary differential equation (NODE)–based **A**ctor-**C**ritic (**MAS^{4}NAC**) framework. Agents are updated sequentially, as detailed in Algorithm 6.2.

applied to unconstrained (standard) RL formulations simply by replacing the Lagrangian loss with an appropriate objective (e.g., negative action-value). In this sense, the proposed heavy-ball–driven refinement should be viewed as a general action-level optimization layer, which can be instantiated with different loss functions depending on the investigated RL problem, rather than as a mechanism tied exclusively to constraint handling.

6.2.4 Practical Implementation

Based on all aforementioned subsections, we propose the complete framework termed **M**ulti-**A**gent **S**afe **S**table **S**calable and **S**ample-efficient **N**eural ordinary differential equation (NODE)–based **A**ctor-**C**ritic (**MAS^{4}NAC**), shown as Algorithm 6.2 and illustrated in Figure 6.2. This algorithm essentially requires that, within each update e, the most recently updated $\{\theta_p^{e+1}\}_{p\in\{1,\ldots,i-1\}}$, instead of $\{\theta_p^{e}\}_{p\in\{1,\ldots,i-1\}}$, are used when updating θ_i for agent i.

6.3 Experiments

In this section, we present extensive empirical evaluations of MAS^{4}NAC to validate its effectiveness across a diverse range of MARL tasks. The experiments are designed to address:

Algorithm 6.2 **M**ulti-**A**gent **S**afe **S**table **S**calable and **S**ample-efficient **N**eural ordinary differential equation (NODE)–based **A**ctor-**C**ritic (**MAS^{4}NAC**)

Input: $\eta_1(e)$ for updating the action-value networks, barrier, and Lyapunov certificates; $\eta_{2,i}(e)$ for updating the individual control policy π^i; and $\eta_{3,i}(e)$ for updating Lagrangian multipliers for agent i. Initialized parameters $\{\phi_i^0\}_{i\in\mathcal{N}}$, $\{\omega_i^0\}_{i\in\mathcal{N}}$, $\{\chi_i^0\}_{i\in\mathcal{N}}$, $\{\theta_i^0\}_{i\in\mathcal{N}}$, $\{\lambda_i^0\}_{i\in\mathcal{N}}$ for corresponding neural networks; maximum iterations E; policy update delay step n_p; Polyak coefficient τ, and target neural networks $\{\phi_{i,targ}^0\}_{i\in\mathcal{N}} \leftarrow \{\phi_i^0\}_{i\in\mathcal{N}}$, $\{\omega_{i,targ}^0\}_{i\in\mathcal{N}} \leftarrow \{\omega_i^0\}_{i\in\mathcal{N}}$, $\{\chi_{i,targ}^0\}_{i\in\mathcal{N}} \leftarrow \{\chi_i^0\}_{i\in\mathcal{N}}$. $\eta_1(e) > \eta_{2,1}(e), \eta_{2,i}(e) > \eta_{3,i}(e), \forall i \in \mathcal{N}$, and $\eta_{3,i-1}(e) > \eta_{2,i}(e), \forall i \in \{2,\ldots,N\}$, heavy-ball coefficient ν, NODE-based refinement steps M, constant coefficient c_{NODE}, $\{\beta_i\}_{i\in\mathcal{N}}$ and $\delta_{i,j}$ used for graph definition, radius k for state and action communication.

[Optional] Practical measures mentioned in Remark 6.2 can be adopted in the following process to further mitigate the error of the surrogate policy gradient estimation when only local observations are available.

for $e = 0, 1, \ldots, E$ **do**

  Collect transitions $(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{x}_{t+1}, \boldsymbol{r}_t, \boldsymbol{b}_t, \boldsymbol{c}_t)$ when global information is available, or $\{(o_t^i, u_t^i, o_{t+1}^i, r_t^i, b_t^i, c_t^i)\}_{i\in\mathcal{N}}$ when only local observations are available, by using the joint policy $\boldsymbol{\pi}$ parameterized by $\{\theta_i^e\}_{i\in\mathcal{N}}$.

  Store transitions into the replay buffer.

  Sample a random minibatch $\mathcal{D}$ of transitions from the replay buffer.

  for $i = 1, 2, \ldots, N$ **do**

    Update agent i's action-value network, barrier certificate, and Lyapunov certificate with the learning rate $\eta_1(e)$ and loss functions detailed in Section 6.5.1:
    $\phi_i^{e+1} \leftarrow \phi_i^e - \eta_1(e)\nabla_{\phi_i} J_{Q^i}(Q^i)$;
    $\omega_i^{e+1} \leftarrow \omega_i^e - \eta_1(e)\nabla_{\omega_i} J_{H^i}(H^i)$;
    $\chi_i^{e+1} \leftarrow \chi_i^e - \eta_1(e)\nabla_{\chi_i} J_{V^i}(V^i)$.

    if $e \bmod n_p = 0$ **then**

      Due to the NODE-based policy refinement design, with V^j, $j \neq i$, from other agents, obtain the loss function $J_{\text{total},i}$ (6.14).

      Update agent i's control policy with the learning rate $\eta_{2,i}(e)$ and the loss function $J_{\text{total},i}$:
      $\theta_i^{e+1} \leftarrow \theta_i^e - \eta_{2,i}(e)\nabla_{\theta_i} J_{\text{total},i}(\theta_i)$.

      Update agent i's Lagrangian multipliers with the learning rate $\eta_{3,i}(e)$:
      $\lambda_{i,1}^{e+1} \leftarrow \left[\lambda_{i,1}^e + \eta_{3,i}(e)\nabla_{\lambda_{i,1}} J_{\text{total},i}(\lambda_i)\right]_+$;
      $\lambda_{i,2}^{e+1} \leftarrow \left[\lambda_{i,2}^e + \eta_{3,i}(e)\nabla_{\lambda_{i,2}} J_{\text{total},i}(\lambda_i)\right]_+$;
      $\lambda_{i,3}^{e+1} \leftarrow \left[\lambda_{i,3}^e + \eta_{3,i}(e)\nabla_{\lambda_{i,3}} J_{\text{total},i}(\lambda_i)\right]_+$

      Update agent i's target networks:
      $\phi_{i,targ}^{e+1} \leftarrow (1-\tau)\phi_{i,targ}^e + \tau\phi_i^{e+1}$;
      $\omega_{i,targ}^{e+1} \leftarrow (1-\tau)\omega_{i,targ}^e + \tau\omega_i^{e+1}$;
      $\chi_{i,targ}^{e+1} \leftarrow (1-\tau)\chi_{i,targ}^e + \tau\chi_i^{e+1}$.

    end if

  end for

  Apply linear decay to all learning rates.

end for

Output: $\{\theta_i^E\}_{i\in\mathcal{N}}$

1. Q1: In cooperative MARL tasks with global information, where high rewards are achieved when the MAS converges to the desired equilibrium (e.g., navigation), can MAS^{4}NAC outperform baselines in reward maximization and safety constraint satisfaction?
2. Q2: In more challenging settings where agents operate under limited observability and potentially conflicting objectives, can MAS^{4}NAC consistently achieve high rewards and low safety violations compared to baselines, for each individual agent?
3. Q3: Does the incorporation of the NODE–based policy refinement effectively improve sample efficiency and control performance relative to the standard policy structure?
4. Q4: How does the heavy-ball optimization mechanism influence the performance of the learned control policy? Furthermore, how does the size of an agent' s local observation affect overall performance?

6.3.1 Baselines

We compare MAS^{4}NAC against both safe MARL algorithms and standard model-free MARL baselines. Details can be found in Section 5.6.6.

6.3.2 Environment Settings

To rigorously evaluate the proposed MAS^{4}NAC framework, we construct a comprehensive suite of safe and stable MARL benchmarks encompassing both cooperative and decentralized non-cooperative settings. These environments are designed to capture a wide range of safety–stability interactions and information-sharing regimes. Full details are provided in Appendix 5.6.7.

Remark 6.5. *The benchmarks considered in this chapter employ two different information structures. The cooperative benchmarks, including Cooperative Ant, Cooperative HalfCheetah, and Cooperative Coupled HalfCheetah, use global information and are primarily intended to evaluate safety, stability, and learning performance. By contrast, the Non-cooperative Ant and Non-cooperative Coupled HalfCheetah benchmarks use local observations only and therefore serve as the primary testbeds for evaluating decentralized training and decentralized execution under limited information. All compared methods use the observation structure provided by the corresponding environment.*

Remark 6.6. *Compared with the global-information settings considered previously, the DTDE paradigm introduces several practical challenges. Since different agents may observe different numbers of neighbouring agents, zero padding is used to align observation dimensions and enable a unified neural-network architecture, at the cost of increased input dimension and computational burden. In addition, the neighbourhood size used to construct local observations affects the trade-off between information availability and computational efficiency, while the inter-agent gain functions $\delta_{ij} \in \mathcal{K}$ require careful design to avoid numerical instability during training. Experimental results indicate that neighbourhood size is one of the most influential hyperparameters in practice. This suggests that neighbouring information is important for enforcing inter-agent safety constraints, although relatively small neighbourhoods are already sufficient to achieve strong performance. Overall, the proposed framework exhibits moderate sensitivity to the neighbourhood size and provides a practical balance between computational efficiency and learning performance.*

6.3.3 Experimental Setup

We employ two metrics to evaluate the performance of MAS^{4}NAC and all baseline algorithms in experiments:

1. **Average Episode Reward**: This measures the task performance of the learned policies. Since the reward functions in all tasks are coupled with convergence toward the desired equilibrium, this metric also reflects stability: a stable controller drives the system efficiently toward the equilibrium and thus leads to faster improvement in the reward curve. Moreover, a stable controller helps prevent large deviations that could cause severe performance degradation, as reflected in the learning curves, during training.

2. **Average Episode Safety Cost**: This quantifies the cumulative safety constraint violations, thereby reflecting the safety-preserving capability of the learned policies.

Notably, in addition to treating the stability parameter α_i as part of the output of policy π^i (i.e., the full version, where α_i is trainable via the loss function incorporating terms based on constraints (6.2c) and (6.2d)), we can still apply the new algorithm with NODE-based refinement mechanism to the original version introduced in Chapter 5 where α_i is treated as a fixed and predefined value. The latter setting may require less computational effort per

update, as its loss function contains fewer terms. Here, we implement both variants: the newly proposed full version in which α_i is trainable, and the original version from Chapter 5 where α_i is fixed and predefined, and combine each with the NODE-based refinement design introduced in this chapter. The corresponding results are provided in the following subsection.

6.3.4 Results

Main result: To answer **Q1**, we first evaluate MAS^{4}NAC on the three cooperative benchmarks where agents operate under centralized training with full global observability. For the Ant and HalfCheetah tasks, we employ the simplified variant with predefined stability parameters α_i, while for Coupled HalfCheetah, the full version with adaptive α_i is used. As illustrated in Figure 6.3, MAS^{4}NAC consistently achieves the most favorable balance between reward maximization and safety preservation across all environments. Since in each task rewards are tightly coupled with convergence toward and maintaining the desired equilibrium, the strong reward performance of the proposed MAS^{4}NAC, supported by efficient and steady improvement in its average episode reward curves, indicates its ability to help guarantee stability for the systems to converge toward the desired equilibrium with fewer safety violations compared to many baselines. Specifically, in both cooperative Ant and HalfCheetah tasks, MAS^{4}NAC attains the highest rewards and the fewest safety violations, and in the cooperative Coupled HalfCheetah environment, although standard unconstrained MARL methods, such as HASAC and HAPPO, achieve competitive reward levels, they fail to respect safety, showing large cumulative safety costs during the whole training. Conversely, constrained MARL baselines (e.g., MAFOCOPS, MACPO) prioritize constraint satisfaction but suffer from pronounced reward degradation due to their conservative updates. MAS^{4}NAC, by contrast, achieves low safety costs while maintaining high rewards, demonstrating its ability to regulate stability and safety through either predefined or adaptive stability constraints when accompanied by safety conditions. These results confirm that MAS^{4}NAC can effectively coordinate agents in cooperative settings to achieve safe and stable control, and further quantitative comparisons are provided in Table 6.1.

To answer **Q2**, we first evaluate MAS^{4}NAC on the decentralized non-cooperative Coupled HalfCheetah task. Here, the full version with adaptive stability parameters α_i is used, and

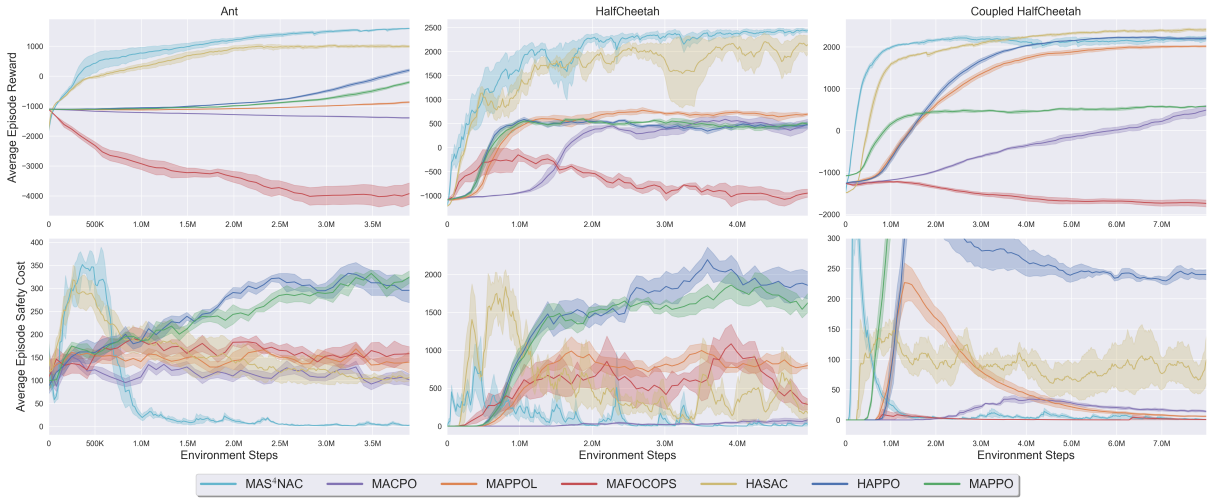


Figure 6.3: Average episode reward and average episode safety cost in the three cooperative tasks, comparing MAS^{4}NAC with baselines. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

Table 6.1: Average episode reward and average episode safety cost in the three cooperative tasks, comparing MAS^{4}NAC with baselines. Each data illustrates the average across 5 experiments employing different random seeds.

	Ant		HalfCheetah		Coupled HalfCheetah	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
HAPPO	41.29 ± 215.89	311.96 ± 67.96	450.48 ± 193.02	1864.28 ± 499.46	2195.35 ± 115.24	248.04 ± 74.57
HASAC	993.56 ± 87.49	107.49 ± 22.64	2160.65 ± 399.88	179.32 ± 283.95	$\mathbf{2409.76 \pm 102.30}$	108.58 ± 119.22
MACPO	-1381.66 ± 37.86	105.24 ± 50.66	450.28 ± 303.31	70.10 ± 88.80	600.08 ± 363.06	13.37 ± 11.21
MAFOCOPS	-3965.01 ± 720.78	156.47 ± 77.84	-996.92 ± 332.76	401.17 ± 420.83	-1740.68 ± 186.66	$\mathbf{0.00 \pm 0.00}$
MAPPO	-354.58 ± 203.39	322.31 ± 64.85	459.99 ± 154.13	1678.18 ± 452.72	574.57 ± 175.37	1579.87 ± 415.07
MAPPOL	-903.98 ± 67.67	142.52 ± 58.00	673.95 ± 154.83	811.79 ± 277.64	2020.18 ± 66.65	5.20 ± 4.26
MAS^{4}NAC	$\mathbf{1589.60 \pm 26.65}$	$\mathbf{3.29 \pm 4.35}$	$\mathbf{2442.66 \pm 101.77}$	$\mathbf{14.80 \pm 59.21}$	2213.30 ± 199.12	0.05 ± 0.34

each agent has access only to local observations. The observation size is treated as a tunable hyperparameter (see the ablation study and Appendix 6.5.5 (ii) for details). Each agent is trained with its own reward, safety, and stability cost signals, and rewards remain closely tied to the multi-agent system's ability to converge to the desired equilibrium. In this setting, agents must make decisions independently based on their own objectives, which increases the complexity due to partial observability and potential conflicts among individual goals. As shown in Figure 6.4 and Table 6.2, despite the aforementioned challenges, MAS^{4}NAC consistently achieves high final rewards and low safety violations for *each* agent, outperforming the baselines. To be more specific, while some baselines exhibit strong performance for certain agents in either reward or safety cost (e.g., HASAC performs well in reward, and MACPO excels in safety preservation), they fail to perform better in both aspects (reward and safety preservation) consistently across all agents compared to the proposed MAS^{4}NAC. Moreover, these methods

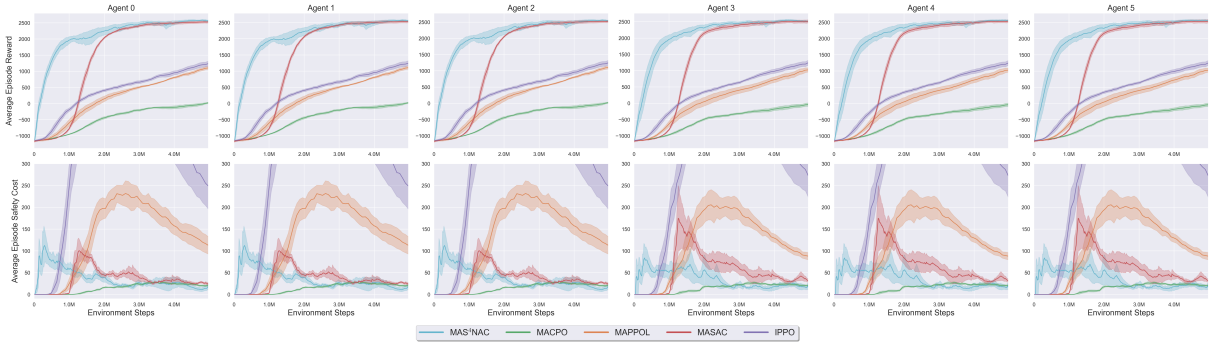


Figure 6.4: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC against baselines. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

often trade off stability for safety or vice versa: the average episode safety cost of HASAC is higher than those of MAS^{4}NAC and MACPO, while MACPO clearly underperforms in reward. In contrast, MAS^{4}NAC demonstrates a superior learning process, ensuring that every agent benefits from both high final rewards and strong constraint satisfaction. Furthermore, the proposed MAS^{4}NAC exhibits strong sample efficiency compared to all baselines, including HASAC. This is supported by the fact that the reward curves for MAS^{4}NAC rise rapidly across all agents, indicating that the algorithm begins effective exploration and learns from limited data early in training. Consequently, MAS^{4}NAC achieves a favorable balance between reward maximization and safety preservation with fewer environment interactions, highlighting its potential applicability to real-world scenarios.

The evaluation of MAS^{4}NAC on the decentralized non-cooperative Ant task is presented in Appendix 6.5.5 (i). These results further highlight the ability of MAS^{4}NAC to generalize beyond cooperative tasks and effectively handle decentralized, non-cooperative MARL settings.

Ablation studies: To answer **Q3**, we conduct ablation studies comparing the full MAS^{4}NAC framework with a simplified version that removes the NODE-based policy refinement module. The experiments are performed on two cooperative tasks, Ant and HalfCheetah (as shown in Figure 6.5 and Table 6.3), and on the decentralized non-cooperative Coupled HalfCheetah task (as shown in Figure 6.6 and Table 6.4) for completeness.

For the cooperative tasks, where global information is available and centralized training paradigm is applied (Figure 6.5), the inclusion of the NODE-based refinement design allows

Table 6.2: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC against baselines. Each data illustrates the average across 5 experiments employing different random seeds.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	902.29 ± 386.41	451.38 ± 305.37	904.99 ± 389.86	451.38 ± 305.37	905.91 ± 388.62	451.38 ± 305.37
MACPO	-140.28 ± 276.72	21.48 ± 25.26	-141.91 ± 276.68	21.48 ± 25.26	-142.75 ± 276.65	21.48 ± 25.26
MAPPOL	735.92 ± 416.97	174.96 ± 143.68	735.77 ± 417.75	174.96 ± 143.68	732.35 ± 417.00	174.96 ± 143.68
MASAC	2499.75 ± 78.45	25.25 ± 20.76	2503.90 ± 84.95	25.25 ± 20.76	2489.93 ± 81.61	25.25 ± 20.76
MAS^{4}NAC	**2570.14 ± 100.52**	**16.73 ± 23.47**	**2557.34 ± 102.21**	**16.73 ± 23.47**	**2545.48 ± 101.84**	**16.73 ± 23.47**
	Agent 3		**Agent 4**		**Agent 5**	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	898.48 ± 395.57	432.92 ± 273.22	900.26 ± 397.68	432.92 ± 273.22	903.87 ± 397.26	432.92 ± 273.22
MACPO	-215.51 ± 277.43	21.14 ± 27.75	-216.04 ± 277.56	21.14 ± 27.75	-215.42 ± 277.29	21.14 ± 27.75
MAPPOL	646.70 ± 464.02	150.76 ± 134.02	648.57 ± 465.17	150.76 ± 134.02	648.17 ± 466.71	150.76 ± 134.02
MASAC	2479.16 ± 102.04	35.51 ± 26.71	2502.30 ± 96.44	35.51 ± 26.71	2501.68 ± 108.95	35.51 ± 26.71
MAS^{4}NAC	**2533.35 ± 82.94**	**20.11 ± 27.52**	**2554.02 ± 87.70**	**20.11 ± 27.52**	**2563.70 ± 85.28**	**20.11 ± 27.52**

MAS^{4}NAC to achieve higher rewards with effective improvement in reward curves, while maintaining lower safety costs compared to its non-NODE counterpart. This demonstrates that the NODE-aided design enhances sample efficiency: with a relatively smaller number of environment interactions (e.g., 4M–5M steps instead of 10M), the proposed NODE-aided design achieves superior joint policy training in terms of both reward maximization and safety preservation. Such an advantage is particularly valuable for safety-critical real-world applications, where data collection is expensive and sample efficiency is crucial.

For the non-cooperative task, where agents operate under a decentralized training decentralized execution paradigm with partial observability and conflicting objectives (Figure 6.6), the performance gap becomes even more pronounced. With the NODE-based design, each agent' s reward increases rapidly and reaches substantially higher levels than without it. In contrast, the non-NODE variant exhibits much slower improvement and lower final performance, accompanied by more stagnant reward and safety cost curves in the early training phase, indicating inefficient exploration. This behavior may be undesirable in practical scenarios where interaction budgets are limited, as it reflects poor sample efficiency.

The underlying reason for this contrast may lie in the nature of decentralized learning. In cooperative settings, each agent is updated to maximize a joint objective, often yielding monotonic and steady improvement even without the NODE refinement. However, in decentralized settings, each agent optimizes its own objective, which may lead to conflicting updates and

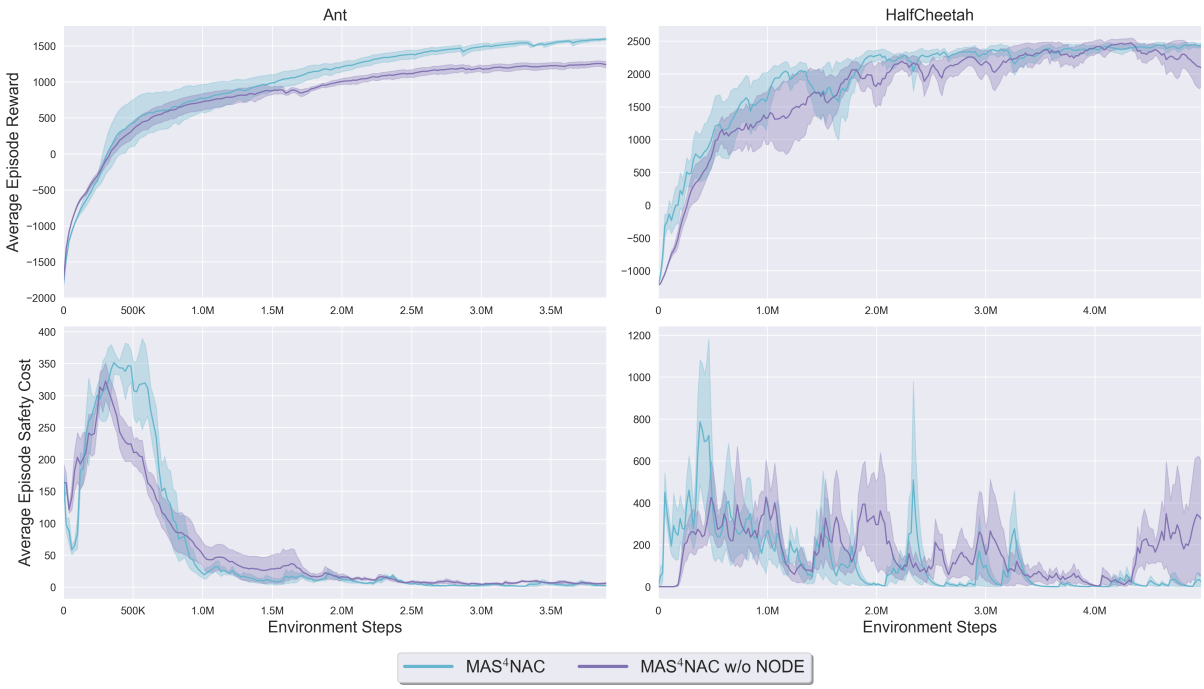


Figure 6.5: Average episode reward and average episode safety cost in the cooperative Ant and HalfCheetah tasks, comparing MAS^{4}NAC with its simplified variant without the NODE-based policy refinement design. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

non-monotonic learning dynamics. Consequently, efficient individual policy updates become especially critical for achieving stable convergence toward a good feasible local Nash equilibrium. The NODE-based refinement effectively addresses this challenge, thereby producing a more pronounced difference between the full and non-NODE versions than that observed in the cooperative tasks.

Table 6.3: Average episode reward and average episode safety cost in the cooperative Ant and HalfCheetah tasks, comparing MAS^{4}NAC with its simplified variant without the NODE-based policy refinement design. Each data illustrates the average across 5 experiments employing different random seeds.

	Ant		HalfCheetah	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**
MAS^{4}NAC w/o NODE	1247.18 ± 85.45	6.17 ± 5.21	2169.05 ± 574.07	273.14 ± 516.66
MAS^{4}NAC	$\mathbf{1589.60 \pm 26.65}$	$\mathbf{3.29 \pm 4.35}$	$\mathbf{2442.66 \pm 101.77}$	$\mathbf{14.80 \pm 59.21}$

To answer **Q4**, we first conduct an ablation study on the cooperative Coupled HalfCheetah benchmark to investigate how the heavy-ball optimization mechanism influences the performance of the learned control policy. As shown in Figure 6.7 and Table 6.5, varying the coefficient of the heavy-ball term within the NODE-based policy refinement significantly affects the overall performance. When the coefficient is small (e.g., 0.0 and 0.2), the momentum term

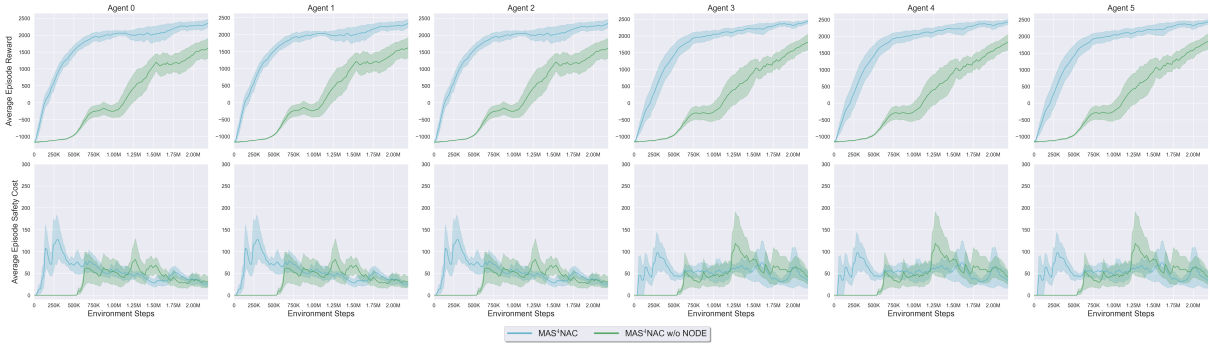


Figure 6.6: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC with its simplified variant without the NODE-based policy refinement design. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

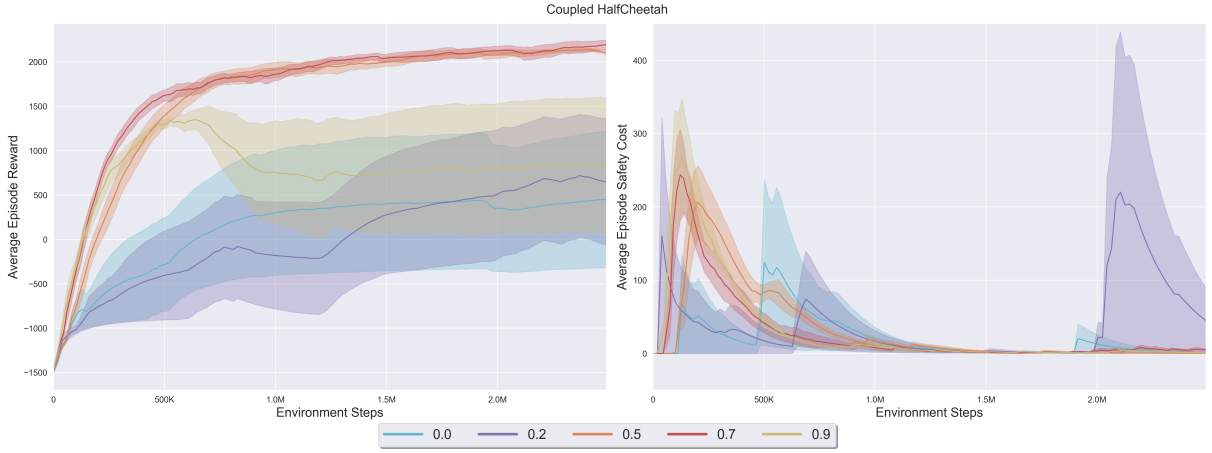


Figure 6.7: Average episode reward and average episode safety cost in the cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC with different coefficients (0.0, 0.2, 0.5, 0.7, and 0.9) for the heavy-ball method within the NODE-based policy refinement design. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

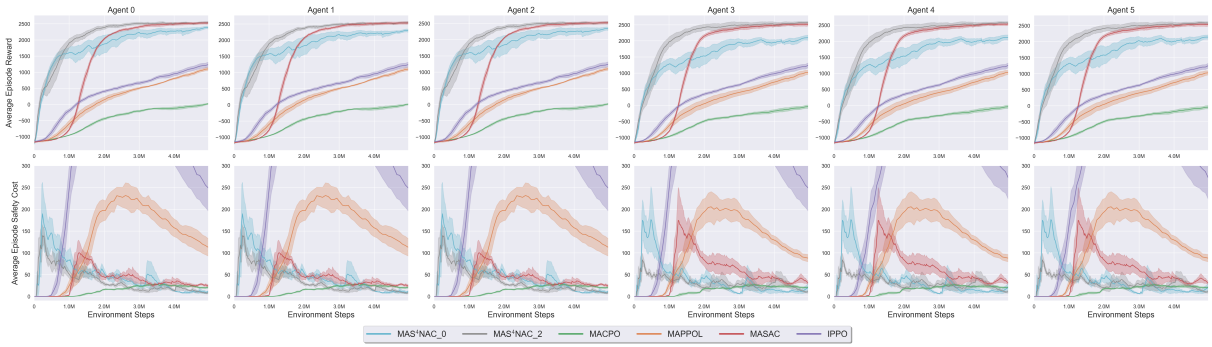


Figure 6.8: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC with different neighbor sizes against baselines. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

Table 6.4: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC with its simplified variant without the NODE-based policy refinement design. Each data illustrates the average across 5 experiments employing different random seeds.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
MAS^{4}NAC w/o NODE	1230.0 ± 783.88	43.53 ± 71.62	1245.87 ± 786.15	43.53 ± 71.62	1237.08 ± 788.10	43.53 ± 71.62
MAS^{4}NAC	$\mathbf{2177.34 \pm 434.58}$	$\mathbf{36.06 \pm 37.00}$	$\mathbf{2152.53 \pm 425.31}$	$\mathbf{36.06 \pm 37.00}$	$\mathbf{2169.14 \pm 431.79}$	$\mathbf{36.06 \pm 37.00}$
	Agent 3		**Agent 4**		**Agent 5**	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
MAS^{4}NAC w/o NODE	1267.11 ± 807.24	67.99 ± 124.14	1274.98 ± 809.75	67.99 ± 124.14	1298.50 ± 817.53	67.99 ± 124.14
MAS^{4}NAC	$\mathbf{2292.15 \pm 274.90}$	$\mathbf{50.65 \pm 79.98}$	$\mathbf{2304.65 \pm 299.23}$	$\mathbf{50.65 \pm 79.98}$	$\mathbf{2294.88 \pm 305.44}$	$\mathbf{50.65 \pm 79.98}$

Table 6.5: Average episode reward and average episode safety cost in the cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC with different coefficients (0.0, 0.2, 0.5, 0.7, and 0.9) for the heavy-ball method within the NODE-based policy refinement design. Each data illustrates the average across 5 experiments employing different random seeds.

	HalfCheetah	
Beta	**Reward**	**Cost**
0.0	408.24 ± 1510.80	4.59 ± 63.01
0.2	591.59 ± 1465.98	66.97 ± 464.55
0.5	2108.63 ± 158.76	$\mathbf{0.59 \pm 5.58}$
0.7	$\mathbf{2148.90 \pm 169.97}$	4.74 ± 14.77
0.9	821.54 ± 1503.96	1.60 ± 7.69

Table 6.6: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC with different neighbor sizes against baselines. Each data illustrates the average across 5 experiments employing different random seeds.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	902.29 ± 386.41	451.38 ± 305.37	904.99 ± 389.86	451.38 ± 305.37	905.91 ± 388.62	451.38 ± 305.37
MACPO	-140.28 ± 276.72	21.48 ± 25.26	-141.91 ± 276.68	21.48 ± 25.26	-142.75 ± 276.65	21.48 ± 25.26
MAPPOL	735.92 ± 416.97	174.96 ± 143.68	735.77 ± 417.75	174.96 ± 143.68	732.35 ± 417.00	174.96 ± 143.68
MASAC	2499.75 ± 78.45	25.25 ± 20.76	2503.90 ± 84.95	25.25 ± 20.76	2489.93 ± 81.61	25.25 ± 20.76
MAS^{4}NAC_0	2332.84 ± 220.28	12.86 ± 33.84	2248.56 ± 206.08	12.86 ± 33.84	2294.51 ± 236.89	12.86 ± 33.84
MAS^{4}NAC_2	$\mathbf{2526.48 \pm 101.59}$	$\mathbf{10.08 \pm 22.30}$	$\mathbf{2527.93 \pm 99.32}$	$\mathbf{10.08 \pm 22.30}$	$\mathbf{2516.60 \pm 99.68}$	$\mathbf{10.08 \pm 22.30}$
	Agent 3		**Agent 4**		**Agent 5**	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	898.48 ± 395.57	451.38 ± 305.37	900.26 ± 397.68	432.92 ± 273.22	903.87 ± 397.26	432.92 ± 273.22
MACPO	-215.51 ± 277.43	21.14 ± 27.75	-216.04 ± 277.56	21.14 ± 27.75	-215.42 ± 277.29	21.14 ± 27.75
MAPPOL	646.70 ± 464.02	150.76 ± 134.02	648.57 ± 465.17	150.76 ± 134.02	648.17 ± 466.71	150.76 ± 134.02
MASAC	2479.16 ± 102.04	35.51 ± 26.71	2502.30 ± 96.44	35.51 ± 26.71	2501.68 ± 108.95	35.51 ± 26.71
MAS^{4}NAC_0	2036.21 ± 270.92	21.07 ± 41.11	2063.64 ± 278.06	21.07 ± 41.11	2087.34 ± 275.09	21.07 ± 41.11
MAS^{4}NAC_2	$\mathbf{2550.25 \pm 121.21}$	$\mathbf{14.16 \pm 26.26}$	$\mathbf{2555.47 \pm 119.70}$	$\mathbf{14.16 \pm 26.26}$	$\mathbf{2571.82 \pm 119.56}$	$\mathbf{14.16 \pm 26.26}$

contributes minimally to the update dynamics, reducing the refinement to a near–first-order scheme. As a consequence, the policy updates lack sufficient guidance and acceleration, making it difficult to escape shallow local minima, which leads to poor exploration and slow reward improvement. Moreover, when the coefficient is 0.2, large final safety violations still occur in

the later stage of training, partially because the convergence is not efficient and stable enough.

Conversely, when the coefficient is too large (e.g., 0.9), the policy becomes overly driven by accumulated momentum. Although the high coefficient initially encourages exploration and causes the reward to increase rapidly at the beginning, the reward curve then deteriorates greatly without converging to a satisfactory final reward. This behavior reflects the intrinsic property of the heavy-ball method: excessive dependence on accumulated momentum can cause the trajectory to overshoot minima repeatedly, potentially leading to divergence from the optimal solution. In contrast, when the coefficient is moderate (e.g., 0.5 or 0.7), the momentum term provides sufficient historical information to smooth and accelerate policy updates, yet remains adequately damped to ensure steady convergence toward the optimal solution. This can result in the best overall trade-off between reward maximization and safety preservation, with strong sample efficiency, as reflected in Figure 6.7. Furthermore, the result demonstrates that the performance with respect to the coefficient is non-monotonic: neither excessively small nor large values are desirable. The optimal choice of the coefficient is task-dependent, and consequently, some hyperparameter tuning may be required in practice to achieve superior performance.

Then, we conduct an ablation study on the decentralized Coupled HalfCheetah benchmark to examine how the **neighbor size**, i.e., the radius of each agent's local detection of neighboring agents' states and actions, affects performance. The communication of Lyapunov values through the class-$\mathcal{K}$ gain function δ_{ij} remains identical across all cases, regardless of the used neighbor size [3]. Figure 6.8 and Table 6.6 compare MAS^{4}NAC with neighbor size 0 (detection of self-state and action only) and 2, against baselines that use neighbor size 3. Notably, MAS^{4}NAC with neighbor size 2 achieves reward levels comparable to MASAC (which purely maximizes reward with neighbor size 3), while maintaining lower safety costs, outperforming all baselines in balancing safety and reward. Additionally, with neighbor size 2, the performance in reward is clearly higher than that of the neighbor size 0 configuration. Nevertheless, even under this most restrictive setting (neighbor size 0), MAS^{4}NAC remains competitive with some baselines like

[3]The communication of Lyapunov values through class-$\mathcal{K}$ gain functions involves only scalar stability values without direct state-action information, thereby preserving the decentralized nature of policy learning, since agents cannot reconstruct others' state-action pairs from these exchanges, and only the state-action information within the detection area specified by the neighbor size is available for each agent's training.

MAPPOL and IPPO, demonstrating the robustness of the framework under severely limited information. While MASAC achieves relatively strong results compared to other baselines like IPPO, its sample efficiency remains inferior to MAS^{4}NAC, as evidenced by the slower rise in reward curves during early training. Additional results with neighbor sizes $(0, 1, 3)$ are provided in Appendix 6.5.5 (ii), further discussing the effect of neighbor sizes and confirming the consistent advantage of MAS^{4}NAC across varying levels of partial observability.

6.4 Conclusions

This work presented a unified framework for safe and stable multi-agent reinforcement learning (MARL), termed **MAS^{4}NAC**, which provides state-wise safety and stability guarantees under stated assumptions in general model-free multi-agent settings. The proposed approach integrates neural barrier certificates for safety with input-to-state stability (ISS) conditions for stability, forming a cohesive theoretical foundation for multi-agent control under unknown dynamics. By incorporating adaptive coefficients that can dynamically regulate the stability constraint, the framework achieves the satisfaction of both constraints with flexible safety–stability trade-offs across tasks. A complete theoretical analysis establishes the convergence of the joint policy update to a feasible local Nash equilibrium under both state-wise safety and stability conditions, ensuring principled learning.

In addition to the centralized paradigm, the framework introduces a scalable decentralized training decentralized execution formulation supported by an analytical upper bound on the surrogate policy gradient estimation error, offering a theoretical justification for off-policy RL under limited communication and partial observability. Furthermore, a NODE–aided policy refinement mechanism, equipped with the heavy-ball method, markedly improves sample efficiency and thus enables applicability to large and complex multi-agent systems. Extensive experiments on both cooperative and general-sum benchmarks demonstrate that **MAS^{4}NAC** consistently achieves a strong balance between reward maximization and constraint satisfaction, while ablation studies confirm that the NODE-based policy design considerably enhances sample efficiency compared to baselines.

6.5 Appendix

6.5.1 A. Loss Functions for Updates

In this part, we provide the explicit forms of loss functions used in the update. The loss functions to update the action-value function Q^i parameterized by ϕ_i, the barrier certificate H^i parameterized by ω_i, and the Lyapunov certificate V^i parameterized by χ_i for agent i are:

$$J_{Q^i}(Q^i) = \mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{u}_t, r_t^i, \boldsymbol{x}_{t+1}) \sim \mathcal{D}} \big[r_t^i + \gamma Q_{\phi_i, targ}(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) - Q_{\phi_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) \big]^2, \tag{6.15}$$

$$J_{H^i}(H^i) = \mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{u}_t, b_t^i, \boldsymbol{x}_{t+1}) \sim \mathcal{D}} \big[b_t^i + \gamma H_{\omega_i, targ}(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) - H_{\omega_i}(\boldsymbol{x}_t, \boldsymbol{u}_t) \big]^2, \tag{6.16}$$

$$J_{V^i}(V^i) = \mathbb{E}_{(x_t^i, u_t^i, c_t^i, x_{t+1}^i) \sim \mathcal{D}} \big[c_t^i + \gamma V_{\chi_i, targ}(x_{t+1}^i, \tilde{u}_{t+1}^i) - V_{\chi_i}(x_t^i, u_t^i) \big]^2, \tag{6.17}$$

where $\tilde{\boldsymbol{u}}_{t+1}$, which includes $\tilde{u}_{t+1}^i$, is generated by the current joint policy $\hat{\boldsymbol{\pi}}_i = (\boldsymbol{\pi}_{e+1}^{1:i-1}, \boldsymbol{\pi}_e^{i:N})$, and $\mathcal{D}$ denotes the replay buffer. V^i is implemented as an action-value network. Target networks $Q_{\phi_i, targ}$, $H_{\omega_i, targ}$ and $V_{\chi_i, targ}$ are used in practical training. Since all these three critic functions are represented by neural networks, the loss function to train the parameter θ_i for policy π^i, and its corresponding Lagrangian multiplier λ_i, derived from (6.3), is:

$$\begin{aligned}
J_{\theta_i}(\theta_i) = J_{\lambda_i}(\lambda_i) = \mathbb{E}_{\mathcal{D}} \big[& - Q^i(\boldsymbol{x}_t, \hat{\boldsymbol{u}}_t; \phi_i) + \lambda_{i,1} \sum_{t=0}^{\infty} \gamma^t \max\{0, -H^i(\boldsymbol{x}_{t+1}, \hat{\boldsymbol{u}}_{t+1}; \omega_i) + (1-\epsilon) H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i)\} \\
& + \frac{\lambda_{i,1}}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, -H^i(\boldsymbol{x}_{t+1}, \hat{\boldsymbol{u}}_{t+1}; \omega_i) + (1-\epsilon) H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i)\}]^2 + \lambda_{i,2} \sum_{t=0}^{\infty} \gamma^t \max\{0, \\
& (1+\beta_i)(\sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)) - \alpha_i(\boldsymbol{x}_t) V^i(x_t^i, u_t^i; \chi_i)\} + \frac{\lambda_{i,2}}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, (1+\beta_i)(\sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)) \\
& - \alpha_i(\boldsymbol{x}_t) V^i(x_t^i, u_t^i; \chi_i)\}]^2 + \lambda_{i,3} \sum_{t=0}^{\infty} \gamma^t \max\{0, V^i(x_{t+1}^i, \hat{u}_{t+1}^i; \chi_i) - (1 - \alpha_i(\boldsymbol{x}_t)) V^i(x_t^i, u_t^i; \chi_i) - \sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)\} \\
& + \frac{\lambda_{i,3}}{2} \sum_{t=0}^{\infty} \gamma^t [\max\{0, V^i(x_{t+1}^i, \hat{u}_{t+1}^i; \chi_i) - (1 - \alpha_i(\boldsymbol{x}_t)) V^i(x_t^i, u_t^i; \chi_i) - \sum_{j \neq i} \delta_{ij} V^j(x_t^j, u_t^j; \chi_j)\}]^2 \big].
\end{aligned} \tag{6.18}$$

Here $\hat{\boldsymbol{u}}_t$, $\hat{\boldsymbol{u}}_{t+1}$ and $\hat{u}_{t+1}^i$ are actions output by the policy network with gradient information for update θ_i. $\lambda_{i,1}$ and $\lambda_{i,2}$ are Lagrangian multipliers for safety and stability constraints, respectively. When global information is not available, all inputs are replaced by local observations.

6.5.2 B. Assumptions

Many assumptions used in this chapter are not the same as those utilized in Chapter 5, and therefore, for completeness, here we still summarize assumptions employed in the theoretical analysis for this chapter. Some notations are formally introduced later in the proof sections, but are still included here to provide a self-contained presentation.

Assumption 6.3. *There exists an upper bound T such that $\forall i \in \mathcal{N}$, all trajectories starting from $\boldsymbol{x}_0 \in \boldsymbol{\mathcal{X}}_{\text{irrec}}^{\boldsymbol{\pi},i}$ will result in $\Pr(b^i(\boldsymbol{x}_t) < 0) > 0$ at least once within T steps under the controller π^i when all other agents follow the fixed policy $\boldsymbol{\pi}^{-i}$.*

This assumes that states within the irrecoverable unsafe set $\boldsymbol{\mathcal{X}}_{irrec}^{\boldsymbol{\pi},i}$ lead to a safety violation with non-zero probability within a bounded number of steps. This is a mild assumption since it does not require deterministic violations, but only requires that safety violations occur with positive probability under the given policy.

Assumption 6.4. *Define $\eta_1(e)$ as the learning rate for updating the action-value functions and barrier certificates of all agents. For each agent $i \in \mathcal{N}$, let $\eta_{2,i}(e)$ and $\eta_{3,i}(e)$ denote the learning rates for updating the policy and the Lagrangian multiplier, respectively. The following conditions hold:*

1. $\sum_e \eta_1(e) = \sum_e \eta_{2,i}(e) = \sum_e \eta_{3,i}(e) = \infty, \quad \forall i \in \mathcal{N}$
2. $\sum_e \eta_1(e)^2 < \infty, \quad \sum_e \eta_{2,i}(e)^2 < \infty, \quad \sum_e \eta_{3,i}(e)^2 < \infty, \quad \forall i \in \mathcal{N}$
3. $\eta_{2,1}(e) = o(\eta_1(e)); \quad \eta_{3,i}(e) = o(\eta_{2,i}(e)), \quad \forall i \in \mathcal{N}; \ \eta_{2,i}(e) = o(\eta_{3,i-1}(e)), \ \forall i \in \{2, \ldots, N\}$.

This assumption adopts a multi-timescale learning framework widely used in prior constrained reinforcement learning studies. It ensures that the critic and barrier functions for all agent are updated on the fastest timescale, while for each agent, the policy is updated on a faster timescale than its associated Lagrangian multiplier; and the update of the Lagrangian multiplier of agent i is faster than the update of the policy of the following agent $i+1$. This hierarchy enables effective tracking of inter-agent constraints, as in our case, and addresses non-stationarity, thereby improving learning stability in multi-agent systems.

In implementation, it is possible to apply linearly decaying learning rates with different initial values to ensure $\eta_1(e) > \eta_{2,i}(e) > \eta_{3,i}(e), \forall e, \forall i \in \mathcal{N}$ and $\eta_{3,i-1}(e) > \eta_{2,i}(e), \forall e, \forall i \in \{2, \ldots, N\}$ to approximate the required relationships among the learning rates, which has been empirically observed to be sufficient to support stable and efficient training across all benchmarks compared to all baselines.

Theoretically, to satisfy Assumption 6.4, the following family of polynomially decaying learning-rate schedules can be adopted: let e denote the iteration index and N denote the number of agents. One valid learning-rate design is given by $\eta_1(e) = \frac{c_1}{\left(1+\frac{e}{K}\right)^{\beta}}$, $\eta_{2,i}(e) = \frac{c_{2,i}}{\left(1+\frac{e}{K}\right)^{\beta\left(1+\frac{2i-1}{2N+2}\right)}}$, $\eta_{3,i}(e) = \frac{c_{3,i}}{\left(1+\frac{e}{K}\right)^{\beta\left(1+\frac{2i}{2N+2}\right)}}$, where $i \in \mathcal{N}$, $c_1, c_{2,i}, c_{3,i}, K > 0$ are constants, and $\beta = \frac{1}{2} + \frac{1}{8N+8}$. Here we also provide the proof that this design meets all requirements in the assumption:

().1 Proof. Let

$$p_1 = \beta, \quad p_{2,i} = \beta\Big(1 + \frac{2i-1}{2N+2}\Big), \quad p_{3,i} = \beta\Big(1 + \frac{2i}{2N+2}\Big).$$

Since $(1+e/K)^{-p} = \Theta(e^{-p})$ as $e \to +\infty$, the convergence of the sums reduces to the behavior of p-series. We have $p_1 > 1/2$, and $p_{2,i} > p_1$, and $p_{3,i} > p_1$ for all i, hence $p_{2,i} > 1/2$, and $p_{3,i} > 1/2$ for all i. Moreover, for the largest exponent $p_{3,N}$,

$$p_{3,N} = \beta\frac{2N+1}{N+1} < 1,$$

so $\forall i \in \{1, \ldots, N\}$, $\frac{1}{2} < p_1 < p_{2,i} < p_{3,i} < 1$. Thus $\sum_e \eta_1(e) = \sum_e \eta_{2,i}(e) = \sum_e \eta_{3,i}(e) = \infty, \forall i \in \{1, \ldots, N\}$, and $\sum_e \eta_1(e)^2 < \infty, \sum_e \eta_{2,i}(e)^2 < \infty, \sum_e \eta_{3,i}(e)^2 < \infty, \forall i \in \{1, \ldots, N\}$. For any $p' > p$,

$$\lim_{e\to+\infty} \frac{(1+e/K)^{-p'}}{(1+e/K)^{-p}} = \lim_{e\to+\infty} (1+e/K)^{-(p'-p)} = 0.$$

Thus, given $p_{2,1} > p_1$, and $p_{3,i} > p_{2,i}, \forall i \in \{1, \ldots, N\}$, and $p_{2,i} > p_{3,i-1}, \forall i \in \{2, \ldots, N\}$, we have $\eta_{2,1}(e) = o(\eta_1(e)); \eta_{3,i}(e) = o(\eta_{2,i}(e)), \forall i \in \{1, \ldots, N\}$; $\eta_{2,i}(e) = o(\eta_{3,i-1}(e))$, $\forall i \in \{2, \ldots, N\}$. The positive constants $c_1, c_{2,i}, c_{3,i}$ do not affect these asymptotic limits. Thus, all conditions in the assumption are satisfied. It is noteworthy that the convergence analysis is

Table 6.7: Reward and safety cost comparison for two tasks with different K values for learning rate design

Algorithm	Ant		Coupled HalfCheetah	
	Reward	Cost	Reward	Cost
HAPPO	41.29 ± 215.89	311.96 ± 67.96	2197.22 ± 112.77	230.28 ± 67.11
$K = 1000$	698.50 ± 181.70	4.27 ± 6.67	2567.26 ± 51.97	17.96 ± 32.99
$K = 2500$	970.57 ± 92.38	$\mathbf{0.59 \pm 1.45}$	2609.67 ± 49.43	0.80 ± 3.42
$K = 5000$	$\mathbf{1064.21 \pm 84.91}$	3.14 ± 11.29	$\mathbf{2611.20 \pm 92.02}$	$\mathbf{0.19 \pm 0.50}$

asymptotic and formally assumes infinitely many updates. In practice, however, only a finite number of updates can be performed. Consequently, practical convergence toward a feasible local Nash equilibrium is approximated by running a sufficiently large number of updates while allowing the learning rates to evolve according to the theoretically justified schedule. In particular, the hyperparameter K controls the decay speed of the learning rates. If K is chosen too small, the learning rates decay rapidly and may prevent sufficient adaptation. Conversely, if K is too large, the learning rates remain relatively large for a prolonged period, potentially increasing oscillatory behaviour. Therefore, selecting K requires balancing and hyperparameter tuning. Here we provide some experimental results on the choice of K in Table 6.7. The results indicate that the proposed learning-rate design remains effective across a range of K values, although the specific choice of K influences the values of metrics.

To mitigate potential slow convergence due to timescale separation, we incorporate additional squared penalty terms into the constraint loss, which encourage and accelerate the update toward feasible solutions. Based on the experimental results, our algorithm has a better sample efficiency compared to baselines in quickly achieving higher rewards while maintaining very few or near-zero safety violations with fewer samples, allowing early stopping to reduce computational burden. Moreover, for homogeneous agents, the parameter sharing method can be applied to reduce the number of trainable parameters and distinct learning rates, which further improves the training efficiency. However, further investigation under the multi-timescale framework, including more techniques for enhancing sample efficiency, remains an important direction for future work.

Assumption 6.5. *For each agent i, the defined signals $\hat{r}_t^i$, $\hat{r}_t^{i,*}$, $\hat{d}_t^i$, $\hat{d}_t^{i,*}$, $\hat{c}_t^i$, and $\hat{c}_t^{i,*}$, as well*

as the function $Q_H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_i)$, $Q_\alpha^i(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{i,t}; \hat{\psi}_i)$, *and* $Q_V^i(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{i,t}; \hat{\chi}_i)$, *and the space* $\hat{\Omega}_i$, $\hat{\Psi}_i$, $\hat{X}_i$ *are bounded. Besides,* $\nabla_{\hat{\omega}_i} Q_H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_i)$ *is a Lipschitz function in* $\hat{\omega}_i$ *for all* $\boldsymbol{x}_t$ *and* $\boldsymbol{u}_t$, $\nabla_{\hat{\psi}_i} Q_\alpha^i(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{i,t}; \hat{\psi}_i)$ *is a Lipschitz function in* $\hat{\psi}_i$ *for all* $\boldsymbol{x}_t$, $\boldsymbol{u}_t$ *and* $\alpha_{i,t}$, *and* $\nabla_{\hat{\chi}_i} Q_V^i(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{i,t}; \hat{\chi}_i)$ *is a Lipschitz function in* $\hat{\chi}_i$ *for all* $\boldsymbol{x}_t$, $\boldsymbol{u}_t$ *and* $\alpha_{i,t}$.

This assumption guarantees the boundedness of signals and critic functions for agents, which is a standard requirement in actor-critic methods to ensure stable training and convergence. The Lipschitz condition, which is also commonly used in previous RL studies, on the gradient promotes well-behaved updates. For neural networks, approximate Lipschitz continuity can be induced through techniques such as weight decay and clipping, making this assumption realistic in practice.

Note: The terms $\hat{r}_t^i$, $\hat{r}_t^{i,*}$, Q_H^i, $\hat{\omega}_i$, *and* $\hat{\Omega}_i$ *are formally introduced in Section 6.5.3. They are referenced here to provide a complete overview of all theoretical assumptions, even though they are not defined in the main body of the submission.*

Assumption 6.6. *For each agent* i, *there always exists a policy* π^i *such that*

$$H_{\boldsymbol{\pi}}^i(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) - H_{\boldsymbol{\pi}}^i(\boldsymbol{x}_t, \boldsymbol{u}_t) \geq -\epsilon H_{\boldsymbol{\pi}}^i(\boldsymbol{x}_t, \boldsymbol{u}_t)$$

for all consecutive pairs $(\boldsymbol{x}_t, \boldsymbol{u}_t)$ *and* $(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1})$, $\forall t \geq 0$, *along with trajectories generated under any joint policy* $\boldsymbol{\pi}^{-i}$. *Further, all initial states are within the safe set* $\boldsymbol{\mathcal{X}}_{safe}$.

This assumption rules out scenarios where the environment, including other agents' fixed policies, renders safety of agent i impossible, namely situations where no admissible action can help to satisfy safety requirements for agent i. Since the other agents' policies $\boldsymbol{\pi}^{-i}$ are treated as part of the environment in this formulation, this feasibility condition is effectively an assumption for single-agent tasks. It ensures the problem is well-posed and that a safe policy is learnable and achievable under the given dynamics. The requirement that all initial states lie within the safe set is also standard in constrained control setups, as starting from unsafe states would inevitably lead to safety constraint violations.

Assumption 6.7. *The state space and action space for each agent are bounded. Besides, the parameter space* Θ_i, *Lagrangian multiplier* λ_i, *and* α_i *for each agent* i *are also bounded. Further,*

$\nabla_{u^i} Q^i(\boldsymbol{x}, \boldsymbol{u}; \phi_i)$, $\nabla_{u^i} Q^i_H(\boldsymbol{x}, \boldsymbol{u}; \hat{\omega}_i)$, $\nabla_{u^i} Q^i_V(\boldsymbol{x}, \boldsymbol{u}, \alpha_i; \hat{\chi}_i)$, *and* $\nabla_{u^i} Q^i_\alpha(\boldsymbol{x}, \boldsymbol{u}, \alpha_i; \hat{\psi}_i)$ *are Lipschitz functions in* u^i_t *for all* $\boldsymbol{x}$, ϕ_i, $\hat{\omega}_i$, $\hat{\chi}_i$ *and* $\hat{\psi}_i$, *and are also Lipschitz functions in* ϕ_i, $\hat{\omega}_i$, $\hat{\chi}_i$ *and* $\hat{\psi}_i$, *respectively, for all* $\boldsymbol{x}$, $\boldsymbol{u}$ *and* α_i. *Such Lipschitz conditions also apply to the derivatives of these functions with respect to* α_i. $\nabla_{\theta_i} u^i(\boldsymbol{x}; \theta_i)$ *and* $\nabla_{\theta_i} \alpha_i(\boldsymbol{x}; \theta_i)$ *are Lipschitz functions in* θ_i.

This assumption ensures the boundedness of the relevant spaces and the smoothness of gradients used in the policy update. Bounded state and action spaces are standard in RL and control theory to avoid unbounded optimization landscapes. Bounding the policy parameter and Lagrange multiplier spaces helps the dual updates and prevents divergence. The Lipschitz continuity of the critic gradients and policy gradient indicates well-behaved gradient-based updates during policy training. In practice, neural networks with smooth activations and regularization techniques can effectively lead to approximate Lipschitz properties, which makes the assumption mild and practical.

Assumption 6.8. *For each agent i, the function* $Q^i_H(\boldsymbol{x}, \boldsymbol{u}; \hat{\omega}_i)$, $Q^i_V(\boldsymbol{x}, \boldsymbol{u}, \alpha_i; \hat{\chi}_i)$ *and* $Q^i_\alpha(\boldsymbol{x}, \boldsymbol{u}, \alpha_i; \hat{\psi}_i)$ *are Lipschitz function in* u^i_t *for all* $\boldsymbol{x}$ *and their parameterizations, respectively. Moreover, the gradients* $\nabla_{u^i} Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})$, $\nabla_{u^i} Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u})$, $\nabla_{u^i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i)$, *and* $\nabla_{u^i} Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i)$ *are Lipschitz functions in the joint policy* $\bar{\boldsymbol{\pi}}_i$. *Such Lipschitz conditions also apply to the derivatives of these functions with respect to* α_i.

This assumption ensures local smoothness of the value function and its gradients with respect to the agent's action. It is essential for bounding the approximation error during policy updates, and reflects a standard smoothness assumption in theoretical RL, capturing the idea that small changes in the policy lead to proportionally bounded changes in value estimates. In practice, the approximate Lipschitz property of value networks can also be induced through methods like regularization techniques, as stated in previous explanations.

Note: Related terms and gradients are formally introduced in Section 6.5.3. They are included here for completeness, as this appendix summarizes all theoretical assumptions supporting the theoretical analysis.

Assumption 6.9. *For functions* $Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})$, $Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u})$, $Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i)$, $Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i)$, *their derivatives* $\nabla_{u^i} Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})$, $\nabla_{u^i} Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u})$, $\nabla_{u^i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i)$ $\nabla_{u^i} Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i)$ *are Lipschitz functions in* $Q^i_{\bar{\boldsymbol{\pi}}_i}$, $Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}$, $Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}$, $Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}$, *respectively, and action* u^i. *Additionally,* $\nabla_{\theta_i} u^i(\boldsymbol{x}; \theta_i)$ *and*

$\nabla_{\theta_i}\alpha_i(\boldsymbol{x};\theta_i)$ *are Lipschitz functions in* π^i *parameterized by* θ_i.

This assumption again requires local smoothness of critic functions' gradients. Still, in practice, neural networks with smooth activations and regularization techniques can effectively lead to approximate Lipschitz properties, which makes the assumption mild and practical.

6.5.3 C. Formal Version and Proof for Theorem 6.2

Theorem 6.4. *Under Assumptions 6.3-6.8, the joint policy updated in Algorithm 6.1 will almost surely converge to a feasible local Nash equilibrium* $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$, *where* $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) \geq 0$ *for all* $(\boldsymbol{x}_t, \boldsymbol{u}_t)$ *along the trajectory induced by* $\boldsymbol{\pi}^*$, *for every* $i \in \mathcal{N}$ *and* $t \geq 0$. *Define* $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) := \min_{i\in\mathcal{N}} H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$, *then* $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ *is a global barrier certificate for the multi-agent system. Furthermore, in the absence of exogenous signal* $\mathbf{d}_t$, *every agent* i *will converge to its desired state (goal),* $x^i_{desired}$, *under the policy* $\boldsymbol{\pi}^*$.

Proof. This proof is extended from [136, 137], but considers a different problem with additional squared terms to better achieve and maintain the satisfaction of the constraints, and include multiple agents, which necessitates additional analysis when updating the policy of agent i to account for the inaccuracy caused by the updates of previous agents (namely from 1 to $i-1$). Additionally, multiple constraints are considered here. The structure of the proof is as follows:

1. For the first updated agent, we first demonstrate that its critic, barrier, and Lyapunov certificates will converge to the fixed solution corresponding to the policy parameterized by $\{\theta_i^k\}_{i\in\mathcal{N}}$.

2. We show that the policy θ_1 and the corresponding Lagrangian multiplier λ_1 will converge to a local saddle point with respect to $(\{\theta_i^k\}_{i\in\{2,\ldots,N\}}, \{\lambda_i^k\}_{i\in\{2,\ldots,N\}})$.

3. For the second updated agent, we similarly prove that the policy θ_2 and λ_2 will converge to a local saddle point with respect to $(\{\theta_i^k\}_{i\in\{3,\ldots,N\}}, \{\lambda_i^k\}_{i\in\{3,\ldots,N\}})$. By induction, finally, the algorithm converge to a feasible local Nash equilibrium asymptotically.

First, we investigate the first agent's update: When update the first agent, because all other agents are updated on slower timescales, we treat $\{\theta_i^k\}_{i\in\{2,\ldots,N\}}$ (namely $\boldsymbol{\theta}_{-1}$) and $\{\lambda_i^k\}_{i\in\{2,\ldots,N\}}$

as certain fixed values, and thus let $\bar{\boldsymbol{\pi}}_1 = (\theta_1^k, \theta_2^k, \ldots, \theta_N^k)$ where θ_1^k is being updated and $\{\theta_i^k\}_{i\in\{2,\ldots,N\}}$ are fixed constants. Later, we have the following different timescales:

Convergence of ϕ_i, ω_i and χ_i, $\forall i \in \mathcal{N}$: The Assumption 6.4 indicates that $\forall i \in \mathcal{N}$, the updates of ϕ_i^k, ω_i^k and χ_i^k converge on a faster timescale than policies $\{\theta_i^k\}_{i\in\mathcal{N}}$ and $\{\lambda_i^k\}_{i\in\mathcal{N}}$. According to [138], we can deem $\{\theta_i^k\}_{i\in\mathcal{N}}$ and $\{\lambda_i^k\}_{i\in\mathcal{N}}$ as arbitrary fixed values when updating ϕ_i^k, ω_i^k and χ_i^k, and therefore, with the standard convergence results in [30], we can easily know that for any policy $\bar{\boldsymbol{\pi}}$ and any $i \in \mathcal{N}$,

$$\begin{aligned}
\mathcal{B}^{\bar{\boldsymbol{\pi}}}[Q^i](\boldsymbol{x}_t, \boldsymbol{u}_t) &= r^i + \gamma \mathbb{E}_{\boldsymbol{u}_{t+1}\sim\bar{\boldsymbol{\pi}}}\big[Q^i(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1})\big] \\
\mathcal{B}^{\bar{\boldsymbol{\pi}}}[H^i](\boldsymbol{x}_t, \boldsymbol{u}_t) &= b^i + \gamma \mathbb{E}_{\boldsymbol{u}_{t+1}\sim\bar{\boldsymbol{\pi}}}\big[H^i(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1})\big] \\
\mathcal{B}^{\bar{\boldsymbol{\pi}}}[V^i](x_t^i, u_t^i) &= c^i + \gamma \mathbb{E}_{\boldsymbol{u}_{t+1}\sim\bar{\boldsymbol{\pi}}}\big[V^i(x_{t+1}^i, u_{t+1}^i)\big]
\end{aligned} \tag{6.19}$$

are $\gamma-$contraction mappings, and therefore when $k \to \infty$, for a certain joint policy $\bar{\boldsymbol{\pi}}$ parameterized by arbitrary $\{\theta_i^k\}_{i\in\mathcal{N}}$, we have $\phi_i^k \to \phi_i^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $Q^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \phi_i^k) \to Q^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \phi_i^*) = Q_{\bar{\boldsymbol{\pi}}}^i(\boldsymbol{x}_t, \boldsymbol{u}_t)$, $\omega_i^k \to \omega_i^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i^k) \to H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_i^*) = H_{\bar{\boldsymbol{\pi}}}^i(\boldsymbol{x}_t, \boldsymbol{u}_t)$, and $\chi_i^k \to \chi_i^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $V^i(x_t^i, u_t^i; \chi_i^k) \to V^i(x_t^i, u_t^i; \chi_i^*) = V_{\bar{\boldsymbol{\pi}}}^i(x_t^i, u_t^i)$. Thus, the convergence of this timescale is proved. Furthermore, for the agent 1, when $\bar{\boldsymbol{\pi}}_1$ is considered, define

$$\begin{aligned}
\hat{r}_t^1(\boldsymbol{x}_t, \boldsymbol{u}_t) = \mathbb{E}_{\tilde{\boldsymbol{u}}_{t+1}\sim\bar{\boldsymbol{\pi}}_1}\big[&\max\{0, -H^1(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}; \omega_1^k) + H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_1^k) - \epsilon H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_1^k)\} \\
&+ \frac{1}{2}[\max\{0, -H^1(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}; \omega_1^k) + H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_1^k) - \epsilon H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_1^k)\}]^2\big]
\end{aligned} \tag{6.20}$$

and

$$\begin{aligned}
\hat{r}_t^{1,*}(\boldsymbol{x}_t, \boldsymbol{u}_t) = \mathbb{E}_{\tilde{\boldsymbol{u}}_{t+1}\sim\bar{\boldsymbol{\pi}}_1}\big[&\max\{0, -H_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) + H_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_t, \boldsymbol{u}_t) - \epsilon H_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)\} \\
&+ \frac{1}{2}[\max\{0, -H_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_{t+1}, \tilde{\boldsymbol{u}}_{t+1}) + H_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_t, \boldsymbol{u}_t) - \epsilon H_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)\}]^2\big],
\end{aligned} \tag{6.21}$$

with

$$Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}, \boldsymbol{u}) = \mathbb{E}_{\tau\sim\bar{\boldsymbol{\pi}}_1}\big[\sum_{t=0}^{\infty} \gamma^t \hat{r}_t^{1,*}(\boldsymbol{x}_t, \boldsymbol{u}_t) | \boldsymbol{x}_0 = \boldsymbol{x}, \boldsymbol{u}_0 = \boldsymbol{u}\big], \tag{6.22}$$

then, given a step size $\eta_c(e)$ such that $\sum_e \eta_c(e) = \infty$, $\sum_e \eta_c(e)^2 < \infty$, $\eta_c(e) = o(\eta_1(e))$, $\eta_{2,1}(e) = o(\eta_c(e))$, if parameterize this $Q^1_{H_{\boldsymbol{\pi}_1}}$ with $\hat{\omega}_1$, we have the update by using some sampled data:

$$\hat{\omega}_1^{k+1} = \Gamma_{\hat{\Omega}_1}\big[\hat{\omega}_1^k - \eta_c \nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k}\big]. \tag{6.23}$$

and an ideal update

$$\hat{\omega}_1^{k+1} = \Gamma_{\hat{\Omega}_1}\big[\hat{\omega}_1^k - \eta_c \nabla_{\hat{\omega}_1}(\hat{r}_t^{1,*} + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k}\big]. \tag{6.24}$$

The difference between (6.23) and (6.24) is the use of signal ($\hat{r}_t^1$ or $\hat{r}_t^{1,*}$), which reveals the inaccuracy caused in the update of ω_1 in the previous timescale. Then, we can rewrite the real update when the sampled batch D follows the distribution caused by $\bar{\boldsymbol{\pi}}_1$ to be:

$$\begin{aligned}\hat{\omega}_1^{k+1} = \Gamma_{\hat{\Omega}_1}\big[\hat{\omega}_1^k - \eta_c \nabla_{\hat{\omega}_1}\big((\hat{r}_t^{1,*} + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k} \\ + \delta\hat{\omega}_1^{k+1} + \delta\hat{\omega}_{1,\varepsilon}\big)\big].\end{aligned} \tag{6.25}$$

where

$$\begin{aligned}\delta\hat{\omega}_1^{k+1} = -\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k}\big] \\ + \nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k}\end{aligned} \tag{6.26}$$

and

$$\begin{aligned}\delta\hat{\omega}_{1,\varepsilon} = \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k} \\ - \nabla_{\hat{\omega}_1}(\hat{r}_t^{1,*} + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k}\big]\end{aligned} \tag{6.27}$$

are two terms representing the difference between the real update and the ideal update.

Under Assumption 6.5:

1. We show that $\delta\hat{\omega}_1^{k+1}$ is square integrable first. To be specific,

$$\begin{aligned}\mathbb{E}\big[||\delta\hat{\omega}_1^{k+1}||^2|\mathcal{F}_{\hat{\omega}_1,k}\big] \leq C_1||\nabla_{\hat{\omega}_1} Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k)||_\infty^2 \times (||\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1^k)||_\infty^2 \\ + ||Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k)||_\infty^2).\end{aligned} \tag{6.28}$$

$\mathcal{F}_{\hat{\omega}_1,k} = \sigma(\hat{\omega}_1^m, \delta\hat{\omega}_1^m, m \leq k)$ is the filtration of $\hat{\omega}_1^k$ generated by different independent trajectories [137]. Further, Assumption 6.5 requires that $||\nabla_{\hat{\omega}_1} Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1)||_\infty^2 \leq K_1(1 + ||\hat{\omega}_1||_\infty^2)$ and other terms are bounded, hence, $\mathbb{E}\big[||\delta\hat{\omega}_1^{k+1}||^2|\mathcal{F}_{\hat{\omega}_1,k}\big] < \infty$, and $\delta\hat{\omega}_1^{k+1}$ is square integrable.

2. We then show that $\delta\hat{\omega}_{1,\varepsilon} \to 0$:

$$
\begin{aligned}
||\delta\hat{\omega}_{1,\varepsilon}|| \leq C_2||\nabla_{\hat{\omega}_1} Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1)||_\infty||\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[2(\hat{r}_t^{1,*} + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1^k) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k)) \\
- 2(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1^k) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k))\big]||
\end{aligned}
$$

Since $||\nabla_{\hat{\omega}_1} Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1)||$ is upper bounded, and we have $\hat{r}_t^1 \to \hat{r}_t^{1,*}$ given $H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_1^k) \to H_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)$ according to the analysis in the previous timescale, we have $||\delta\hat{\omega}_{1,\varepsilon}|| \to 0$, and therefore $\delta\hat{\omega}_{1,\varepsilon} \to 0$.

3. Because the $\nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k}$ is a sample of $\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k}\big]$, we can conclude that $\mathbb{E}\big[\delta\hat{\omega}_1^{k+1}|\mathcal{F}_{\hat{\omega}_1,k}\big] = 0$.

When $\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\omega}_1}(\hat{r}_t^1 + \gamma Q_H^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}; \hat{\omega}_1) - Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1))^2|_{\hat{\omega}_1 = \hat{\omega}_1^k}\big]$ is a Lipschitz function in $\hat{\omega}_1$, following a similar way which will be detailed in the update of policy θ_1 in the next step, we conclude that $\hat{\omega}_1^k$ converges to a minimizer $\hat{\omega}_1^*$. Following the analysis of policy evaluation in previous studies, assume that a sufficiently expressive function approximator is used such that there is only one local minimum, within the space $\hat{\Omega}_1$, which minimizes the TD error to 0, then we have shown that $\hat{\omega}_1^k \to \hat{\omega}_1^*(\{\theta_i^k\}_{i \in \mathcal{N}})$ where $Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^k) \to Q_H^1(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_1^*) = Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t, \boldsymbol{u}_t)$.

Later, we consider another constraint. Define:

$$
\begin{aligned}
\hat{d}_t^1(\boldsymbol{x}_t, \boldsymbol{u}_t) = \mathbb{E}_{\alpha_1 \sim \bar{\boldsymbol{\pi}}_1}\big[\max\{0, (1+\beta_1)(\sum_{j \neq 1} \delta_{1j} V^j(x_t^j, u_t^j; \chi_j^k)) - \alpha_1(\boldsymbol{x}_t) V^1(x_t^i, u_t^i; \chi_1^k)\} \\
+ \frac{1}{2}[\max\{0, (1+\beta_1)(\sum_{j \neq 1} \delta_{1j} V^j(x_t^j, u_t^j; \chi_j^k)) - \alpha_1(\boldsymbol{x}_t) V^1(x_t^i, u_t^i; \chi_1^k)\}]^2\big]
\end{aligned}
\tag{6.29}
$$

and

$$\begin{aligned}\hat{d}_t^{1,*}(\boldsymbol{x}_t, \boldsymbol{u}_t) = \mathbb{E}_{\alpha_1 \sim \bar{\boldsymbol{\pi}}_1}\big[&\max\{0, (1+\beta_1)(\sum_{j\neq 1} \delta_{1j} V_{\bar{\boldsymbol{\pi}}_1}^j(x_t^j, u_t^j)) - \alpha_1(\boldsymbol{x}_t) V_{\bar{\boldsymbol{\pi}}_1}^1(x_t^i, u_t^i)\} \\ &+ \frac{1}{2}[\max\{0, (1+\beta_1)(\sum_{j\neq 1} \delta_{1j} V_{\bar{\boldsymbol{\pi}}_1}^j(x_t^j, u_t^j)) - \alpha_1(\boldsymbol{x}_t) V_{\bar{\boldsymbol{\pi}}_1}^1(x_t^i, u_t^i;)\}]^2\big],\end{aligned} \tag{6.30}$$

with

$$Q_{\alpha_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}, \boldsymbol{u}, \alpha_1) = \mathbb{E}_{\tau \sim \bar{\boldsymbol{\pi}}_1}\big[\sum_{t=0}^{\infty} \gamma^t \hat{d}_t^{1,*}(\boldsymbol{x}_t, \boldsymbol{u}_t) | \boldsymbol{x}_0 = \boldsymbol{x}, \boldsymbol{u}_0 = \boldsymbol{u}\big], \tag{6.31}$$

since α_1 is also incorporated in the $\hat{d}_t^{1,*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ signal, and α_1 is also one part of the output of the policy $\bar{\boldsymbol{\pi}}_1$, together with u_t^i. Parameterize this $Q_{\alpha_{\bar{\boldsymbol{\pi}}_1}}^1$ with $\hat{\psi}_1$, we have the update by using some sampled data:

$$\hat{\psi}_1^{k+1} = \Gamma_{\hat{\Psi}_1}\big[\hat{\psi}_1^k - \eta_c \nabla_{\hat{\psi}_1}(\hat{d}_t^1 + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1))^2|_{\hat{\psi}_1 = \hat{\psi}_1^k}\big]. \tag{6.32}$$

and an ideal update

$$\hat{\psi}_1^{k+1} = \Gamma_{\hat{\Psi}_1}\big[\hat{\psi}_1^k - \eta_c \nabla_{\hat{\psi}_1}(\hat{d}_t^{1,*} + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1))^2|_{\hat{\psi}_1 = \hat{\psi}_1^k}\big]. \tag{6.33}$$

The difference between (6.32) and (6.33) is the use of signal ($\hat{d}_t^1$ or $\hat{d}_t^{1,*}$). Then, we can rewrite the real update for $\hat{\psi}_1$ as follows:

$$\begin{aligned}\hat{\psi}_1^{k+1} = \Gamma_{\hat{\Psi}_1}\big[\hat{\psi}_1^k - \eta_c \nabla_{\hat{\psi}_1}\big((\hat{d}_t^{1,*} + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1))^2|_{\hat{\psi}_1 = \hat{\psi}_1^k} \\ + \delta\hat{\psi}_1^{k+1} + \delta\hat{\psi}_{1,\varepsilon}\big)\big].\end{aligned} \tag{6.34}$$

where

$$\begin{aligned}\delta\hat{\psi}_1^{k+1} = &-\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\psi}_1}(\hat{d}_t^1 + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1))^2|_{\hat{\psi}_1=\hat{\psi}_1^k}\big] \\ &+ \nabla_{\hat{\psi}_1}(\hat{d}_t^1 + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1))^2|_{\hat{\psi}_1=\hat{\psi}_1^k}\end{aligned} \tag{6.35}$$

and

$$\begin{aligned}\delta\hat{\psi}_{1,\varepsilon} = &\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\psi}_1}(\hat{d}_t^1 + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1))^2|_{\hat{\psi}_1=\hat{\psi}_1^k} \\ &- \nabla_{\hat{\psi}_1}(\hat{d}_t^{1,*} + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1))^2|_{\hat{\psi}_1=\hat{\psi}_1^k}\big]\end{aligned} \tag{6.36}$$

are two terms representing the difference between the real update and the ideal update.

Under Assumption 6.5:

1.

$$\begin{aligned}\mathbb{E}\big[||\delta\hat{\psi}_1^{k+1}||^2|\mathcal{F}_{\hat{\psi}_1,k}\big] \leq C_3||\nabla_{\hat{\psi}_1}Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1^k)||_\infty^2 \times (||\hat{d}_t^1 + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1^k)||_\infty^2 \\ + ||Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1^k)||_\infty^2)\end{aligned} \tag{6.37}$$

where $\mathcal{F}_{\hat{\psi}_1,k} = \sigma(\hat{\psi}_1^m, \delta\hat{\psi}_1^m, m \leq k)$ is the filtration of $\hat{\psi}_1^k$ generated by different independent trajectories. Furthermore, Assumption 6.5 requires that $||\nabla_{\hat{\psi}_1}Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1)||_\infty^2 \leq K_2(1 + ||\hat{\psi}_1||_\infty^2)$ and other terms are bounded, therefore, $\mathbb{E}\big[||\delta\hat{\psi}_1^{k+1}||^2|\mathcal{F}_{\hat{\psi}_1,k}\big] < \infty$, and $\delta\hat{\psi}_1^{k+1}$ is square integrable.

2. We then show that $\delta\hat{\psi}_{1,\varepsilon} \to 0$:

$$\begin{aligned}||\delta\hat{\psi}_{1,\varepsilon}|| \leq C_4||\nabla_{\hat{\psi}_1}Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1)||_\infty||\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[2(\hat{d}_t^{1,*} + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1^k) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1^k)) \\ - 2(\hat{d}_t^1 + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1^k) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1^k))\big]||\end{aligned}$$

Since $||\nabla_{\hat{\psi}_1}Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1)||$ is upper bounded, and we have $\hat{d}_t^1 \to \hat{d}_t^{1,*}$ given $\forall i \in \mathcal{N}$, $V^i(x_t^i, u_t^i; \chi_i^k) \to V_{\bar{\boldsymbol{\pi}}_1}^i(x_t^i, u_t^i)$, based on the analysis in the previous timescale, we have $||\delta\hat{\psi}_{1,\varepsilon}|| \to 0$, and therefore $\delta\hat{\psi}_{1,\varepsilon} \to 0$.

3. Because the $\nabla_{\hat{\psi}_1}(\hat{d}_t^1 + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1))^2|_{\hat{\psi}_1=\hat{\psi}_1^k}$ is a

sample of $\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\psi}_1}(\hat{d}_t^1 + \gamma Q_\alpha^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\psi}_1) - Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1))^2|_{\hat{\psi}_1=\hat{\psi}_1^k}\big]$, we can conclude that $\mathbb{E}\big[\delta\hat{\psi}_1^{k+1}|\mathcal{F}_{\hat{\psi}_1,k}\big] = 0$.

Then, according to the same analysis of $\hat{\omega}_1$, we conclude that $\hat{\psi}_1^k$ converges to a minimizer $\hat{\psi}_1^*$, namely $\hat{\psi}_1^k \to \hat{\psi}_1^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1^k) \to Q_\alpha^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\psi}_1^*) = Q_{\alpha_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t})$.

We then work on the last constraint. Define:

$$\hat{c}_t^1(\boldsymbol{x}_t, \boldsymbol{u}_t) = \mathbb{E}_{(\tilde{\boldsymbol{u}}_{t+1},\alpha_1)\sim\bar{\boldsymbol{\pi}}_1}\big[\max\{0, V^1(x_{t+1}^1, \tilde{u}_{t+1}^1; \chi_1^k) - (1-\alpha_1(\boldsymbol{x}_t))V^1(x_t^1, u_t^1; \chi_1^k) - \sum_{j\neq 1}\delta_{1j}V^j(x_t^j, u_t^j; \chi_j^k)\}$$
$$+ \frac{1}{2}[\max\{0, V^1(x_{t+1}^1, \tilde{u}_{t+1}^1; \chi_1^k) - (1-\alpha_1(\boldsymbol{x}_t))V^1(x_t^1, u_t^1; \chi_1^k) - \sum_{j\neq 1}\delta_{1j}V^j(x_t^j, u_t^j; \chi_j^k)\}]^2\big], \tag{6.38}$$

and

$$\hat{c}_t^{1,*}(\boldsymbol{x}_t, \boldsymbol{u}_t) = \mathbb{E}_{(\tilde{\boldsymbol{u}}_{t+1},\alpha_1)\sim\bar{\boldsymbol{\pi}}_1}\big[\max\{0, V_{\bar{\boldsymbol{\pi}}_1}^1(x_{t+1}^1, \tilde{u}_{t+1}^1) - (1-\alpha_1(\boldsymbol{x}_t))V_{\bar{\boldsymbol{\pi}}_1}^1(x_t^1, u_t^1) - \sum_{j\neq 1}\delta_{1j}V_{\bar{\boldsymbol{\pi}}_1}^j(x_t^j, u_t^j)\}$$
$$+ \frac{1}{2}[\max\{0, V_{\bar{\boldsymbol{\pi}}_1}^1(x_{t+1}^1, \tilde{u}_{t+1}^1) - (1-\alpha_1(\boldsymbol{x}_t))V_{\bar{\boldsymbol{\pi}}_1}^1(x_t^1, u_t^1) - \sum_{j\neq 1}\delta_{1j}V_{\bar{\boldsymbol{\pi}}_1}^j(x_t^j, u_t^j)\}]^2\big], \tag{6.39}$$

with

$$Q_{V_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}, \boldsymbol{u}, \alpha_1) = \mathbb{E}_{\tau\sim\bar{\boldsymbol{\pi}}_1}\big[\sum_{t=0}^{\infty}\gamma^t\hat{c}_t^{1,*}(\boldsymbol{x}_t, \boldsymbol{u}_t)|\boldsymbol{x}_0 = \boldsymbol{x}, \boldsymbol{u}_0 = \boldsymbol{u}\big], \tag{6.40}$$

since α_1 is also incorporated in the $\hat{c}_t^{1,*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ signal, and α_1 is also one part of the output of the policy $\bar{\boldsymbol{\pi}}_1$. Again, if we parameterize this $Q_{V\bar{\boldsymbol{\pi}}_1}^1$ with $\hat{\chi}_1$, we have the update by using some sampled data:

$$\hat{\chi}_1^{k+1} = \Gamma_{\hat{X}_1}\big[\hat{\chi}_1^k - \eta_c\nabla_{\hat{\chi}_1}(\hat{c}_t^1 + \gamma Q_V^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\chi}_1) - Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1))^2|_{\hat{\chi}_1=\hat{\chi}_1^k}\big]. \tag{6.41}$$

and an ideal update

$$\hat{\chi}_1^{k+1} = \Gamma_{\hat{\chi}_1}\big[\hat{\chi}_1^k - \eta_c \nabla_{\hat{\chi}_1}(\hat{c}_t^{1,*} + \gamma Q_V^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\chi}_1) - Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1))^2|_{\hat{\chi}_1=\hat{\chi}_1^k}\big]. \tag{6.42}$$

The difference between (6.41) and (6.42) is the use of signal ($\hat{c}_t^1$ or $\hat{c}_t^{1,*}$). Then, we can rewrite the real update for $\hat{\psi}_1$ as follows:

$$\begin{aligned}\hat{\chi}_1^{k+1} = \Gamma_{\hat{\chi}_1}\big[\hat{\chi}_1^k - \eta_c \nabla_{\hat{\chi}_1}\big((\hat{c}_t^{1,*} + \gamma Q_V^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\chi}_1) - Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1))^2|_{\hat{\chi}_1=\hat{\chi}_1^k} \\ + \delta\hat{\chi}_1^{k+1} + \delta\hat{\chi}_{1,\varepsilon}\big)\big].\end{aligned} \tag{6.43}$$

where

$$\begin{aligned}\delta\hat{\chi}_1^{k+1} = -\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\chi}_1}(\hat{c}_t^1 + \gamma Q_V^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\chi}_1) - Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1))^2|_{\hat{\chi}_1=\hat{\chi}_1^k}\big] \\ + \nabla_{\hat{\chi}_1}(\hat{c}_t^1 + \gamma Q_V^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\chi}_1) - Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1))^2|_{\hat{\chi}_1=\hat{\chi}_1^k}\end{aligned} \tag{6.44}$$

and

$$\begin{aligned}\delta\hat{\chi}_{1,\varepsilon} = \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\chi}_1}(\hat{c}_t^1 + \gamma Q_V^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\chi}_1) - Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1))^2|_{\hat{\chi}_1=\hat{\chi}_1^k} \\ - \nabla_{\hat{\chi}_1}(\hat{c}_t^{1,*} + \gamma Q_V^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\chi}_1) - Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1))^2|_{\hat{\chi}_1=\hat{\chi}_1^k}\big]\end{aligned} \tag{6.45}$$

Under Assumption 6.5:

1.

$$\begin{aligned}\mathbb{E}\big[||\delta\hat{\chi}_1^{k+1}||^2|\mathcal{F}_{\hat{\chi}_1,k}\big] \leq & C_5||\nabla_{\hat{\chi}_1} Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1^k)||_\infty^2 \times (||\hat{c}_t^1 + \gamma Q_V^1(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}, \alpha_{1,t+1}; \hat{\chi}_1^k)||_\infty^2 \\ & + ||Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1^k)||_\infty^2)\end{aligned} \tag{6.46}$$

where $\mathcal{F}_{\hat{\chi}_1,k} = \sigma(\hat{\chi}_1^m, \delta\hat{\chi}_1^m, m \leq k)$ is the filtration of $\hat{\chi}_1^k$ generated by different independent trajectories. Furthermore, Assumption 6.5 requires that $||\nabla_{\hat{\chi}_1} Q_V^1(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{1,t}; \hat{\chi}_1)||_\infty^2 \leq K_3(1 + ||\hat{\chi}_1||_\infty^2)$ with other terms bounded, therefore, $\mathbb{E}\big[||\delta\hat{\chi}_1^{k+1}||^2|\mathcal{F}_{\hat{\chi}_1,k}\big] < \infty$, and $\delta\hat{\chi}_1^{k+1}$ is square

integrable.

2. We then show that $\delta\hat{\chi}_{1,\varepsilon} \to 0$:

$$||\delta\hat{\chi}_{1,\varepsilon}|| \leq C_6||\nabla_{\hat{\chi}_1}Q_V^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\chi}_1)||_\infty||\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[2(\hat{c}_t^{1,*}+\gamma Q_V^1(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1},\alpha_{1,t+1};\hat{\chi}_1^k)-Q_V^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\chi}_1^k))$$
$$-2(\hat{c}_t^1+\gamma Q_V^1(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1},\alpha_{1,t+1};\hat{\chi}_1^k)-Q_V^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\chi}_1^k))\big]||$$

Since $||\nabla_{\hat{\chi}_1}Q_V^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\chi}_1)||$ is upper bounded, and we have $\hat{c}_t^1 \to \hat{c}_t^{1,*}$ given $\forall i \in \mathcal{N}$, $V^i(x_t^i,u_t^i;\chi_i^k) \to V_{\bar{\boldsymbol{\pi}}_1}^i(x_t^i,u_t^i)$, based on the analysis in the previous timescale, we have $||\delta\hat{\chi}_{1,\varepsilon}|| \to 0$, and therefore $\delta\hat{\chi}_{1,\varepsilon} \to 0$.

3. Because the $\nabla_{\hat{\chi}_1}(\hat{c}_t^1+\gamma Q_V^1(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1},\alpha_{1,t+1};\hat{\chi}_1)-Q_V^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\chi}_1))^2|_{\hat{\chi}_1=\hat{\chi}_1^k}$ is a sample of $\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\hat{\chi}_1}(\hat{c}_t^1+\gamma Q_V^1(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1},\alpha_{1,t+1};\hat{\chi}_1)-Q_V^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\chi}_1))^2|_{\hat{\chi}_1=\hat{\chi}_1^k}\big]$, we can conclude that $\mathbb{E}\big[\delta\hat{\chi}_1^{k+1}|\mathcal{F}_{\hat{\chi}_1,k}\big]=0$.

Then, according to the same analysis of $\hat{\omega}_1$, we conclude that $\hat{\chi}_1^k$ converges to a minimizer $\hat{\chi}_1^*$, namely $\hat{\chi}_1^k \to \hat{\chi}_1^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $Q_V^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\chi}_1^k) \to Q_V^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\chi}_1^*) = Q_{V_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t})$.

Convergence of θ_1: Since $\{\theta_i^k\}_{i\in\{2,\ldots,N\}}$ and $\{\lambda_i^k\}_{i\in\{1,\ldots,N\}}$ are updated on slower timescales than θ_1, again, according to [138],we treat $\{\theta_i^k\}_{i\in\{2,\ldots,N\}}$ and $\{\lambda_i^k\}_{i\in\{1,\ldots,N\}}$ as fixed values when update θ_1, and additionally, according to aforementioned analysis we have $||Q^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\phi_1^k)-Q_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x}_t,\boldsymbol{u}_t)|| \to 0$, $||Q_H^1(\boldsymbol{x}_t,\boldsymbol{u}_t;\hat{\omega}_1^k)-Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t,\boldsymbol{u}_t)|| \to 0$, $||Q_\alpha^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\psi}_1^k)-Q_{\alpha_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t})|| \to 0$, $||Q_V^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t};\hat{\chi}_1^k)-Q_{V_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t})|| \to 0$ almost surely. With the function $Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t,\boldsymbol{u}_t)$, $Q_{\alpha_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t})$ and $Q_{V_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x}_t,\boldsymbol{u}_t,\alpha_{1,t})$, by using data following $\bar{\boldsymbol{\pi}}_1$ and extending the constraint from being based on $(\boldsymbol{x}_0,\boldsymbol{u}_0)$ to being based on general $(\boldsymbol{x},\boldsymbol{u})$, the ideal Lagrangian function can be written as:

$$\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)=\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[-Q_{\bar{\boldsymbol{\pi}}_1}^1(\boldsymbol{x},\boldsymbol{u})+\lambda_{1,1}Q_{H_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x},\boldsymbol{u})+\lambda_{1,2}Q_{V_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x},\boldsymbol{u},\alpha_1)+\lambda_{1,3}Q_{\alpha_{\bar{\boldsymbol{\pi}}_1}}^1(\boldsymbol{x},\boldsymbol{u},\alpha_1)\big]. \tag{6.47}$$

It is noteworthy that, although looks like constraints commonly used in CMDP settings, due to the application of the max function, only the states where constraints are violated will be taken into consideration, and therefore safety constraints are imposed on each state. Besides,

because of the use of barrier certificate $H^1_{\bar{\boldsymbol{\pi}}_1}$, one violation of safety constraints happening at timestep t can be recorded and reflected in all $\hat{r}^{1,*}_k, k \leq t$ through the calculation of difference (6.39), and therefore this violation can be sufficiently suppressed during the update. Later, by using the sampled data to update θ_1, we have:

$$\begin{aligned}
\theta_1^{k+1} =& \Gamma_{\Theta_1}\big[\theta_1^k - \eta_{2,1}(\nabla_{u^1}(-Q^1(\boldsymbol{x},\boldsymbol{u};\phi_1^k) + \lambda_{1,1}^k Q_H^1(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_1^k) + \lambda_{1,2}^k Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k) \\
&+ \lambda_{1,3}^k Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k))|_{u^1=u^1(\boldsymbol{x};\theta_1^k)}\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k} + \nabla_{\alpha_1}(\lambda_{1,2}^k Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k) \\
&+ \lambda_{1,3}^k Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k))|_{\alpha_1=\alpha_1(\boldsymbol{x};\theta_1^k)}\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k})\big] \\
=& \Gamma_{\Theta_1}\big[\theta_1^k - \eta_{2,1}[\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^k} + \delta\theta_1^{k+1} + \delta\theta_{1,\varepsilon}]\big]
\end{aligned} \tag{6.48}$$

where

$$\begin{aligned}
\delta\theta_1^{k+1} =& -\mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{u^1}(-Q^1(\boldsymbol{x},\boldsymbol{u};\phi_1^k) + \lambda_{1,1}^k Q_H^1(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_1^k) + \lambda_{1,2}^k Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k) \\
&+ \lambda_{1,3}^k Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k))|_{u^1=u^1(\boldsymbol{x};\theta_1^k)}\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k} + \nabla_{\alpha_1}(\lambda_{1,2}^k Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k) \\
&+ \lambda_{1,3}^k Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k))|_{\alpha_1=\alpha_1(\boldsymbol{x};\theta_1^k)}\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k}\big] \\
&+ \nabla_{u^1}(-Q^1(\boldsymbol{x},\boldsymbol{u};\phi_1^k) + \lambda_{1,1}^k Q_H^1(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_1^k) + \lambda_{1,2}^k Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k) \\
&+ \lambda_{1,3}^k Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k))|_{u^1=u^1(\boldsymbol{x};\theta_1^k)}\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k} + \nabla_{\alpha_1}(\lambda_{1,2}^k Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k) \\
&+ \lambda_{1,3}^k Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k))|_{\alpha_1=\alpha_1(\boldsymbol{x};\theta_1^k)}\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k}
\end{aligned} \tag{6.49}$$

and

$$\begin{aligned}
\delta\theta_{1,\varepsilon} =& \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{u^1}(-Q^1(\boldsymbol{x},\boldsymbol{u};\phi_1^k) + \lambda_{1,1}^k Q_H^1(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_1^k) + \lambda_{1,2}^k Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k) \\
&+ \lambda_{1,3}^k Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k))|_{u^1=u^1(\boldsymbol{x};\theta_1^k)}\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k} + \nabla_{\alpha_1}(\lambda_{1,2}^k Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k) \\
&+ \lambda_{1,3}^k Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k))|_{\alpha_1=\alpha_1(\boldsymbol{x};\theta_1^k)}\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k} \\
&- \nabla_{u^1}(-Q^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x},\boldsymbol{u}) + \lambda_{1,1}^k Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u}) + \lambda_{1,2}^k Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1) + \lambda_{1,3}^k Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1))|_{u^1=u^1(\boldsymbol{x};\theta_1^k)} \\
&\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k} - \nabla_{\alpha_1}(\lambda_{1,2}^k Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1) + \lambda_{1,3}^k Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1))|_{\alpha_1=\alpha_1(\boldsymbol{x};\theta_1^k)}\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k}\big]
\end{aligned} \tag{6.50}$$

Under Assumption 6.7:

1. We show that $\delta\theta_1^{k+1}$ is square integrable first. To be specific,

$$\begin{aligned}
&\mathbb{E}\big[||\delta\theta_1^{k+1}||^2|\mathcal{F}_{\theta_1,k}\big] \leq C_7||\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k}||_\infty^2 \times (||\nabla_{u^1}Q^1(\boldsymbol{x},\boldsymbol{u};\phi_1^k)||_\infty^2 \\
&+ ||\lambda_{1,1}||_\infty^2||\nabla_{u^1}Q_H^1(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_1^k)||_\infty^2 + ||\lambda_{1,2}||_\infty^2||\nabla_{u^1}Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k)||_\infty^2 + ||\lambda_{1,3}||_\infty^2||\nabla_{u^1}Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k)||_\infty^2) \\
&+ C_8||\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k}||_\infty^2 \times (||\lambda_{1,2}||_\infty^2||\nabla_{\alpha_1}Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k)||_\infty^2 + ||\lambda_{1,3}||_\infty^2||\nabla_{\alpha_1}Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k)||_\infty^2)
\end{aligned} \tag{6.51}$$

where $\mathcal{F}_{\theta_1,k} = \sigma(\theta_1^m, \delta\theta_1^m, m \leq k)$ is the filtration of θ_1^k generated by different independent trajectories. Based on Assumption 6.7, we know:

$$\begin{aligned}
||\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k}||_\infty^2 &\leq K_4(1+||\theta_i^k||_\infty^2) \\
||\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta_1^k}||_\infty^2 &\leq K_5(1+||\theta_i^k||_\infty^2) \\
||\nabla_{u^1}Q^1(\boldsymbol{x},\boldsymbol{u};\phi_1^k)||_\infty^2 &\leq K_6(1+||u^1||_\infty^2) \\
||\nabla_{u^1}Q_H^1(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_1^k)||_\infty^2 &\leq K_7(1+||u^1||_\infty^2) \\
||\nabla_{u^1}Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k)||_\infty^2 &\leq K_8(1+||u^1||_\infty^2) \\
||\nabla_{u^1}Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k)||_\infty^2 &\leq K_9(1+||u^1||_\infty^2) \\
||\nabla_{\alpha_1}Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k)||_\infty^2 &\leq K_{10}(1+||\alpha_1||_\infty^2) \\
||\nabla_{\alpha_1}Q_\alpha^1(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k)||_\infty^2 &\leq K_{11}(1+||\alpha_1||_\infty^2)
\end{aligned} \tag{6.52}$$

where $K_4 - K_{11}$ are constants, and $||\lambda_{1,1}||_\infty^2$, $||\lambda_{1,2}||_\infty^2$, and $||\lambda_{1,3}||_\infty^2$ are bounded. Therefore, $\mathbb{E}\big[||\delta\theta_1^{k+1}||^2|\mathcal{F}_{\theta_1,k}\big] \leq C_7K_4(1+||\theta_i^k||_\infty^2)[(K_6 + ||\lambda_{1,1}||_\infty^2K_7 + ||\lambda_{1,2}||_\infty^2K_8 + ||\lambda_{1,3}||_\infty^2K_9)(1 + ||u^1||_\infty^2)] + C_8K_5(1+||\theta_i^k||_\infty^2)[(||\lambda_{1,2}||_\infty^2K_{10} + ||\lambda_{1,3}||_\infty^2K_{11})(1+||\alpha_1||_\infty^2)] < \infty$, and $\delta\theta_1^{k+1}$ is square integrable.

2. Later, we have:

$$\delta\theta_{1,\varepsilon} = \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{u^1}(Q^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x},\boldsymbol{u}) - Q^1(\boldsymbol{x},\boldsymbol{u};\phi^k_1) + \lambda^k_{1,1}(Q^1_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}^k_1) - Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u})) + \lambda^k_{1,2}(Q^1_V(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}^k_1)$$
$$- Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1)) + \lambda^k_{1,3}(Q^1_\alpha(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}^k_1) - Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1)))|_{u^1=u^1(\boldsymbol{x};\theta^k_1)}\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta^k_1}$$
$$+ \nabla_{\alpha_1}(\lambda^k_{1,2}(Q^1_V(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}^k_1) - Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1)) + \lambda^k_{1,3}(Q^1_\alpha(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}^k_1) - Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1)))|_{\alpha_1=\alpha_1(\boldsymbol{x};\theta^k_1)}$$
$$\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta^k_1}\big]$$
$$= \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{u^1}(Q^1(\boldsymbol{x},\boldsymbol{u};\phi^*_i(\{\theta^k_i\}_{i\in\mathcal{N}})) - Q^1(\boldsymbol{x},\boldsymbol{u};\phi^k_1) + \lambda^k_{1,1}(Q^1_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}^k_1) - Q^1_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}^*_1(\{\theta^k_i\}_{i\in\mathcal{N}})))$$
$$+ \lambda^k_{1,2}(Q^1_V(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}^k_1) - Q^1_V(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}^*_1(\{\theta^k_i\}_{i\in\mathcal{N}}))) + \lambda^k_{1,3}(Q^1_\alpha(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}^k_1)$$
$$- Q^1_\alpha(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}^*_1(\{\theta^k_i\}_{i\in\mathcal{N}})))|_{u^1=\pi^1(\boldsymbol{x};\theta^k_1)}\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta^k_1} + \nabla_{\alpha_1}(\lambda^k_{1,2}(Q^1_V(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}^k_1)$$
$$- Q^1_V(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}^*_1(\{\theta^k_i\}_{i\in\mathcal{N}}))) + \lambda^k_{1,3}(Q^1_\alpha(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}^k_1) - Q^1_\alpha(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}^*_1(\{\theta^k_i\}_{i\in\mathcal{N}})))|_{\alpha_1=\alpha_1(\boldsymbol{x};\theta^k_1)}$$
$$\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta^k_1}\big]$$
$$\leq \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\theta_1}u^1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta^k_1}](K_{12}||\phi^k_1 - \phi^*_1||_\infty + K_{13}||\hat{\omega}^k_1 - \hat{\omega}^*_1||_\infty + K_{14}||\hat{\chi}^k_1 - \hat{\chi}^*_1||_\infty + K_{15}||\hat{\psi}^k_1 - \hat{\psi}^*_1||_\infty)$$
$$+ \mathbb{E}_{\bar{\boldsymbol{\pi}}_1}\big[\nabla_{\theta_1}\alpha_1(\boldsymbol{x};\theta_1)|_{\theta_1=\theta^k_1}](K_{16}||\hat{\chi}^k_1 - \hat{\chi}^*_1||_\infty + K_{17}||\hat{\psi}^k_1 - \hat{\psi}^*_1||_\infty) \to 0, \tag{6.53}$$

where $K_{12} - K_{17}$ are constants, since in the previous timescale, we showed the convergence of $\phi^k_1 \to \phi^*_1(\{\theta^k_i\}_{i\in\mathcal{N}})$, $\hat{\omega}^k_1 \to \hat{\omega}^*_1(\{\theta^k_i\}_{i\in\mathcal{N}})$,$\hat{\chi}^k_1 \to \hat{\chi}^*_1(\{\theta^k_i\}_{i\in\mathcal{N}})$, and $\hat{\psi}^k_1 \to \hat{\psi}^*_1(\{\theta^k_i\}_{i\in\mathcal{N}})$ when $k \to \infty$.

3. Because $\nabla_{\theta_1}(-Q^1(\boldsymbol{x},\boldsymbol{u};\phi^k_1)+\lambda^k_{1,1}Q^1_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}^k_1)+\lambda^k_{1,2}Q^1_V(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}^k_1)+\lambda^k_{1,3}Q^1_\alpha(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}^k_1))$ is a sample of $\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta^k_1}$, we have that $\mathbb{E}\big[\delta\theta^{k+1}_1|\mathcal{F}_{\theta_1,k}\big] = 0$

Due to these three facts, the policy θ_1 update is a stochastic approximation of a continuous system $\theta_i(t)$ [138] represented by:

$$\dot{\theta}_1(t) = \Upsilon_{\Theta_1}\big[-\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)\big] \tag{6.54}$$

where

$$\Upsilon_{\Theta_1}\big[F(\theta_1)\big] \triangleq \lim_{\eta\to 0^+}\frac{\Gamma_{\Theta_1}(\theta_1 + \eta F(\theta_1)) - \Gamma_{\Theta_1}(\theta_1)}{\eta} \tag{6.55}$$

is the left directional derivative of the function $\Gamma_{\Theta_1}(\theta_1)$ in the direction of $F(\theta_1)$. According to [137], we have:

$$\frac{d\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)}{dt} = \nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)^T\Upsilon_{\Theta_1}\big[-\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)\big] \leq 0 \tag{6.56}$$

which is non-zero if $||\Upsilon_{\Theta_1}\big[-\nabla_{\theta_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)\big]|| \neq 0$. Therefore, for given fixed $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\mathcal{N}}$, define a Lyapunov function:

$$L_{\{\theta_i^k\}_{i\in\{2,\dots,N\}},\{\lambda_i^k\}_{i\in\mathcal{N}}}(\theta_1) = \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1^k) - \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*,\lambda_1^k) \tag{6.57}$$

where θ_1^* is a local minimum point. Then, there exists a ball centered at θ_1^* with a radius r_{θ_1} such that $\forall\theta_1 \in B_{r_{\theta_1}}(\theta_1^*) = \{\theta_1 \mid ||\theta_1-\theta_1^*|| \leq r_{\theta_1}\}$, $L_{\{\theta_i^k\}_{i\in\{2,\dots,N\}},\{\lambda_i^k\}_{i\in\mathcal{N}}}(\theta_1) \geq 0$. Given $d\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)/dt \leq 0$, according to the Lyapunov theory for stable systems [139], for any initial $\theta_1(0) \in B_{r_{\theta_1}}(\theta_1^*)$, the trajectory $\theta_1(t)$ converges to θ_1^*, and thus $\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*,\lambda_1) \leq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1(t),\lambda_1) \leq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1(0),\lambda_1)$, $\forall t \geq 0$. Based on all aforementioned analysis and [137, Proposition 17], as well as the fact that θ_1^k is bounded in set Θ_1, according to [138, Theorem 2, Chapter 6], the sequence $\{\theta_1^k\}$ converges almost surely to the locally minimum point θ_1^*.

Convergence of λ_1: Since λ_1 is updated at a slower timescale than previously analyzed parameters, we have that, given fixed $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{2,\dots,N\}}$, $||\theta_1^k-\theta_1^*(\lambda_1^k)|| = 0$, $||Q^1(\boldsymbol{x},\boldsymbol{u};\phi_1^k) - Q^1_{\bar{\boldsymbol{\pi}}_1}(\boldsymbol{x},\boldsymbol{u})|| = 0$, $||Q^1_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_1^k) - Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u})|| = 0$, $||Q^1_\alpha(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k) - Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1)|| \to 0$, and $||Q^1_V(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k) - Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x},\boldsymbol{u},\alpha_1)|| \to 0$. With the continuity of $\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)$, we have $||\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^k,\lambda_1=\lambda_1^k} - \nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1^k),\lambda_1=\lambda_1^k}|| = 0$ almost surely. Thus, the update of λ_1 based on the sampled data is:

$$\begin{aligned}\lambda_1^{k+1} = \begin{bmatrix}\lambda_{1,1}^{k+1}\\ \lambda_{1,2}^{k+1}\\ \lambda_{1,3}^{k+1}\end{bmatrix} &= \Gamma_{\Lambda_1}\Big[\begin{bmatrix}\lambda_{1,1}^{k}\\ \lambda_{1,2}^{k}\\ \lambda_{1,3}^{k}\end{bmatrix} + \eta_{3,1}\begin{bmatrix}Q^1_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_1^k)\\ Q^1_V(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\chi}_1^k)\\ Q^1_\alpha(\boldsymbol{x},\boldsymbol{u},\alpha_1;\hat{\psi}_1^k)\end{bmatrix}\Big]\\ &= \Gamma_{\Lambda_1}\Big[\begin{bmatrix}\lambda_{1,1}^{k}\\ \lambda_{1,2}^{k}\\ \lambda_{1,3}^{k}\end{bmatrix} + \eta_{3,1}\Big[\begin{bmatrix}\nabla_{\lambda_{1,1}}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)\\ \nabla_{\lambda_{1,2}}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)\\ \nabla_{\lambda_{1,3}}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)\end{bmatrix}\Big|_{\theta_1=\theta_1^*(\lambda_1^k),\lambda_1=\lambda_1^k} + \delta\lambda_1^{k+1}\Big]\Big]\end{aligned} \tag{6.58}$$

where

$$\delta\lambda_1^{k+1} = \begin{bmatrix} \delta\lambda_{1,1}^{k+1} \\ \delta\lambda_{1,2}^{k+1} \\ \delta\lambda_{1,3}^{k+1} \end{bmatrix} = \begin{bmatrix} -\mathbb{E}\big[Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] + Q^1_H(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}); \hat{\omega}_1^k) \\ -\mathbb{E}\big[Q^1_{V_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] + Q^1_V(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}); \hat{\chi}_1^k) \\ -\mathbb{E}\big[Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] + Q^1_\alpha(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}); \hat{\psi}_1^k) \end{bmatrix}$$
$$= \begin{bmatrix} -\mathbb{E}\big[Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] + Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x})) - Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x})) + Q^1_H(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}); \hat{\omega}_1^k) \\ -\mathbb{E}\big[Q^1_{V_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] + Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x})) - Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x})) + Q^1_V(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}); \hat{\chi}_1^k) \\ -\mathbb{E}\big[Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] + Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x})) - Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x})) + Q^1_\alpha(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}); \hat{\psi}_1^k) \end{bmatrix} \tag{6.59}$$

Here, $\bar{\boldsymbol{\pi}}_1^*$ is the joint policy parameterized by $\{\theta_1^*(\lambda_1^k), \theta_2^k, \ldots, \theta_N^k\}$ where $\theta_1^*(\lambda_1^k)$ is the locally minimum solution given a fixed λ_1^k, and $\{\theta_i^k\}_{i\in\{2,\ldots,N\}}$ are certain policies for other agents. Similarly, here we have for $p \in \{1, 2, 3\}$:

$$\mathbb{E}\big[||\delta\lambda_{1,p}^{k+1}||^2 | \mathcal{F}_{\lambda_{1,p},k}\big] < \infty \tag{6.60}$$

according to Assumption 6.5, where $\mathcal{F}_{\lambda_{1,p},k} = \sigma(\lambda_{1,p}^m, \delta\lambda_{1,p}^m, m \leq k)$ is the filtration of $\lambda_{1,p}^k$ generated by different independent trajectories, and since $||Q^1_H(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}); \hat{\omega}_1^k) - Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}))|| = 0$, $||Q^1_V(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}); \hat{\chi}_1^k) - Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}))|| = 0$, and $||Q^1_\alpha(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}); \hat{\psi}_1^k) - Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}))|| = 0$, and also because $Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}))$ is a sample of $\nabla_{\lambda_{1,1}} \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1, \lambda_1)|_{\theta_1=\theta_1^*(\lambda_1^k), \lambda_1=\lambda_1^k}$, $Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}))$ is a sample of $\nabla_{\lambda_{1,2}} \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1, \lambda_1)|_{\theta_1=\theta_1^*(\lambda_1^k), \lambda_1=\lambda_1^k}$, and $Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_1(\boldsymbol{x}))$ is a sample of $\nabla_{\lambda_{1,3}} \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1, \lambda_1)|_{\theta_1=\theta_1^*(\lambda_1^k), \lambda_1=\lambda_1^k}$, we have $\mathbb{E}\big[\delta\lambda_{1,p}^{k+1} | \mathcal{F}_{\lambda_{1,p},k}\big] = 0$, $\forall p \in \{1, 2, 3\}$, almost surely. Thus, similar to previous analysis, the λ_1's update is a stochastic approximation of the following system:

$$\dot{\lambda}_1(t) = \Upsilon_{\Lambda_1}\big[\nabla_{\lambda_1} \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1, \lambda_1)|_{\theta_1=\theta_1^*(\lambda_1)}\big]. \tag{6.61}$$

Similarly, we have:

$$\frac{d\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1), \lambda_1)}{dt} = \nabla_{\lambda_1} \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1, \lambda_1)|_{\theta_1=\theta_1^*(\lambda_1)}{}^T \Upsilon_{\Lambda_1}\big[\nabla_{\lambda_1} \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1, \lambda_1)|_{\theta_1=\theta_1^*(\lambda_1)}\big] \geq 0 \tag{6.62}$$

For a local maximum point λ_1^*, define the following Lyapunov function:

$$L_{\{\theta_i^k\}_{i\in\{2,\dots,N\}},\{\lambda_i^k\}_{i\in\{2,\dots,N\}}}(\lambda_1) = \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1^*),\lambda_1^*) - \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1),\lambda_1), \tag{6.63}$$

there exists a ball centered at λ_1^* with a radius r_{λ_1} such that $\forall\lambda_1 \in B_{r_{\lambda_1}}(\lambda_1^*) = \{\lambda_1 \mid ||\lambda_1 - \lambda_1^*|| \leq r_{\lambda_1}\}$, $L_{\{\theta_i^k\}_{i\in\{2,\dots,N\}},\{\lambda_i^k\}_{i\in\{2,\dots,N\}}}(\lambda_1) \geq 0$. Furthermore,

$$dL_{\{\theta_i^k\}_{i\in\{2,\dots,N\}},\{\lambda_i^k\}_{i\in\{2,\dots,N\}}}(\lambda_1)/dt = -d\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1),\lambda_1)/dt \leq 0. \tag{6.64}$$

Therefore, starting from $\lambda_1(0) \in B_{r_{\lambda_1}}(\lambda_1^*)$, the trajectory $\lambda_1(t)$ converges to λ_1^*, which is a locally maximum point. Following the similar process in the update of θ_1, we conclude that $\{\lambda_1^k\}$ converges almost surely to λ_1^*, with $\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1^*),\lambda_1^*) \geq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1),\lambda_1)$, $\forall t \geq 0$.

Further, we show that, with the policy of agent 1 being parameterized by $\theta_1^*(\lambda_1^*)$, $\mathbb{E}\big[Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] = \mathbb{E}\big[Q^1_{V_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] = \mathbb{E}\big[Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] = 0$. Recall that $\Upsilon_{\Lambda_1}\big[\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1),\lambda_1=\lambda_1^*}\big] = 0$ based on previous analysis, suppose there exist $(\boldsymbol{x},\boldsymbol{u},\alpha_1)$ generated by $\bar{\boldsymbol{\pi}}_1^*$ such that $\mathbb{E}\big[Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] \neq 0$ or $\mathbb{E}\big[Q^1_{V_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] \neq 0$ or $\mathbb{E}\big[Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] \neq 0$, then, this implies that $\exists p \in \{1,2,3\}$ such that for $\lambda_{1,p}^* \in [0,\lambda_{1,p}^{\max})$, we have $\Gamma_{\Lambda_1}\big[\lambda_{1,p}^* + \eta_{3,1}\nabla_{\lambda_{1,p}}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1),\lambda_1=\lambda_1^*}\big] = \lambda_{1,p}^* + \eta_{3,1}\mathbb{E}\big[Q_q^1(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big]$ for some sufficiently small $\eta_{3,1} > 0$ and $q \in \{H_{\bar{\boldsymbol{\pi}}_1^*}, V_{\bar{\boldsymbol{\pi}}_1^*}, \alpha_{\bar{\boldsymbol{\pi}}_1^*}\}$, since $\mathbb{E}\big[Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] \geq 0$, or $\mathbb{E}\big[Q^1_{V_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] \geq$, or $\mathbb{E}\big[Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] \geq 0$ according to their definitions given $\hat{r}_t^i$, $\hat{d}_t^i$, and $\hat{c}_t^i$ are all non-negative. Therefore, $\Upsilon_{\Lambda_1}\big[\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1),\lambda_1=\lambda_1^*}\big] \neq 0$, which contradicts with the fact that $\Upsilon_{\Lambda_1}\big[\nabla_{\lambda_1}\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1,\lambda_1)|_{\theta_1=\theta_1^*(\lambda_1),\lambda_1=\lambda_1^*}\big] = 0$. Thus, $\mathbb{E}\big[Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] = \mathbb{E}\big[Q^1_{V_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] = \mathbb{E}\big[Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] = 0$, when the policy is parameterized by $\theta_1^*(\lambda_1^*)$, and

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1^*),\lambda_1^*) &= \mathbb{E}_{\bar{\boldsymbol{\pi}}_1^*}\big[-Q^1_{\bar{\boldsymbol{\pi}}_1^*}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x})) + \lambda_{1,1}^* Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x})) \\
&\quad + \lambda_{1,2}^* Q^1_{V_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x})) + \lambda_{1,3}^* Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] \\
&\geq \mathbb{E}_{\bar{\boldsymbol{\pi}}_1^*}\big[-Q^1_{\bar{\boldsymbol{\pi}}_1^*}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x})) + \lambda_{1,1} Q^1_{H_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x})) \\
&\quad + \lambda_{1,2} Q^1_{V_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x})) + \lambda_{1,3} Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1^*}}(\boldsymbol{x},\bar{\boldsymbol{\pi}}_1^*(\boldsymbol{x}))\big] \\
&= \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1^*),\lambda_1)
\end{aligned} \tag{6.65}$$

Therefore, $\mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1^*), \lambda_1) \leq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1^*(\lambda_1^*), \lambda_1^*) \leq \mathcal{L}_{\boldsymbol{\theta}^{-1}}(\theta_1, \lambda_1^*)$, and thus $(\theta_1^*, \lambda_1^*)$ is a locally saddle point for the certain fixed $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{2,\dots,N\}}$, and the update of θ_1 and λ_1 will converge to the feasible locally optimal policy and multiplier with respect to its value function for the certain fixed set of policies $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and multipliers $\{\lambda_i^k\}_{i\in\{2,\dots,N\}}$ of other agents that have not been updated.

Later we analysis the update of the second agent. For the agent 2, according to the previous analysis, it is already obtained that $\lambda_1^k \to \lambda_1^*(\{\theta_i^k\}_{i\in\{2,\dots,N\}}, \{\lambda_i^k\}_{i\in\{2,\dots,N\}})$ and $\theta_1^k \to \theta_1^*(\{\theta_i^k\}_{i\in\{2,\dots,N\}}, \{\lambda_i^k\}_{i\in\{2,\dots,N\}})$ when we update θ_2^k and λ_2^k. Since agent 2 is updated at a quicker timescale than agent 3 to agent N, here we still have $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$ as fixed sets. Let $\bar{\boldsymbol{\pi}}_2 = (\theta_1^k, \theta_2^k, \dots, \theta_N^k)$ where $\theta_1^k \to \theta_1^*(\theta_2^k, \lambda_2^k)$ and $\lambda_1^k \to \lambda_1^*(\theta_2^k, \lambda_2^k)$, then, similar to the previous proof, we have $\phi_2^k \to \phi_2^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $Q^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \phi_2^k) \to Q^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \phi_2^*) = Q^2_{\bar{\boldsymbol{\pi}}_2}(\boldsymbol{x}_t, \boldsymbol{u}_t)$, $\omega_2^k \to \omega_2^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $H^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_2^k) \to H^2(\boldsymbol{x}_t, \boldsymbol{u}_t; \omega_2^*) = H^2_{\bar{\boldsymbol{\pi}}_2}(\boldsymbol{x}_t, \boldsymbol{u}_t)$. Further, define $\hat{r}_t^2$, $\hat{d}_t^2$,$\hat{c}_t^2$ and $Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}$, $Q^2_{V_{\bar{\boldsymbol{\pi}}_2}}$, $Q^2_{\alpha_{\bar{\boldsymbol{\pi}}_2}}$ similarly to $\hat{r}_t^1$, $\hat{d}_t^1$, $\hat{c}_t^1$ and $Q^1_{H_{\bar{\boldsymbol{\pi}}_1}}$, $Q^1_{V_{\bar{\boldsymbol{\pi}}_1}}$, $Q^1_{\alpha_{\bar{\boldsymbol{\pi}}_1}}$, and we use $\hat{\omega}_2$, $\hat{\chi}_2$, $\hat{\psi}_2$ to parameterize $Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}$, $Q^2_{V_{\bar{\boldsymbol{\pi}}_2}}$, $Q^2_{\alpha_{\bar{\boldsymbol{\pi}}_2}}$, respectively, under Assumption 6.5, we can conclude that $\hat{\omega}_2^k \to \hat{\omega}_2^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ where $Q^2_H(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_2^k) \to Q^2_H(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_2^*) = Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \boldsymbol{u}_t)$, and similarly $\hat{\chi}_2^k \to \hat{\chi}_2^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ with $Q^2_V(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{2,t}; \hat{\chi}_2^k) \to Q^2_V(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{2,t}; \hat{\chi}_2^*) = Q^2_{V_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{2,t})$, and $\hat{\psi}_2^k \to \hat{\psi}_2^*(\{\theta_i^k\}_{i\in\mathcal{N}})$ with $Q^2_\alpha(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{2,t}; \hat{\psi}_2^k) \to Q^2_\alpha(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{2,t}; \hat{\psi}_2^*) = Q^2_{\alpha_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \boldsymbol{u}_t, \alpha_{2,t})$. Then, we can analyze the convergence of θ_2 and λ_2.

Convergence of θ_2: For the convergence of θ_2, define $\bar{\boldsymbol{\pi}}_2^* = (\theta_1^*, \{\theta_i^k\}_{i\in\{2,\dots,N\}})$ which differs from $\bar{\boldsymbol{\pi}}_2 = (\theta_1^k, \theta_2^k, \dots, \theta_N^k)$ by θ_1^* and θ_1^k. Based on previous analysis, here we have $\theta_1^k \to \theta_1^*$ and therefore $\bar{\boldsymbol{\pi}}_2 \to \bar{\boldsymbol{\pi}}_2^*$ for θ_2^k which is updated in this timescale under certain fixed $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$. The ideal Lagrangian function to use now is:

$$\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2, \lambda_2) = \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[- Q^2_{\bar{\boldsymbol{\pi}}_2^*}(\boldsymbol{x}, \boldsymbol{u}) + \lambda_{2,1} Q^2_{H_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x}, \boldsymbol{u}) + \lambda_{2,2} Q^2_{V_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_2) + \lambda_{2,3} Q^2_{\alpha_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_2)\big]. \tag{6.66}$$

where $\boldsymbol{\theta}^{-2}$ means θ_2 is updated, and during its update, $\theta_1^k = \theta_1^*(\{\theta_i^k\}_{i\in\{2,\dots,N\}}, \{\lambda_i^k\}_{i\in\{2,\dots,N\}})$ with $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$ being certain fixed values. Then, by sampling data from

the distribution to update θ_2, we have:

$$\begin{aligned}\theta_2^{k+1} =&\Gamma_{\Theta_2}\big[\theta_2^k - \eta_{2,2}(\nabla_{u^2}(-Q^2(\boldsymbol{x},\boldsymbol{u};\phi_2^k) + \lambda_{2,1}^k Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k) + \lambda_{2,2}^k Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)\\ &+ \lambda_{2,3}^k Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k))|_{u^2=u^2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k} + \nabla_{\alpha_2}(\lambda_{2,2}^k Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)\\ &+ \lambda_{2,3}^k Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k))|_{\alpha_2=\alpha_2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k})\big]\\ =&\Gamma_{\Theta_2}\big[\theta_2^k - \eta_{2,2}[\nabla_{\theta_2}\boldsymbol{\mathcal{L}}_{\boldsymbol{\theta}^{-2}}(\theta_2,\lambda_2)|_{\theta_2=\theta_2^k} + \delta\theta_2^{k+1} + \delta\theta_{2,\varepsilon}]\big]\end{aligned} \tag{6.67}$$

where

$$\begin{aligned}\delta\theta_2^{k+1} =& -\mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{u^2}(-Q^2(\boldsymbol{x},\boldsymbol{u};\phi_2^k) + \lambda_{2,1}^k Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k) + \lambda_{2,2}^k Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)\\ &+ \lambda_{2,3}^k Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k))|_{u^2=u^2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k} + \nabla_{\alpha_2}(\lambda_{2,2}^k Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)\\ &+ \lambda_{2,3}^k Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k))|_{\alpha_2=\alpha_2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}\big]\\ &+ \nabla_{u^2}(-Q^2(\boldsymbol{x},\boldsymbol{u};\phi_2^k) + \lambda_{2,1}^k Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k) + \lambda_{2,2}^k Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}2^k)\\ &+ \lambda_2^k Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k))|_{u^2=u^2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k} + \nabla_{\alpha_2}(\lambda_{2,2}^k Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)\\ &+ \lambda_{2,3}^k Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k))|_{\alpha_2=\alpha_2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}\end{aligned} \tag{6.68}$$

and

$$\begin{aligned}\delta\theta_{2,\varepsilon} =& \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{u^2}(-Q^2(\boldsymbol{x},\boldsymbol{u};\phi_2^k) + \lambda_{2,1}^k Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k) + \lambda_{2,2}^k Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)\\ &+ \lambda_{2,3}^k Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k))|_{u^2=u^2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k} + \nabla_{\alpha_2}(\lambda_{2,2}^k Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)\\ &+ \lambda_{2,3}^k Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k))|_{\alpha_2=\alpha_2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}\\ &- \nabla_{u^2}(-Q_{\bar{\boldsymbol{\pi}}_2^*}^2(\boldsymbol{x},\boldsymbol{u}) + \lambda_{2,1}^k Q_{H_{\bar{\boldsymbol{\pi}}_2^*}}^2(\boldsymbol{x},\boldsymbol{u}) + \lambda_{2,2}^k Q_{V_{\bar{\boldsymbol{\pi}}_2^*}}^2(\boldsymbol{x},\boldsymbol{u},\alpha_2) + \lambda_{2,3}^k Q_{\alpha_{\bar{\boldsymbol{\pi}}_2^*}}^1(\boldsymbol{x},\boldsymbol{u},\alpha_2))|_{u^2=u^2(\boldsymbol{x};\theta_2^k)}\\ &\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k} - \nabla_{\alpha_2}(\lambda_{2,2}^k Q_{V_{\bar{\boldsymbol{\pi}}_2^*}}^2(\boldsymbol{x},\boldsymbol{u},\alpha_2) + \lambda_{2,3}^k Q_{\alpha_{\bar{\boldsymbol{\pi}}_2^*}}^2(\boldsymbol{x},\boldsymbol{u},\alpha_2))|_{\alpha_2=\alpha_2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}\big]\end{aligned} \tag{6.69}$$

1. We have the following:

$$\mathbb{E}\big[||\delta\theta_2^{k+1}||^2|\mathcal{F}_{\theta_2,k}\big] \leq C_9||\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}||_\infty^2 \times (||\nabla_{u^2}Q^2(\boldsymbol{x},\boldsymbol{u};\phi_2^k)||_\infty^2$$
$$+ ||\lambda_{2,1}||_\infty^2||\nabla_{u^2}Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k)||_\infty^2 + ||\lambda_{2,2}||_\infty^2||\nabla_{u^2}Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)||_\infty^2 + ||\lambda_{2,3}||_\infty^2||\nabla_{u^2}Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k)||_\infty^2)$$
$$+ C_{10}||\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}||_\infty^2 \times (||\lambda_{2,2}||_\infty^2||\nabla_{\alpha_2}Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)||_\infty^2 + ||\lambda_{2,3}||_\infty^2||\nabla_{\alpha_2}Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k)||_\infty^2) \tag{6.70}$$

where $\mathcal{F}_{\theta_2,k} = \sigma(\theta_2^m, \delta\theta_2^m, m \leq k)$ is the filtration of θ_2^k generated by different independent trajectories. Similar to the analysis of $\delta\theta_1^{k+1}$, we know $\mathbb{E}\big[||\delta\theta_2^{k+1}||^2|\mathcal{F}_{\theta_2,k}\big] < \infty$, and thus $\delta\theta_2^{k+1}$ is square integrable.

2. Under Assumption 6.8, we have:

$$\delta\theta_{2,\varepsilon} = \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{u^2}(Q^2_{\bar{\boldsymbol{\pi}}_2^*}(\boldsymbol{x},\boldsymbol{u}) - Q^2(\boldsymbol{x},\boldsymbol{u};\phi_2^k) + \lambda_{2,1}^k(Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k) - Q^2_{H_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x},\boldsymbol{u})) + \lambda_{2,2}^k(Q_V^1(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)$$
$$- Q^2_{V_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x},\boldsymbol{u},\alpha_2)) + \lambda_{2,3}^k(Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k) - Q^2_{\alpha_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x},\boldsymbol{u},\alpha_2)))|_{u^2=u^2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}$$
$$+ \nabla_{\alpha_2}(\lambda_{2,2}^k(Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k) - Q^2_{V_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x},\boldsymbol{u},\alpha_2)) + \lambda_{2,3}^k(Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k) - Q^2_{\alpha_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x},\boldsymbol{u},\alpha_2)))|_{\alpha_2=\alpha_2(\boldsymbol{x};\theta_2^k)}$$
$$\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}\big]. \tag{6.71}$$

Here we choose some terms as examples for a simpler derivation.

$$\mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{u^2}(Q^2_{\bar{\boldsymbol{\pi}}_2^*}(\boldsymbol{x},\boldsymbol{u}) - Q^2(\boldsymbol{x},\boldsymbol{u};\phi_2^k) + \lambda_{2,1}^k(Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k) - Q^2_{H_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x},\boldsymbol{u})))|_{u^2=u^2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}$$
$$+ \nabla_{\alpha_2}(\lambda_{2,2}^k(Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k) - Q^2_{V_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x},\boldsymbol{u},\alpha_2)))|_{\alpha_2=\alpha_2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}\big]$$
$$= \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{u^2}(Q^2_{\bar{\boldsymbol{\pi}}_2^*}(\boldsymbol{x},\boldsymbol{u}) - Q^2_{\bar{\boldsymbol{\pi}}_2}(\boldsymbol{x},\boldsymbol{u}) + Q^2_{\bar{\boldsymbol{\pi}}_2}(\boldsymbol{x},\boldsymbol{u}) - Q^2(\boldsymbol{x},\boldsymbol{u};\phi_2^k) + \lambda_{2,1}^k(Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k) - Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x},\boldsymbol{u})$$
$$+ Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x},\boldsymbol{u}) - Q^2_{H_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x},\boldsymbol{u})))|_{u^2=u^2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k} + \nabla_{\alpha_2}(\lambda_{2,2}^k(Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)$$
$$- Q^2_{V_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x},\boldsymbol{u},\alpha_2) + Q^2_{V_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x},\boldsymbol{u},\alpha_2) - Q^2_{V_{\bar{\boldsymbol{\pi}}_2^*}}(\boldsymbol{x},\boldsymbol{u},\alpha_2)))|_{\alpha_2=\alpha_2(\boldsymbol{x};\theta_2^k)}\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}\big]$$
$$\leq \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{\theta_2}u^2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}](K_{18}||\bar{\boldsymbol{\pi}}_2^* - \bar{\boldsymbol{\pi}}_2||_\infty + K_{19}||\phi_2^k - \phi_2^*||_\infty + K_{20}||\hat{\omega}_2^k - \hat{\omega}_2^*||_\infty)$$
$$+ \mathbb{E}_{\bar{\boldsymbol{\pi}}_2^*}\big[\nabla_{\theta_2}\alpha_2(\boldsymbol{x};\theta_2)|_{\theta_2=\theta_2^k}](K_{21}||\bar{\boldsymbol{\pi}}_2^* - \bar{\boldsymbol{\pi}}_2||_\infty + K_{22}||\hat{\chi}_2^k - \hat{\chi}_2^*||_\infty) \to 0. \tag{6.72}$$

The addition of remaining terms such as Q_α^2 does not change the logic of this derivation, and thus we have $\delta\theta_{2,\varepsilon} \to 0$.

3. Since $\nabla_{\theta_2}(-Q^2(\boldsymbol{x},\boldsymbol{u};\phi_2^k)+\lambda_{2,1}^k Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k)+\lambda_{2,2}^k Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)+\lambda_{2,3}^k Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k))$ is a sample of $\nabla_{\theta_2}\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2,\lambda_2)|_{\theta_2=\theta_2^k}$ given $\bar{\boldsymbol{\pi}}_2 \to \bar{\boldsymbol{\pi}}_2^*$, one can obtain that $\mathbb{E}\big[\delta\theta_2^{k+1}|\mathcal{F}_{\theta_2,k}\big]=0$ almost surely.

Then, similar to the previous section using Lyapunov function, we conclude that the sequence $\{\theta_2^k\}$ converges to a locally minimum point θ_2^* such that $\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2^*,\lambda_2) \le \mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2(t),\lambda_2) \le \mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2(0),\lambda_2),\ \forall t \ge 0$.

Convergence of λ_2: Similar to the previous section on λ_1, we can treat $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$ as fixed when update λ_2, and $||\theta_2^k-\theta_2^*||=0$. Then, the update of λ_2 can be written as:

$$
\begin{aligned}
\lambda_2^{k+1} = \begin{bmatrix}\lambda_{2,1}^{k+1}\\ \lambda_{2,2}^{k+1}\\ \lambda_{2,3}^{k+1}\end{bmatrix} &= \Gamma_{\Lambda_2}\Big[\begin{bmatrix}\lambda_{2,1}^{k}\\ \lambda_{2,2}^{k}\\ \lambda_{2,3}^{k}\end{bmatrix} + \eta_{3,2}\begin{bmatrix} Q_H^2(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_2^k)\\ Q_V^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\chi}_2^k)\\ Q_\alpha^2(\boldsymbol{x},\boldsymbol{u},\alpha_2;\hat{\psi}_2^k)\end{bmatrix}\Big]\\
&=\Gamma_{\Lambda_2}\Big[\begin{bmatrix}\lambda_{2,1}^{k}\\ \lambda_{2,2}^{k}\\ \lambda_{2,3}^{k}\end{bmatrix} + \eta_{3,2}\Big[\begin{bmatrix}\nabla_{\lambda_{2,1}}\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2,\lambda_2)\\ \nabla_{\lambda_{2,2}}\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2,\lambda_2)\\ \nabla_{\lambda_{2,3}}\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2,\lambda_2)\end{bmatrix}\Big|_{\theta_2=\theta_2^*(\lambda_2^k),\lambda_2=\lambda_2^k} + \delta\lambda_2^{k+1}\Big]\Big]
\end{aligned}
\tag{6.73}
$$

where

$$\delta\lambda_2^{k+1} = \begin{bmatrix} \delta\lambda_{2,1}^{k+1} \\ \delta\lambda_{2,2}^{k+1} \\ \delta\lambda_{2,3}^{k+1} \end{bmatrix} = \begin{bmatrix} -\mathbb{E}\big[Q^2_{H_{\bar{\boldsymbol{\pi}}^*_{2,*}}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}^*_{2,*}(\boldsymbol{x}))\big] + Q^2_H(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}); \hat{\omega}_2^k) \\ -\mathbb{E}\big[Q^2_{V_{\bar{\boldsymbol{\pi}}^*_{2,*}}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}^*_{2,*}(\boldsymbol{x}))\big] + Q^2_V(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}); \hat{\chi}_2^k) \\ -\mathbb{E}\big[Q^2_{\alpha_{\bar{\boldsymbol{\pi}}^*_{2,*}}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}^*_{2,*}(\boldsymbol{x}))\big] + Q^2_\alpha(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}); \hat{\psi}_2^k) \end{bmatrix}$$
$$= \begin{bmatrix} -\mathbb{E}\big[Q^2_{H_{\bar{\boldsymbol{\pi}}^*_{2,*}}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}^*_{2,*}(\boldsymbol{x}))\big] + Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x})) - Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x})) + Q^2_H(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}); \hat{\omega}_2^k) \\ -\mathbb{E}\big[Q^2_{V_{\bar{\boldsymbol{\pi}}^*_{2,*}}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}^*_{2,*}(\boldsymbol{x}))\big] + Q^2_{V_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x})) - Q^2_{V_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x})) + Q^2_V(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}); \hat{\chi}_2^k) \\ -\mathbb{E}\big[Q^2_{\alpha_{\bar{\boldsymbol{\pi}}^*_{2,*}}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}^*_{2,*}(\boldsymbol{x}))\big] + Q^2_{\alpha_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x})) - Q^2_{\alpha_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x})) + Q^2_\alpha(\boldsymbol{x}, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}); \hat{\psi}_2^k) \end{bmatrix} \tag{6.74}$$

where $\bar{\boldsymbol{\pi}}^*_{2,*} = (\theta_1^*, \theta_2^*(\lambda_2^k), \{\theta_i^k\}_{i\in\{3,\dots,N\}})$, $\bar{\boldsymbol{\pi}}_2 = (\theta_1^k, \theta_2^k, \{\theta_i^k\}_{i\in\{3,\dots,N\}})$ where $\theta_1^k \to \theta_1^*$ and $\theta_2^k \to \theta_2^*(\lambda_2^k)$ according to previous analysis. Therefore, similarly, here we have for $p \in \{1,2,3\}$:

$$\mathbb{E}\big[||\delta\lambda_{2,p}^{k+1}||^2 | \mathcal{F}_{\lambda_{2,p},k}\big] < \infty, \tag{6.75}$$

according to Assumption 6.5. Since based on aforementioned analysis, $||Q^2_H(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t); \hat{\omega}_2^k) - Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t))|| \to 0$, and $Q^2_{H_{\bar{\boldsymbol{\pi}}_2}}(\boldsymbol{x}_t, \bar{\boldsymbol{\pi}}_2(\boldsymbol{x}_t))$ is a sample of $\nabla_{\lambda_2}\mathcal{L}_{\boldsymbol{\theta}^{-2}}(\theta_2, \lambda_2)|_{\theta_2=\theta_2^*(\lambda_2^k),\lambda_2=\lambda_2^k}$, and such properties also apply to Q^2_V and Q^2_α, we have $\mathbb{E}\big[\delta\lambda_{2,p}^{k+1} | \mathcal{F}_{\lambda_{2,p},k}\big] = 0$, $\forall p \in \{1,2,3\}$, almost surely.

Then, similar to the previous section, we have that λ_2^k converges to λ_2^* which is a local maximum point, and then, $(\theta_2^*, \lambda_2^*)$ is a locally saddle point for the certain fixed $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$ of other agents that have not been updated, while θ_1 and λ_1 have already converged to the local optimal policy and multiplier for the agent 1 with respect to $\{\theta_i^k\}_{i\in\{2,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{2,\dots,N\}}$. Namely, we finally obtain (θ_1^*, θ_2^*) and $(\lambda_1^*, \lambda_2^*)$ with respect to $\{\theta_i^k\}_{i\in\{3,\dots,N\}}$ and $\{\lambda_i^k\}_{i\in\{3,\dots,N\}}$. This process will iterate until the agent N finishes its update, and we have a feasible joint policy $\{\theta_i^*\}_{i\in\mathcal{N}}$, where for each i, θ_i^* is a feasible policy. Furthermore, considering this obtained joint policy $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$ parameterized by $\{\theta_i^*\}_{i\in\mathcal{N}}$, denote $\boldsymbol{\theta}^{-i,*}$ as the obtained joint policy except agent i, then, consider another joint policy $\boldsymbol{\pi}^*_{-i}$ parameterized by $\{\theta_i, \boldsymbol{\theta}^{-i,*}\}$

where $\theta_i \in B_{r_{\theta_i}}(\theta_i^*) \backslash \theta_i^*$ is a feasible policy, we have :

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\pi}^*}\big[-Q^i_{\boldsymbol{\pi}^*}(\boldsymbol{x},\boldsymbol{u})\big] = \mathbb{E}_{\boldsymbol{\pi}^*}\big[-Q^i_{\boldsymbol{\pi}^*}(\boldsymbol{x},\boldsymbol{u}) + \lambda^*_{i,1}Q^i_{H_{\boldsymbol{\pi}^*}}(\boldsymbol{x},\boldsymbol{u}) + \lambda^*_{i,2}Q^i_{V_{\boldsymbol{\pi}^*}}(\boldsymbol{x},\boldsymbol{u},\alpha_i) + \lambda^*_{i,3}Q^i_{\alpha_{\boldsymbol{\pi}^*}}(\boldsymbol{x},\boldsymbol{u},\alpha_i)\big] \\
&= \mathcal{L}_{\boldsymbol{\theta}^{-i,*}}(\theta_i^*,\lambda_i^*) \leq \mathcal{L}_{\boldsymbol{\theta}^{-i,*}}(\theta_i,\lambda_i^*) = \mathbb{E}_{\boldsymbol{\pi}^*_{-i}}\big[-Q^i_{\boldsymbol{\pi}^*_{-i}}(\boldsymbol{x},\boldsymbol{u}) + \lambda^*_{i,1}Q^i_{H_{\boldsymbol{\pi}^*_{-i}}}(\boldsymbol{x},\boldsymbol{u}) + \lambda^*_{i,2}Q^i_{V_{\boldsymbol{\pi}^*_{-i}}}(\boldsymbol{x},\boldsymbol{u},\alpha_i) \\
&+ \lambda^*_{i,3}Q^i_{\alpha_{\boldsymbol{\pi}^*_{-i}}}(\boldsymbol{x},\boldsymbol{u},\alpha_i)\big] = \mathbb{E}_{\boldsymbol{\pi}^*_{-i}}\big[-Q^i_{\boldsymbol{\pi}^*_{-i}}(\boldsymbol{x},\boldsymbol{u})\big],
\end{aligned} \tag{6.76}$$

since $\theta_i^* = \underset{\theta_i \in B_{r_{\theta_i}}(\theta_i^*)}{\arg\min}\ \mathcal{L}_{\boldsymbol{\theta}^{-i,*}}(\theta_i,\lambda_i^*)$. Thus, $\mathbb{E}_{\boldsymbol{\pi}^*}\big[Q^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t)\big] \geq \mathbb{E}_{\boldsymbol{\pi}^*_{-i}}\big[Q^i_{\boldsymbol{\pi}^*_{-i}}(\boldsymbol{x}_t,\boldsymbol{u}_t)\big]$ for all feasible policy $\theta_i \in B_{r_{\theta_i}}(\theta_i^*)\backslash\theta_i^*$, $\forall i \in \mathcal{N}$. Therefore, the obtained joint policy $\boldsymbol{\pi}^* = \{\pi_i^*\}_{i\in\mathcal{N}}$ is a feasible local Nash equilibrium.

Now we show that $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) \geq 0$ for $(\boldsymbol{x}_t,\boldsymbol{u}_t)$ along the trajectory under $\boldsymbol{\pi}^*$, $\forall t \geq 0$ and $\forall i \in \mathcal{N}$: According to the definition, $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0) \leq 0$, $\forall i \in \mathcal{N}$. Consider any $i \in \mathcal{N}$ and let $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0) = p \leq 0$ with $\boldsymbol{u}_0$ being any control signal generated by the joint policy at $\boldsymbol{x}_0$, according to the barrier certificate, we have $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_1,\boldsymbol{u}_1) \geq (1-\epsilon)H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0)$ for all following $(\boldsymbol{x}_1,\boldsymbol{u}_1)$. Furthermore, according to the definition of barrier certificate, $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0) = b^i_0 + \gamma\mathbb{E}_{\boldsymbol{x}_1,\boldsymbol{u}_1}[H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_1,\boldsymbol{u}_1)] \geq (1-\epsilon)\gamma p$ given $b^i_0 = 0$. Thus, we have $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_0,\boldsymbol{u}_0) = p \geq (1-\epsilon)\gamma p$ at convergence. Since $p \leq 0$ and $1-\epsilon \in (0,1)$, $\gamma \in (0,1)$, we have $p = 0$, for all $i \in \mathcal{N}$ and all possible generated control signal $\boldsymbol{u}_0$. The similar process applies for all $t \geq 0$, and we have $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) \geq 0$, $\forall t \geq 0$ and $\forall i \in \mathcal{N}$.

Furthermore, for a state $\boldsymbol{x}_t$ along the trajectory under the control policy $\boldsymbol{\pi}^*$, let $\boldsymbol{x}_{t+1}$ represent the next state following $\boldsymbol{u}_t$ generated by $\boldsymbol{\pi}^*$, and let i_A and i_B denote the index of the maximum $H_{\boldsymbol{\pi}^*}$ at $(\boldsymbol{x}_t,\boldsymbol{u}_t)$ and $(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1})$, respectively, namely:

$$H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) = \min_{i\in\mathcal{N}} H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) = H^{i_A}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t,\boldsymbol{u}_t) \tag{6.77}$$

$$H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1}) = \min_{i\in\mathcal{N}} H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1}) = H^{i_B}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1},\boldsymbol{u}_{t+1}) \tag{6.78}$$

Then, given $1-\epsilon \in \mathcal{K}_\infty$, we have:

$$\begin{aligned} H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) &= H^{i_B}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) \geq (1-\epsilon) H^{i_B}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) \\ &\geq (1-\epsilon) H^{i_A}_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) = (1-\epsilon) H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t), \end{aligned} \tag{6.79}$$

since $\forall i \in \mathcal{N}$ and $\forall t \geq 0$, $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) \geq (1-\epsilon) H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ for all $\boldsymbol{u}_t$ generated by $\boldsymbol{\pi}^*$. Besides, it is easily known that $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) \geq 0$ implies that $\forall i \in \mathcal{N}$, $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) \geq 0$, which means safety is maintained for all agents. Additionally, $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}) < 0$ if $\exists i \in \mathcal{N}$ such that $H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t) < 0$, which means at least one agent will experience safety violation with a positive probability. Therefore, $H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}_t, \boldsymbol{u}_t)$ is a global barrier certificate for the whole multi-agent system, and the corresponding safe set is given as:

$$\mathcal{X}^g := \{\boldsymbol{x} \mid H^g_{\boldsymbol{\pi}^*}(\boldsymbol{x}, \boldsymbol{\pi}^*(\boldsymbol{x})) \geq 0\} = \cap_{i \in \mathcal{N}} \{\boldsymbol{x} \mid H^i_{\boldsymbol{\pi}^*}(\boldsymbol{x}, \boldsymbol{\pi}^*(\boldsymbol{x})) \geq 0\}. \tag{6.80}$$

Then, for each agent i, because the final converged individual policy π^*_i satisfies both the safety and stability constraints, and the joint control signal $\boldsymbol{u}$, generated by $\boldsymbol{\pi}^*$, is a function of states, the whole multi-agent system becomes an autonomous system without external control input. In the absence of exogenous signal $\boldsymbol{d}_t$, we have $||\boldsymbol{x}_t|| \leq \xi(||\boldsymbol{x}_0||, t)$, which further implies that $\lim_{t \to \infty} ||\boldsymbol{x}_t|| = 0$. Therefore, the whole multi-agent system converges to the desired equilibrium (goal), and so does each subsystem (agent). □

Remark 6.7. *The graph topology and the multi-timescale learning hierarchy play different roles in the proposed framework. The graph topology is primarily used to define the communication structure, local information available to each agent, and the inter-agent coupling terms appearing in the ISS-based stability constraints. Consequently, the graph influences the construction of the constrained optimization problem and the detailed form of several assumptions, such as Assumption 6.5. By contrast, the convergence analysis focuses on how the resulting constrained optimization problem is solved and is primarily driven by the imposed multi-timescale stochastic approximation framework. Therefore, the graph topology affects convergence indirectly through the optimization problem being solved, rather than directly through the convergence mechanism itself. The update ordering adopted in this chapter is introduced for analytical tractability and is not specifically*

designed according to the underlying graph topology.

Remark 6.8. *The scalability analysis and convergence analysis in this chapter address different questions. Theorem 6.3 studies the error introduced when global information is replaced by local observations within a neighbourhood of size k. Consequently, the dimensions of the policy and critic inputs, together with the communication burden among agents, depend primarily on the selected neighbourhood size rather than on the full system size. By contrast, the convergence analysis focuses on how the resulting constrained optimization problem is solved and is primarily driven by the multi-timescale stochastic approximation framework. Therefore, the scalability contribution of this chapter should be interpreted mainly in terms of information, communication, and computational efficiency, rather than a reduction in the complexity of the convergence proof itself.*

6.5.4 D. Formal Version and Proof for Theorem 6.3

Theorem 6.5. *Suppose that Assumption 6.1, 6.2 and 6.9 hold, then, under the multi-timescale update scheme used in Algorithm 6.1, if sizes of all signals, e.g., reward, barrier and cost signals, are upper bounded for each agent, we have:*

$$\|\nabla_{\theta_i} J^{real}_{\theta_i}(\theta_i) - \nabla_{\theta_i} \tilde{J}_{\theta_i}(\theta_i)\|_2 \leq E_{i,1} e^{-lk} + E_{i,2} e^{-wk} = \mathcal{O}(e^{-\min\{l,w\}k}), \tag{6.81}$$

where $E_{i,1}$ and $E_{i,2}$ are constants.

Proof. Similar to the proof of Theorem 6.2, by using Q^i_H Q^i_V and Q^i_α to represent different constraint terms, and use $(\boldsymbol{x}, \boldsymbol{u})$, where $\boldsymbol{x} = (\boldsymbol{x}_{N^i_{k,t}}, \boldsymbol{x}_{N^{-i}_{k,t}})$, $\boldsymbol{u} = (\boldsymbol{u}_{N^i_{k,t}}, \boldsymbol{u}_{N^{-i}_{k,t}})$, and $(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})$, where $\breve{\boldsymbol{x}} = (\boldsymbol{x}_{N^i_{k,t}}, \breve{\boldsymbol{x}}_{N^{-i}_{k,t}})$, $\breve{\boldsymbol{u}} = (\boldsymbol{u}_{N^i_{k,t}}, \breve{\boldsymbol{u}}_{N^{-i}_{k,t}})$, to define two different state-action pairs, then, in the multi-timescale update scheme, we have:

$$J^{\text{real}}_{\theta_i}(\theta_i) = \mathbb{E}\big[- Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) + \lambda_{i,1} Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}) + \lambda_{i,2} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) + \lambda_{i,3} Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x}))\big] \tag{6.82}$$

while with $(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})$, which does not reflect the true state-action pair of further agents outside

$N^i_{k,t}$, and parameterization, we similarly have the surrogate function:

$$\tilde{J}_{\theta_i}(\theta_i) = \mathbb{E}\big[- Q^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \phi_i) + \lambda_{i,1} Q^i_H(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \hat{\omega}_i) + \lambda_{i,2} Q^i_V(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}); \hat{\chi}_i) + \lambda_{i,3} Q^i_\alpha(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}); \hat{\psi}_i)\big]. \tag{6.83}$$

Therefore,

$$\begin{aligned}
&\|\nabla_{\theta_i} J^{\text{real}}_{\theta_i}(\theta_i) - \nabla_{\theta_i} \tilde{J}_{\theta_i}(\theta_i)\|_2 = \|\mathbb{E}\big[\nabla_{\theta_i}(Q^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \phi_i) - Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})) + \nabla_{\theta_i}\lambda_{i,1}(Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}) - Q^i_H(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \hat{\omega}_i)) \\
&+ \nabla_{\theta_i}\lambda_{i,2}(Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) - Q^i_V(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}); \hat{\chi}_i)) + \nabla_{\theta_i}\lambda_{i,3}(Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) - Q^i_\alpha(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}); \hat{\psi}_i))\big]\| \\
&\leq \mathbb{E}\|\nabla_{\theta_i}(Q^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \phi_i) - Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}))\| + \mathbb{E}\|\nabla_{\theta_i}\lambda_{i,1}(Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}) - Q^i_H(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \hat{\omega}_i))\| \\
&+ \mathbb{E}\|\nabla_{\theta_i}\lambda_{i,2}(Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) - Q^i_V(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}); \hat{\chi}_i))\| + \mathbb{E}\|\nabla_{\theta_i}\lambda_{i,3}(Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) - Q^i_\alpha(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}); \hat{\psi}_i))\|,
\end{aligned} \tag{6.84}$$

by using triangle inequality. Later we show that each of these three terms can be upper bounded. At first,

$$\begin{aligned}
&\mathbb{E}\|\nabla_{\theta_i}(Q^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \phi_i) - Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}))\| \\
&= \mathbb{E}\|\nabla_{\theta_i}(Q^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \phi_i) - Q^i(\boldsymbol{x}, \boldsymbol{u}; \phi_i) + Q^i(\boldsymbol{x}, \boldsymbol{u}; \phi_i) - Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}))\| \\
&\leq \underbrace{\mathbb{E}\|\nabla_{\theta_i}(Q^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \phi_i) - Q^i(\boldsymbol{x}, \boldsymbol{u}; \phi_i))\|}_{\mathcal{T}_1} + \underbrace{\mathbb{E}\|\nabla_{\theta_i}(Q^i(\boldsymbol{x}, \boldsymbol{u}; \phi_i) - Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}))\|}_{\mathcal{T}_2}
\end{aligned} \tag{6.85}$$

The second term $\mathcal{T}_2 \to 0$ according to the proof of Theorem 6.2 since when calculating the gradient with respect to θ_i, the parameter ϕ_i has converged under Assumption 6.4. For $\mathcal{T}_1$, we have:

$$\begin{aligned}
&\mathbb{E}\|\nabla_{\theta_i}(Q^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \phi_i) - Q^i(\boldsymbol{x}, \boldsymbol{u}; \phi_i))\| \\
&\leq \sup_{\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}} \|\nabla_{\theta_i}(Q^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}; \phi_i) - Q^i(\boldsymbol{x}, \boldsymbol{u}; \phi_i))\|
\end{aligned} \tag{6.86}$$

Then, according to [149], define $d_{t,i}(\boldsymbol{x}, \boldsymbol{u})$ and $\breve{d}_{t,i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})$ to be the distributions of (x^i, u^i) at time t when starting from $(\boldsymbol{x}, \boldsymbol{u})$ and $(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})$, respectively, we have $TV(d_{t,i}, \breve{d}_{t,i}) \leq (\frac{q}{2})^t e^{-lk}$ under Assumption 6.1 where TV denotes total variation distance. Therefore, with $\|\cdot\|_\infty$

denoting the maximum value of the norms over the space:

$$
\begin{aligned}
&\|Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u}) - Q^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}})\|_\infty \\
&\leq \sum_{t=0}^{\infty} \|\mathbb{E}[\gamma^t r^i|\boldsymbol{x}_0=\boldsymbol{x},\boldsymbol{u}_0=\boldsymbol{u}] - \mathbb{E}[\gamma^t r^i|\boldsymbol{x}_0=\breve{\boldsymbol{x}},\boldsymbol{u}_0=\breve{\boldsymbol{u}}]\|_\infty \\
&= \sum_{t=0}^{\infty} \|\gamma^t \mathbb{E}_{d_{t,i}} r^i - \gamma^t \mathbb{E}_{\breve{d}_{t,i}} r^i\|_\infty = \sum_{t=0}^{\infty} \gamma^t \|r\|_\infty \|\mathbb{E}_{d_{t,i}} \frac{r^i}{\|r\|_\infty} - \mathbb{E}_{\breve{d}_{t,i}} \frac{r^i}{\|r\|_\infty}\|_\infty \\
&\leq \sum_{t=0}^{\infty} \gamma^t \|r\|_\infty \sup_{\|f\|_\infty \leq 1} \|\mathbb{E}_{d_{t,i}} f - \mathbb{E}_{\breve{d}_{t,i}} f\|_\infty = \sum_{t=0}^{\infty} \gamma^t \|r\|_\infty 2TV(d_{t,i},\breve{d}_{t,i}) \\
&\leq \sum_{t=0}^{\infty} \gamma^t \|r\|_\infty 2(\frac{q}{2})^t e^{-lk} = 2\|r\|_\infty e^{-lk} \sum_{t=0}^{\infty} (\frac{q\gamma}{2})^t = 2\|r\|_\infty e^{-lk} \frac{2}{2-q\gamma} \\
&= \frac{4\|r\|_\infty}{2-q\gamma} e^{-lk}.
\end{aligned}
\tag{6.87}
$$

This derivation is because $\|\frac{r^i}{\|r\|_\infty}\|_\infty \leq 1$ where $||r||_\infty$ is the maximum of the norm of the reward that can be obtained, and $TV(d_{t,i},\breve{d}_{t,i}) = \frac{1}{2}\sup_{|f|\leq 1} |\mathbb{E}_{d_{t,i}} f - \mathbb{E}_{\breve{d}_{t,i}} f| = \frac{1}{2}\sup_{\|f\|_\infty \leq 1} \|\mathbb{E}_{d_{t,i}} f - \mathbb{E}_{\breve{d}_{t,i}} f\|_\infty$ where $f = \frac{r^i}{\|r\|_\infty}$. Also, since $Q^i(\boldsymbol{x},\boldsymbol{u};\phi_i) \to Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u})$, we can further rewrite and upper bound $\mathcal{T}_1$ as follows:

$$
\begin{aligned}
\mathcal{T}_1 &\leq \|\nabla_{\theta_i} u^i(\breve{\boldsymbol{x}};\theta_i) \nabla_{u^i} Q^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}}) - \nabla_{\theta_i} u^i(\breve{\boldsymbol{x}};\theta_i) \nabla_{u^i} Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u}) \\
&\quad + \nabla_{\theta_i} u^i(\breve{\boldsymbol{x}};\theta_i) \nabla_{u^i} Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u}) - \nabla_{\theta_i} u^i(\boldsymbol{x}_t;\theta_i) \nabla_{u^i} Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u})\| \\
&\leq \|\nabla_{\theta_i} u^i(\breve{\boldsymbol{x}};\theta_i)\|_\infty \|\nabla_{u^i} Q^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}}) - \nabla_{u^i} Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u})\|_\infty + \\
&\|\nabla_{\theta_i} u^i(\breve{\boldsymbol{x}};\theta_i) - \nabla_{\theta_i} u^i(\boldsymbol{x};\theta_i)\|_\infty \|\nabla_{u^i} Q^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u})\|_\infty \\
&\leq d_1||\theta_i||_\infty d_2 \frac{4||r||_\infty}{2-q\gamma} e^{-lk} + d_3 d_4 ||u^i||_\infty v e^{-wk}
\end{aligned}
\tag{6.88}
$$

where d_1, d_2, d_3 and d_4 are constants related to Lipschitz continuity under Assumption 6.9 given the parameter and action spaces are bounded. The second inequality comes from triangle inequality, and the third one follows from Assumption 6.1, and Assumption 6.2 that requires $\|\pi^i(\breve{\boldsymbol{x}};\theta_i) - \pi^i(\boldsymbol{x};\theta_i)\|_\infty$ to be upper bounded by ve^{-wk}, with u^i being part of the output π^i, together with Lipschitz condition under Assumption 6.9. Therefore, the first part in (6.84) is upper bounded. For simplicity, we abbreviate the upper bound as $D_1 e^{-lk} + D_2 e^{-wk}$. Then,

we derive the upper bound for the second part of (6.84). Similarly,

$$\begin{aligned}
&\mathbb{E}\|\nabla_{\theta_i}\lambda_{i,1}(Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u}) - Q^i_H(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}};\hat{\omega}_i))\| \\
&= \mathbb{E}\|\lambda_{i,1}(\nabla_{\theta_i}Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u}) - \nabla_{\theta_i}Q^i_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_i)) + \lambda_{i,1}(\nabla_{\theta_i}Q^i_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_i) - \nabla_{\theta_i}Q^i_H(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}};\hat{\omega}_i))\| \\
&\leq \underbrace{\mathbb{E}\|\lambda_{i,1}(\nabla_{\theta_i}Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u}) - \nabla_{\theta_i}Q^i_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_i))\|}_{\mathcal{T}_3} + \underbrace{\mathbb{E}\|\lambda_{i,1}(\nabla_{\theta_i}Q^i_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_i) - \nabla_{\theta_i}Q^i_H(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}};\hat{\omega}_i))\|}_{\mathcal{T}_4}
\end{aligned} \tag{6.89}$$

The term $\mathcal{T}_3 \to 0$ according to the proof in Theorem 6.2since when calculating the gradient with respect to θ_i the parameter $\hat{\omega}_i$ has converged under Assumption 6.4. For $\mathcal{T}_4$, we have:

$$\begin{aligned}
&\mathbb{E}\|\lambda_{i,1}(\nabla_{\theta_i}Q^i_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_i) - \nabla_{\theta_i}Q^i_H(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}};\hat{\omega}_i))\| \\
&\leq \sup_{\breve{\boldsymbol{x}},\breve{\boldsymbol{u}}} \|\lambda_{i,1}(\nabla_{\theta_i}Q^i_H(\boldsymbol{x},\boldsymbol{u};\hat{\omega}_i) - \nabla_{\theta_i}Q^i_H(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}};\hat{\omega}_i))\|.
\end{aligned} \tag{6.90}$$

Since for all states and actions,

$$H^i_{\bar{\boldsymbol{\pi}}_i} \leq \sum_{t=0}^{\infty}\gamma^t\|b\|_\infty = \frac{\|b\|_\infty}{1-\gamma}, \tag{6.91}$$

then, according to the definition in the proof of Theorem 6.2,

$$\hat{r}^i_t \leq (2+\epsilon)\frac{\|b\|_\infty}{1-\gamma} + \frac{1}{2}[(2+\epsilon)\frac{\|b\|_\infty}{1-\gamma}]^2 := \|\hat{r}^i_t\|_\infty \tag{6.92}$$

by using triangle inequality, where ϵ is the parameter used for the safety constraint and $\|b\|_\infty$ is the maximum of the norm of the barrier signal that can be obtained, given ω_i has converged such that $H^i(\boldsymbol{x},\boldsymbol{u};\omega_i) \to H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u})$. Furthermore, define $\hat{r}^{i'}_t$ to be the value calculated by using $H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}})$, then, for analysis, without loss of generality, let $-H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}',\boldsymbol{u}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x},\boldsymbol{u}) \geq -H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}',\breve{\boldsymbol{u}}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}})$, where the prime denotes the next state

and action, we have:

$$
\begin{aligned}
&\mathbb{E}[\max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})\} - \max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\}] \\
&\leq \mathbb{E}[-H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) + H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') - H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) + \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})] \\
&= \mathbb{E}[H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') - H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}')] + \mathbb{E}[H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})] + \epsilon\mathbb{E}[H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})] \\
&= \frac{1}{\gamma}\mathbb{E}[H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})] + \mathbb{E}[H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})] + \epsilon\mathbb{E}[H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})] \\
&= \underbrace{(\frac{1}{\gamma} - 1 + \epsilon)\mathbb{E}[H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})]}_{\mathcal{T}_5},
\end{aligned}
\tag{6.93}
$$

where $(\boldsymbol{x}', \boldsymbol{u}')$ and $(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}')$ are the next state-action pairs for $(\boldsymbol{x}, \boldsymbol{u})$ and $(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})$, respectively. The first inequality is from the fact that $\text{ReLU}(a) - \text{ReLU}(b) \leq a - b$ for $a \geq b$, and the equality is from the definition of barrier certificate and the fact that the barrier signal depends on local information. Then, we have:

$$
\begin{aligned}
\mathcal{T}_5 &\leq (\frac{1}{\gamma} - 1 + \epsilon)\|H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})\|_\infty \\
&\leq (\frac{1}{\gamma} - 1 + \epsilon)\frac{4||b||_\infty}{2 - q\gamma}e^{-lk}
\end{aligned}
\tag{6.94}
$$

Then, since:

$$
\begin{aligned}
&\mathbb{E}[\max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})\} + \max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\}] \\
&\leq \| - H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})\|_\infty + \| - H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\|_\infty \\
&\leq (4 + 2\epsilon)\frac{\|b\|_\infty}{1 - \gamma},
\end{aligned}
\tag{6.95}
$$

we have:

$$
\begin{aligned}
&\|\mathbb{E}[\max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})\}^2 - \max\{0, -H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') + H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - \epsilon H^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\}^2]\| \\
&\leq (4 + 2\epsilon)\frac{\|b\|_\infty}{1 - \gamma}(\frac{1}{\gamma} - 1 + \epsilon)\frac{4||b||_\infty}{2 - q\gamma}e^{-lk}
\end{aligned}
\tag{6.96}
$$

Finally, we have that:

$$\|\hat{r}_t^i - \hat{r}_t^{i'}\| \leq \|\mathcal{T}_5\|_\infty + \frac{1}{2} \times (4+2\epsilon)\frac{\|b\|_\infty}{1-\gamma}(\frac{1}{\gamma} - 1 + \epsilon)\frac{4||b||_\infty}{2-q\gamma}e^{-lk} = [(2+\epsilon)\frac{\|b\|_\infty}{1-\gamma} + 1](\frac{1}{\gamma} - 1 + \epsilon)\frac{4||b||_\infty}{2-q\gamma}e^{-lk}, \tag{6.97}$$

which means the difference between $\hat{r}_t^i$ and $\hat{r}_t^{i'}$ is upper bounded. For simplicity, we denote it as $\|\hat{r}_t^i - \hat{r}_t^{i'}\| \leq d_5 e^{-lk}$. Further, since $Q_H^i(\boldsymbol{x}_t, \boldsymbol{u}_t; \hat{\omega}_i) \to Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\boldsymbol{x}_t, \boldsymbol{u}_t)$, the term $\mathcal{T}_4$ can be rewritten and upper bounded according to:

$$\begin{aligned}
&\mathbb{E}\|\lambda_{i,1}(\nabla_{\theta_i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\boldsymbol{x}, \boldsymbol{u}) - \nabla_{\theta_i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}))\| \leq \|\lambda_{i,1}\|_\infty \|\nabla_{\theta_i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\boldsymbol{x}, \boldsymbol{u}) - \nabla_{\theta_i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\|_\infty \\
&= \|\lambda_{i,1}\|_\infty \|\nabla_{u^i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\boldsymbol{x}, \boldsymbol{u}) \nabla_{\theta_i} u^i(\boldsymbol{x}; \theta_i) - \nabla_{u^i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) \nabla_{\theta_i} u^i(\boldsymbol{x}; \theta_i) \\
&+ \nabla_{u^i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) \nabla_{\theta_i} u^i(\boldsymbol{x}; \theta_i) - \nabla_{u^i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) \nabla_{\theta_i} u^i(\breve{\boldsymbol{x}}; \theta_i)\|_\infty \\
&\leq \|\lambda_{i,1}\|_\infty \|\nabla_{\theta_i} u^i(\boldsymbol{x}; \theta_i)\|_\infty \|\nabla_{u^i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\boldsymbol{x}, \boldsymbol{u}) - \nabla_{u^i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\|_\infty \\
&+ \|\lambda_{i,1}\|_\infty \|\nabla_{u^i} Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\|_\infty \|\nabla_{\theta_i} u^i(\boldsymbol{x}; \theta_i) - \nabla_{\theta_i} u^i(\breve{\boldsymbol{x}}; \theta_i)\|_\infty \\
&\leq d_6 \|\theta_i\|_\infty d_7 \|Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\boldsymbol{x}, \boldsymbol{u}) - Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\|_\infty + d_8 \|u^i\|_\infty d_9 \|u^i(\boldsymbol{x}; \theta_i) - u^i(\breve{\boldsymbol{x}}; \theta_i)\|_\infty,
\end{aligned} \tag{6.98}$$

where d_6, d_7, d_8 and d_9 are constants related to Lipschitz continuity under Assumption 6.9 given the parameter and action spaces are bounded. Later, for $\|Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\boldsymbol{x}, \boldsymbol{u}) - Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\|$, we show it is upper bounded by $= \mathcal{O}(e^{-lk})$ as follows:

$$\begin{aligned}
&\|Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\boldsymbol{x}, \boldsymbol{u}) - Q_{H_{\bar{\boldsymbol{\pi}}_i}}^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\| \\
&= \|\sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{r}_t^i \mid \boldsymbol{x}(0) = \boldsymbol{x}, \boldsymbol{u}(0) = \boldsymbol{u}] - \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{r}_t^{i'} \mid \boldsymbol{x}(0) = \breve{\boldsymbol{x}}, \boldsymbol{u}(0) = \breve{\boldsymbol{u}}]\| \\
&= \|\sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{r}_t^i \mid \boldsymbol{x}(0) = \boldsymbol{x}, \boldsymbol{u}(0) = \boldsymbol{u}] - \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{r}_t^{i'} \mid \boldsymbol{x}(0) = \boldsymbol{x}, \boldsymbol{u}(0) = \boldsymbol{u}] \\
&+ \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{r}_t^{i'} \mid \boldsymbol{x}(0) = \boldsymbol{x}, \boldsymbol{u}(0) = \boldsymbol{u}] - \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{r}_t^{i'} \mid \boldsymbol{x}(0) = \breve{\boldsymbol{x}}, \boldsymbol{u}(0) = \breve{\boldsymbol{u}}]\| \\
&\leq \|\hat{r}_t^i - \hat{r}_t^{i'}\|_\infty \sum_{t=0}^{\infty} \gamma^t + \|\hat{r}_t^{i'}\|_\infty \frac{4}{2-q\gamma} e^{-lk} \\
&\leq \frac{1}{1-\gamma} d_5 e^{-lk} + \|\hat{r}_t^{i'}\|_\infty \frac{4}{2-q\gamma} e^{-lk}.
\end{aligned} \tag{6.99}$$

Under Assumption 6.2, we further have $\|\pi^i(\boldsymbol{x};\theta_i) - \pi^i(\breve{\boldsymbol{x}};\theta_i)\| \leq ve^{-wk}$, therefore, given u^i being part of the output π^i, together with (6.98) and (6.99), we have

$$\mathbb{E}\|\nabla_{\theta_i}\lambda_{i,1}(Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u}) - Q^i_H(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}};\hat{\omega}_i))\| \leq D_3e^{-lk} + D_4e^{-wk} \tag{6.100}$$

where D_3 and D_4 are constants.

Now we consider $\mathbb{E}\|\nabla_{\theta_i}\lambda_{i,2}(Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x})) - Q^i_V(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}});\hat{\chi}_i))\|$:

$$\begin{aligned}
&\mathbb{E}\|\nabla_{\theta_i}\lambda_{i,2}(Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x})) - Q^i_V(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}});\hat{\chi}_i))\| \\
&= \mathbb{E}\|\lambda_{i,2}(\nabla_{\theta_i}Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x})) - \nabla_{\theta_i}Q^i_V(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x});\hat{\chi}_i)) + \lambda_{i,2}(\nabla_{\theta_i}Q^i_V(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x}));\hat{\chi}_i) - \nabla_{\theta_i}Q^i_V(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}});\hat{\chi}_i))\| \\
&\leq \underbrace{\mathbb{E}\|\lambda_{i,2}(\nabla_{\theta_i}Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x})) - \nabla_{\theta_i}Q^i_V(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x});\hat{\chi}_i))\|}_{\mathcal{T}_6} + \underbrace{\mathbb{E}\|\lambda_{i,2}(\nabla_{\theta_i}Q^i_V(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x});\hat{\chi}_i) - \nabla_{\theta_i}Q^i_V(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}});\hat{\chi}_i))\|}_{\mathcal{T}_7}
\end{aligned} \tag{6.101}$$

Similarly, the term $\mathcal{T}_6 \to 0$ according to the proof in Theorem 6.2 since when calculating the gradient with respect to θ_i the parameter $\hat{\chi}_i$ has converged under Assumption 6.4. For $\mathcal{T}_7$, we have:

$$\begin{aligned}
&\mathbb{E}\|\lambda_{i,2}(\nabla_{\theta_i}Q^i_V(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x});\hat{\chi}_i) - \nabla_{\theta_i}Q^i_V(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}});\hat{\chi}_i))\| \\
&\leq \sup_{\breve{\boldsymbol{x}},\breve{\boldsymbol{u}}} \|\lambda_{i,2}(\nabla_{\theta_i}Q^i_V(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x});\hat{\chi}_i) - \nabla_{\theta_i}Q^i_V(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}});\hat{\chi}_i))\|.
\end{aligned} \tag{6.102}$$

When analyzing the upper bound of the error caused by different global state-action pairs $(\boldsymbol{x},\boldsymbol{u})$ and $(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}})$, consider $V^i_{\bar{\boldsymbol{\pi}}}(x^i,u^i) = \mathbb{E}_{(\boldsymbol{x}^{-i},\boldsymbol{u}^{-i})\sim\bar{\boldsymbol{\pi}}^{-i}}\left[V^{\text{full},i}_{\bar{\boldsymbol{\pi}}}(x^i,\boldsymbol{x}^{-i},u^i,\boldsymbol{u}^{-i})\right]$, where $(\boldsymbol{x}^{-i},\boldsymbol{u}^{-i})$ is the joint state-action of all other agents and driven by $\bar{\boldsymbol{\pi}}^{-i}$. $V^{\text{full},i}_{\bar{\boldsymbol{\pi}}}$ has the same definition as $V^i_{\bar{\boldsymbol{\pi}}}$ (value function associated with cost signal c^i), but with a specific global state-action pair $(\boldsymbol{x},\boldsymbol{u}) = (x^i,\boldsymbol{x}^{-i},u^i,\boldsymbol{u}^{-i})$, which is used as the input, as the start of the trajectory, instead of only using agent i's own state-action pair. This means the original Lyapunov certificate $V^i_{\bar{\boldsymbol{\pi}}}$ of agent i, when only using its own state-action (x^i,u^i) as the input and the joint control policy is a fixed $\bar{\boldsymbol{\pi}}$, can be understood as an expected value over the distribution of all other agents. Conceptually, such $V^{\text{full},i}_{\bar{\boldsymbol{\pi}}}$ is a variant of $V^i_{\bar{\boldsymbol{\pi}}}$ by chaning the input from own observation to global information, with the updating rule via Bellman operator being unchanged, and

thus the convergence analysis in the multi-timescale method remains valid. When deriving the upper bound of the error caused in this part due to the difference between global state-action pairs, we instead analyze $V_{\bar{\boldsymbol{\pi}}}^{\text{full},i}$ to account for the difference between $(\boldsymbol{x}, \boldsymbol{u})$ and $(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})$. For simplicity, in the later content we use $V_{\bar{\boldsymbol{\pi}}}^{i}(\boldsymbol{x}, \boldsymbol{u})$ to represent this variant since global state-action pair $(\boldsymbol{x}, \boldsymbol{u})$ is explicitly used as the input, and following the same analysis for (with a little bit abuse) χ_i via the Bellman operator and its update on a faster timescale, we have $V^i(\boldsymbol{x}, \boldsymbol{u}; \chi_i^k) \to V^i(\boldsymbol{x}, \boldsymbol{u}; \chi_i^*) = V_{\bar{\boldsymbol{\pi}}}^{i}(\boldsymbol{x}, \boldsymbol{u})$.

Later, since for all states and actions,

$$V_{\bar{\boldsymbol{\pi}}_i}^{i}(\boldsymbol{x}, \boldsymbol{u}) \leq \sum_{t=0}^{\infty} \gamma^t \|c\|_\infty = \frac{\|c\|_\infty}{1-\gamma}, \tag{6.103}$$

then, according to the similar definition in the proof of Theorem 6.2,

$$\hat{c}_t^i \leq (2 + \|\alpha_i\|_\infty + \sum_{j \neq i} \delta_{ij}) \frac{\|c\|_\infty}{1-\gamma} + \frac{1}{2}[(2 + \|\alpha_i\|_\infty + \sum_{j \neq i} \delta_{ij}) \frac{\|c\|_\infty}{1-\gamma}]^2 := \|\hat{c}_t^i\|_\infty \tag{6.104}$$

by using triangle inequality, where $\|\alpha_i\|_\infty$ and $\|c\|_\infty$ are the maximum of the norm of α_i and the stability cost signal that can be obtained, and given $V^i(\boldsymbol{x}, \boldsymbol{u}; \chi_i) \to V_{\bar{\boldsymbol{\pi}}_i}^{i}(\boldsymbol{x}, \boldsymbol{u})$. Here δ_{ij} are treated as one hyperparameter (coefficient) to construct proportional functions. Furthermore, define $\hat{c}_t^{i'}$ to be the value calculated by using $V_{\bar{\boldsymbol{\pi}}_i}^{i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})$, then, again, for analysis, without loss of generality, let $V_{\bar{\boldsymbol{\pi}}_i}^{i}(\boldsymbol{x}', \boldsymbol{u}') - V_{\bar{\boldsymbol{\pi}}_i}^{i}(\boldsymbol{x}, \boldsymbol{u}) + \alpha_i(\boldsymbol{x}) V_{\bar{\boldsymbol{\pi}}_i}^{i}(\boldsymbol{x}, \boldsymbol{u}) - \sum_{j \neq i} \delta_{ij} V_{\bar{\boldsymbol{\pi}}_i}^{j}(\boldsymbol{x}, \boldsymbol{u}) \geq V_{\bar{\boldsymbol{\pi}}_i}^{i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') - V_{\bar{\boldsymbol{\pi}}_i}^{i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) + \alpha_i(\breve{\boldsymbol{x}}) V_{\bar{\boldsymbol{\pi}}_i}^{i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - \sum_{j \neq i} \delta_{ij} V_{\bar{\boldsymbol{\pi}}_i}^{j}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})$, where the prime denotes the next state and

action, we have:

$$
\begin{aligned}
&\mathbb{E}[\max\{0, V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}') - V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) + \alpha_i(\boldsymbol{x}) V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \sum_{j \neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})\} - \max\{0, V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') \\
&- V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) + \alpha_i(\breve{\boldsymbol{x}}) V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - \sum_{j \neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\}] \\
&\leq \mathbb{E}[V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}') - V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') - (V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})) - \sum_{j \neq i} \delta_{ij} (V^j_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - V^j_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}))] \\
&+ \mathbb{E}[\alpha_i(\boldsymbol{x}) V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \alpha_i(\breve{\boldsymbol{x}}) V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})] \\
&= \underbrace{\mathbb{E}[(\frac{1}{\gamma} - 1)(V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})) - \sum_{j \neq i} \delta_{ij} (V^j_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - V^j_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}))] + \mathbb{E}[\alpha_i(\boldsymbol{x}) V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \alpha_i(\breve{\boldsymbol{x}}) V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})]}_{\mathcal{T}_8},
\end{aligned}
\tag{6.105}
$$

Then, we have:

$$
\begin{aligned}
\mathcal{T}_8 &\leq (\frac{1}{\gamma} - 1 + \sum_{j \neq i} \delta_{ij}) \| V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) \|_\infty + \| \alpha_i(\boldsymbol{x}) V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \alpha_i(\breve{\boldsymbol{x}}) V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) \|_\infty \\
&= (\frac{1}{\gamma} - 1 + \sum_{j \neq i} \delta_{ij}) \frac{4||c||_\infty}{2 - q\gamma} e^{-lk} + \| \alpha_i(\boldsymbol{x}) V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \alpha_i(\boldsymbol{x}) V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) + \alpha_i(\boldsymbol{x}) V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - \alpha_i(\breve{\boldsymbol{x}}) V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) \|_\infty \\
&\leq (\frac{1}{\gamma} - 1 + \sum_{j \neq i} \delta_{ij}) \frac{4||c||_\infty}{2 - q\gamma} e^{-lk} + \| \alpha_i(\boldsymbol{x}) \|_\infty \| V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) \|_\infty + \| \alpha_i(\boldsymbol{x}) - \alpha_i(\breve{\boldsymbol{x}}) \|_\infty \| V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) \|_\infty \\
&= (\frac{1}{\gamma} - 1 + \sum_{j \neq i} \delta_{ij}) \frac{4||c||_\infty}{2 - q\gamma} e^{-lk} + \| \alpha_i \|_\infty \frac{4||c||_\infty}{2 - q\gamma} e^{-lk} + \frac{\|c\|_\infty}{1 - \gamma} v e^{-wk} \\
&= (\frac{1}{\gamma} - 1 + \sum_{j \neq i} \delta_{ij} + \| \alpha_i \|_\infty) \frac{4||c||_\infty}{2 - q\gamma} e^{-lk} + \frac{\|c\|_\infty}{1 - \gamma} v e^{-wk},
\end{aligned}
\tag{6.106}
$$

which follows a similar process used to analyze $\mathcal{T}_5$, given α_i being part of the output π^i. Then,

since:

$$\begin{aligned}
&\mathbb{E}[\max\{0, V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}') - V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) + \alpha_i(\boldsymbol{x}) V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \sum_{j \neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})\} + \max\{0, V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') \\
&- V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) + \alpha_i(\breve{\boldsymbol{x}}) V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - \sum_{j \neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\}] \\
&\leq (4 + 2\|\alpha_i\|_\infty + 2 \sum_{j \neq i} \delta_{ij}) \frac{\|c\|_\infty}{1 - \gamma},
\end{aligned} \tag{6.107}$$

we have:

$$\begin{aligned}
&\|\mathbb{E}[\max\{0, V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}', \boldsymbol{u}') - V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) + \alpha_i(\boldsymbol{x}) V^i_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u}) - \sum_{j \neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(\boldsymbol{x}, \boldsymbol{u})\}^2 - \max\{0, V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}', \breve{\boldsymbol{u}}') \\
&- V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) + \alpha_i(\breve{\boldsymbol{x}}) V^i_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}) - \sum_{j \neq i} \delta_{ij} V^j_{\bar{\boldsymbol{\pi}}_i}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}})\}^2]\| \\
&\leq (4 + 2\|\alpha_i\|_\infty + 2 \sum_{j \neq i} \delta_{ij}) \frac{\|c\|_\infty}{1 - \gamma} \big[(\frac{1}{\gamma} - 1 + \sum_{j \neq i} \delta_{ij} + \|\alpha_i\|_\infty) \frac{4||c||_\infty}{2 - q\gamma} e^{-lk} + \frac{\|c\|_\infty}{1 - \gamma} v e^{-wk}\big]
\end{aligned} \tag{6.108}$$

Finally, we have that:

$$\|\hat{c}^i_t - \hat{c}^{i'}_t\| \leq \big[(\frac{1}{\gamma} - 1 + \sum_{j \neq i} \delta_{ij} + \|\alpha_i\|_\infty) \frac{4||c||_\infty}{2 - q\gamma} e^{-lk} + \frac{\|c\|_\infty}{1 - \gamma} v e^{-wk}\big] \big[(2 + \|\alpha_i\|_\infty + \sum_{j \neq i} \delta_{ij}) \frac{\|c\|_\infty}{1 - \gamma} + 1\big] \tag{6.109}$$

which means the difference between $\hat{c}^i_t$ and $\hat{c}^{i'}_t$ is upper bounded. For simplicity, we denote it as $\|\hat{c}^i_t - \hat{c}^{i'}_t\| \leq d_6 e^{-lk} + d_7 e^{-wk}$. Further, since $Q^i_V(\boldsymbol{x}, \boldsymbol{u}, \alpha_i; \hat{\omega}_i) \to Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i)$, the term

$\mathcal{T}_7$ can be rewritten and upper bounded according to:

$$
\begin{aligned}
&\mathbb{E}\|\lambda_{i,2}(\nabla_{\theta_i} Q_V^i(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x}); \hat{\chi}_i) - \nabla_{\theta_i} Q_V^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}); \hat{\chi}_i))\| \leq \|\lambda_{i,2}\|_\infty \|\nabla_{\theta_i} Q_V^i(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x}); \hat{\chi}_i) \\
&- \nabla_{\theta_i} Q_V^i(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}); \hat{\chi}_i)\|_\infty \\
&= \|\lambda_{i,2}\|_\infty \|\nabla_{u^i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) \nabla_{\theta_i} u^i(\boldsymbol{x}; \theta_i) + \nabla_{\alpha_i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) \nabla_{\theta_i} \alpha_i(\boldsymbol{x}; \theta_i) \\
&- \nabla_{u^i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}) \nabla_{\theta_i} u^i(\breve{\boldsymbol{x}}, ; \theta_i) - \nabla_{\alpha_i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}) \nabla_{\theta_i} \alpha_i(\breve{\boldsymbol{x}}; \theta_i)\|_\infty \\
&= \|\lambda_{i,2}\|_\infty \|\nabla_{u^i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) \nabla_{\theta_i} u^i(\boldsymbol{x}; \theta_i) - \nabla_{u^i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) \nabla_{\theta_i} u^i(\breve{\boldsymbol{x}}; \theta_i) \\
&+ \nabla_{u^i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) \nabla_{\theta_i} u^i(\breve{\boldsymbol{x}}; \theta_i) - \nabla_{u^i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}) \nabla_{\theta_i} u^i(\breve{\boldsymbol{x}}; \theta_i) \\
&+ \nabla_{\alpha_i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) \nabla_{\theta_i} \alpha_i(\boldsymbol{x}; \theta_i) - \nabla_{\alpha_i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) \nabla_{\theta_i} \alpha_i(\breve{\boldsymbol{x}}; \theta_i) \\
&+ \nabla_{\alpha_i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) \nabla_{\theta_i} \alpha_i(\breve{\boldsymbol{x}}; \theta_i) - \nabla_{\alpha_i} Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}) \nabla_{\theta_i} \alpha_i(\breve{\boldsymbol{x}}; \theta_i)\|_\infty \\
&\leq d_8 \|u^i(\boldsymbol{x}; \theta_i) - u^i(\breve{\boldsymbol{x}}; \theta_i)\|_\infty + d_9 \|Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) - Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}))\|_\infty \\
&+ d_{10} \|\alpha_i(\boldsymbol{x}; \theta_i) - \alpha_i(\breve{\boldsymbol{x}}; \theta_i)\|_\infty + d_{11} \|Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) - Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}))\|_\infty,
\end{aligned}
\tag{6.110}
$$

according to assumptions on boundedness and Lipschitz, where d_{10}, d_{11}, d_{12} and d_{13} are constants. Later, for $\|Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) - Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}))\|$, we show it is upper bounded as follows:

$$
\begin{aligned}
&\|Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x}, \boldsymbol{u}, \alpha_i(\boldsymbol{x})) - Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\breve{\boldsymbol{x}}, \breve{\boldsymbol{u}}, \alpha_i(\breve{\boldsymbol{x}}))\| \\
&= \|\sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{c}_t^i \mid \boldsymbol{x}(0) = \boldsymbol{x}, \boldsymbol{u}(0) = \boldsymbol{u}, \alpha_i(0) = \alpha_i(\boldsymbol{x})] - \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{c}_t^{i'} \mid \boldsymbol{x}(0) = \breve{\boldsymbol{x}}, \boldsymbol{u}(0) = \breve{\boldsymbol{u}}, \alpha_i(0) = \alpha_i(\breve{\boldsymbol{x}})]\| \\
&= \|\sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{c}_t^i \mid \boldsymbol{x}(0) = \boldsymbol{x}, \boldsymbol{u}(0) = \boldsymbol{u}, \alpha_i(0) = \alpha_i(\boldsymbol{x})] - \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{c}_t^{i'} \mid \boldsymbol{x}(0) = \boldsymbol{x}, \boldsymbol{u}(0) = \boldsymbol{u}, \alpha_i(0) = \alpha_i(\boldsymbol{x})] \\
&+ \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{c}_t^{i'} \mid \boldsymbol{x}(0) = \boldsymbol{x}, \boldsymbol{u}(0) = \boldsymbol{u}, \alpha_i(0) = \alpha_i(\boldsymbol{x})] - \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[\hat{c}_t^{i'} \mid \boldsymbol{x}(0) = \breve{\boldsymbol{x}}, \boldsymbol{u}(0) = \breve{\boldsymbol{u}}, \alpha_i(0) = \alpha_i(\breve{\boldsymbol{x}})]\| \\
&\leq \|\hat{c}_t^i - \hat{c}_t^{i'}\|_\infty \sum_{t=0}^{\infty} \gamma^t + \|\hat{c}_t^{i'}\|_\infty \frac{4}{2 - q\gamma} e^{-lk} \leq \frac{1}{1-\gamma}(d_6 e^{-lk} + d_7 e^{-wk}) + \|\hat{c}_t^{i'}\|_\infty \frac{4}{2 - q\gamma} e^{-lk} \\
&= d_{12} e^{-lk} + d_{13} e^{-wk}
\end{aligned}
\tag{6.111}
$$

Under Assumption 6.2, we further have $\|\pi^i(\boldsymbol{x}; \theta_i) - \pi^i(\breve{\boldsymbol{x}}; \theta_i)\| \leq v e^{-wk}$, therefore, given u^i

and α_i being part of the output π^i, together with (6.110) and (6.111), we have

$$\mathbb{E}\|\nabla_{\theta_i}\lambda_{i,2}(Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x}))-Q^i_V(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}});\hat{\chi}_i))\|\leq D_5e^{-lk}+D_6e^{-wk} \tag{6.112}$$

For the final part $\mathbb{E}\|\nabla_{\theta_i}\lambda_{i,3}(Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x}))-Q^i_\alpha(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}});\hat{\psi}_i))\|$ in (6.84), a similar analysis follows, and we have:

$$\|\hat{d}^i_t-\hat{d}^{i'}_t\|\leq\big[(1+\beta_i)\sum_{j\neq i}\delta_{ij}+\|\alpha_i\|_\infty\big]\frac{\|c\|_\infty}{1-\gamma}+1\big]\big[(1+\beta_i)\sum_{j\neq i}\delta_{ij}+\|\alpha_i\|_\infty\big]\frac{4\|c\|_\infty}{2-q\gamma}e^{-lk}+\frac{\|c\|_\infty}{1-\gamma}ve^{-wk}\big] \tag{6.113}$$

For simplicity, we denote it as $\|\hat{d}^i_t-\hat{d}^{i'}_t\|\leq d_{14}e^{-lk}+d_{15}e^{-wk}$. Also, $\|Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x}))-Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}}))\|$ can be similar proved to be upper bounded by $d_{16}e^{-lk}+d_{17}e^{-wk}$, and therefore, we can obtain $\mathbb{E}\|\nabla_{\theta_i}\lambda_{i,3}(Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}(\boldsymbol{x},\boldsymbol{u},\alpha_i(\boldsymbol{x}))-Q^i_\alpha(\breve{\boldsymbol{x}},\breve{\boldsymbol{u}},\alpha_i(\breve{\boldsymbol{x}});\hat{\psi}_i))\|\leq D_7e^{-lk}+D_8e^{-wk}$.

In sum, under the multi-timescale scheme where all corresponding parameters have been updated to correctly parameterize the critic-like functions such as $Q^i_{\bar{\boldsymbol{\pi}}_i}$, $Q^i_{H_{\bar{\boldsymbol{\pi}}_i}}$, $Q^i_{V_{\bar{\boldsymbol{\pi}}_i}}$, and $Q^i_{\alpha_{\bar{\boldsymbol{\pi}}_i}}$,we have:

$$\begin{aligned}\|\nabla_{\theta_i}J^{\text{real}}_{\theta_i}(\theta_i)-\nabla_{\theta_i}\tilde{J}_{\theta_i}(\theta_i)\|_2&\leq(D_1+D_3+D_5+D_7)e^{-lk}+(D_2+D_4+D_6+D_8)e^{-wk}\\&=E_{i,1}e^{-lk}+E_{i,2}e^{-wk}=\mathcal{O}(e^{-\min\{l,w\}k}),\end{aligned} \tag{6.114}$$

where D_1-D_8, $E_{i,1}$, and $E_{i,2}$ are constants that can increase when quantities like the maximum absolute values of reward, barrier, and cost signals increase (more details can be founded in Remark 6.2). Therefore, the norm of the error between the true gradient using the global information and the surrogate gradient using only the local information for scalability to update θ_i under the multi-timescale framework is upper bounded by the neighborhood size k as an exponential decay.

□



Figure 6.9: Average episode reward and average episode safety cost in the non-cooperative Ant task, comparing MAS^{4}NAC against baselines. Each curve illustrates the average across 5 experiments employing different random seeds, with the shaded area denoting the standard error.

6.5.5 E. Additional Experimental Results

(i) Additional Main Results on the Non-cooperative Ant task

Table 6.8: Average episode reward and average episode safety cost in the non-cooperative Ant task for the first four agents, comparing MAS^{4}NAC against baselines. Each data illustrates the average across 5 experiments employing different random seeds.

	Agent 0		Agent 1	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	-584.96 ± 42.70	89.55 ± 44.08	-585.34 ± 43.48	89.55 ± 44.08
MACPO	-671.63 ± 27.45	62.15 ± 37.39	-672.40 ± 28.53	62.15 ± 37.39
MAPPOL	-601.25 ± 31.58	78.30 ± 41.72	-602.86 ± 31.74	78.30 ± 41.72
MASAC	-562.29 ± 27.19	109.21 ± 49.32	-573.49 ± 28.11	109.21 ± 49.32
MAS^{4}NAC	**-142.15 ± 265.60**	**30.57 ± 34.09**	**-146.72 ± 281.23**	**30.57 ± 34.09**
	Agent 2		**Agent 3**	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**
IPPO	-611.69 ± 38.46	87.60 ± 41.88	-613.56 ± 37.73	87.60 ± 41.88
MACPO	-693.07 ± 27.67	63.00 ± 36.67	-704.13 ± 34.71	63.00 ± 36.67
MAPPOL	-628.94 ± 29.07	76.40 ± 40.52	-630.16 ± 28.91	76.40 ± 40.52
MASAC	-605.29 ± 27.26	108.05 ± 48.81	-606.12 ± 26.97	108.05 ± 48.81
MAS^{4}NAC	**-170.32 ± 249.68**	**22.70 ± 25.25**	**-194.48 ± 268.94**	**22.70 ± 25.25**

We further evaluate MAS^{4}NAC in the decentralized non-cooperative Ant environment. Here, the version with predefined stability parameters α_i is used, and each agent has access only to local observations. Each agent receives its own reward, safety, and stability cost signals, with rewards still remaining closely related to the multi-agent system's ability to converge toward the desired equilibrium. As shown in Figure 6.9 and Table 6.8, MAS^{4}NAC consistently achieves obviously the highest rewards and the lowest safety violations for *each* agent compared to all baselines. Specifically, MAS^{4}NAC's reward curves exhibit fast and steady improvement, while its safety cost curves decrease meaningfully and remain well below those of competing methods across all agents. In contrast, baseline algorithms exhibit noticeable weaknesses. MASAC fails to improve rewards effectively, likely due to the inherently non-cooperative nature of this task,

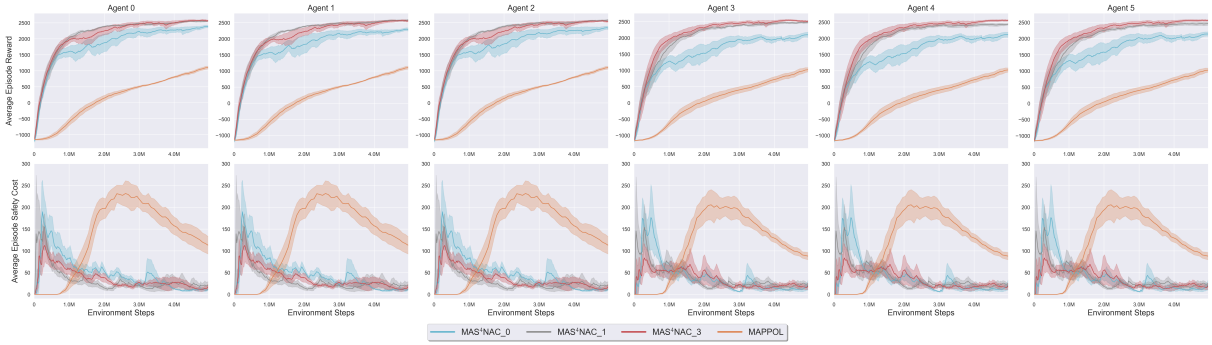


Figure 6.10: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC with more different neighbor sizes (1 and 3).

which limits its coordination capability. Constrained MARL algorithms fail to decrease the safety violations effectively. Particularly, MAPPOL leads to increasing safety costs, reflecting poor constraint handling. This failure can be attributed to the weaker CMDP setting and inefficient training via the traditional dual ascent update, as also noted in [66]. Overall, the superior performance of MAS^{4}NAC demonstrates its strong sample efficiency and capability of safe and stable control in decentralized, non-cooperative multi-agent systems compared to all baselines.

(ii) Additional Ablation Studies on Neighbor Size

Table 6.9: Average episode reward and average episode safety cost in the non-cooperative Coupled HalfCheetah task, comparing MAS^{4}NAC with different neighbor sizes against baselines. Each data illustrates the average across 5 experiments employing different random seeds.

	Agent 0		Agent 1		Agent 2	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
MAPPOL	735.92 ± 416.97	174.96 ± 143.68	735.77 ± 417.75	174.96 ± 143.68	732.35 ± 417.00	174.96 ± 143.68
MAS^{4}NAC_0	2332.84 ± 220.28	$\mathbf{12.86 \pm 33.84}$	2248.56 ± 206.08	$\mathbf{12.86 \pm 33.84}$	2294.51 ± 236.89	$\mathbf{12.86 \pm 33.84}$
MAS^{4}NAC_1	2550.69 ± 73.95	21.46 ± 38.91	2574.62 ± 72.07	21.46 ± 38.91	2563.94 ± 74.45	21.46 ± 38.91
MAS^{4}NAC_3	$\mathbf{2570.14 \pm 100.52}$	16.73 ± 23.47	$\mathbf{2574.62 \pm 102.21}$	16.73 ± 23.47	$\mathbf{2545.48 \pm 101.84}$	16.73 ± 23.47
	Agent 3		**Agent 4**		**Agent 5**	
Algorithm	**Reward**	**Cost**	**Reward**	**Cost**	**Reward**	**Cost**
MAPPOL	646.70 ± 464.02	150.76 ± 134.02	648.57 ± 465.17	150.76 ± 134.02	648.17 ± 466.71	150.76 ± 134.02
MAS^{4}NAC_0	2036.21 ± 270.92	21.07 ± 41.11	2063.64 ± 278.06	21.07 ± 41.11	2087.34 ± 275.09	21.07 ± 41.11
MAS^{4}NAC_1	2473.72 ± 79.03	20.11 ± 27.52	2431.33 ± 94.45	20.11 ± 27.52	2459.23 ± 114.29	20.11 ± 27.52
MAS^{4}NAC_3	$\mathbf{2533.35 \pm 82.94}$	$\mathbf{19.56 \pm 32.75}$	$\mathbf{2554.02 \pm 87.70}$	$\mathbf{19.56 \pm 32.75}$	$\mathbf{2563.70 \pm 85.28}$	$\mathbf{19.56 \pm 32.75}$

We further provide extended results on the effect of neighbor size in the decentralized Coupled HalfCheetah benchmark, shown in Figure 6.10 and Table 6.9. Across all cases, MAS^{4}NAC consistently outperforms the baseline MAPPOL that uses neighbor size 3, confirming its robustness under varying information availability. Within MAS^{4}NAC itself, differences can be

observed across neighbor sizes. With neighbor size 0, the algorithm performs poorly in terms of rewards, as relying only on self-information may fail to capture the inherently inter-agent nature to optimize individual objectives. Comparing neighbor size 1 and 3, larger neighborhoods do not necessarily lead to clearly and significantly better performance. Although a neighbor size of 3 yields slightly higher rewards and marginally lower final safety violations for agents 3-5, the benefits are relatively minor considering the increased communication and computational burdens. Larger neighborhoods may also introduce more complex input, requiring more sophisticated learning. Moreover, each task may have an upper bound on the useful neighbor size, beyond which additional information does not provide proportional benefits. This is evidenced by the result in Figure 6.10, where the improvement in reward between neighbor sizes 1 and 3 is significantly smaller than that between sizes 0 and 1, indicating that moderate local observability already captures most of the inter-agent nature, and can provide sufficiently appropriate surrogate policy gradient estimation for effective updates.

CHAPTER 7

Summary and Future Research Directions

This thesis examined the problem of enforcing safety and stability constraints in reinforcement learning–based control systems. While reinforcement learning has been widely adopted for complex decision-making and control, its application to safety-critical systems is hindered by the absence of mechanisms that ensure constraint satisfaction at the level of individual system states. The work presented in this thesis addressed this limitation by developing a sequence of reinforcement learning frameworks that explicitly incorporate state-wise safety and stability constraints, progressing from single-agent control to general multi-agent settings, and from model-based formulations to fully model-free learning approaches. Below we present a summary of the main results of this thesis, also discussing future research directions in which the scope of this thesis can be extended.

7.1 Chapter 3: State-Wise Safe and Stable Reinforcement Learning in White-box Settings

Chapter 3 presented the first contribution of the thesis, focusing on safe and stable reinforcement learning in white-box settings where nominal system dynamics and predefined safety certificates are available. In this chapter, a Barrier–Lyapunov Actor–Critic framework was proposed by integrating Lyapunov certificates and predefined control barrier functions into an actor–critic reinforcement learning algorithm via the augmented Lagrangian formulation, which enabled the simultaneous enforcement of state-wise safety and stability constraints during policy learning. Compared to many prior studies based on CMDPs, this framework imposed constraints for each step in the trajectory instead of only the trajectory expectation, and thus enforced stricter conditions, which is crucial in real-world safety-critical application domains. For infeasible states

in which safety and stability constraints conflict, a safety-prioritized backup controller was introduced. Simulation results demonstrated that the proposed framework significantly reduced safety violations while guiding the system towards the desired equilibrium.

Several possible improvements can be considered to further develop the framework introduced in this chapter. One important direction concerns the treatment of the unknown part of the system dynamics. In the current formulation, Gaussian process regression is employed to learn this unknown part and to quantify the uncertainty through confidence intervals defined by the GP posterior mean and variance. While Gaussian process provides principled uncertainty estimates, it is a nonparametric regression method whose computational complexity can grow rapidly with the number of training samples, which can limit scalability in long-horizon or data-intensive reinforcement learning settings. Moreover, Gaussian process–based uncertainty quantification typically relies on Gaussian assumptions on distributions. To alleviate both the reliance on such distributional assumptions and the scalability limitations, alternative uncertainty quantification techniques may be considered. In particular, conformal prediction [153, 154] offers a distribution-free approach to constructing prediction intervals with probabilistic guarantees using finite samples under mild assumptions such as exchangeability. When combined with a lightweight predictor, conformal prediction can provide valid probabilistic guarantees without requiring Gaussian noise or expensive kernel matrix computations, making it a promising direction for improving the practicality of the proposed framework in large-scale or real-world applications. Besides the uncertainty in system dynamics, another promising direction for improving the proposed framework is to adopt distributional reinforcement learning [155, 156] in place of expectation-based critic learning. Unlike classical approaches that estimate only the expected action-value and neural certificate, distributional reinforcement learning models the full return distribution, thereby capturing uncertainty that is critical in constrained settings. By replacing the scalar action-value and certificate critics with distributional counterparts, the framework can explicitly account for uncertainty reflected in the critic functions themselves, thereby enforcing probabilistic constraints.

7.2 Chapter 4: State-Wise Safe and Stable Reinforcement Learning for Gray-box Settings

Chapter 4 further extended the framework developed in Chapter 3 to gray-box settings. This chapter removed the strict requirement for known system dynamics and predefined safety certificates, and introduced learnable neural barrier certificates with theoretical analysis. Issues of infeasibility arising from conflicting safety and stability requirements were analyzed, and it was shown that the learning process naturally avoids such states as training progresses. A Neural ODE–based Lyapunov–Barrier Actor–Critic (NLBAC) algorithm designed for both state-wise safety and stability for controlled systems by leveraging the augmented Lagrangian method (ALM) and neural ODEs, was further proposed to improve modeling accuracy and sample efficiency, with either predefined control barrier functions or learned barrier certificates. Empirical evaluations on standard benchmarks demonstrated that NLBAC achieved higher cumulative rewards, fewer safety violations, and improved sample efficiency compared to baseline methods. Various ablation studies were also included to show the better performance of the ALM for non-CMDP constrained RL problems compared to the widely-used dual ascent update, and the robustness of control policies after being trained with NLBAC in the presence of external disturbance and parameter uncertainties.

While the effectiveness of the proposed algorithm has been validated across a range of tasks, further reducing the discrepancy between real system dynamics and NODE-based models remains an important direction for future work. Potential approaches include employing ensemble NODE models, and using Gaussian dynamic models, which can be trained using maximum likelihood. Also, although the frameworks developed in this thesis are instantiated using Soft Actor-Critic (SAC), the underlying constrained reinforcement learning methodology is not fundamentally tied to a particular learning algorithm. Future research may investigate the integration of the proposed constraint formulations with alternative reinforcement learning methods, including DDPG, TD3, PPO, and TRPO. Second, the present thesis focuses primarily on barrier certificates and Lyapunov-based notions of safety and stability. Future work may consider incorporating other certificate-based formulations, such as contraction metrics [157], which provide a framework for analyzing trajectory-tracking behaviour. More broadly,

the proposed constrained learning framework may be extended to incorporate different classes of learnable certificates and desired system properties beyond those considered in this thesis.

7.3 Chapter 5: State-Wise Safe and Stable Reinforcement Learning for General Multi-agent Settings in a Totally Model-free Manner

Chapter 5 developed, to the best of our knowledge, the first framework that helps to obtain a joint controller with post-convergence state-wise safety and stability guarantees under mild assumptions for general multi-agent RL tasks in a fully model-free manner. The proposed framework addresses key challenges arising from non-stationarity, agent interactions, and partial observability in multi-agent systems. State-wise safety was enforced through learnable neural barrier certificates, and a key contribution of this chapter was the establishment of sufficient local stability conditions, developed based on the concept of input-to-state stability (ISS), that enable each agent to ensure system-level stability using limited information. Moreover, the learning process of the proposed framework was analyzed theoretically, and the convergence of the update to a feasible local Nash equilibrium was formally established. The chapter further studied the issue of infeasibility under state-wise safety and stability constraints, showing that this issue does not prevent the framework from finding safe and stable controllers for MAS. To support empirical evaluation, new environments for benchmarking safe and stable MARL algorithms were introduced. Extensive experiments on both cooperative and non-cooperative multi-agent benchmarks showed that the proposed framework achieves a favorable balance between reward maximization and strict constraint satisfaction, highlighting its applicability to complex and safety-critical multi-agent environments.

Several promising extensions of this framework can be explored. One direction is to generalize the proposed framework to other reinforcement learning domains where multiple objectives coexist and are required to be optimized. For instance, in adversarial reinforcement learning [158], an adversary policy is introduced to apply external disturbances that degrade the return of the controlled agent and potentially induce unsafe or unstable behaviors, while the primary policy must actively counteract these adverse effects. This naturally leads to a two-player game formulation, to which the framework developed in this chapter may be applied to inves-

tigate whether strong empirical performance can be achieved alongside theoretical analysis on convergence, safety, and stability. Furthermore, inspired by the model-free reinforcement learning framework proposed in this chapter for control with guarantees under mild assumptions, another promising direction is to integrate model-based and model-free reinforcement learning within a unified control framework that combines the advantages of both paradigms. Specifically, a model-based safe policy can be incorporated as a module to correct potentially unsafe actions at runtime, while a model-free actor–critic algorithm is employed to optimize long-term performance. A key point that requires careful consideration in future work is that, in contrast to approaches that apply control barrier function–based safety filters at each timestep, which can be overly conservative or induce abrupt changes in control actions, the proposed integration should aim to enable smooth and continuous transitions between safe and exploratory regimes, while maintaining rigorous safety assurance and performance analysis.

7.4 Chapter 6: Improvements regarding Flexibility, Scalability, and Sample Efficiency

Chapter 6 extended the multi-agent framework developed in the previous chapter by further addressing practical challenges related to flexibility, scalability, and sample efficiency in safe and stable multi-agent reinforcement learning. To support the decentralized training and decentralized execution paradigm, a theoretical bound was derived on the error of surrogate policy gradient updates caused by replacing global information with locally available observations. This result provided a rigorous justification for scalable learning under limited communication and computational resources. In addition, this chapter proposed a neural ordinary differential equation (NODE)–based policy refinement mechanism, incorporating momentum-based dynamics, to improve sample efficiency, particularly in scenarios where data collection is expensive or constrained. The resulting MAS^{4}NAC unified safety, stability, scalability, and efficiency within a single model-free multi-agent learning algorithm. Extensive empirical evaluations across diverse cooperative and general-sum benchmarks demonstrated that MAS^{4}NAC consistently achieves better constraint satisfaction while maintaining competitive or superior reward performance and enhanced sample efficiency compared to existing safe multi-agent reinforcement learning baseline approaches.

Future work includes deploying the proposed framework in real-world safety-critical domains, such as multi-robot coordination and power grid management. Moreover, it is possible to further enhance the adaptability of the framework by allowing additional coefficients, such as the inter-agent gain functions, to be learned jointly with the existing adaptive stability parameters. This extension would enable the framework to dynamically adjust stability and interaction coefficients, leading to a more flexible and general approach to scalable safe and stable multi-agent learning. Furthermore, beyond neural ordinary differential equations, other techniques, such as diffusion models [159] and flow matching [160], have recently demonstrated promising compatibility with reinforcement learning due to their strong expressive power. The integration of diffusion-based and flow-matching-based policy representations into reinforcement learning remains an emerging research direction, and their potential benefits for constrained reinforcement learning across diverse task settings merit further investigation.

In summary, this thesis developed a coherent family of reinforcement learning frameworks that integrate control-theoretic guarantees under mild assumptions into learning-based control, enabling state-wise safe and stable decision-making in increasingly general and sophisticated settings. By bridging control theory and modern reinforcement learning, the thesis provides both theoretical insights and practical tools for reliable learning-based control in safety-critical single-agent and multi-agent systems.

Bibliography

1. V. Mnih *et al.* Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).
2. D. Silver *et al.* Mastering the game of Go with deep neural networks and tree search. *nature* **529,** 484–489 (2016).
3. D. Silver *et al.* Mastering the game of go without human knowledge. *nature* **550,** 354–359 (2017).
4. O. Vinyals *et al.* Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* **575,** 350–354 (2019).
5. J. Kober, J. A. Bagnell & J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* **32,** 1238–1274 (2013).
6. Y. Guan *et al.* Centralized cooperation for connected and automated vehicles at intersections by proximal policy optimization. *IEEE Transactions on Vehicular Technology* **69,** 12597–12608 (2020).
7. P. Zhang *et al.* Reinforcement learning-based end-to-end parking for automatic parking system. *Sensors* **19,** 3996 (2019).
8. F. Richter, R. K. Orosco & M. C. Yip. Open-sourced reinforcement learning environments for surgical robotics. *arXiv preprint arXiv:1903.02090* (2019).
9. L. Brunke *et al.* Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems* **5,** 411–444 (2022).
10. J. F. Fisac *et al.* A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control* **64,** 2737–2752 (2018).
11. J. Garcıa & F. Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research* **16,** 1437–1480 (2015).
12. E. Altman. *Constrained Markov decision processes: stochastic modeling* (Routledge, 1999).
13. J. Achiam, D. Held, A. Tamar & P. Abbeel. *Constrained policy optimization* in *Proceedings of the 34th International Conference on Machine Learning* **70** (PMLR, Sydney, Australia, 2017), 22–31.
14. T. Yang, J. Rosca, K. Narasimhan & P. Ramadge. *Projection-based constrained policy optimization* 2020.
15. C. Tessler, D. Mankowitz & S. Mannor. *Reward constrained policy optimization* 2018.
16. S. Paternain, L. Chamon, M. Calvo-Fullana & A. Ribeiro. *Constrained reinforcement learning has zero duality gap* in *Advances in Neural Information Processing Systems* **32** (Curran Associates, Inc., 2019).
17. S. Gu *et al.* Safe multi-agent reinforcement learning for multi-robot control. *Artificial Intelligence* **319,** 103905 (2023).
18. W. Zhao, T. He, R. Chen, T. Wei & C. Liu. *State-wise safe reinforcement learning: A survey* in *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence* (International Joint Conferences on Artificial Intelligence Organization, Macau, SAR China, 2023), 6814–6822.
19. H. Ma, C. Liu, S. E. Li, S. Zheng & J. Chen. *Joint synthesis of safety certificate and safe control policy using constrained reinforcement learning* in *Learning for Dynamics and Control Conference* (2022), 97–109.
20. S. Zhang, Y. Xiu, G. Qu & C. Fan. *Compositional neural certificates for networked dynamical systems* in *Learning for Dynamics and Control Conference* (2023), 272–285.
21. Y. Chow, O. Nachum, A. Faust, E. Duenez-Guzman & M. Ghavamzadeh. Lyapunov-based safe policy optimization for continuous control. *arXiv preprint arXiv:1901.10031* (2019).
22. M. Han, L. Zhang, J. Wang & W. Pan. Actor-critic reinforcement learning for control with stability guarantee. *IEEE Robotics and Automation Letters* **5,** 6217–6224 (2020).
23. M. Reis, P. Aguiar & P. Tabuada. Control barrier function-based quadratic programs introduce undesirable asymptotically stable equilibria. *IEEE Control Systems Letters* **5,** 731–736 (2020).
24. A. Wong, T. Bäck, A. V. Kononova & A. Plaat. Deep multiagent reinforcement learning: Challenges and directions. *Artificial Intelligence Review* **56,** 5023–5056 (2023).
25. W. Chen *et al. Adaptive primal-dual method for safe reinforcement learning* 2024.
26. S. Gronauer & K. Diepold. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review* **55,** 895–943 (2022).
27. T. Haarnoja *et al.* Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905* (2018).
28. A. D. Ames *et al. Control barrier functions: Theory and applications* in *2019 18th European control conference* (2019), 3420–3431.

29. C. Dawson, B. Lowenkamp, D. Goff & C. Fan. Learning safe, generalizable perception-based hybrid control with certificates. *IEEE Robotics and Automation Letters* **7,** 1904–1911 (2022).
30. R. S. Sutton & A. G. Barto. *Reinforcement Learning: An Introduction* (MIT Press, 2018).
31. Shweta Bhatt. *5 Things You Need to Know about Reinforcement Learning* 2018.
32. D. Silver. *Lectures on Reinforcement Learning* URL: `https://www.davidsilver.uk/teaching/`. 2015.
33. F. Berkenkamp, M. Turchetta, A. Schoellig & A. Krause. Safe model-based reinforcement learning with stability guarantees. *Advances in neural information processing systems* **30** (2017).
34. S.-Y. Su, X. Li, J. Gao, J. Liu & Y.-N. Chen. Discriminative deep dyna-q: Robust planning for dialogue policy learning. *arXiv preprint arXiv:1808.09442* (2018).
35. C. J. Watkins & P. Dayan. Q-learning. *Machine learning* **8,** 279–292 (1992).
36. R. Cheng, G. Orosz, R. Murray & J. Burdick. *End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks* in *Proceedings of the AAAI Conference on Artificial Intelligence* **33** (2019), 3387–3395.
37. C. K. Williams & C. E. Rasmussen. *Gaussian processes for machine learning* **3** (MIT press Cambridge, MA, 2006).
38. Wikipedia contributors. *Model-free (reinforcement learning) — Wikipedia, The Free Encyclopedia* [Online; accessed 6-August-2022]. 2021.
39. T. P. Lillicrap *et al.* Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).
40. J. Schulman, S. Levine, P. Abbeel, M. Jordan & P. Moritz. *Trust region policy optimization* in *International conference on machine learning* (2015), 1889–1897.
41. J. Schulman, F. Wolski, P. Dhariwal, A. Radford & O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).
42. S. Fujimoto, H. Hoof & D. Meger. *Addressing function approximation error in actor-critic methods* in *International conference on machine learning* (2018), 1587–1596.
43. T. Haarnoja, A. Zhou, P. Abbeel & S. Levine. *Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor* in *International conference on machine learning* (2018), 1861–1870.
44. Y. Duan, X. Chen, R. Houthooft, J. Schulman & P. Abbeel. *Benchmarking deep reinforcement learning for continuous control* in *International conference on machine learning* (2016), 1329–1338.
45. G. Brockman *et al. OpenAI Gym* 2016.
46. S. Gu *et al.* A review of safe reinforcement learning: Methods, theories, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **46,** 11216–11235 (2024).
47. A. Sootla *et al. Sauté RL: Almost surely safe reinforcement learning using state augmentation* in *International Conference on Machine Learning* (2022), 20423–20443.
48. Y. Guan *et al. Learn collision-free self-driving skills at urban intersections with model-based reinforcement learning* in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)* (2021), 3462–3469.
49. D. Ding, K. Zhang, T. Basar & M. Jovanovic. Natural policy gradient primal-dual method for constrained markov decision processes. *Advances in Neural Information Processing Systems* **33,** 8378–8390 (2020).
50. S. M. Kakade. A natural policy gradient. *Advances in neural information processing systems* **14** (2001).
51. D. Ding, X. Wei, Z. Yang, Z. Wang & M. Jovanovic. *Provably efficient safe exploration via primal-dual policy optimization* in *International Conference on Artificial Intelligence and Statistics* (2021), 3304–3312.
52. Q. Liang, F. Que & E. Modiano. Accelerated primal-dual policy optimization for safe reinforcement learning. *arXiv preprint arXiv:1802.06480* (2018).
53. S. Paternain, M. Calvo-Fullana, L. F. Chamon & A. Ribeiro. Safe policies for reinforcement learning via primal-dual methods. *IEEE Transactions on Automatic Control* **68,** 1321–1336 (2022).
54. A. Stooke, J. Achiam & P. Abbeel. *Responsive safety in reinforcement learning by pid lagrangian methods* in *International Conference on Machine Learning* (2020), 9133–9143.
55. Y. Zhang, Q. Vuong & K. Ross. First order constrained optimization in policy space. *Advances in Neural Information Processing Systems* **33,** 15338–15349 (2020).
56. M. Han, Y. Tian, L. Zhang, J. Wang & W. Pan. Reinforcement learning control of constrained dynamic systems with uniformly ultimate boundedness stability guarantee. *Automatica* **129,** 109689 (2021).
57. Y. Chow, O. Nachum, E. Duenez-Guzman & M. Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. *Advances in neural information processing systems* **31** (2018).
58. W. Saunders, G. Sastry, A. Stuhlmueller & O. Evans. Trial without error: Towards safe reinforcement learning via human intervention. *arXiv preprint arXiv:1707.05173* (2017).

59. D. Mguni *et al.* DESTA: A Framework for Safe Reinforcement Learning with Markov Games of Intervention. *arXiv preprint arXiv:2110.14468* (2021).
60. A. Sootla, A. I. Cowen-Rivers, J. Wang & H. B. Ammar. Enhancing Safe Exploration Using Safety State Augmentation. *arXiv preprint arXiv:2206.02675* (2022).
61. H. Ma *et al.* Feasible actor-critic: Constrained reinforcement learning for ensuring statewise safety. *arXiv preprint arXiv:2105.10682* (2021).
62. H. Ma *et al.* Learn Zero-Constraint-Violation Policy in Model-Free Constrained Reinforcement Learning. *arXiv preprint arXiv:2111.12953* (2021).
63. M. Samvelyan *et al.* The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043* (2019).
64. S. Shalev-Shwartz, S. Shammah & A. Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295* (2016).
65. D. Biagioni *et al. Powergridworld: A framework for multi-agent reinforcement learning in power systems* in *Proceedings of the thirteenth ACM international conference on future energy systems* (2022), 565–570.
66. Y. Zhao, Y. Yang, Z. Lu, W. Zhou & H. Li. Multi-agent first order constrained optimization in policy space. *Advances in Neural Information Processing Systems* **36,** 39189–39211 (2023).
67. L. Zhang *et al.* Scalable constrained policy optimization for safe multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* **37,** 138698–138730 (2024).
68. D. Ding, X. Wei, Z. Yang, Z. Wang & M. Jovanovic. *Provably efficient generalized lagrangian policy optimization for safe multi-agent reinforcement learning* in *Learning for dynamics and control conference* (2023), 315–332.
69. S. Lu, K. Zhang, T. Chen, T. Başar & L. Horesh. *Decentralized policy gradient descent ascent for safe multi-agent reinforcement learning* in *Proceedings of the AAAI conference on artificial intelligence* **35** (2021), 8767–8775.
70. R. Lowe *et al.* Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems* **30** (2017).
71. L. Hewing, K. P. Wabersich, M. Menner & M. N. Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems* **3,** 269–296 (2020).
72. J.-J. E. Slotine, W. Li, *et al. Applied nonlinear control* **1** (Prentice hall Englewood Cliffs, NJ, 1991).
73. R. A. Freeman & P. V. Kokotovic. Inverse optimality in robust stabilization. *SIAM journal on control and optimization* **34,** 1365–1391 (1996).
74. A. D. Ames, K. Galloway, K. Sreenath & J. W. Grizzle. Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics. *IEEE Transactions on Automatic Control* **59,** 876–891 (2014).
75. A. D. Ames, X. Xu, J. W. Grizzle & P. Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control* **62,** 3861–3876 (2016).
76. Y. Emam, P. Glotfelter & M. Egerstedt. *Robust barrier functions for a fully autonomous, remotely accessible swarm-robotics testbed* in *2019 IEEE 58th Conference on Decision and Control (CDC)* (2019), 3984–3990.
77. W. Xiao *et al.* Differentiable Control Barrier Functions for Vision-based End-to-End Autonomous Driving. *arXiv preprint arXiv:2203.02401* (2022).
78. A. Anand *et al.* Safe learning for control using control Lyapunov functions and control barrier functions: A review. *Procedia Computer Science* **192,** 3987–3997 (2021).
79. A. Taylor, A. Singletary, Y. Yue & A. Ames. *Learning for safety-critical control with control barrier functions* in *Learning for Dynamics and Control* (2020), 708–717.
80. Y. Emam, G. Notomista, P. Glotfelter, Z. Kira & M. Egerstedt. Safe reinforcement learning using robust control barrier functions. *IEEE Robotics and Automation Letters*, 1–8 (2022).
81. L. Zhao, K. Gatsis & A. Papachristodoulou. *Stable and Safe Reinforcement Learning via a Barrier-Lyapunov Actor-Critic Approach* in *2023 62nd IEEE Conference on Decision and Control (CDC)* (IEEE, Singapore, 2023), 1320–1325.
82. A. Chriat & C. Sun. On the optimality, stability, and feasibility of control barrier functions: An adaptive learning-based approach. *IEEE Robotics and Automation Letters* **8,** 7865–7872 (2023).
83. C. Dawson, Z. Qin, S. Gao & C. Fan. *Safe nonlinear control using robust neural lyapunov-barrier functions* in *Conference on Robot Learning* (2022), 1724–1735.
84. Y. Yang, Y. Jiang, Y. Liu, J. Chen & S. E. Li. Model-free safe reinforcement learning through neural barrier certificate. *IEEE Robotics and Automation Letters* **8,** 1295–1302 (2023).

85. D. Du *et al. Reinforcement learning for safe robot control using control Lyapunov barrier functions* in *2023 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, London, UK, 2023), 9442–9448.
86. Y. Wang *et al. Enforcing hard constraints with soft barriers: Safe reinforcement learning in unknown stochastic environments* in *International Conference on Machine Learning* (2023), 36593–36604.
87. Z. Li, C. Hu, Y. Wang, Y. Yang & S. Li. Safe reinforcement learning with dual robustness. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **46,** 10876–10890 (2024).
88. D. Yu *et al.* Safe model-based reinforcement learning with an uncertainty-aware reachability certificate. *IEEE Transactions on Automation Science and Engineering* **21,** 4129–4142 (2023).
89. Y. Yang *et al.* Synthesizing control barrier functions with feasible region iteration for safe reinforcement learning. *IEEE Transactions on Automatic Control* **69,** 2713–2720 (2023).
90. Z. Li & N. Azizan. Safe multi-agent reinforcement learning with convergence to generalized nash equilibrium. *arXiv preprint arXiv:2411.15036* (2024).
91. S. Zhang, O. So, M. Black & C. Fan. *Discrete GCBF Proximal Policy Optimization for Multi-agent Safe Optimal Control* in *The Thirteenth International Conference on Learning Representations* (2025).
92. Z. Qin, K. Zhang, Y. Chen, J. Chen & C. Fan. *Learning Safe Multi-agent Control with Decentralized Neural Barrier Certificates* in *International Conference on Learning Representations* (2021).
93. Y.-C. Chang, N. Roohi & S. Gao. Neural lyapunov control. *Advances in neural information processing systems* **32** (2019).
94. L. Yang *et al.* Lyapunov-stable neural control for state and output feedback: A novel formulation. *arXiv preprint arXiv:2404.07956* (2024).
95. J. Yao, M. Han & X. Yin. Lyapunov-based distributed reinforcement learning control with stability guarantee. *Computers & Chemical Engineering* **195,** 108979 (2025).
96. K. He, X. Zhang, S. Ren & J. Sun. *Deep residual learning for image recognition* in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), 770–778.
97. R. T. Chen, Y. Rubanova, J. Bettencourt & D. K. Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems* **31** (2018).
98. Y. Liang *et al. Modeling Trajectories with Neural Ordinary Differential Equations.* in *IJCAI* (2021), 1498–1504.
99. F. Djeumou, C. Neary, E. Goubault, S. Putot & U. Topcu. *Neural networks with physics-informed architectures and constraints for dynamical systems modeling* in *Learning for Dynamics and Control Conference* (2022), 263–277.
100. V. M. M. Alvarez, R. Roşca & C. G. Fălcuţescu. Dynode: Neural ordinary differential equations for dynamics modeling in continuous control. *arXiv preprint arXiv:2009.04278* (2020).
101. L. Zhao, K. Miao, H. Cao, K. Gatsis & A. Papachristodoulou. NLBAC: A neural ODE-based algorithm for state-wise stable and safe reinforcement learning. *Neurocomputing* **638,** 130041 (2025).
102. Y. Wang *et al. ODE-based Smoothing Neural Network for Reinforcement Learning Tasks* in *The Thirteenth International Conference on Learning Representations* (2025).
103. Y. Wang & D. Boyle. Trustworthy Reinforcement Learning for Quadrotor UAV Tracking Control Systems. *arXiv preprint arXiv:2302.11694* (2023).
104. Y. Wang, J. O' Keeffe, Q. Qian & D. Boyle. *QuaDUE-CCM: Interpretable Distributional Reinforcement Learning using Uncertain Contraction Metrics for Precise Quadrotor Trajectory Tracking* in *Conference on Robot Learning* (2023), 2306–2316.
105. T. Koller, F. Berkenkamp, M. Turchetta & A. Krause. *Learning-based model predictive control for safe exploration* in *2018 IEEE conference on decision and control* (2018), 6059–6066.
106. T. Wei & C. Liu. *Safe control algorithms using energy functions: A unified framework, benchmark, and new directions* in *2019 IEEE 58th Conference on Decision and Control* (2019), 238–243.
107. H. Wang, K. Margellos & A. Papachristodoulou. Safety verification and controller synthesis for systems with input constraints. *arXiv preprint arXiv:2204.09386* (2022).
108. A. A. do Nascimento, A. Papachristodoulou & K. Margellos. A game theoretic approach for safe and distributed control of unmanned aerial vehicles (2023).
109. H. Cao, Y. Mao, L. Sha & M. Caccamo. Physical Deep Reinforcement Learning Towards Safety Guarantee. *arXiv preprint arXiv:2303.16860* (2023).
110. Y.-C. Chang & S. Gao. *Stabilizing neural control using self-learned almost Lyapunov critics* in *2021 IEEE International Conference on Robotics and Automation (ICRA)* (2021), 1803–1809.

111. X. Tan & D. V. Dimarogonas. On the undesired equilibria induced by control barrier function based quadratic programs. *Automatica* **159,** 111359 (2024).
112. J. Choi, F. Castaneda, C. J. Tomlin & K. Sreenath. Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions. *arXiv preprint arXiv:2004.07584* (2020).
113. C. Dawson, S. Gao & C. Fan. Safe Control with Learned Certificates: A Survey of Neural Lyapunov, Barrier, and Contraction methods. *arXiv preprint arXiv:2202.11762* (2022).
114. S. Meyn. *Control systems and reinforcement learning* (Cambridge University Press, 2022).
115. S. Wang *et al.* A RL-based Policy Optimization Method Guided by Adaptive Stability Certification. *arXiv preprint arXiv:2301.00521* (2023).
116. J. Li, D. Fridovich-Keil, S. Sojoudi & C. J. Tomlin. *Augmented Lagrangian Method for Instantaneously Constrained Reinforcement Learning Problems* in *2021 60th IEEE Conference on Decision and Control* (2021), 2982–2989.
117. A. Ray, J. Achiam & D. Amodei. Benchmarking Safe Exploration in Deep Reinforcement Learning (2019).
118. W. Zhao, R. Chen, Y. Sun, T. Wei & C. Liu. *State-wise constrained policy optimization* 2023.
119. L. Zhao, K. Miao, K. Gatsis & A. Papachristodoulou. *Stable and safe human-aligned reinforcement learning through neural ordinary differential equations* 2024.
120. G. Thomas, Y. Luo & T. Ma. *Safe reinforcement learning by imagining the near future* in *Advances in Neural Information Processing Systems* **34** (Curran Associates, Inc., 2021), 13859–13869.
121. Y. Xiong, D. Zhai, M. Tavakoli & Y. Xia. Discrete-time control barrier function: High-order case and adaptive case. *IEEE Transactions on Cybernetics* **53,** 3231–3239 (2022).
122. X. Tan, W. S. Cortez & D. Dimarogonas. High-order barrier functions: Robustness, safety, and performance-critical control. *IEEE Transactions on Automatic Control* **67,** 3021–3028 (2021).
123. W. Xiao & C. Belta. High-order control barrier functions. *IEEE Transactions on Automatic Control* **67,** 3655–3662 (2022).
124. D. Tan, F. Acero, R. McCarthy, D. Kanoulas & Z. Li. *Value functions are control barrier functions: Verification of safe policies using control theory* 2023.
125. D. Bertsekas. *Nonlinear Programming* (Athena Scientific, Belmont, Massachusetts, 2016).
126. Y. Yang *et al. Feasible policy iteration* 2023.
127. A. Jayant & S. Bhatnagar. *Model-based safe deep reinforcement learning via a constrained proximal policy optimization algorithm* in *Advances in Neural Information Processing Systems* **35** (Curran Associates, Inc., New Orleans, LA, USA, 2022), 24432–24445.
128. J. Wu, H. Zhang & Y. Vorobeychik. *Verified safe reinforcement learning for neural network dynamic models* 2024.
129. Z. Yuan *et al.* Safe-control-gym: A unified benchmark suite for safe learning-based control and reinforcement learning in robotics. *IEEE Robotics and Automation Letters* **7,** 11142–11149 (2022).
130. B. Peng *et al.* Facmac: Factored multi-agent centralised policy gradients. *Advances in Neural Information Processing Systems* **34,** 12208–12221 (2021).
131. D. Ying, Y. Zhang, Y. Ding, A. Koppel & J. Lavaei. Scalable primal-dual actor-critic method for safe multi-agent rl with general utilities. *Advances in Neural Information Processing Systems* **36,** 36524–36539 (2023).
132. E. D. Sontag & Y. Wang. On characterizations of the input-to-state stability property. *Systems & Control Letters* **24,** 351–359 (1995).
133. Z.-P. Jiang & T. Liu. Small-gain theory for stability and control of dynamical networks: A Survey. *Annual Reviews in Control* **46,** 58–79 (2018).
134. L. J. Ratliff, S. A. Burden & S. S. Sastry. On the characterization of local Nash equilibria in continuous games. *IEEE transactions on automatic control* **61,** 2301–2307 (2016).
135. J. Foerster, I. A. Assael, N. De Freitas & S. Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *Advances in neural information processing systems* **29** (2016).
136. D. Yu, H. Ma, S. Li & J. Chen. *Reachability Constrained Reinforcement Learning* in *Proceedings of the 39th International Conference on Machine Learning* **162** (PMLR, 2022), 25636–25655.
137. Y. Chow, M. Ghavamzadeh, L. Janson & M. Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *Journal of Machine Learning Research* **18,** 1–51 (2018).
138. V. S. Borkar. *Stochastic approximation: a dynamical systems viewpoint* (Springer, 2008).
139. H. K. Khalil & J. Grizzle. *Nonlinear Systems* (Prentice Hall, Upper Saddle River, 2002).

140. H. Ito, S. Dashkovskiy & F. Wirth. Capability and limitation of max-and sum-type construction of Lyapunov functions for networks of iISS systems. *Automatica* **48,** 1197–1204 (2012).
141. Y. Zhong *et al.* Heterogeneous-Agent Reinforcement Learning. *Journal of Machine Learning Research* **25,** 1–67 (2024).
142. J. G. Kuba *et al. Trust Region Policy Optimisation in Multi-Agent Reinforcement Learning* in *International Conference on Learning Representations* (2022).
143. C. Yu *et al.* The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in neural information processing systems* **35,** 24611–24624 (2022).
144. C. S. De Witt *et al.* Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533* (2020).
145. G. Qu, Y. Lin, A. Wierman & N. Li. Scalable multi-agent reinforcement learning for networked systems with average reward. *Advances in Neural Information Processing Systems* **33,** 2074–2086 (2020).
146. A. Das *et al. Tarmac: Targeted multi-agent communication* in *International Conference on machine learning* (2019), 1538–1546.
147. J. Foerster, G. Farquhar, T. Afouras, N. Nardelli & S. Whiteson. *Counterfactual multi-agent policy gradients* in *Proceedings of the AAAI conference on artificial intelligence* **32** (2018).
148. L. Canese *et al.* Multi-agent reinforcement learning: A review of challenges and applications. *Applied Sciences* **11,** 4948 (2021).
149. C. Alfano & P. Rebeschini. Dimension-free rates for natural policy gradient in multi-agent reinforcement learning. *arXiv preprint arXiv:2109.11692* (2021).
150. J. K. Gupta, M. Egorov & M. Kochenderfer. *Cooperative multi-agent control using deep reinforcement learning* in *International conference on autonomous agents and multiagent systems* (2017), 66–83.
151. V. Apidopoulos, N. Ginatta & S. Villa. Convergence rates for the heavy-ball continuous dynamics for non-convex optimization, under Polyak–Łojasiewicz condition. *Journal of Global Optimization* **84,** 563–589 (2022).
152. K. Okamura, N. Marumo & A. Takeda. Heavy-ball Differential Equation Achieves $O(\varepsilon^{-7/4})$ Convergence for Nonconvex Functions. *arXiv preprint arXiv:2406.06100* (2024).
153. R. Tibshirani. Conformal prediction. *UC Berkeley* (2023).
154. L. Lindemann, M. Cleaveland, G. Shim & G. J. Pappas. Safe planning in dynamic environments using conformal prediction. *IEEE Robotics and Automation Letters* **8,** 5116–5123 (2023).
155. M. G. Bellemare, W. Dabney & R. Munos. *A distributional perspective on reinforcement learning* in *International conference on machine learning* (2017), 449–458.
156. W. Dabney, M. Rowland, M. Bellemare & R. Munos. *Distributional reinforcement learning with quantile regression* in *Proceedings of the AAAI conference on artificial intelligence* **32** (2018).
157. C. Dawson, S. Gao & C. Fan. Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control. *IEEE Transactions on Robotics* **39,** 1749–1767 (2023).
158. L. Pinto, J. Davidson, R. Sukthankar & A. Gupta. *Robust adversarial reinforcement learning* in *International Conference on Machine Learning* (2017), 2817–2826.
159. Z. Zhu *et al.* Diffusion models for reinforcement learning: A survey. *arXiv preprint arXiv:2311.01223* (2023).
160. D. McAllister *et al.* Flow matching policy gradients. *arXiv preprint arXiv:2507.21053* (2025).