UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

LEARNING-BASED MIN-MAX DIFFERENTIAL DYNAMIC PROGRAMMING

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By Mohammad Sarbaz

Norman, Oklahoma

2026

LEARNING-BASED MIN-MAX DIFFERENTIAL DYNAMIC PROGRAMMING

A DISSERTATION APPROVED FOR THE

SCHOOL OF AEROSPACE AND MECHANICAL ENGINEERING

BY THE COMMITTEE CONSISTING OF

Dr. Wei Sun, Chair

Dr. Bin Xu

Dr. Diogo Merguizo Sanchez

Dr. Theodore B. Trafalis

Acknowledgments

During my time as a graduate student, I have been fortunate to have encountered remarkable individuals who have consistently provided me with unwavering support and encouragement when I needed it most. Their unwavering positivity and strength have played a pivotal role in shaping my scientific experience, as well as fostering my personal growth and eventual accomplishments.

First and foremost, I extend my heartfelt appreciation to my advisor, Dr. Wei Sun. His dedication, mentorship, and profound investment in my academic growth have been instrumental in shaping the trajectory of my research. Dr. Sun's unwavering support, insightful feedback, and encouragement have continuously motivated me to strive for excellence in my work. The time and effort that Dr. Sun dedicated to nurturing my scientific curiosity have been pivotal in enabling me to successfully complete this task. I am sincerely grateful for his mentorship and belief in my abilities.

I would like to express my gratitude to my esteemed committee members, Dr. Bin Xu, Dr. Diogo Merguizo Sanchez, and Dr. Theodore B. Trafalis. Their valuable insights, expertise, and constructive feedback have significantly enhanced the quality and rigor of this dissertation. Their collective wisdom and guidance have broadened my perspective and enriched the overall contributions of this research.

I want to express my deepest gratitude to my mother, Fatemeh, my father, Mohammadreza, and my brother Alireza, whose love, encouragement, and guidance kept me motivated in pursuing my dreams. Above all, I am extremely grateful for the continuous support, encouragement.

To my Mom and Dad, for sacrificing their lives to support and nurture my dreams. This achievement is as much yours as it is mine.

Table of Contents

List of Tables

List of Figures

Abstract

Designing optimal control strategies for dynamic systems operating in uncertain and adversarial environments remains a fundamental challenge in modern control theory and reinforcement learning. Traditional control methods often assume precise knowledge of system dynamics, but real-world systems frequently operate with incomplete information, disturbances, and unpredictable adversarial influences. This has led to the development of robust and adaptive control strategies that can optimize performance while mitigating worst-case uncertainties. Among these, Min-Max Differential Dynamic Programming (Min-Max DDP) stands out as an effective approach, leveraging the principles of dynamic programming to iteratively refine control policies while explicitly accounting for worst-case adversarial disturbances. By framing the control problem as a minimization–maximization game, Min-Max DDP explicitly considers worst-case disturbances, providing robustness in safety-critical and adversarial settings. Despite its advantages, Min-Max DDP faces several key challenges that limit its practical applicability. However, conventional implementations rely on exact system models, limiting their applicability to systems with unknown or stochastic dynamics. Additionally, extending Min-Max DDP to large-scale, multi-agent, and output feedback systems introduces further complexity. On the other hand, ensuring safety in the framework's performance is a crucial aspect that cannot be neglected. This research systematically addresses these challenges by developing a series of novel methodologies that enhance the adaptability, robustness, and scalability of Min-Max DDP. To overcome the limitation of requiring exact dynamics, data-driven Min-Max DDP is introduced, employing Gaussian Process Regression to approximate system dynamics and compute optimal control updates in the absence of explicit models. The frame-

work is further extended to a stochastic game-theoretic Min-Max DDP formulation that models both drift and diffusion components. Gaussian Process Regression is used to approximate the drift, while a binning-based nonparametric method is employed to estimate the diffusion dynamics, enabling a fully data-driven framework for stochastic Min-Max control. To enhance scalability, the research develops a distributed Min-Max DDP framework enabling decentralized coordination among interconnected subsystems. This approach allows interconnected agents to collaborate effectively while maintaining robustness against uncertainty. Additionally, Min-Max DDP is applied to stochastic H_∞ control as a robust strategy to mitigate disturbances and ensure stability. When direct state observation is unavailable, the research introduces sliding mode observer-based Min-Max DDP, integrating sliding mode observers to estimate system states and provide robust output feedback control, ensuring effective performance despite limited observability. The research further extends Min-Max DDP to nonzero-sum and multiplayer differential games, enabling optimal strategies in competitive and cooperative multi-agent settings. Additinally, the study investigates Min-Max adaptive dynamic programming for zero-sum differential games, incorporating adaptive learning mechanisms to enhance performance in adversarial settings. Finally, this study incorporates a chance-constrained framework to ensure that the trajectory optimization satisfies safety requirements. By systematically addressing these challenges, the work transforms Min-Max DDP into a versatile framework for data-driven, robust, and multi-agent control. Through the integration of machine learning, stochastic modeling, and distributed control, these contributions advance the frontiers of optimal control, making Min-Max Differential Dynamic Programming more applicable to real-world problems characterized by uncertainty, partial observability, and adversarial interactions. The proposed methods are validated on diverse dynamic systems, including the inverted pendulum, double inverted pendulum, mass–spring–damper, power sys-

tems, two-degree-of-freedom helicopter, and nonlinear Quadrotor. Additional tests on vertical flight regulation, cart-pole, unicycle robot, and load-frequency control further demonstrate their effectiveness across varied control scenarios.

Chapter 1

Introduction

1.1 Background and Motivation

Designing optimal control strategies for dynamic systems operating under uncertainty has been a central topic in control theory and reinforcement learning for decades. The goal of optimal control is to determine a control law that minimizes a performance index while ensuring system stability and constraint satisfaction [1]. Classical optimal control methods, such as Linear Quadratic Regulation (LQR) and Model Predictive Control (MPC), rely on known system dynamics and often assume linearity or small deviations from equilibrium [2]. Although these methods provide satisfactory performance for well-modeled systems, their effectiveness degrades significantly when faced with modeling errors, stochastic disturbances, or adversarial influences commonly present in real-world applications.

Dynamic Programming (DP), introduced by Bellman in the 1950s [3], laid the theoretical foundation for solving optimal control problems by decomposing them into smaller subproblems. However, DP suffers from the "curse of dimensionality," making it computationally impractical for high-dimensional nonlinear systems. To address this issue, Jacobson and Mayne proposed Differential Dynamic Programming (DDP) [4], which combines second-order local approximations with iterative optimization to efficiently solve nonlinear control problems. DDP has since become a cornerstone in

nonlinear optimal control and trajectory optimization due to its ability to generate near-optimal control laws with relatively low computational effort [5]. Its applications span robotics, aerospace, energy systems, and autonomous vehicles, where nonlinearities and complex dynamics are prevalent [6].

Despite its advantages, conventional DDP assumes precise knowledge of system dynamics, which limits its applicability in practical scenarios. Real systems are often subject to uncertainties, external disturbances, and unmodeled dynamics. These factors can cause significant performance degradation or even instability when the nominal model deviates from the true system. To mitigate such effects, robust control techniques—such as H_∞ control and sliding-mode control—have been developed [7], offering guaranteed performance under bounded uncertainties. However, these methods typically rely on conservative design assumptions and may sacrifice optimality to ensure robustness.

To explicitly handle worst-case disturbances within the optimal control framework, the Min-Max Differential Dynamic Programming (Min-Max DDP) method has emerged [8]. This approach formulates the control problem as a two-player zero-sum differential game between a control input (the minimizing player) and an adversarial disturbance (the maximizing player). The resulting min-max optimization enables the control policy to anticipate and counteract the worst-case effects of uncertainty. By embedding game-theoretic reasoning into the DDP structure, Min-Max DDP achieves robustness against disturbances while retaining the efficiency of dynamic programming. This makes it highly relevant for safety-critical applications such as aerospace guidance, multi-robot coordination, and autonomous navigation.

Nevertheless, Min-Max DDP in its traditional form still faces several challenges. First, it requires exact knowledge of the system dynamics, making it unsuitable for data-driven or partially known systems. Second, its computational cost grows rapidly

with system dimensionality, limiting its scalability to large-scale or distributed systems. Third, real-world control problems often involve stochastic uncertainties and partial observability, where state information is limited or corrupted by measurement noise. Finally, ensuring safety during learning and control execution remains an open challenge, particularly in environments where violations of physical or operational constraints are unacceptable.

These limitations motivate the development of learning-based and robust variants of Min-Max DDP that integrate machine learning, adaptive control, and stochastic modeling. Recent progress in data-driven methods such as Gaussian Process Regression (GPR) and neural-network-based function approximation [9, 10] offers promising tools to estimate unknown dynamics directly from data. Likewise, distributed optimization and chance-constrained control frameworks enable scalable and safe coordination of multiple agents under uncertainty [11, 12]. Building on these advances, this research aims to develop a unified, learning-based Min-Max DDP framework that enhances adaptability, scalability, and safety, enabling robust optimal control in complex, uncertain, and adversarial environments.

1.2 Emergence of Min-Max Differential Dynamic Programming

Differential Dynamic Programming (DDP) has long been recognized as one of the most efficient second-order methods for nonlinear optimal control and trajectory optimization. It refines control policies iteratively through backward–forward propagation of the value function and system dynamics, providing both feedforward and feedback gains at each step [5, 6]. While DDP offers excellent convergence and computational efficiency, it traditionally assumes deterministic and fully known dynamics. In practi-

cal settings—such as autonomous vehicles, aerial robotics, and power systems—control performance is often degraded by model uncertainties, process noise, and unpredictable external disturbances. This need for robustness and worst-case performance guarantees led to the integration of game-theoretic principles into the DDP framework.

To explicitly address adversarial disturbances, the optimal control problem can be reformulated as a two-player zero-sum differential game, in which the controller seeks to minimize a cost functional while a virtual adversary maximizes it. This min–max structure results in the Hamilton–Jacobi–Isaacs (HJI) equation—a generalization of the Hamilton–Jacobi–Bellman (HJB) equation that governs standard DDP. Solving the HJI equation iteratively through local quadratic expansions yields the Min-Max Differential Dynamic Programming (Min-Max DDP) algorithm. The method explicitly anticipates the worst-case disturbance during optimization, thereby producing control policies that are inherently robust to uncertainty and model mismatch.

The first formal development of Min-Max DDP was presented by [8] in "Min-Max Differential Dynamic Programming: Continuous and Discrete Time Formulations". That seminal work derived the complete set of backward differential equations for both continuous- and discrete-time settings and established the theoretical connection between Min-Max DDP and the HJI framework. It also demonstrated that Min-Max DDP can achieve H_∞-like robustness without resorting to conservative worst-case linearization, marking the first systematic integration of game-theoretic reasoning into DDP. This publication is now widely recognized as the foundation of the Min-Max DDP family of methods.

Following the 2018 formulation, researchers began extending the approach to broader and more realistic contexts. [13] refined the algorithm for nonlinear stochastic systems, incorporating stochastic disturbances within the min–max recursion. [14] applied Min-Max DDP to trajectory optimization under model uncertainty, improving numerical

conditioning of the backward pass and convergence speed. Subsequent studies have explored risk-sensitive, robust, and safe variants, where probabilistic constraints or risk measures are embedded into the cost function [15]. Recent advances [16, 17] integrated chance-constrained and barrier function as safety-critical formulations into the DDP structure, enabling real-time trajectory optimization under uncertainty.

Parallel to these algorithmic developments, researchers have also investigated the theoretical links between Min-Max DDP, robust dynamic programming, and H_∞ control. [18] demonstrated that Min-Max DDP can be interpreted as a dynamic programming realization of H_∞ optimal control, bridging classical robust control with modern game-theoretic optimization. This perspective positions Min-Max DDP as a unifying framework between robust control theory and reinforcement learning, where the adversarial agent plays a role analogous to the environment in worst-case policy optimization.

In summary, the emergence of Min-Max DDP represents a major milestone in optimal control research. By extending classical DDP to a two-player game structure, it achieves robustness against uncertainty while maintaining computational efficiency. The ongoing progression of Min-Max DDP—toward stochastic, risk-aware, and safety-critical systems—forms the foundation upon which this dissertation builds.

1.3 Literature Review

1.3.1 Classical and Modern Differential Dynamic Programming Methods

Dynamic Programming (DDP) stands as a powerful optimization framework utilized to solve complex control problems in dynamic systems [19]. At its core, DDP operates

by iteratively refining control policies through the application of dynamic programming principles. By discretizing the system dynamics and employing local quadratic approximations, DDP efficiently computes optimal control trajectories while considering system constraints [20, 21]. This iterative approach enables DDP to handle nonlinear dynamics, non-convex cost functions, and state and control constraints effectively [22]. Moreover, DDP exhibits versatility in various domains, from robotics to aerospace engineering, enabling the design of optimal control strategies for a wide range of applications[23]. Despite its computational demands, DDP's ability to exploit system dynamics and provide near-optimal control policies makes it a cornerstone in the field of optimal control theory. Furthermore, recent advancements in computational techniques and parallel computing have enhanced the scalability and applicability of DDP, opening doors for its integration into real-time control systems and reinforcement learning frameworks. Specifically, Differential Dynamic Programming serves as a fundamental tool for solving challenging optimization problems in dynamic systems, offering a pathway towards robust and efficient control strategies across diverse domains [24]. Due to its capability for solving complex problems, DDP has been used in many works and application over several decades and has become one of the most popular algorithms in modern optimal control. A detailed comparison between Newton's method, as applied to discrete-time unconstrained optimal control problems and the DDP has been done in [25, 26] and later the Quasi-Newton DDP for robust low-thrust optimization is studied in [27]. DDP algorithm for unsteady, nonlinear, groundwater management problems used in [28]. Another application of DDP is presented for solving the hydroelectric generation scheduling problem (HSP) in [29] and for multi-reservoir system control in [30, 31]. There have been also some extensions of DDP recently like receding horizon, an extension of classic DDP [32, 33]. Solving problem of air traffic control by DDP has been considered in [34]. There have been also many

other utilization of DDP in various fields and works due to its strength and ability like using DDP for trajectory optimization of autonomous driving [35], torque control on actuators embedded in a TALOS humanoid robot [36], multi-phase rigid contact dynamics [37], rigid-body system [38], whole-body motion planning of legged robots [39], and many other applications in industry [40, 41, 42, 43, 44]. From the academic view, due to flexibility of DDP, it has been used for many extensions and with many control algorithm to solve issues are arisen in research. For instance, the DDP for stochastic nonlinear dynamics is utilized in [45] to diminish the impact of noise. Probabilistic trajectory optimization framework for systems with unknown dynamics, called Probabilistic Differential Dynamic Programming (PDDP) based on Gaussian processes (GPs) is introduced in [46]. The hierarchical scheme of the DDP to improve the computational speed is studied in [47]. Distributed fom of the DDP for large-scale system is introduced in [48]. Combination of neural network and DDP for reinforcement learning problem has been proposed in [49]. To accommodate inequality constrains, DDP with nonlinear constraints is considered in [50]. For unconstrained nonlinear dynamic games, the Newton's method with DDP is presented in [51]. Also many other methods and approached have been considered with DDP such as DDP with terminal constraints [52], data-driven DDP using Gaussian process [53], robust DDP [54], and distributed robust DDP with Wasserstein distance in [55]. With all the works mentioned in both academic and industry, it is evident that the DDP is a powerful and capable approach for many problems. However, there are some issues with this approach that indicate a need to explore other algorithms. One of the innovative and prominent extensions of the DDP is min-max Differential Dynamic Programming (min-max DDP) or Game-Theoretic Differential Dynamic Programming (GT-DDP) which is a powerful tool to encounter disturbance, uncertainties, and complexity of dynamics due to existence of intelligent disturbance or destabilizing controller in the dynamics [8].

1.3.2 Game-Theoretic and Min-Max Differential Dynamic Programming Approaches

The Min-Max Differential Dynamic Programming (Min-Max DDP) framework extends the classical DDP to address control and decision-making problems in uncertain or adversarial environments. While traditional DDP assumes a benign or stochastic setting, Min-Max DDP introduces a game-theoretic formulation in which the controller minimizes the performance index while an adversarial input or disturbance seeks to maximize it. This two-player zero-sum structure leads to a min–max optimization problem that anticipates the worst-case disturbance and generates robust feedback laws. The algorithm integrates principles from dynamic programming and differential games to improve resilience against uncertainty and adversarial perturbations. The first comprehensive formulation of Min-Max DDP, in both continuous- and discrete-time settings, was introduced in [8, 56], where backward recursive equations were derived to approximate the Hamilton–Jacobi–Isaacs (HJI) solution using local quadratic expansions. This framework laid the foundation for extending DDP into game-theoretic domains where robustness and strategic adaptation are critical.

Following this foundational development, several extensions and applications of Min-Max DDP have been proposed. The approach has been utilized for biped walking control under model uncertainties [57, 57], where the adversarial formulation enables robust gait stabilization. In [13], Min-Max DDP was extended to stochastic dynamic systems to account for process noise and probabilistic disturbances. More recent efforts include neural and learning-based extensions, such as the Min-Max robust control framework developed using continuous differential neural networks [58], and a game-theoretic model predictive control approach to mitigate model inaccuracy by representing control inputs and disturbances as competing agents [59]. The method has also

been applied to pursuit–evasion problems, including multiple-pursuit-evasion in three-dimensional flow fields [60] and multi-pursuer single-evader interactions in dynamic environments [61]. Moreover, Min-Max DDP has been adopted as an H_∞-type robust control method to attenuate external disturbances and improve stability margins [62]. These developments collectively demonstrate the versatility of Min-Max DDP across both theoretical and practical control problems.

Despite these advancements, several key challenges remain in the Min-Max DDP framework that motivate further research. Most existing formulations rely on the assumption of fully known system dynamics, which limits their applicability to real-world systems with modeling errors or incomplete information. This motivates the need for data-driven Min-Max DDP approaches capable of learning or approximating system dynamics from observed data, such as through Gaussian Process regression or other machine learning estimators. In addition, conventional Min-Max DDP methods typically assume full-state feedback, whereas many practical systems only provide partial or noisy output measurements, necessitating observer-based and output-feedback Min-Max DDP formulations that can estimate the unmeasured states in real time. Scalability also remains an open problem for high-dimensional or interconnected systems, leading to growing interest in distributed and large-scale Min-Max DDP methods that decompose global optimization across subsystems. Furthermore, robust control under stochastic disturbances and safety-critical operation has drawn attention to stochastic and chance-constrained Min-Max DDP formulations that guarantee safe trajectory optimization under uncertainty. Finally, the extension of the Min-Max DDP paradigm to continuous-time and nonzero-sum differential games provides a foundation for analyzing cooperative and competitive interactions among multiple intelligent agents. Collectively, these research directions form the basis of the methodologies developed and analyzed in the subsequent chapters of this dissertation.

1.3.3 Adaptive and Reinforcement Learning-Based Control

Adaptive control and reinforcement learning (RL) have become two converging for achieving robust and optimal control in uncertain and nonlinear dynamic systems. Adaptive control methods focus on online identification and adjustment of controller parameters to ensure stability and tracking performance even when the system dynamics are partially unknown or time-varying. In contrast, RL offers a data-driven framework for learning optimal control policies through interaction with the environment, without explicit knowledge of the system model. Recent studies have emphasized that these two frameworks are not mutually exclusive but rather complementary: adaptive control provides guaranteed stability and convergence properties, while RL enhances performance and generalization in high-dimensional tasks [63, 64]. The work by [65] highlights this synergy, showing that RL can be interpreted as an adaptive dynamic programming process with policy and value iteration embedded within a feedback learning loop.

Recent research has focused on hybrid and model-based RL techniques that integrate adaptive control theory with RL's data-driven exploration to achieve stability, sample efficiency, and safety. For instance, adaptive actor–critic algorithms have been proposed to handle nonlinear systems with unknown dynamics while maintaining Lyapunov stability guarantees [66, 67]. Neural network–based adaptive dynamic programming (ADP) frameworks have been employed for continuous-time nonlinear control, leveraging online learning to update both policy and value networks [68]. Deep reinforcement learning has also been applied to robotic and autonomous systems, combining adaptive control structures with data-efficient policy updates to overcome the challenges of limited training data and environmental uncertainty [69, 70]. Moreover, safety-aware adaptive RL frameworks have emerged to address real-world deployment

issues by enforcing stability and constraint satisfaction during policy learning [71, 72].

Overall, adaptive and reinforcement learning-based control methods provide a theoretical and algorithmic foundation for intelligent decision-making in complex systems with incomplete information. By unifying stability-guaranteed adaptive control with data-driven learning and policy optimization, these approaches have advanced the capability of modern control frameworks toward real-time adaptation, robustness, and safe autonomy — establishing a conceptual bridge between classical control and the learning-based extensions explored in subsequent chapters of this dissertation.

1.3.4 Observer-Based and Output-Feedback Control Strategies

In many real-world control systems, the full state vector cannot be directly measured, or the measurements are corrupted by noise, delays or partial observability. This constraint gives rise to the challenge of output-feedback control, where an estimator (observer) must reconstruct the unmeasured states from available output data, and the controller must be designed based on these estimated states. Classical linear results exploit the separation principle, but for nonlinear and uncertain systems this separation often fails, necessitating integrated observer-controller design and robust estimation schemes [73]. Recent advances focus on high-gain observers, neural-network based state estimators, and hybrid measurement/observer architectures that deal with sampling, quantization, and sensor failures.

For instance, a 2024 study [74] investigates an observer-based exponential stabilization scheme for switched Takagi–Sugeno fuzzy systems with unmeasurable premise variables, showing how reduced-order observers can be designed for nonlinear systems under output feedback. Another very recent work, [75], proposes a neural-network

prescribed-time observer for uncertain pure-feedback nonlinear systems, demonstrating finite-time convergence of the state estimates despite high nonlinearities and measurement challenges. These works illustrate the growing use of learning-based estimation embedded in output-feedback control frameworks. Meanwhile, [76] studies output-feedback control for two-time-scale systems using high-gain observers that link reduced-order estimation with control design under multiscale dynamics. Together, these approaches reflect the trend toward observer-based designs that are robust, adaptive, and suitable for output-feedback control in modern uncertain systems.

Despite these developments, several open issues remain particularly relevant for robust, game-theoretic, or large-scale control frameworks. First, the interplay between observer convergence speed and controller robustness in adversarial or worst-case disturbance settings is still underexplored. Second, when applying such designs in high-dimensional or interconnected systems, scalability and computational complexity of the observer-controller pair can become prohibitive. Third, most existing output-feedback methods assume known or structured dynamics, making their adaptation to truly unknown or data-driven settings problematic. Addressing these gaps is critical when output feedback must be integrated into frameworks like Min-Max Differential Dynamic Programming, distributed control or safe trajectory optimization under uncertainty.

1.3.5 Data-Driven and Gaussian Process Regression

In complex real-world systems, precise mathematical models are often unavailable or only partially known due to nonlinearities, stochastic effects, and unmodeled dynamics. This challenge has led to the rise of data-driven control methods, which aim to identify, approximate, or directly learn control policies from observed data rather than relying on complete analytical models. Data-driven approaches combine elements of system identification, adaptive control, and machine learning to build predictive models that

can generalize across operating regimes while providing feedback for control design [77]. Within this context, Gaussian Process Regression (GPR) has emerged as one of the most promising tools for learning continuous-time nonlinear system dynamics, offering both mean predictions and uncertainty quantification in a Bayesian framework [78]. This probabilistic feature allows controllers to explicitly account for model uncertainty, making GPR particularly suited for safety-critical and robust control applications.

Recent studies have explored the integration of GPR with model predictive control (MPC) and differential dynamic programming (DDP) to handle unknown or partially known system dynamics. For example, Liu et al. [79] developed a dual-Gaussian-process MPC framework for online learning and prediction, demonstrating improved control accuracy and adaptation under dynamic uncertainty. [80] proposed an experimental design-based GPR control strategy that optimizes data collection to minimize model error during learning. Earlier work by [53] introduced one of the first data-driven DDP formulations using Gaussian Processes, enabling efficient trajectory optimization when explicit system models are unavailable. More recent efforts, such as [81], provide finite-sample reachability guarantees for systems controlled using GPR models, addressing safety and stability in learning-based controllers. Additionally, [82] presented an event-triggered multi-agent learning scheme that employs GPR to improve cooperative control performance while reducing communication costs.

Despite the rapid progress in GPR-based control, several fundamental challenges remain. The computational scalability of standard GPR, with its cubic dependence on dataset size, limits its real-time applicability in high-dimensional systems. To address this, sparse and variational GPR approximations have been proposed, though they often trade off accuracy for speed [83]. Another open problem concerns multi-step uncertainty propagation, where posterior variance growth can lead to overly conservative control policies in predictive frameworks like MPC or DDP [84]. Furthermore, most

existing works treat uncertainty as stochastic rather than adversarial, leaving room for further integration of GPR with robust and game-theoretic methods such as Min-Max DDP. Continued research in these directions aims to bridge data-driven learning and robust optimal control, making GPR a central component in future adaptive and safe control architectures.

1.3.6 Stochastic Control Frameworks

Stochastic control provides a rigorous mathematical framework for decision-making in systems subject to randomness, disturbances, and measurement noise. Unlike deterministic control, stochastic control explicitly accounts for the probabilistic nature of system evolution, allowing controllers to optimize performance under uncertainty. Classical stochastic optimal control methods rely on dynamic programming and the stochastic Hamilton–Jacobi–Bellman (HJB) equation, which defines the optimal policy in expectation form [64]. However, solving the HJB equation analytically is often intractable for nonlinear or high-dimensional systems, motivating approximate and computationally tractable approaches such as stochastic differential dynamic programming (SDDP) and risk-sensitive control [85]. Early work by Theodorou and Todorov established stochastic DDP as an extension of deterministic DDP that includes second-order noise terms to model diffusion processes [45]. This formulation has since been applied in robotics, aerospace, and energy systems where environmental and sensor noise significantly influence control performance.

In recent years, there has been a resurgence of interest in stochastic control driven by advances in machine learning and probabilistic modeling. Study such as [13] extended the Min-Max DDP framework to stochastic nonlinear systems under adversarial disturbances, offering robustness guarantees through a min–max stochastic game formulation. Similarly, research [86] proposed a stochastic policy gradient control method

that approximates the stochastic HJB solution using deep learning for continuous control problems. Risk-sensitive and entropy-regularized formulations have also gained traction, providing a principled way to balance exploration and exploitation in uncertain environments [87]. Meanwhile, the integration of stochastic modeling with model predictive control (MPC) has produced predictive controllers that account for noise statistics and covariance propagation during real-time optimization [88]. Such approaches have been applied to UAVs, autonomous driving, and robotic manipulation, where controllers must act under dynamic and uncertain conditions.

Despite these significant developments, several challenges remain in stochastic control theory and application. The computation of multi-step expected costs and covariances remains computationally intensive, limiting scalability for large-scale systems. The derivation of closed-form policies under correlated noise and partial observability also remains difficult, particularly when system dynamics are nonlinear and non-Gaussian [89]. Recent research explores stochastic control under partial observability, combining Kalman-type filters or particle filters with stochastic DDP and stochastic ADP to enable estimation-based optimal control [90]. Another emerging direction involves the unification of stochastic and robust control, where the controller minimizes expected cost while accounting for worst-case uncertainties, bridging stochastic optimization and Min-Max game theory [91]. Collectively, these developments have revitalized stochastic control as a cornerstone for modern adaptive and learning-based frameworks, forming the probabilistic foundation upon which robust, data-driven, and safe control strategies are built.

1.3.7 Safe and Chance-Constrained Control Methods

Safety has become a fundamental requirement in modern control systems, particularly in autonomous and learning-based applications where controllers operate under

uncertainty, nonlinearity, and environmental disturbances. Traditional optimal control focuses on minimizing expected cost, but it does not guarantee the satisfaction of safety constraints such as collision avoidance, actuator saturation, or state bounds. To address these challenges, chance-constrained control introduces probabilistic constraints that ensure the probability of violating safety conditions remains below a prescribed threshold. This formulation enables the controller to explicitly account for uncertainty while maintaining tractable optimization [92]. The integration of chance constraints with model predictive control (MPC) has led to powerful frameworks for stochastic systems, allowing the computation of control inputs that satisfy safety constraints in expectation or with high confidence [93]. These methods are particularly effective in robotic motion planning, autonomous driving, and aerial vehicle navigation, where safety must be balanced with performance and adaptability.

Recent research has focused on extending chance-constrained control to nonlinear, time-varying, and learning-based systems. For instance, [94] proposed a scalable convex approximation of chance constraints for nonlinear systems with Gaussian noise, improving computational tractability in real-time applications. [95] developed a chance-constrained differential dynamic programming (CC-DDP) algorithm that ensures probabilistic safety in nonlinear trajectory optimization problems, demonstrating its effectiveness in robot motion planning. [96] integrated chance constraints into a stochastic dynamic programming framework for continuous-time systems, providing recursive feasibility guarantees and stability under bounded uncertainty. Moreover, recent works have incorporated distributionally robust optimization (DRO) to handle ambiguous or unknown uncertainty distributions, resulting in distributionally robust MPC (DR-MPC) formulations that maintain safety even when the underlying disturbance statistics are uncertain [97, 98].

Parallel advances have also emerged in safe reinforcement learning (Safe RL) and

risk-sensitive control, where safety constraints are enforced during learning or decision-making. Methods such as control barrier functions (CBFs) [99] and Lyapunov-based constraint enforcement [100] have been integrated with model-based and model-free reinforcement learning to ensure safety during policy adaptation. [101] Introduced an adaptive chance-constrained RL framework that ensures state and input constraints are satisfied with high probability while improving sample efficiency. These developments highlight a convergence between safe control, chance-constrained optimization, and learning-based methods. Despite notable progress, key challenges remain — including scalability to high-dimensional systems, handling non-Gaussian uncertainties, and guaranteeing recursive feasibility in online settings. Addressing these challenges continues to drive research in safe control theory and its integration into data-driven, robust, and stochastic optimization frameworks.

1.3.8 Distributed and Large-Scale Differential Game Control

In recent years, control and game-theoretic frameworks have shifted from centralized architectures to distributed and large-scale systems due to the proliferation of interconnected agents, networked subsystems, and cyber-physical infrastructures. Traditional differential-game formulations often assume that all players share global information and act in a centralized manner, which becomes impractical as the system size increases and subsystems are geographically dispersed. As a result, research in distributed differential games focuses on decentralizing decision-making, leveraging local communication, and designing scalable equilibrium or worst-case strategies under network constraints [102]. The interplay between local objectives and global coupling through interconnections introduces new analytical and computational challenges in stability, convergence, and equilibrium synthesis.

Several recent works have addressed these issues by developing architectures that

enable local agents to participate in differential games while preserving overall system performance. For instance, [48] proposed Distributed Differential Dynamic Programming Architectures for large-scale multi-agent systems, where each agent solves a local DDP subproblem and communicates consensus variables to neighbours, yielding scalable solutions that extend to thousands of vehicles or robots. [103] Studied differential-game-based distributed vibration control for large-scale structural systems, formulating the problem as a network of adversarial subsystems and deriving distributed Nash or min-max strategies. Surveys on aggregative games and networked control [104] further provide a broad perspective on how distributed algorithms for large networks and games behave under directed/undirected graphs, asynchronous updates, and disturbances. These developments demonstrate that game-theoretic control methods are being adapted to large-scale interconnected systems with scalability and locality in mind.

Despite these advances, key challenges remain at the intersection of large-scale control and differential games. First, the scalability of equilibrium computation remains a major hurdle: many methods still rely on solving high-dimensional coupled Riccati or Hamilton–Jacobi-Isaacs (HJI) equations which scale poorly. Second, information structure constraints (such as local sensing, delays, packet drops) in distributed systems complicate the design of consistent strategies and may compromise worst-case guarantees. Third, the integration of adversarial disturbances or uncertainties within the distributed game framework is underexplored, especially for nonlinear dynamics and heterogeneous agents. Finally, the transition from theory to real-world implementation—for example in smart grids, multi-vehicle fleets, and large industrial networks—demands algorithms that are both communication-efficient and robust to topology changes, disturbances, and cyber-attacks. Addressing these gaps is essential for advancing robust, scalable differential game control in large-scale networked

systems.

1.3.9 Nonzero-Sum and Multiplayer Differential Games

While zero-sum differential games have been widely studied for adversarial interactions, many real-world systems involve multiple decision-makers with distinct and sometimes partially cooperative objectives. These interactions are modeled as nonzero-sum differential games, where each player minimizes its own cost functional without necessarily maximizing the opponent's cost. This formulation captures realistic behaviors in multi-agent systems such as autonomous vehicle coordination, smart grid management, and networked robotics, where competition and cooperation coexist. The main analytical challenge in nonzero-sum differential games lies in solving coupled Hamilton–Jacobi (HJ) or Hamilton–Jacobi–Bellman (HJB) equations, whose solutions characterize the feedback Nash equilibrium strategies. The classical theory provides existence and uniqueness results for linear-quadratic games, but extending these to nonlinear and uncertain systems remains complex [105].

Recent advances have leveraged adaptive dynamic programming and reinforcement learning to approximate Nash equilibria in nonlinear and stochastic nonzero-sum settings. For example, [106] developed a Nash equilibrium control method for uncertain delay dynamic systems using adaptive parameterization to handle unknown delays and disturbances. [107] proposed an online Q-learning algorithm for linear-quadratic nonzero-sum stochastic differential games with unknown dynamics, where actor–critic neural networks approximate equilibrium strategies in continuous time. Similar learning-based formulations have been explored for partially observable systems and cooperative games with communication constraints [108]. In the stochastic domain, [109] investigated nonzero-sum differential games in investment, consumption, and reinsurance problems, deriving equilibrium strategies under risk-sensitive utilities.

In parallel, distributed and multi-agent game-theoretic formulations have emerged, where agents interact through shared state variables or coupling constraints, requiring decentralized computation of Nash equilibria in large-scale dynamic systems [110, 111].

Despite these promising developments, several challenges persist. The coupled nature of Nash conditions complicates convergence guarantees for nonlinear and data-driven games. Moreover, most current frameworks assume perfect information exchange or complete state observability, which is rarely available in practice. Achieving real-time scalability, incorporating uncertainty and safety constraints, and integrating learning-based adaptation into multi-player nonzero-sum dynamics remain active areas of research. Addressing these challenges is key to extending differential game theory toward robust, scalable, and cooperative–competitive control architectures for future autonomous and interconnected systems.

1.4 Research Gaps and Motivation Summary

Although substantial progress has been achieved in optimal control, adaptive control, stochastic optimization, and game-theoretic formulations, several critical research gaps remain unresolved. Existing Min-Max Differential Dynamic Programming (DDP) frameworks primarily rely on known system models and deterministic dynamics, limiting their applicability to real-world systems characterized by model uncertainties, stochastic disturbances, and partial observability. Moreover, most formulations address only a single aspect of robustness or scalability rather than providing a unified framework capable of handling multiple sources of uncertainty simultaneously. Specifically, the following gaps can be identified from the literature:

1-Model Dependence: Conventional Min-Max DDP requires explicit knowledge of system dynamics, making it unsuitable for data-driven or unknown-model systems.

2-Lack of Stochastic Robustness: Few existing approaches integrate stochastic dynamics, particularly those involving both drift and diffusion terms, into the Min-Max optimization framework.

3-Limited Observability Handling: Most works assume full-state feedback, whereas real-world systems often operate with partial or noisy measurements requiring output-feedback control.

4-Scalability Challenges: Current methods are computationally intensive and not easily extendable to large-scale or distributed systems with interacting subsystems.

5-Safety and Constraint Satisfaction: Existing formulations rarely incorporate probabilistic or chance-constrained safety guarantees in trajectory optimization.

6-Lack of Unified Structure: There is no comprehensive Min-Max DDP framework that simultaneously addresses data-driven modeling, stochasticity, output feedback, large-scale interactions, and safety considerations.

Motivated by these limitations, this dissertation aims to establish a unified, learning-based Min-Max DDP framework that integrates robustness, adaptability, and safety into a cohesive design. The overall objective is to make Min-Max DDP more data-driven, scalable, and practically implementable for uncertain and adversarial dynamical systems.

1.5 Research Objectives and Contributions

The primary objective of this dissertation is to develop a unified and learning-based Min-Max Differential Dynamic Programming (DDP) framework capable of addressing uncertainty, stochasticity, scalability, safety, and limited observability in nonlinear dynamic systems. Building upon the foundational principles of robust and game-theoretic control, the research advances Min-Max DDP into a comprehensive methodology suit-

able for real-world applications.

The specific objectives and corresponding contributions of this dissertation are summarized as follows:

1-Develop a Data-Driven Min-Max DDP Framework Using Gaussian Process Regression (GPR): Formulate a data-driven Min-Max DDP algorithm that removes the dependency on explicit system models. The unknown drift dynamics are approximated using Gaussian Process Regression, enabling the computation of optimal control updates directly from data.

2-Extend the Framework to Stochastic Systems with Drift and Diffusion Estimation: Introduce a data-driven stochastic Min-Max DDP method capable of handling stochastic dynamics by separately estimating drift and diffusion components. The drift is modeled using GPR, while a nonparametric binning estimator captures diffusion characteristics, allowing robust optimization under process noise.

3-Design a Sliding Mode Observer Based Min-Max DDP for Output Feedback Systems: Develop an observer-based Min-Max DDP that integrates a Higher-Order Sliding Mode (HOSM) observer to reconstruct unmeasured states. This formulation enables robust output-feedback control for systems with limited or noisy state information.

4-Formulate a Distributed Min-Max DDP for Large-Scale Interconnected Systems: Extend the Min-Max DDP algorithm to distributed architectures, enabling decentralized coordination among subsystems with mismatched interconnections. This distributed formulation ensures scalability while preserving local autonomy and robustness.

5-Introduce a Chance-Constrained Min-Max DDP for Safe Trajectory Optimization: Incorporate probabilistic safety guarantees through chance-constrained optimization, ensuring that the likelihood of constraint violation remains below a spec-

ified threshold. This approach integrates safety considerations directly within the Min-Max DDP framework.

6-Explore Adaptive and Nonzero-Sum Extensions for Multi-Agent Differential Games: Formulate and analyze adaptive and nonzero-sum variants of Min-Max DDP to handle cooperative–competitive multi-agent interactions. These extensions broaden the applicability of DDP to complex differential games and multi-player systems.

Collectively, these contributions advance the state of Min-Max DDP from a deterministic, model-dependent approach toward a data-driven, stochastic, scalable, safe, and adaptive control framework, bridging theoretical rigor with practical implementation.

1.6 Dissertation Organization

This dissertation is organized into eight chapters, each addressing a specific extension of the Min-Max Differential Dynamic Programming (DDP) framework to overcome key limitations identified in the literature. The remainder structure and focus of each chapter are summarized as follows:

Chapter 2 presents the **Min-Max Adaptive Dynamic Programming (ADP)** framework for zero-sum differential games. The chapter develops an adaptive critic-based formulation to approximate the Hamilton–Jacobi–Isaacs (HJI) solution iteratively, enabling robust control for nonlinear systems with adversarial disturbances.

Chapter 3 introduces the **Sliding-Mode-Observer-Based Min-Max DDP** for output feedback control problems. A Higher-Order Sliding Mode (HOSM) observer is employed to estimate unmeasured states, providing robustness against model uncertainties and measurement noise.

Chapter 4 develops a **Data-Driven Min-Max DDP** approach for nonlinear systems with unknown dynamics. Gaussian Process Regression (GPR) is utilized to approximate system drift dynamics, allowing the control law to be synthesized directly from data without requiring explicit model knowledge.

Chapter 5 extends the data-driven concept to stochastic environments by formulating a **Data-Driven Stochastic Game-Theoretic Min-Max DDP**. Both drift and diffusion dynamics are estimated—using GPR and a nonparametric binning method, respectively—yielding a fully data-driven stochastic optimal control strategy.

Chapter 6 proposes a **Distributed Min-Max DDP** for large-scale systems with mismatched interconnections. The framework decomposes the global optimization problem into local subproblems, enabling scalable and decentralized control of multi-agent or networked systems.

Chapter 7 introduces a **Chance-Constrained Game-Theoretic Min-Max DDP** for safe trajectory optimization under uncertainty. Probabilistic safety constraints are embedded into the optimization process to ensure that the trajectory remains within acceptable risk bounds with high confidence.

Chapter 8 concludes with the development of a **Continuous-Time Nonzero-Sum Differential Game DDP**, extending the Min-Max DDP methodology to multi-player settings where agents exhibit both cooperative and competitive interactions. The chapter also discusses potential future directions in adaptive, safe, and distributed control.

Chapter 9, finally, concludes the dissertation with a summary of findings, key theoretical insights, and recommendations for future research in extending Min-Max DDP to broader applications in robotics, autonomous systems, and large-scale stochastic control.

Chapter 2

Min-Max Adaptive Dynamic Programming for Zero-Sum Differential Games

2.1 Introduction

Adaptive Dynamic Programming (ADP) has gained much attention for its ability to find approximate solution to optimal control problems with nonlinear systems, and has been applied to many problems in both industry and academia [112, 113, 114]. ADP algorithms utilize a function approximation structure to estimate the Hamilton–Jacobi–Bellman (HJB) equation [115] associated with the optimal control problem, and they can solve the problem through an online data-based method where the detailed knowledge of the plant is unknown. The optimality and learning capabilities of ADP has been explored in many systems including power systems, engine control, robotics, etc. [116, 117]. For instance, an ADP algorithm for optimal control of an infinite-horizon discrete nonlinear system with finite approximation errors is proposed in [118]. In [119], a Recurrent Neural Network (RNN) based ADP is designed to solve the optimal problem of a class of nonlinear discrete systems with control constraints. The control problem for a low-level controller of an unmanned vehicle using reinforcement

learning is investigated in [120]. Several variations of ADP have also been developed such as the heuristic dynamic programming (HDP) [121], the dual heuristic dynamic programming (DHP) [122], and the globalized dual heuristic dynamic programming (GDHP) [123, 124], which are generalization of dynamic programming with neural reinforcement learning approaches. In addition, ADP control policies can be applied to the systems where the model is unknown. To solve the ADP problems with unknown dynamics, Reinforcement Learning (RL) is commonly used [125]. RL is a machine learning method that learns the optimal control policies based on observations received from the environment [126, 127]. One of the significant methods in RL is the policy iteration (PI), which includes policy evaluation and policy improvement [128]. An online policy iteration based on H_∞ robust optimization is developed for nonlinear systems in [129]. The PI method for problem of optimal control tracking without system information is studied in [130].

One extension of the optimal control problem is the min-max or zero-sum differential game [131], which involves two players in conflict to optimize a performance index in the context of dynamical systems. The zero-sum differential game also has a direct connection with robust control [132, 133]. Several works have been done to extend the ADP algorithm to solve zero-sum differential game problems. In [134], ADP is applied to an infinite-horizon discrete-time system by solving the associated Hamilton-Jacobi-Isaacs (HJI) equation, where the dynamics and value function are known. Another ADP algorithm is designed for discrete-time affine systems with known dynamics in [135]. [136] focuses on a model-free approach for linear systems and utilizes the ADP method to find optimal control solutions. In [137], an iterative ADP technique with only the critic neural network structure that approximates the value function is proposed for systems with known dynamics. Additionally, [138] addresses the Nash equilibrium solution for two-player zero-sum games with known linear dynamics, while

[139] proposes an ADP method for systems with known value function and nonlinear models. Adaptive critic approximate dynamic programming designs are studied in ([140]) to solve the discrete-time zero-sum game. But this algorithm is not applicable to complex systems with unknown dynamics. In ([141]), adaptive critic design is proposed to approximate the online Nash equilibrium solution for the robust trajectory tracking control of non-zero-sum (NZS) games for continuous-time uncertain nonlinear systems. However, this algorithm cannot solve problems of zero-sum games with disturbances and uncertainties. In ([139]), ADP is proposed to solve a continuous-time nonlinear two-person zero-sum differential games problem with known dynamics. Similarly, ([134]) proposes the ADP for infinite-horizon discrete-time zero-sum games with known dynamics. The stability proof based on the Lyapunov theory is not discussed in this Chapter. In ([117]), the problem of constrained cost for discrete-time nonlinear system is studied, where the neural network is used to approximate only the value function. There are other works that have been done in ADP, nonzero-sum games and zero-sum games ([142, 143, 141, 144, 145, 146]), but there has been no investigation on an ADP algorithm that solves for the zero-sum differential game with approximation on both the value function and nonlinear dynamics. In this Chapter, the problem of min-max adaptive dynamic programming for zero-sum differential games based on the PI algorithm using recurrent neural network and critic neural network for the unknown dynamics and value function is investigated.

The contribution of the paper is divided into several aspects

1) A min-max zero-sum differential game with unknown nonlinear dynamics is solved to obtain the control policies through the HJI equation.

2) The PI approach is used to solve the HJI equation associated with the zero-sum differential game.

3) The online recurrent neural network is employed to approximate the nonlinear

dynamics of the system.

4) The neural network is employed to approximate the value function.

5) The error between the approximated dynamics by RNN and dynamics of the system is proved to converge to 0 as $t \to \infty$.

6) The Uniform Ultimate Bounded (UUB) stability of the closed-loop system is proved based on the Lyapunov theory for the zero-sum game problem.

The rest of this Chapter is organized as follows. In Section 2.2, the primary problem is formulated. In Section 2.3, an online recurrent neural network is employed to approximate the dynamics of the system. The online PI algorithm for the zero-sum games and the critic neural network for the estimation of the value function are presented in 2.4. The stability analysis of the method is shown in 2.5. Finally, in section 2.6, simulation results of the proposed algorithm are demonstrated.

2.2 Problem Formulation

Consider the dynamics of the zero-sum differential game problem as follows

$$\dot{x}(t) = f(x(t)) + g_1(x(t))u(t) + g_2(x(t))v(t), \tag{2.1}$$

where $x(t) \in \Re^n$, $t \in [t_0, t_f]$, is the state. $u \subset \Re^{m_u}$ and $v \subset \Re^{m_v}$ are the control policies. $f \in \Re^n$, $g_1 \in \Re^{n \times m_u}$ and $g_2 \in \Re^{n \times m_v}$ are polynomial mapping with $f(0) = 0$. It is assumed that $f(x(t))$, $g_1(x(t))$ and $g_2(x(t))$ are unknown, continuous, and the system is stabilizable on a compact set Ω.

The cost function of the differential game is defined as

$$J(x_0, u, v) = \int_{t_0}^{t_f} \mathcal{L}(x(t), u(t), v(t))dt, \tag{2.2}$$

where u tries to minimize the cost function while v tries to maximize it. The running cost with respect to the control policies is introduced as

$$\mathcal{L}(x(t), u(t), v(t)) = x(t)^T P x(t) + u(t)^T R u(t) - v(t)^T Q v(t), \tag{2.3}$$

where $P \in \Re^{n \times n}$, $R \in \Re^{m_u \times m_u}$, and $Q \in \Re^{m_v \times m_v}$ are positive definite matrices.

The optimal control policies correspond to the Nash equilibrium of the differential game [147], and they can be computed through the following Hamiltonian-Jacobi-Isaacs (HJI) equation

$$0 = \mathcal{L}(x, u, v) + (\nabla_x V)^T (f(x) + g_1(x)u + g_2(x)v), \tag{2.4}$$

where the time dependency on the variables are omitted for brevity. In (2.4), V is the value function, $\nabla_x V$ is gradient of value function based on x, and $V(x_0)$ is the optimal cost function at the initial condition x_0, that is

$$V(x_0) = \min_u \max_v J(x_0, u, v) = \min_u \max_v \int_{t_0}^{\infty} (x^T P x + u^T R u - v^T Q v) d\tau. \tag{2.5}$$

The corresponding Hamiltonian function is defined as

$$H = \mathcal{L}(x, u, v) + (\nabla_x V)^T (f(x) + g_1(x)u + g_2(x)v). \tag{2.6}$$

The optimal control policies can be obtained by setting

$$\frac{\partial H}{\partial u} = 0, \qquad \frac{\partial H}{\partial v} = 0, \tag{2.7}$$

from which we have

$$u = -\frac{1}{2}R^{-1}g_1^T(x)\nabla_x V(x), \tag{2.8}$$

$$v = \frac{1}{2}Q^{-1}g_2^T(x)\nabla_x V(x). \tag{2.9}$$

Substituting (2.8) and (2.9) into (2.4), the HJI equation can then be expressed as

$$\begin{aligned} 0 &= x^T Px + (\nabla_x V)^T f(x) \\ &- \frac{1}{4}(\nabla_x V)^T g_1(x)R^{-1}g_1^T(x)\nabla_x V + \frac{1}{4}(\nabla_x V)^T g_2(x)Q^{-1}g_2^T(x)\nabla_x V, \end{aligned} \tag{2.10}$$

where $V(0) = 0$.

2.3 Online recurrent neural network for the system dynamics

Since the model of the system is assumed to be unknown, we need to approximate the dynamics of the system. In particular, we utilize the recurrent neural network (RNN) to learn the dynamics. From input-output data, the nonlinear dynamics of the system are approximated. Specifically, the unknown dynamics of the system is assumed to have the following form

$$\dot{x} = C^T x(t) + A^T \sigma_f(x(t)) + C_u^T \sigma_{g1}(x(t))u(t) + C_v^T \sigma_{g2}(x(t))v(t) + A_u^T - \epsilon(t), \tag{2.11}$$

where $\epsilon(t)$ is bounded and C^T, A^T, C_u^T, C_v^T, and A_u are unknown weigh matrices. σ_f, σ_{g1} and σ_{g2} represent the activation functions.

Assumption 1. *The term $\epsilon(t)$ is considered as a upper bounded by a function of*

modeling error such that

$$\epsilon^T(t)\epsilon(t) \leq \epsilon_M(t) \triangleq \lambda e_m^T(t)e_m(t), \tag{2.12}$$

where $e_m(t) = x(t) - \hat{x}(t)$ is system modeling error, $\epsilon_M(t)$ is the upper bounded by a function that is equal to $\lambda e_m^T(t)e_m(t)$, and λ is the bounded constant target value.

The RNN approximation model for the system is then represented by

$$\begin{aligned}\dot{\hat{x}} &= \hat{C}^T(t)\hat{x}(t) + \hat{A}^T(t)\sigma_f(\hat{x}(t)) \\ &+ \hat{C}_u^T(t)\sigma_{g1}(\hat{x}(t))u(t) + \hat{C}_v^T(t)\sigma_{g2}(\hat{x}(t))v(t) + \hat{A}_u^T(t) - \mathcal{V}(t),\end{aligned} \tag{2.13}$$

where $\hat{x}$ is the estimate of the state vector, $\hat{C}^T, \hat{A}^T, \hat{C}_u^T, \hat{C}_v^T$, and $\hat{A}_u$ are estimates of weight matrices, and $\mathcal{V}(t)$ is defined as

$$\mathcal{V}(t) = Se_m(t) + \frac{\hat{\lambda}(t)e_m(t)}{e_m^T(t)e_m(t) - \zeta}, \tag{2.14}$$

where $e_m(t) = x(t) - \hat{x}(t)$ is the system modeling error, S is a designed matrix, $\hat{\lambda}$ is an adjustable parameter, and $\zeta > 1$ is a constant parameter.

From equations (2.11) and (2.13), the modeling error dynamics of $e_m(t)$ can be calculated as

$$\begin{aligned}\dot{e}_m(t) &= C^T e_m(t) + \tilde{C}^T(t)\hat{x}(t) + A^T\tilde{\sigma}_f(e_m(t)) + \tilde{A}^T(t)\sigma_f(\hat{x}(t)) \\ &+ C_u^T\tilde{\sigma}_{g1}(e_m(t)) + \tilde{C}_u^T(t)\sigma_{g1}(\hat{x}(t)) + C_v^T\tilde{\sigma}_{g2}(e_m(t)) + \tilde{C}_v^T(t)\sigma_{g2}(\hat{x}(t)) \\ &+ \tilde{A}_u^T(t) + \epsilon(t) + Se_m(t) - \frac{\tilde{\lambda}(t)e_m(t)}{e_m^T(t)e_m(t) + \zeta} + \frac{\lambda e_m(t)}{e_m^T(t)e_m(t) + \zeta},\end{aligned} \tag{2.15}$$

where $\tilde{C}(t) = C - \hat{C}(t)$, $\tilde{A}(t) = A - \hat{A}(t)$, $\tilde{C}_u(t) = C_u - \hat{C}_u(t)$, $\tilde{C}_v(t) = C_v - \hat{C}_v(t)$,

$\tilde{A}_u(t) = A_u - \hat{A}_u(t)$, $\tilde{\sigma}_f(e_m(t)) = \sigma_f(x(t)) - \sigma_f(\hat{x}(t))$, $\tilde{\sigma}_{g1}(e_m(t)) = \sigma_{g1}(x(t)) - \sigma_{g1}(\hat{x}(t))$, $\tilde{\sigma}_{g2}(e_m(t)) = \sigma_{g2}(x(t)) - \sigma_{g2}(\hat{x}(t))$, and $\tilde{\lambda}(t) = \lambda - \hat{\lambda}(t)$. The activation functions are assumed to be increasing functions that satisfy the following,

$$
\begin{aligned}
&0 \leq \sigma_f(x) - \sigma_f(y) \leq k(x-y), \\
&0 \leq \sigma_{g1}(x) - \sigma_{g1}(y) \leq k(x-y), \\
&0 \leq \sigma_{g2}(x) - \sigma_{g2}(y) \leq k(x-y), \\
&\forall x, y \in R \text{ and } x \geq y, k > 0.
\end{aligned} \tag{2.16}
$$

Some extra assumptions are made to help us prove later that the modeling error converges to zero.

Assumption 2. *All the weights defined in the approximated dynamics (2.13) are bounded.*

Assumption 3. *The control inputs are bounded.*

Theorem 1. *Let the parameters $\hat{C}^T, \hat{A}^T, \hat{C}_u^T, \hat{C}_v^T$, $\hat{A}_u$ and $\hat{\lambda}$ in (2.13) be updated iteratively as follows*

$$\dot{\hat{C}}(t) = \Gamma_1 \hat{x}(t) e_m^T(t), \tag{2.17a}$$

$$\dot{\hat{A}}(t) = \Gamma_2 \sigma_f(\hat{x}(t)) e_m^T(t), \tag{2.17b}$$

$$\dot{\hat{C}}_u(t) = \Gamma_3 \sigma_{g1}(\hat{x}(t)) u(t) e_m^T(t), \tag{2.17c}$$

$$\dot{\hat{C}}_v(t) = \Gamma_4 \sigma_{g2}(\hat{x}(t)) v(t) e_m^T(t), \tag{2.17d}$$

$$\dot{\hat{A}}_u(t) = \Gamma_5 e_m^T(t), \tag{2.17e}$$

$$\dot{\hat{\lambda}} = \Gamma_6 \frac{e_m^T(t) e_m(t)}{e_m^T(t) e_m(t) + \zeta}, \tag{2.17f}$$

where Γ_1, Γ_2, Γ_3, Γ_4, Γ_5, and Γ_6 are positive definite matrices. Then, the modeling

error $\tilde{x}$ converges to 0 asymptotically as $t \to \infty$.

Proof. Consider the Lyapunov function

$$V(t) = V_1(t) + V_2(t), \tag{2.18}$$

where

$$V_1(t) = \frac{1}{2} e_m^T(t) e_m(t), \tag{2.19}$$

$$V_2(t) = \frac{1}{2} tr\{\tilde{C}^T(t)\Gamma_1^{-1}\tilde{C}(t) + \tilde{A}^T(t)\Gamma_2^{-1}\tilde{A}(t) + \tilde{C}_u^T(t)\Gamma_3^{-1}\tilde{C}_u(t) + \tilde{C}_v^T(t)\Gamma_4^{-1}\tilde{C}_v(t) + \tilde{A}_u^T(t)\Gamma_5^{-1}\tilde{A}_u(t)\} + \frac{1}{2}\tilde{\lambda}^T(t)\Gamma_6^{-1}\tilde{\lambda}(t). \tag{2.20}$$

By taking the time derivation of (2.18) along the trajectory of the error system (2.15), we have

$$\begin{aligned}\dot{V}_1(t) &= e_m^T(t)C^T e_m(t) + e_m^T(t)\tilde{C}^T(t)\hat{x}(t) + e_m^T(t)A^T\tilde{\sigma}_f(e_m(t)) + e_m^T(t)\tilde{A}^T(t)\sigma_f(\hat{x}(t)) \\ &+ e_m^T(t)C_u^T\tilde{\sigma}_{g1}(e_m(t)) + e_m^T(t)\tilde{C}_u^T(t)\sigma_{g1}(\hat{x}(t)) + e_m^T(t)C_v^T\tilde{\sigma}_{g2}(e_m(t)) \\ &+ e_m^T(t)\tilde{C}_v^T(t)\sigma_{g2}(\hat{x}(t)) + e_m^T(t)\tilde{A}_u^T(t) + e_m^T(t)\epsilon(t) + e_m^T(t)Se_m(t) \\ &- \frac{e_m^T(t)\tilde{\lambda}(t)e_m(t)}{e_m^T(t)e_m(t)+\zeta} + \frac{\lambda e_m^T(t)e_m(t)}{e_m^T(t)e_m(t)+\zeta}.\end{aligned} \tag{2.21}$$

From (2.16) we get

$$e_m^T(t)A^T\tilde{\sigma}_f(e_m(t)) \le \frac{1}{2}e_m^T(t)A^TAe_m(t) + \frac{1}{2}k^2 e_m^T(t)e_m(t), \tag{2.22a}$$

$$e_m^T(t)C_u^T\tilde{\sigma}_{g1}(e_m(t)) \le \frac{1}{2}e_m^T(t)C_u^TC_ue_m(t) + \frac{1}{2}k^2 e_m^T(t)e_m(t), \tag{2.22b}$$

$$e_m^T(t)C_u^T\tilde{\sigma}_{g2}(e_m(t)) \le \frac{1}{2}e_m^T(t)C_v^TC_ve_m(t) + \frac{1}{2}k^2 e_m^T(t)e_m(t), \tag{2.22c}$$

From Assumption 1, we have

$$e_m^T(t)\epsilon(t) \leq \frac{1}{2}e_m^T(t)e_m(t) + \frac{1}{2}\epsilon^T(t)\epsilon(t) \leq \frac{1}{2}e_m^T(t)e_m(t) + \frac{1}{2}\lambda e_m^T(t)e_m(t). \tag{2.23}$$

Substitute (2.22) and (2.22) into (2.21), we obtain

$$\begin{aligned}\dot{V}_1(t) \leq\ & e_m^T(t)C^T e_m(t) + e_m^T(t)\tilde{C}^T(t)\hat{x}(t) + \frac{1}{2}e_m^T(t)A^T A e_m(t) + \frac{1}{2}e_m^T(t)C_u^T C_u e_m(t) \\ & + \frac{1}{2}e_m^T(t)C_v^T C_v e_m(t) + \left(\frac{1}{2} + \frac{1}{2}\lambda + \frac{3}{2}k^2\right)e_m^T(t)e_m(t) + e_m^T(t)\tilde{A}^T(t)\sigma_f(\hat{x}(t)) \\ & + e_m^T(t)\tilde{C}_u^T(t)\sigma_{g1}(\hat{x}(t)) + e_m^T(t)\tilde{C}_v^T(t)\sigma_{g2}(\hat{x}(t)) + e_m^T(t)\tilde{A}_u^T(t) + e_m^T(t)Se_m(t) \\ & - \frac{e_m^T(t)\tilde{\lambda}(t)e_m(t)}{e_m^T(t)e_m(t)+\zeta} + \frac{\lambda e_m^T(t)e_m(t)}{e_m^T(t)e_m(t)+\zeta}.\end{aligned} \tag{2.24}$$

Computation on the time derivative of $V_2(t)$ yields

$$\begin{aligned}V_2(t) = tr\Big\{ & \tilde{C}^T(t)\Gamma_1^{-1}\dot{\tilde{C}}(t) + \tilde{A}^T(t)\Gamma_2^{-1}\dot{\tilde{A}}(t) + \tilde{C}_u^T(t)\Gamma_3^{-1}\dot{\tilde{C}}_u(t) \\ & + \tilde{C}_v^T(t)\Gamma_4^{-1}\dot{\tilde{C}}_v(t) + \tilde{A}_u^T(t)\Gamma_5^{-1}\dot{\tilde{A}}_u(t)\Big\} + \tilde{\lambda}^T(t)\Gamma_6^{-1}\dot{\tilde{\lambda}}(t),\end{aligned} \tag{2.25}$$

Combining (2.24) and (2.25), we get

$$\begin{aligned}\dot{V}(t) \leq e_m^T(t)\Bigg(& C^T + \frac{1}{2}A^T A + \frac{1}{2}C_u^T C_u + \frac{1}{2}C_v^T C_v + \left(\frac{1}{2} + \frac{1}{2}\lambda + \frac{1}{2}k^2\right)I_n \\ & + S + \frac{\lambda I_n}{e_m^T(t)e_m(t)+\zeta}\Bigg)e_m(t),\end{aligned} \tag{2.26}$$

where I_n is the identity matrix.

Since $\zeta > 1$, let

$$\Pi = C^T + \frac{1}{2}A^T A + \frac{1}{2}C_u^T C_u + \frac{1}{2}C_v^T C_v + \left(\frac{1}{2} + \frac{3}{2}\lambda + \frac{1}{2}k^2\right)I_n + S. \tag{2.27}$$

When the designed matrix S is selected such that $\Pi < 0$, then it can be concluded that $\dot{V} \leq e_m^T(t)\Pi e_m(t) < 0$. Therefore, $e_m(t)$ converges to 0 as $t \to \infty$. This completes the proof. □

Since $e_m(t) \to 0$ as $t \to \infty$ from Theorem 1, it can be concluded that $\mathcal{V}(t) \to 0$ as $t \to \infty$. Furthermore, $\dot{\hat{C}}(t), \dot{\hat{A}}_u(t), \dot{\hat{C}}_u(t), \dot{\hat{C}}_v(t)$, and $\dot{\hat{A}}(t)$ converge to 0 as $e_m(t) \to 0$, which means that all the weights will converge to constants, and be denoted as $\bar{C}, \bar{A}, \bar{C}_u, \bar{C}_v$, and $\bar{A}_u$, respectively. Henceforth, the system (2.1) is approximated by

$$\dot{\hat{x}}(t) = \bar{C}^T\hat{x}(t) + \bar{A}^T\sigma_f(\hat{x}(t)) + \bar{C}_u^T\sigma_{g1}(\hat{x}(t))u(t) + \bar{C}_v^T\sigma_{g2}(\hat{x}(t))v(t) + \bar{A}_u^T. \tag{2.28}$$

2.4 Online PI algorithm and critic NN for the value function

To find the optimal control policies, the HJI equation associated with the differential game problem need to be solved. However, the HJI equation cannot be solved analytically due to its nonlinear nature. Instead, a numerical online policy iteration (PI) algorithm [148] is utilized for the HJI equation, as shown in Algorithm 1.

Algorithm 1 (Online PI for zero-sum games)

Step 1: Set $k = 0$ *and start with initial admissible control* u^0 *and* v^0*, and* $V^0(.) = 0$.
Step 2: For control policies u^k, v^k*, solve the following nonlinear Lyapunov equation to get* V^{k+1}.
$0 = \mathcal{L}(x, u^k, v^k) + (\nabla_x V^{k+1})^T(f(x) + g_1(x)u + g_2(x)v), \quad V^{k+1}(0) = 0$
Step 3: Update the control policies simultaneously as.
$u^{k+1} = -\frac{1}{2}R^{-1}g_1^T(x)\nabla_x V^{k+1}$
$v^{k+1} = \frac{1}{2}Q^{-1}g_2^T(x)\nabla_x V^{k+1}$
Step 4: If max $(\|V^{k+1} - V^k\|) \leq \epsilon$*, stop and receive the approximate optimal control policies*; *else, set* k=k+1, *and go to step* 2.

In order to apply the PI algorithm to our problem, we need to approximate the

value function of the system. To this end, we utilize the critic NN to estimate the value function. The continuous differentiable value function $V(x)$ can be approximated by the feedforward NN as

$$V = W^T\psi(x) + \delta(x), \tag{2.29}$$

where $W \in R^k$, $\psi(x)$,$\delta(x)$, and k are the weight, activation function, approximation error, and number of neurons in the hidden layer. The derivation of (2.29) with respect to x is

$$\frac{\partial V}{\partial x} = \left(\frac{\partial \psi}{\partial x}\right)^T W + \frac{\partial \delta(x)}{\partial x} = \nabla_x \psi^T W + \nabla_x \delta. \tag{2.30}$$

Hence, the HJI equation (2.4) can be rewritten as

$$0 = \mathcal{L}(x(t), u(t), v(t)) + (W^T\nabla_x\psi + (\nabla_x\delta)^T)\dot{x}. \tag{2.31}$$

Substitute $\dot{x}$ with the RNN model (2.28), we obtain

$$\begin{aligned} 0 = \mathcal{L}(x(t), u(t), v(t)) + (W^T\nabla_x\psi + (\nabla_x\delta)^T)\Big(\bar{C}^T(t)x(t) + \bar{A}^T(t)\sigma_f(x(t)) \\ + \bar{C}_u^T(t)\sigma_{g1}(x(t))u(t) + \bar{C}_v^T\sigma_{g2}(x(t))v(t) + \bar{A}_u^T(t)\Big). \end{aligned} \tag{2.32}$$

Notice that the value function can be approximated as $\hat{V} = \hat{W}^T\psi(x)$, then the approximated Hamiltonian function associated with the HJI equation is given by

$$\begin{aligned} \hat{H} = \mathcal{L}(x(t), u(t), v(t)) + \hat{W}^T\nabla_x\psi\Big(\hat{C}^T(t)\hat{x}(t) + \hat{A}^T(t)\sigma_f(\hat{x}(t)) \\ + \hat{C}_u^T(t)\sigma_{g1}(\hat{x}(t))u(t) + \hat{C}_v^T(t)\sigma_{g2}(\hat{x}(t))v(t) + \hat{A}_u^T(t)\Big). \end{aligned} \tag{2.33}$$

The goal here is to find $\hat{W}$ in order to minimize the following objective function

$$E(\hat{W}) = \frac{1}{2}\hat{H}^T\hat{H}. \tag{2.34}$$

The goal here is to find $\hat{W}$ in order to minimize the following objective function

$$E(\hat{W}) = \frac{1}{2}\hat{H}^T\hat{H}. \tag{2.35}$$

By using the standard gradient descent algorithm in the NN, the tuning law for the weights is given by

$$\dot{\hat{W}} = -\eta\left(\frac{\partial E}{\partial \hat{W}}\right)^T = -\eta\theta(\mathcal{L} + \hat{W}^T\theta), \tag{2.36}$$

where $\eta > 0$ is the learning rate of the critic NN and

$$\begin{aligned}\theta = \nabla_x\psi\Big(&\bar{C}^T(t)x(t) + \bar{A}^T(t)\sigma_f(x(t)) + \bar{C}_u^T(t)\sigma_{g1}(x(t))u(t)\\ &+ \bar{C}_v^T(t)\sigma_{g2}(x(t))v(t) + \bar{A}_u^T(t)\Big).\end{aligned} \tag{2.37}$$

There is a positive constant θ_M such that $\|\theta\| \leq \theta_M$. According to (2.6), the Hamiltonian function become

$$\begin{aligned}H = \mathcal{L}(x(t), u(t), v(t)) + W^T\nabla_x\psi\Big(&\hat{C}^T(t)\hat{x}(t) + \hat{A}^T(t)\sigma_f(\hat{x}(t))\\ &+ \hat{C}_u^T(t)\sigma_{g1}(\hat{x}(t))u(t) + \hat{C}_v^T(t)\sigma_{g2}(\hat{x}(t))v(t) + \hat{A}_u^T(t)\Big) \triangleq e_H,\end{aligned} \tag{2.38}$$

where the residual error due to NN approximation is

$$e_H = -(\nabla_x \delta)^T \Bigg(\hat{C}^T(t)\hat{x}(t) + \hat{A}^T(t)\sigma_f(\hat{x}(t)) + \hat{C}_u^T(t)\sigma_{g1}(\hat{x}(t))u(t) + \hat{C}_v^T(t)\sigma_{g2}(\hat{x}(t))v(t) + \hat{A}_u^T(t) \Bigg). \tag{2.39}$$

Let the weight estimation error of the critic NN be $\tilde{W} = W - \hat{W}$, we can have the following error dynamics

$$\dot{\tilde{W}} = \eta\theta \Bigg(e_H - \tilde{W}^T\theta \Bigg). \tag{2.40}$$

To obtain the minimizing control u and maximizing control v, we substitute (2.8) and (2.9) with the approximated dynamics and value function to get the corresponding approximated control policies

$$\hat{u} = -\frac{1}{2}R^{-1}\bar{C}_u^T(t)\sigma_{g1}(\hat{x}(t))(\nabla_x \psi^T \hat{W}). \tag{2.41}$$

$$\hat{v} = \frac{1}{2}Q^{-1}\bar{C}_v^T(t)\sigma_{g2}(\hat{x}(t))(\nabla_x \psi^T \hat{W}). \tag{2.42}$$

Hence, the structure diagram of the min-max adaptive dynamic programming for the zero-sum differential game with unknown dynamics is illustrated in Fig. 2.1. With the approximation of the dynamics and the value function, the algorithm to solve the two player zero-sum differential game is presented in Algorithm 2.

Remark 1. *In the proposed approach described in Algorithm 2, there are some challenges associated with real-time implementation: 1) Initial conditions for NNs weights are chosen randomly and the implementation need to be run several times to obtain good initial weights; 2) Appropriate activation functions need to be selected for each problem to approximate the dynamics and value function with high accuracy. Hence,*

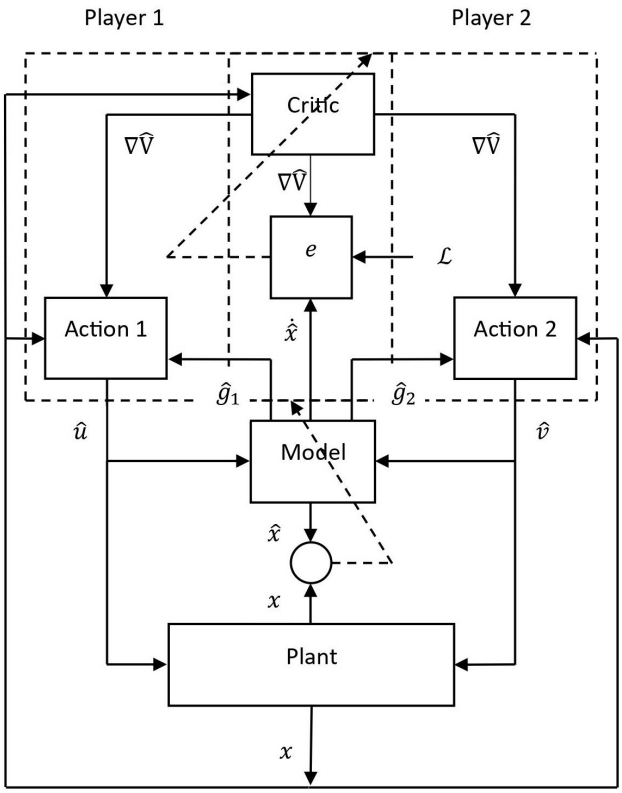


Figure 2.1: Schematic of the min-max ADP for a zero-sum differential game with unknown dynamics

we need to run the process several times with different activation functions to compare results and select the activation functions with the best outcome; 3) In the proposed approach, the algorithm for complex systems typically takes more than real-time to run and the implementation need to be fine tuned towards real-time execution.

Remark 2. *The computational complexity of the RNN algorithms are $O(N^3)$, where N is the number of nodes in the network [149]. The control update step has a complexity of $O(nm)$, where n is the dimension of the state and $m = max m_u, m_v$. m_u and m_v are the dimension of controls u and v, respectively. When N is larger than m and n, the computational complexity in the general case is $O(pN^3)$, where p is the number of iterations needed before it converges.*

Algorithm 2 (Procedure of Algorithm for Zero-Sum Differential Game)

Step 1: Set initial values for $\bar{C}$, $\bar{A}$, $\bar{C}_u$, $\bar{C}_v$, $\bar{A}_u$ *and* $\hat{\lambda}$. *Set a small value* ϵ.
Step 2: Set corresponding activation functions ψ, σ_f, σ_{g1}, *and* σ_{g2} *for critic NN and dynamics.*
Step 3: Compute u and v from equations (2.41) and (2.42), respectively.
Step 4: Compute approximated weights in the approximated dynamics (2.28) from equation (2.17).
Step 5: Solve the equation (2.36) to obtain $\hat{W}$.
Step 6: Integrate Eq. (2.28) forward to find $\hat{x}$.
Step 7: If $\|V^{k+1} - V^k\| \leq \epsilon$, *stop and return the approximate optimal control policies* u, v *and state* $\hat{x}$; *else, go back to step* 3.

2.5 Stability analysis

The Uniform Ultimate Bounded (UUB) stability of the closed-loop system can be proved based on the Lyapunov approach. Towards this end, we first prove that the weight estimate error of the critic NN is UUB.

Theorem 2. *Consider the system (2.28), where the initial weights of the critic NN are chosen to generate an initial admissible control pair. Then, the weight estimate error of the critic NN is UUB, if we update the weights of critic NN by (2.36) and control policies by (2.41) and (2.42).*

Proof. Consider the following Lyapunov function

$$S(t) = \frac{tr(\tilde{W}^T \tilde{W})}{2\eta} \triangleq S(t). \tag{2.43}$$

The time derivatives of the Lyapunov function along the trajectories of the error dy-

namics (2.40) is

$$\begin{aligned}\dot{S}(t) &= \frac{1}{\eta} tr(\tilde{W}^T \dot{\tilde{W}}) \\ &= \frac{1}{\eta} tr\left(\tilde{W}^T [\eta\theta(e_H - \tilde{W}^T\theta)]\right),\end{aligned} \tag{2.44}$$

according to $\theta_m \leq \|\theta\| \leq \theta_M$

$$\dot{S}(t) \leq -\left(\theta_m^2 - \frac{\eta}{2}\theta_M^2\right)\|\tilde{W}\|^2 + \frac{e_H^2}{2\eta}, \tag{2.45}$$

if the learning rate is selecting that satisfy

$$\eta \leq \frac{2\theta_m^2}{\theta_M^2},$$

and given that the following hold

$$\|\tilde{W}\| > \sqrt{\frac{e_H^2}{\eta\left(2\theta_m^2 - \eta\theta_M^2\right)}}, \tag{2.46}$$

then $\dot{S}(t) < 0$. Using Lyapunov theory, it can be concluded that the weight estimation error of the critic NN $\|\tilde{W}\|$ is UUB. □

The UUB stability of the state is addressed in the following theorem.

Theorem 3. *Consider the system (2.28) and $(\xi_1, \xi_2) \in R$ as computable constants, the initial weights of the critic NN are chosen to generate an initial admissible control pair. For initial condition x_0, there exists a time $T(x_0, B)$ such that $x(t)$ is UUB, where the*

bounded B is given by

$$\|x(t)\|\sqrt{\frac{\xi_1+\xi_2}{eig_{\min}\{P\}}} \triangleq B, t \geq t_0+T, (\xi_1,\xi_2) \in R^+, \tag{2.47}$$

if we update the weights of critic NN by (2.36) and control policies by (2.41) and (2.42).

Proof. To illustrate the stability of the approximate control policies (2.41) and (2.42), we take the derivatives of $V(t)$ along the trajectories generated by the approximate control policies $\hat{u}$ and $\hat{v}$ as

$$\dot{V}(t) = (\nabla_x V(x))^T (f(x) + g_1(x)\hat{u} + g_2(x)\hat{v}). \tag{2.48}$$

Considering (2.10)

$$\begin{aligned}\dot{V}(t) &= -x^T P x + \frac{1}{4}(\nabla_x V)^T g_1 R^{-1} g_1^T (\nabla_x V) \\ &- \frac{1}{4}(\nabla_x V)^T g_2 Q^{-1} g_2^T (\nabla_x V) + (\nabla_x V)^T g_1 \hat{u} + (\nabla_x V)^T g_2 \hat{v}.\end{aligned} \tag{2.49}$$

Considering (2.8), (2.9), (2.41), and (2.42) we have

$$\dot{V}(t) = -x^T P x + (\nabla_x V)^T g_1(-\frac{1}{2}u + \hat{u}) + (\nabla_x V)^T g_2(-\frac{1}{2}v + \hat{v}). \tag{2.50}$$

Considering (2.8), (2.9), (2.30), (2.41), (2.42), and the following equations

$$\begin{aligned} g_1 &= C_u^T \sigma_{g1}(x(t)), \\ g_2 &= C_v^T \sigma_{g2}(x(t)), \end{aligned}$$

we have we have

$$\dot{V}(t) = -x^T Px + (\nabla_x \psi^T W + \nabla_x \delta)^T \Big[C_u^T \sigma_{g1} R^{-1} (\frac{1}{4}(C_u^T \sigma_{g1})^T (\nabla_x \psi^T W + \nabla_x \delta)$$
$$- \frac{1}{2}(C_u^T \sigma_{g1})^T (\nabla_x \psi^T \hat{W})) \Big]$$
$$+ (\nabla_x \psi^T W + \nabla_x \delta)^T \Big[C_v^T \sigma_{g2} Q^{-1} (-\frac{1}{4}(C_v^T \sigma_{g2})^T (\nabla_x \psi^T W + \nabla_x \delta)$$
$$+ \frac{1}{2}(C_v^T \sigma_{g2})^T (\nabla_x \psi^T \hat{W}) \Big], \tag{2.51}$$

which (2.51) can be written as

$$\dot{V}(t) = -x^T Px + (\nabla_x \psi^T W + \nabla_x \delta)^T \Big[C_u^T \sigma_{g_1} R^{-1} (\frac{1}{4}(C_u^T \sigma_{g1})^T \nabla_x \psi^T W$$
$$+ \frac{1}{4}(C_u^T \sigma_{g1})^T \nabla_x \delta - \frac{1}{2}(C_u^T \sigma_{g1})^T \nabla_x \psi^T W + \frac{1}{2}(C_u^T \sigma_{g1})^T \nabla_x \psi^T \tilde{W}) \Big]$$
$$+ (\nabla_x \psi^T W + \nabla_x \delta)^T \Big[C_v^T \sigma_{g2} Q^{-1} (-\frac{1}{4}(C_v^T \sigma_{g2})^T \nabla_x \psi^T W$$
$$- \frac{1}{4}(C_v^T \sigma_{g2})^T \nabla_x \delta + \frac{1}{2}(C_v^T \sigma_{g2})^T \nabla_x \psi^T W - \frac{1}{2}(C_v^T \sigma_{g2})^T \nabla_x \psi^T \tilde{W}) \Big], \tag{2.52}$$

Hence, (2.52) would be

$$\dot{V}(t) = -x^T Px + (\nabla_x \psi^T W + \nabla_x \delta)^T \Big[C_u^T \sigma_{g_1} R^{-1} (-\frac{1}{4}(C_u^T \sigma_{g1})^T \nabla_x \psi^T W$$
$$+ \frac{1}{4}(C_u^T \sigma_{g1})^T \nabla_x \delta + \frac{1}{2}(C_u^T \sigma_{g1})^T \nabla_x \psi^T \tilde{W}) \Big]$$
$$+ (\nabla_x \psi^T W + \nabla_x \delta)^T \Big[C_v^T \sigma_{g2} Q^{-1} (\frac{1}{4}(C_v^T \sigma_{g2})^T \nabla_x \psi^T W$$
$$- \frac{1}{4}(C_v^T \sigma_{g2})^T \nabla_x \delta - \frac{1}{2}(C_v^T \sigma_{g2})^T \nabla_x \psi^T \tilde{W}) \Big]$$
$$\triangleq -x^T Px + \varkappa_1 + \varkappa_2, \tag{2.53}$$

where $\tilde{W} = W - \hat{W}$. According to the bounded assumptions on weights are UUB, the

terms $\varkappa_1$ and $\varkappa_2$ have an upper bounded

$$\begin{aligned}
\varkappa_1 &\leq \|(\nabla_x \psi^T W + \nabla_x \delta)^T [C_u^T \sigma_{g_1} R^{-1} (-\frac{1}{4}(C_u^T \sigma_{g1})^T \nabla_x \psi^T W \\
&+ \frac{1}{4}(C_u^T \sigma_{g1})^T \nabla_x \delta + \frac{1}{2}(C_u^T \sigma_{g1})^T \nabla_x \psi^T \tilde{W})\| \leq \xi_1 \\
\varkappa_2 &\leq \|(\nabla_x \psi^T W + \nabla_x \delta)^T C_v^T \sigma_{g2} Q^{-1} (\frac{1}{4}(C_v^T \sigma_{g2})^T \nabla_x \psi^T W \\
&- \frac{1}{4}(C_v^T \sigma_{g2})^T \nabla_x \delta - \frac{1}{2}(C_v^T \sigma_{g2})^T \nabla_x \psi^T \tilde{W})\| \leq \xi_2
\end{aligned} \tag{2.54}$$

where $(\xi_1, \xi_2) \in R$ are a computable constant. Therefore, it can be said that $\dot{V}(t)$ is upper bounded by

$$\dot{V}(t) \leq -x^T P x + \xi_1 + \xi_2 \leq -\text{eig}_{\min}\{P\}\|x\|^2 + \xi_1 + \xi_2, \tag{2.55}$$

where $\text{eig}_{\min}\{P\}$ represents the least eigenvalue of the matrix P. For the value function $V(x(t))$, it can be shown that its derivative $\dot{V}(t)$ is negative whenever $x(t)$ lies outside the compact set $\Omega \triangleq \{x : \|x\| \leq \sqrt{\frac{\xi_1+\xi_2}{\text{eig}_{\min}\{P\}}}\}$. Then, the derivative $\dot{V}(t)$ is negative whenever $x(t)$ lies outside the compact set Ω, i.e., $\|x(t)\|$ is UUB. This completes the proof. □

2.6 Simulation Results

In this section we provide three examples to show the effectiveness of the proposed algorithm.

2.6.1 Mathematical Example

In the first example, a mathematical dynamics system is proposed as

$$\dot{x} = -\frac{3}{4}x + u + 2v, \tag{2.56}$$

with the cost functionx

$$J = \int_{t_0}^{t_f} (2x^2 + Ru^2 - Qv^2)dt, \tag{2.57}$$

For this example, we set the initial conditions for weights as $W = [0\ 0\ 0]^T$ and $x_0 = 1$. The learning rate for critic NN is 0.1 and initial control inputs are $u = v = 0$. Here, the activation function for critic NN is chosen as $\psi = [x^2\ x^4\ x^6]^T$. The weights for u and v are selected as $R = 0.1$ and $Q = 1$. In this example, the objective is to bring the state from $x_o = 1$ to the goal state $x = 0$. The Algorithm 2 is implemented to solve the problem by considering the aforementioned parameters with the initial and goal values.

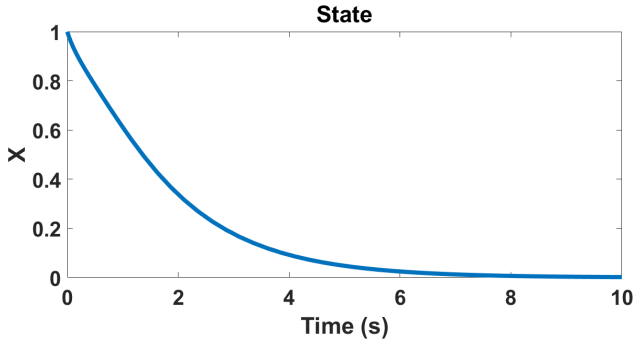


Figure 2.2: The state of the system.

As shown in Fig. 2.2, the state of the system converges to 0 after a few seconds and the system is asymptotically stable. The critic NN weights that are updated during the time are shown in Fig. 2.3 and it can be seen that after about 2 seconds all weights

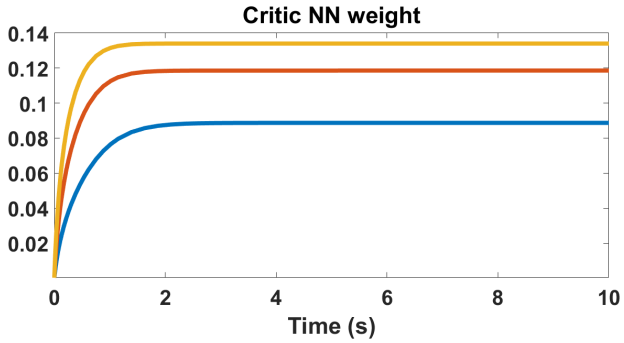


Figure 2.3: Trajectories of critic NN weights.

are tuned properly by the proposed update law. In this example, it is assumed that the dynamics of the system are known and the NN is only used to approximate the value function.

2.6.2 Inverted Pendulum

In the second example, the algorithm is applied to the inverted pendulum with conflicting controls. The dynamics of the system is given by

$$I\ddot{\theta} + b\dot{\theta} - mglsin\theta = u + v, \tag{2.58}$$

where $m = 1kg$, $l = 0.5m$, $b = 0.1(N.s)/m$, $I = ml^2$. $g = 9.81(kg.m)/s^2$. For this example, the initial conditions are selected as $x_0 = [1; 0]$, $W = [-0.305 \quad -1.958 \quad -0.926]^T$, $C = [-0.75 \quad 0.4; -0.19 \quad -0.64]$, $A = [-0.35 \quad 0.26; -0.40 \quad -0.2]$, $C_u = C_v = [-0.5 \quad 0.32; 0.2 \quad -0.9]$, $A_u = [-0.37 \quad -0.06]^T$, and $\lambda = 0$. Also; $Q = 0.1$, $R = 0.5$, $\Gamma_1 = \Gamma_2 = \Gamma_3 = \Gamma_4 = [1 \quad 0.1; 0.1 \quad 1]$, $\Gamma_5 = 0.2$, $\Gamma_6 = 0.1$ and the activation function is $\psi = [x_1^2 \quad x_1x_2 \quad x_2^2]^T$ and $\sigma_f = \sigma_{g1} = \sigma_{g2} = tanh(.)$.

It is assumed in this example that the dynamics of the system is unknown. Instead, it is approximated by the NN. It is illustrated in Fig. 2.4 that the trajectory of the approximated state xh closely resemble the state trajectory x and the dynamics of

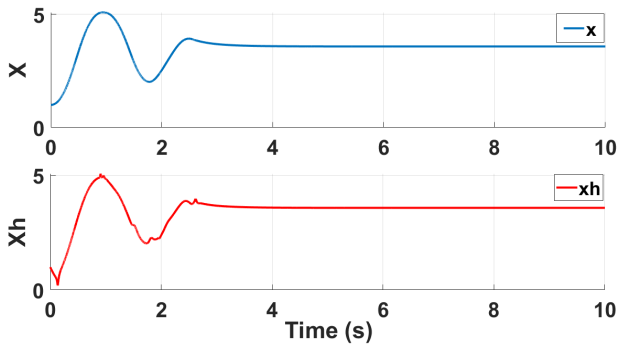


Figure 2.4: State and Approximated State of the system

the system is stabilized as the proposed algorithm is applied. Also, the approximated trajectory of the system goes to the desired state in about 3 seconds, which shows the effectiveness of the method. The trajectories of the critic weights are presented in Fig. 2.5 and it can be seen that all trajectories converge to fix values.

It is noticeable in this example that the trajectories of the approximated states and state are about same and this shows the strength of the proposed recurrent neural network.

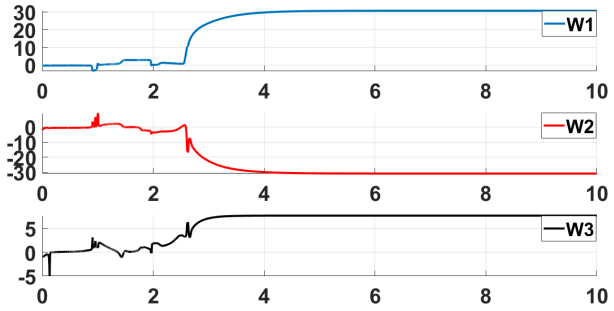


Figure 2.5: Trajectories of critic weights

2.6.3 Chaos System

In the third example, the proposed algorithm is applied to a chaos system [150]. The chaos theory in mechanics and mathematics is the study of apparently random or

unpredictable behavior in systems governed by deterministic laws. Here, the dynamics of the chaotic system is proposed as

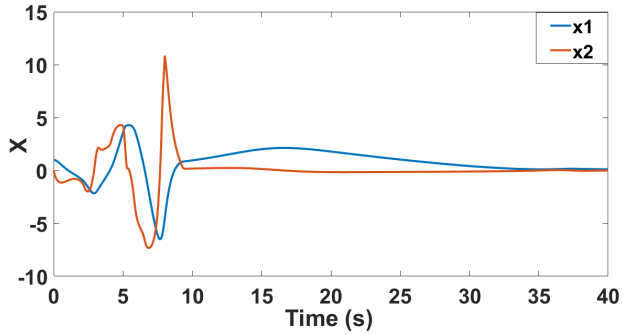


Figure 2.6: State of the system

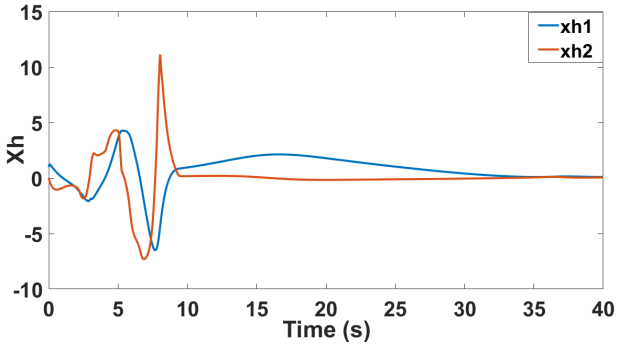


Figure 2.7: Approximated states

$$\dot{x}_1 = x_2,$$

$$\dot{x}_2 = -2E - \mu x_2 + (0.1667x_1 - 0.1667x_1^3) + E(f_m + f_e\Omega^2 cos(\Omega x)) + u + v, \quad (2.59)$$

where $E = 0.01$, $\mu = 9$, $f_m = 1$,$f_e = 30$, and $\Omega = 0.5$. Same as previous example, the initial conditions are selected as $x_0 = [1; 0]$, $W = [-0.305 \quad -1.958 \quad -0.926]^T$, $C = [-0.75 \quad 0.4; -0.19 \quad -0.64]$, $A = [-0.35 \quad 0.26; -0.40 \quad -0.2]$, $C_u = C_v = [-0.5 \quad 0.32; 0.2 \quad -0.9]$, $A_u = [-0.37 \quad -0.06]^T$, and $\lambda = 0$. Also; $Q = 0.1$, $R = 0.5$, $\Gamma_1 = \Gamma_2 = \Gamma_3 = \Gamma_4 = [1 \quad 0.1; 0.1 \quad 1]$, $\Gamma_5 = 0.2$, $\Gamma_6 = 0.1$ and the activation function

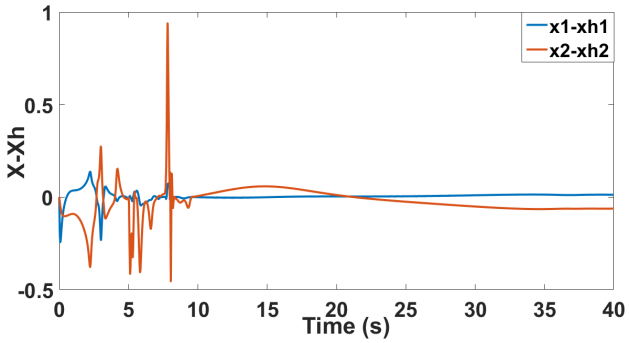


Figure 2.8: State error

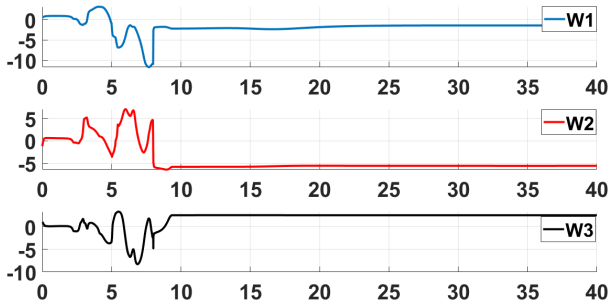


Figure 2.9: Trajectories of Critic NN

is $\psi = [x_1^2 \quad x_1x_2 \quad x_2^2]^T$ and $\sigma_f = \sigma_{g1} = \sigma_{g2} = tanh(.)$. The initial condition for the dynamics of the system is given by $x_0 = [1\ 0]^T$. The objective is to bring the states from $x_0 = [1\ 0]^T$ to the goal state $x = [0\ 0]^T$.

The proposed algorithm is applied to the chaotic system and it can be observed from Fig. 2.8 that the state error converges to 0 as t gets large. The states and the approximated states are stabilized by the proposed method, as shown in Fig. 2.6 and Fig. 2.7, respectively. Also, Fig. 2.9 demonstrates the trajectories of critic NN and all trajectories converge to fixed values.

Chapter 3

Sliding Mode Observer-Based Min-Max Differential Dynamic Programming for Output Feedback Differential Games

3.1 Introduction

Output feedback systems aim to regulate the output of the system based on its measured response rather than its full state [151]. This approach is crucial when full state measurement is impractical or impossible due to sensor limitations, cost, or complexity. However, designing effective output feedback systems presents significant challenges. These include dealing with incomplete or noisy measurement data, ensuring system stability, and achieving desired performance despite uncertainties and external disturbances. Traditional methods often fall short in addressing these issues comprehensively, necessitating advanced techniques that can optimize performance while maintaining robustness. Output feedback systems play a crucial role in a wide range of applications, including aerospace, robotics, automotive systems, and industrial automation [152, 153, 154]. These systems are essential for ensuring precise and reliable

operation in environments where it is impractical or impossible to measure the full state of the system continuously. A multitude of methods have been proposed for output feedback control, reflecting the growing interest and advancements in this field [155, 156]. These approaches range from traditional PID controllers to more sophisticated techniques such as adaptive control, robust control, and intelligent control strategies [157, 158, 159]. More modern control methods like model predictive control for nonlinear large-scale system [160], sliding mode control for unmanned marine vehicles [161], and reinforcement learning-based methods for output feedback systems [158, 162] have also been explored to enhance performance and robustness. These developments aim to address the inherent challenges of output feedback control, including dealing with incomplete state information, improving system stability, and achieving optimal performance under various uncertainties and disturbances. The ongoing research and innovation in this area highlight the critical importance of developing effective output feedback control solutions for a wide array of applications.

Recently, the study of differential games has gained significant attraction in the field of control systems [163]. Differential games extend the framework of optimal control to scenarios involving multiple decision-makers, often with competing objectives [164]. Within this context, output feedback control has been increasingly explored through the framework of differential games, leading to several notable studies and advancements in the field [165]. For instance, in [166], static and dynamic output feedback control schemes using Orientation Cellular Space Module (OCSM) and Rate Cellular Space Module RCSM modules are developed for spacecraft attitude control. However, these approaches may struggle with handling complex nonlinearities and uncertainties inherent in real-world applications. Paper [167], explores distributed differential game tracking for multi-agent systems using adaptive dynamic programming and neural networks. Despite its comprehensive design, the approach could be hin-

dered by the computational demands and reliance on precise model representations. The framework developed in [168], involves state estimators and output feedback controllers for affine nonlinear systems, leveraging zero-sum differential games and state- and measurement-dependent. Yet, the necessity of solving these intricate measurment could pose significant implementation challenges. In [169], an adaptive learning algorithm for solving the online output-feedback optimal control problem in continuous-time two-player zero-sum differential games, utilizing a modified game algebraic Riccati equation (MGARE) is presented. Nonetheless, the approach may face challenges due to the dependence on continuous data acquisition and the intricacy involved in reconstructing the MGARE, potentially hindering real-time implementation. Paper [170], develops a robust optimal control for a multi-input system which a Neural Networks (NNs) is utilized to approximate the Q-functions and solve the the Hamilton-Jacobi-Bellman (HJB) equation considering the unknown disturbance. However, the method of this paper cannot be applied to an output feedback control system. Paper [171], models the non-cooperative target approach of multiple spacecraft as a hierarchical dynamic game, using Stackelberg-Nash-saddle equilibriums for optimal output feedback control strategies. The paper [172] proposes a strategy matrix for ensuring controllability in game-based control systems, linking Nash equilibrium actions to system topology. It develops a multi-agent system to study controllability. In [173] game-based control systems (GBCSs), exploring how a regulator controls macrostates by influencing Nash equilibrium is studied. It provides controllability conditions for linear systems. A limitation in this paper is the reliance on the assumption of effective attacker cooperation, which might not hold in practical scenarios. The application of output feedback system is used for adaptive dynamic programming and quadratic differential game for stochastic systems in [174, 136],

Despite the advancements in output feedback control for differential games, many

of the proposed and mentioned methods face significant issues and often fail to achieve optimal performance. These traditional approaches may struggle with handling uncertainties, ensuring robustness, and maintaining desired performance levels, particularly in complex and dynamic environments. Hence, there is a pressing need for an optimal control technique to address these shortcomings [175]. The min-max Differential Dynamic Programming (DDP) method emerges as a powerful solution, offering several advantages over conventional methods [8]. Min-max DDP introduces a paradigm shift by considering control policies that not only optimize performance objectives, but also strategically anticipate and mitigate adversarial actions or uncertainties within the system. By formulating control decisions through the lens of game theory, the approach aims to propose two controllers which one tries to minimze a cost function, while the other one attempts to maximize the same cost function, thereby enhancing the resilience of control strategies [176, 177]. Unlike conventional DDP methods that often assume perfect knowledge of system dynamics, min-max DDP acknowledges the inherent uncertainties and disturbances present in real-world systems, which motivates the exploration of control policies that proactively address uncertainties by leveraging the min-max framework [178]. One drawback of the Min-max DDP is that it relies on having precise knowledge of the dynamics to compute the control policy. For instance, in [8], the state feedback optimal control policies are obtained through the min-max DDP approach for both continuous and discrete time dynamics with the assumption that there is a full knowledge of dynamics. In [13], the min-max DDP method is considered for the stochastic trajectory optimization where the precise knowledge of the drift and diffusion dynamics are assumed to be known. The application of the min-max DDP for the real world dynamics robust biped walking is discussed in [57]. In many practical applications that we only have access to the outputs of the system, resulting in a lack of direct knowledge of the system dynamics [179]. A reliable tool

in approximating the dynamics of the system solely based on the output information is the High Order Sliding Mode (HOSM) observer [180, 181, 182]. By continuously estimating the unobserved states of the system, the HOSM provides valuable insights into the behavior of the system and enables the construction of a dynamics model. Hence, it is a reliable approximation method among many researchers for obtaining states and dynamics when they are unavailable or cannot be directly measured.

Therefore, this Chapter extends the min-max DDP approach for output feedback control dynamics, where control decisions are computed through the approximated dynamics obtained by HOSM.

The contribution of the paper can be summarized as

1) The output feedback control system for min-max DDP is considered.

2) The High Order Sliding Mode observer is used to approximate the dynamics of the system through measuring the output states.

3) The update laws of the control policies and the set of backward differential equations are provided.

The rest of the paper is organized as follows: In Section 3.2, we introduce important notations. In Section 3.3, the problem formulation is provided. In Section 3.4, the observer design is discussed. In Section 3.5, the optimal control updates and the backward propagation of the value function approximation are studied. Finally, Section 3.6 provides the numerical examples and results.

3.2 Notation

For the reader's convenience, some useful notations and prerequisites are mentioned here. $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{R}^n$, and $\mathbb{R}^{n\times m}$ present the set of real numbers, positive real numbers, $n\times 1$ real column vectors, and $n \times m$ real matrices, respectively. $\mathcal{C}^1(\mathbb{R})$ and $\mathcal{C}^2(\mathbb{R})$ illustrates

sets of functions whose first and second-order derivative exist and is continuous with domain $\mathbb{R}$. For any specified functions $\Psi : \Re^n \times \Re \to \Re$ and $W : \Re^n \times \Re \to \Re^n$, the following are defined

$$\Psi_x(x,t) = \begin{pmatrix} \frac{\partial \Psi(x,t)}{\partial x_1} \\ \vdots \\ \frac{\partial \Psi(x,t)}{\partial x_n} \end{pmatrix},$$

$$\Psi_{xx}(x,t) = \begin{pmatrix} \frac{\partial^2 \Psi(x,t)}{\partial^2 x_1}, & \cdots & , \frac{\partial^2 \Psi(x,t)}{\partial x_1 x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi(x,t)}{\partial x_1 \partial x_n}, & \cdots & , \frac{\partial^2 \Psi(x,t)}{\partial^2 x_n} \end{pmatrix},$$

$$W_x(x,t) = \begin{pmatrix} \frac{\partial W_1(x,t)}{\partial x_1}, & \cdots & , \frac{\partial W_1(x,t)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial W_n(x,t)}{\partial x_1}, & \cdots & , \frac{\partial W_n(x,t)}{\partial x_n} \end{pmatrix},$$

that are partial derivative of Ψ with respect to x, Hessian of Ψ with respect to x, and the Jacobian matrix of W with respect to x, respectively.

3.3 Problem formulation

This Chapter considers the following partially unknown output-feedback nonlinear system

$$\begin{aligned} \frac{dx(t)}{dt} &= F(x(t), u(t), v(t)), \quad x(t_0) = x_0 \\ y_1(t) &= h_1(x(t)), \\ &\vdots \\ y_p(t) &= h_p(x(t)), \end{aligned} \tag{3.1}$$

where $x(t) \in \Re^n$, $t \in [t_0, t_f]$ is the state, $h_i(x(t)) \in \Re, i = 1, ..., p$ is the output of the system, and $u(t) \subset \Re^{m_u}$ and $v(t) \subset \Re^{m_v}$ are control policies.

The cost function of the min-max differential game is defined as

$$J(x, u, v) = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}(x(t), u(t), v(t))dt, \tag{3.2}$$

where $u(t)$ tries to minimize the cost while $v(t)$ tries to maximize it. Hence, $u(t)$ and $v(t)$ are called conflicting controls [56]. Here, $\mathcal{L}(x(t), u(t), v(t))$ is the running cost and $\phi(x(t_f), t_f)$ is the terminal cost.

The value function which represents the min-max value of the cost function is denoted as

$$V(x_0, t_0) = \min_u \max_v \left\{ \phi(x(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}(x(t), u(t), v(t))dt \right\}. \tag{3.3}$$

In output-feedback nonlinear systems, it is important to utilize observers, also known as estimators, to infer the internal state variables or dynamics of a system based on available measurements of states. In our approach, we leverage an observer to approximate the dynamics of the system in order to effectively apply min-max DDP

to find optimal control policies under uncertainty or adversarial behavior. By utilizing an observer, we can bridge the gap between the available measurements and the dynamics required for min-max DDP. Hence, in the following section, we introduce a reliable observer to approximate the dynamics of the system to use in the control design procedure.

3.4 Observer design

Consider the output-feedback nonlinear system (3.1) where

$$F(x(t), u(t), v(t)) = f(x(t)) + B(x(t), u(t)) + \sum_{i=1}^{m} g_i(x(t)) d_i(x(t), u(t), t), \tag{3.4}$$

and $f(x(t))$, $B(x(t), u(t)) \in \Re^n$ and $G(x(t)) = [g_1(x(t)), ..., g_m(x(t))]$ are smooth vector functions. The term $d_i(x(t), u(t), t)$ represents the uncertainty due to modeling error or the unknown input. For simplicity of notation, we henceforth omit the dependence on time t. The following assumptions are provided to facilitate subsequent analyses.

Assumption 4. *We assume that $p \geq m$, and the first m outputs have a vector relative degree $\{q_1, ..., q_m\}$ corresponding to $G(x(t))$ at each point $x_0 \in \Re^n$, that is,*

$$L_{g_j} L_f^k h_i(x) = 0,$$

for all $j = 1, ..., m$, $k < q_i - 1$, $i = 1, ..., m$, and $x \in \Re^n$, where L denotes the Lie

derivative. Further, the $m \times m$ matrix

$$A(x) = \begin{pmatrix} L_{g_1} L_f^{q_1-1} h_1(x), & \cdots & , L_{g_m} L_f^{q_1-1} h_1(x) \\ L_{g_1} L_f^{q_2-1} h_2(x), & \cdots & , L_{g_m} L_f^{q_2-1} h_2(x) \\ \vdots & \ddots & \vdots \\ L_{g_1} L_f^{q_m-1} h_m(x), & \cdots & , L_{g_m} L_f^{q_m-1} h_m(x) \end{pmatrix},$$

is nonsingular at each point x_0.

Remark 3. *Assumption (4) states that if the system given by (3.1) and (3.4) has p outputs $\{y_1, y_2, ..., y_p\}$, there includes m outputs ($p \geq m$) with a relative degree $\{q_1, q_2, ..., q_m\}$ corresponding to the vector fields $\{g_1(x), g_2(x), ..., g_m(x)\}$. For an output $y_i = h_i(x)$, the relative degree q_i is the number of times you need to differentiate $h_i(x)$ with respect to the time until the input u explicitly appears in the resulting expression. Mathematically, this means $L_{g_j} L_f^k h_i(x) = 0$ for all $j = 1, ..., m$ and $k < q_i$. Also, the nonsingularity of matrix A ensures that the output functions provide enough information to uniquely determine the states of the system through the observer. This is a key requirement for the system to be observable and the construction of a valid coordinate transformation. This transformation is used to decompose the system into observable subsystems, which is critical for the observer design process since the observer relies on this assumption to ensure that the estimation error dynamics can be controlled and driven to zero.*

Assumption 5. *The distribution spanned by the vector fields $g_1(x), ..., g_m(x)$ is involutive.*

Remark 4. *Assumption (5) indicates that, in the context of control systems, vector fields $\{g_1(x), ..., g_m(x)\}$ represent directions in which the system can evolve. These vector fields are functions that assign a vector to every point in the state space. The*

distribution is the set of all linear combinations of the vector fields at each point in the state space. Mathematically, at each point x, the distribution is:

$$D(x) = span\{g_1(x), g_2(x), .., g_m(x)\}. \tag{3.5}$$

A distribution is involutive if, for any two vector fields g_i and g_j in the distribution, their Lie bracket is also in the distribution. Essentially, the involutivity condition means that combining the vector fields through the Lie bracket operation results in a vector field that is still within the span of the original vector fields. For designing observers, especially sliding mode observers (SMOs), ensuring that the distribution is involutive allows for the separation of the system into parts influenced by unknown inputs and parts that are not. This separation is critical for creating observers that can estimate the states of the system accurately even in the presence of uncertainties.

If Assumptions (4) and (5) holds, then

$$q_1 + ... + q_m \leq n. \tag{3.6}$$

Since the relative degree indicates how many state variables are involved in defining each output, the sum of all relative degrees $\{q_1, q_2, ..., q_m\}$ should not exceed the total number of state variables n. In other words, the relative degree q_i of an output y_i is the number of times you need to differentiate y_i with respect to the time until the control input u explicitly appears. For each output y_i, differentiating q_i times gives information about the state variables. If the sum of relative degrees exceeded n, it would imply that we have more independent directions in the state space than the state variables available, which is not possible.

In the following, detailed steps for the observer design in our nonlinear output

feedback control system are outlined. It elaborates on how the system can be transformed into a form suitable for observer design, focusing on the decomposition of the state space and the introduction of new coordinates. This transformation isolates the subsystem affected by unknown inputs and prepares the system for the subsequent observer design. The step-by-step transformation ensures that the observer can effectively estimate the states despite the presence of unknown inputs. This lays the groundwork for the practical implementation of the sliding mode observer, and ensures that our control design is reliable, addressing the challenges posed by the nonlinear nature and uncertainties of the system.

For $i = 1, ..., m$, let

$$\begin{aligned} \phi_1^i &= h_i(x), \\ \phi_2^i &= L_f h_i(x), \\ &\vdots \\ \phi_{q_i}^i &= L_f^{q_i - 1} h_i(x). \end{aligned} \tag{3.7}$$

If $q = q_1 + ... + q_m < n$, it is always possible to find $n - q$ more functions $\phi_{q+1}(x), ..., \phi_n(x)$ such that the mapping

$$\Phi(x) = col\left(\phi_1^1(x), ..., \phi_{q_1}^1(x), \phi_1^2(x), ..., \phi_{q_2}^2(x), ..., \phi_1^m(x), ..., \phi_{q_m}^m(x), \phi_{q+1}(x), ..., \phi_n(x)\right), \tag{3.8}$$

has a Jacobian matrix at each x_0 and therefore qualifies as a local coordinate transformation. Moreover, based on assumption (5), it is always possible to choose

$\phi_{q+1}(x), ..., \phi_n(x)$ such that

$$L_{g_j}\phi_i(x) = 0, \tag{3.9}$$

for all $i = q+1, ..., n, j = 1, ..., m$, and all x. Let

$$x_d^i = \begin{pmatrix} x_1^i \\ x_2^i \\ \vdots \\ x_{q_i}^i \end{pmatrix} = \begin{pmatrix} \phi_1^i(x) \\ \phi_2^i(x) \\ \vdots \\ \phi_{q_i}^i(x) \end{pmatrix}, \tag{3.10}$$

for $i = 1, ..., m$, and $x_d = \left((x_d^1)^T, ..., (x_d^m)^T\right)$, and let

$$x_o = \begin{pmatrix} x_o^1 \\ x_o^2 \\ \vdots \\ x_o^{n-q} \end{pmatrix} = \begin{pmatrix} \phi_{q+1}(x) \\ \phi_{q+2}(x) \\ \vdots \\ \phi_n(x) \end{pmatrix}, \tag{3.11}$$

then the equations under new coordinates can be written as

$$\begin{aligned} \dot{x}_1^i &= x_2^i + b_1^i(x_d, x_o, u), \\ &\vdots \\ \dot{x}_{q_i-1}^i &= x_{q_i}^i + b_{q_i-1}^i(x_d, x_o, u), \\ \dot{x}_{q_i}^i &= a_i(x_d, x_o) + b_{q_i}^i(x_d, x_o, u) + \sum_{j=1}^{m} c_{ij}(x_d, x_o)d_j, \\ y_i &= x_1^i, \end{aligned} \tag{3.12}$$

for $i = 1, ..., m$, and

$$\begin{aligned}
\dot{x}_o &= q(x_d, x_o) + p(x_d, x_o, u), \\
y_{m+1} &= h_{m+1}(x_d, x_o), \\
&\vdots \\
y_p &= h_p(x_d, x_o),
\end{aligned} \tag{3.13}$$

where

$$\begin{aligned}
a_i(x_d, x_o) &= L_f^{q_i} h_i(\Phi^{-1}(x_d, x_o)), \\
c_{ij} &= L_{g_j} L_f^{q_i - 1} h_i(\Phi^{-1}(x_d, x_o)),
\end{aligned} \tag{3.14}$$

and

$$b_k(x_d, x_o, u)^i = \frac{\partial (L_f^{k-2} h_i)}{\partial x} B\Big(\Phi^{-1}(x_d, x_o), u\Big). \tag{3.15}$$

Above is a trivial extension of Prop. 5.1.2 in [183]. The nonlinear transformation $\Phi(x)$ decomposes the system (3.1) into two subsystems, where only x_d^i subsystem is effected by unknown inputs. In the context of HOSM observer, "subsystem" refers to the partitioning of the state-space of the system into components like x_d^i and x_o based on their observability. The subsystem x_d^i consists of states directly influenced by the outputs and inputs of the system, while x_o may include states that are less directly observable and can be negligible due to unknown inputs and effects on the performance of the system.

Hence, next we study the observability of the nonlinear system to approximate the dynamics through the HOSM observer for system (3.12) and (3.13).

3.4.1 Observability of subsystems

The observability of nonlinear systems (3.12) and (3.13) is closely tied to their inputs. Inputs that render a nonlinear system unobservable are termed "bad inputs". When dealing with known input signals, one can typically avoid employing the bad inputs. However, unknown inputs, which may arise from faults or other disturbances beyond our control, present a challenge. Thus, ensuring observability for all unknown inputs is generally essential for designing a nonlinear unknown input observer, unless it can be established beforehand that these unknown inputs do not fall into the category of bad inputs. Building upon the findings in [184], we present the following Proposition outlining conditions under which the observability of the subsystem x_d remains unaffected by unknown inputs.

Proposition 1. *Let* $\bar{x}_d^i = \{x_d^1, ..., x_d^{i-1}, x_d^{i+1}, ..., x_d^m\}$. *Assume each* x_d *subsystem in (3.12) has its input term in the following form*

$$
\begin{aligned}
&b_1^i(x_d, x_o, u) = b_1^i(x_1^i, x_2^i, \bar{x}_d^i, x_o, u),\\
&b_2^i(x_d, x_o, u) = b_2^i(x_1^i, x_2^i, x_3^i, \bar{x}_d^i, x_o, u),\\
&\vdots\\
&b_{q_i-1}^i(x_d.x_o, u) = b_{q_i}^i(x_d.x_o, u),\\
&b_{q_i}^i(x_d.x_o, u) = b_{q_i}^i(x_d.x_o, u),
\end{aligned}
\tag{3.16}
$$

and the functions

$$
1 + \frac{\partial b_j^i}{\partial x_{j+1}^i} \neq 0, \quad j = 1, ..., q_i - 1, \tag{3.17}
$$

and state $x_o, \bar{x}_d^i$ is considered as inputs for x_d^i subsystem. Then, x_d^i subsystem is uniformly observable for all inputs $u, d, \bar{x}_d^i$ and x_o.

Proof. To establish observability, we construct the observability matrix for the subsystem x_d^i. Observability is guaranteed if this matrix has full rank.

Step 1: Define the Output and Its Derivatives:

Since $x_1^i = y_i$, we differentiate y_i iteratively:

$$\begin{aligned}
\dot{x}_1^i &= x_2^i + b_1^i(x_d, x_o, u), \\
\dot{x}_2^i &= x_3^i + b_2^i(x_d, x_o, u), \\
&\vdots \\
\dot{x}_{q_i-1}^i &= x_{q_i}^i + b_{q_i-1}^i(x_d, x_o, u), \\
\dot{x}_{q_i}^i &= a^i(x_d, x_o) + b_{q_i}^i(x_d, x_o, u) + \sum_{j=1}^{m} c_j^i(x_d, x_o) d_j.
\end{aligned} \tag{3.18}$$

This gives us the output derivatives up to order q_i

$$\begin{aligned}
y_i &= x_1^i, \\
\dot{y}_i &= x_2^i + b_1^i, \\
\ddot{y}_i &= x_3^i + b_2^i, \\
&\vdots \\
y^{(q_i)} &= a^i(x_d, x_o) + b_{q_i}^i + \sum_{j=1}^{m} c_j^i(x_d, x_o) d_j.
\end{aligned} \tag{3.19}$$

Step 2: Construct the Observability Matrix:

The observability matrix $\mathcal{O}$ is given by

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ \frac{\partial \dot{y}_i}{\partial x_1^i} & \frac{\partial \dot{y}_i}{\partial x_2^i} & \frac{\partial \dot{y}_i}{\partial x_3^i} & \cdots & \frac{\partial \dot{y}_i}{\partial x_{q_i}^i} \\ \frac{\partial \ddot{y}_i}{\partial x_1^i} & \frac{\partial \ddot{y}_i}{\partial x_2^i} & \frac{\partial \ddot{y}_i}{\partial x_3^i} & \cdots & \frac{\partial \ddot{y}_i}{\partial x_{q_i}^i} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y^{(q_i)}}{\partial x_1^i} & \frac{\partial y^{(q_i)}}{\partial x_2^i} & \frac{\partial y^{(q_i)}}{\partial x_3^i} & \cdots & \frac{\partial y^{(q_i)}}{\partial x_{q_i}^i} \end{bmatrix}. \tag{3.20}$$

Using the equations above, we compute the partial derivatives

$$\begin{aligned} &\frac{\partial \dot{y}_i}{\partial x_1^i} = 0, \quad \frac{\partial \dot{y}_i}{\partial x_2^i} = 1, \quad \frac{\partial \dot{y}_i}{\partial x_3^i} = 0, \quad \cdots \quad , \frac{\partial \dot{y}_i}{\partial x_{q_i}^i} = 0, \\ &\frac{\partial \ddot{y}_i}{\partial x_1^i} = 0, \quad \frac{\partial \ddot{y}_i}{\partial x_2^i} = 0, \quad \frac{\partial \ddot{y}_i}{\partial x_3^i} = 1, \quad \cdots \quad , \frac{\partial \ddot{y}_i}{\partial x_{q_i}^i} = 0, \\ &\vdots \\ &\frac{\partial y^{(q_i)}}{\partial x_1^i} = 0, \quad \frac{\partial y^{(q_i)}}{\partial x_2^i} = 0, \quad \frac{\partial y^{(q_i)}}{\partial x_3^i} = 0, \quad \cdots \quad , \frac{\partial y^{(q_i)}}{\partial x_{q_i}^i} = 1 + \frac{\partial b_{q_i-1}^i}{\partial x_{q_i}^i}. \end{aligned} \tag{3.21}$$

Step 3: Check Rank Condition

The observability matrix $\mathcal{O}$ has a triangular structure

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 + \frac{\partial b_{q_i-1}^i}{\partial x_{q_i}^i} \end{bmatrix}. \tag{3.22}$$

For $\mathcal{O}$ to be full rank, its determinant must be nonzero. The determinant of this lower

triangular matrix is

$$\det(\mathcal{O}) = \prod_{j=1}^{q_i-1} \left(1 + \frac{\partial b_j^i}{\partial x_{j+1}^i}\right), \tag{3.23}$$

by assumption

$$1 + \frac{\partial b_j^i}{\partial x_{j+1}^i} \neq 0, \quad j = 1, ..., q_i - 1. \tag{3.24}$$

which guarantees that $\det(\mathcal{O}) \neq 0$, ensuring full rank. Since the observability matrix has full rank for all input conditions, the subsystem x_d^i is uniformly observable for all inputs u, d, x_i^d, and x_o. □

Remark 5. *From Proposition 1, each x_d^i subsystem has input terms b_i structured in a specific manner and these input terms depend on the state variables and control inputs in a hierarchical way. The hierarchical structure ensures that the influence of each state variable can be isolated and measured and the condition (3.17) ensure that each state variable x_{j+1}^i affects the input term b_q^i in a way that can be detected. On the other hand, by ensuring that the input terms depend on the state variables hierarchically, we can use a recursive approach to observe the states. Starting from the first state variable, we can determine each subsequent state variable step by step, making the entire subsystem observable. Hence, the Proposition 1 guarantees that states of the system can be accurately estimated.*

3.4.2 HOSM observer for subsystem with unknown inputs

In this subsection, we build an unknown input independent observer for x_d^i subsystem.

Theorem 4. *Consider the system (3.12). If subsystem x_d has its input term as*

$$\begin{aligned}
&b_1^i(x_d, x_o, u) = b_1^i(y, u),\\
&b_2^i(x_d, x_o, u) = b_2^i(x_2^i, y, u),\\
&b_{q_i-1}^i(x_d, x_o, u) = b_{q_i-1}^i, (x_2^i, ..., x_{q_i-1}^i, y, u),\\
&b_{q_i}^i(x_d, x_o, u) = b_{q_i}^i(x_d, x_o, u),
\end{aligned} \tag{3.25}$$

for $i = 1, ..., m$, then there exists $y_j, j = 1, ..., q_i$ such that the states of the system can be approximated by the following HOSM observer

$$\begin{aligned}
&\dot{\hat{x}}_1^i = \hat{x}_2^i + b_1^i(y, u) + \lambda_1^i sign(y_i - \hat{x}_1^i),\\
&\dot{\hat{x}}_2^i = \hat{x}_3^i + b_2^i(\hat{x}_2^i, y, u) + \lambda_2^i sign(\bar{e}_2^i),\\
&\vdots\\
&\dot{\hat{x}}_{q_i-1}^i = \hat{x}_{q_i}^i + b_{q_i-1}^i(\hat{x}_2^i, ..., \hat{x}_{q_i-1}^i, y, u) + \lambda_{q_i-1}^i sign(\bar{e}_{q_i-1}^i),\\
&\dot{\hat{x}}_{q_i}^i = a_i(\hat{x}_d, \hat{x}_o) + b_{q_i}^i(\hat{x}_d, \hat{x}_o, u) + \lambda_{q_i}^i sign(\bar{e}_{q_i}^i),
\end{aligned} \tag{3.26}$$

where $\bar{e}_j^i = (\lambda_{j-1}^i sign(\bar{e}_{j-1}^i))_{eq}$, $j = 2, ..., q_i$, and $\bar{e}_1^i = e_1^i = y_i - \hat{x}_1^i$ is computed directly.

Proof. If $e_j^i = x_j^i - \hat{x}_j^i, j = 1, ..., q_i$, the following error dynamics is obtain from equations

(3.12), (3.25), and (3.26)

$$
\begin{aligned}
&\dot{e}_1^i = e_2^i - \lambda_1^i sign(e_1^i) \\
&\dot{e}_2^i = e_3^i + b_2^i(x_2^i, y, u) - b_2^i(\hat{x}_2^i, y, u) - \lambda_2^i sign(\bar{e}_2^i) \\
&\vdots \\
&\dot{e}_{q_i-1}^i = e_{q_i}^i + b_{q_i-1}^i(x_2^i, ..., x_{q_i-1}^i, y, u) - b_{q_i-1}^i(\hat{x}_2^i, ..., \hat{x}_{q_i-1}^i, y, u) \\
&- \lambda_{q_i-1}^i sign(\bar{e}_{q_i-1}^i) \\
&\dot{e}_{q_i}^i = a_i(x_d, x_o) + b_{q_i}^i(x_d, x_o, u) + \sum_{j=1}^{m} c_{ij}(x_d, x_o) d_j - a_i(\hat{x}_d, \hat{x}_o) \\
&- b_{q_i}(\hat{x}_d, \hat{x}_o, u) - \lambda_{q_i}^i sign(\bar{e}_{q_i}^i).
\end{aligned} \tag{3.27}
$$

We want to prove that the observation error converge to zero in finite time. Hence, we investigate them separately based on different conditions. For e_1, if it is selected $\lambda_1^i > sup|e_2^i(t)|$, e_1^i converges to zero in finite time t_1. Additionally, after t_1

$$
\bar{e}_2^i = (\lambda_1^i sign(e_1))_{eq} = e_2^i. \tag{3.28}
$$

Thus, the second error equation after t_1 will be

$$
\dot{e}_2^i = e_3^i + b_2^i(x_2^i, y, u) - b_2^i(\hat{x}_2^i, y, u) - \lambda_2^i sign(e_2^i). \tag{3.29}
$$

If $\lambda_2^i > |e_3^i + b_2^i(x_2^i, y, u) - b_2^i(\hat{x}_2^i, y, u)|, e_2^i$ goes to zero in finite time $t_2 > t_1$. Hence, after $t > t_2$

$$
\begin{aligned}
&\bar{e}_3^i = (\lambda_2^i sign(e_2))_{eq} = e_3^i \\
&\dot{e}_3^i = e_4^i + b_3^i(x_2^i, x_3^i, y, u) - b_3^i(\hat{x}_2^i, \hat{x}_3^i, y, u) - \lambda_3^i sign(e_3^i).
\end{aligned} \tag{3.30}
$$

We do this process up to step q_i, Therefore after t_{q_i-1}

$$\dot{e}^i_{q_i} = a_i(x_d, x_o) + b^i_{q_i}(x_d, x_o, u) + \sum_{j=1}^{m} c_{ij}(x_d, x_o)d_j - a_i(\hat{x}_d, \hat{x}_o) - b^i_{q_i}(\hat{x}_d, \hat{x}_o, u) - \lambda^i_{q_i} sign(e^i_{q_i}). \tag{3.31}$$

If $\lambda^i_{q_i} > |a_i(x_d, x_o) + b^i_{q_i}(x_d, x_o, u) + \sum_{j=1}^{m} c_{ij}(x_d, x_o)d_j - a_i(\hat{x}_d, \hat{x}_o) - b^i_{q_i}(\hat{x}_d, \hat{x}_o, u)|$, e_{q_i} converges to zero in finite time $t_{q_i} > t_{q_i-1}$.

Which this completes the proof. □

3.5 Optimal control update and backward propagation of the value function

After obtaining the approximation of nonlinear output feedback control system (3.1) through the HOSM observer method, we compute the optimal control update laws through the min-max DDP approach to obtain the desired output of the system. As mentioned before, in the context of HOSM observer concept, two subsystems x^i_d including states that are directly influenced by the outputs and inputs of the system and x_o consists of states that are less directly observable and can be neglected due to unknown inputs and effects on the performance of the system. The system under consideration, which captures the relevant dynamics for control design can be represented by the observable state vector $x_d = \left((x^1_d)^T, ..., (x^m_d)^T\right)$ and the differential equation governing these dynamics of x^i_d is given by equation (3.12). For simplicity in notation, we denote the observable states x_d as x throughout this section. Thus, the nonlinear

dynamics of x_d can be rewritten as

$$\frac{dx}{dt} = F(\bar{x} + \delta x, \bar{u} + \delta u, \bar{v} + \delta v), \tag{3.32}$$

where $(\bar{x}, \bar{u}, \bar{v})$ represents the state and control histories, $\delta x = x - \bar{x}$, $\delta u = u - \bar{u}$, and $\delta v = v - \bar{v}$.

The control update laws are only applied to x_d^i subsystem due to its influence and role on the performance of the system. The optimal control updates are leveraged by the Hamilton-Jacobi-Bellman-Isaacs (HJBI) Partial Differential Equation (PDE) which is satisfied by the value function of the min-max differential game. To this end, consider the following HJBI

$$-\frac{\partial V(x,t)}{\partial t} = \min_u \max_v \{r(x(t), u(t), v(t)) + V_x(x,t)^T F(x(t), u(t), v(t))\}, \tag{3.33}$$

with the terminal condition $V(x(t_f), t_f) = \phi(x(t_f), t_f)$.

The linearized model around $(\bar{x}, \bar{u}, \bar{v})$ is expressed as

$$\frac{d\delta x}{dt} = \bar{F}_x \delta x + \bar{F}_u \delta u + \bar{F}_v \delta v, \tag{3.34}$$

where

$$\bar{F}_x = F_x(\bar{x}, \bar{u}, \bar{v}).$$

$$\bar{F}_u = F_u(\bar{x}, \bar{u}, \bar{v}).$$

$$\bar{F}_v = F_v(\bar{x}, \bar{u}, \bar{v}).$$

After approximating the dynamics through the HOSM observer method, we can approximate $\bar{F}_x, \bar{F}_u, \bar{F}_v$ with $\hat{F}_x$, $\hat{F}_u$, $\hat{F}_v$ based on the equation (3.26). The linearized

system (3.34) can then be written as

$$\frac{d\hat{\delta}_x}{dt} = \hat{F}_x\hat{\delta}x + \hat{F}_u\hat{\delta}u + \hat{F}_v\hat{\delta}v. \tag{3.35}$$

In this context, all terms denoted with $\hat{\cdot}$ correspond to the approximate dynamics learned through HOSM observer method. The terms $\hat{\delta}u$ and $\hat{\delta}v$ in equation (3.35) are iteratively updated utilizing the approximate system, and the update laws for the optimal control policy are presented in the following theorem.

Theorem 5. *If the approximate dynamics through the HOSM observer algorithm is obtained as*

$$\frac{d\hat{x}}{dt} = \hat{F}(\bar{x} + \hat{\delta}x, \bar{u} + \hat{\delta}u, \bar{v} + \hat{\delta}v), \tag{3.36}$$

then, the update of the optimal control policies $\hat{\delta}u$ and $\hat{\delta}v$ are computed as

$$\hat{\delta}u^* = \hat{I}_u + \hat{K}_u\hat{\delta}x, \tag{3.37a}$$

$$\hat{\delta}v^* = \hat{I}_v + \hat{K}_v\hat{\delta}x, \tag{3.37b}$$

where

$$\hat{I}_u = -(\hat{Q}_{uu} - \hat{Q}_{uv}\hat{Q}_{vv}^{-1}\hat{Q}_{vu})^{-1}(\hat{Q}_u - \hat{Q}_{uv}\hat{Q}_{vv}^{-1}\hat{Q}_v). \tag{3.38}$$

$$\hat{I}_v = -(\hat{Q}_{vv} - \hat{Q}_{vu}\hat{Q}_{uu}^{-1}\hat{Q}_{uv})^{-1}(\hat{Q}_v - \hat{Q}_{vu}\hat{Q}_{uu}^{-1}\hat{Q}_u). \tag{3.39}$$

$$\hat{K}_u = -(\hat{Q}_{uu} - \hat{Q}_{uv}\hat{Q}_{vv}^{-1}\hat{Q}_{vu})^{-1}(\hat{Q}_{ux} - \hat{Q}_{uv}\hat{Q}_{vv}^{-1}\hat{Q}_{vx}). \tag{3.40}$$

$$\hat{K}_v = -(\hat{Q}_{vv} - \hat{Q}_{vu}\hat{Q}_{uu}^{-1}\hat{Q}_{uv})^{-1}(\hat{Q}_{vx} - \hat{Q}_{vu}\hat{Q}_{uu}^{-1}\hat{Q}_{ux}). \tag{3.41}$$

and

$$\hat{Q}_x = \hat{F}_x^T\hat{V}_x + \hat{\mathcal{L}}_x. \tag{3.42a}$$

$$\hat{Q}_u = \hat{F}_u^T\hat{V}_x + \hat{\mathcal{L}}_u. \tag{3.42b}$$

$$\hat{Q}_v = \hat{F}_v^T\hat{V}_x + \hat{\mathcal{L}}_v. \tag{3.42c}$$

$$\hat{Q}_{xx} = \hat{\mathcal{L}}_{xx} + \hat{V}_{xx}\hat{F}_x + \hat{F}_x^T\hat{V}_{xx}. \tag{3.42d}$$

$$\hat{Q}_{uu} = \hat{\mathcal{L}}_{uu}. \tag{3.42e}$$

$$\hat{Q}_{vv} = \hat{\mathcal{L}}_{vv}. \tag{3.42f}$$

$$\hat{Q}_{ux} = \hat{\mathcal{L}}_{ux} + \hat{F}_u^T\hat{V}_{xx}. \tag{3.42g}$$

$$\hat{Q}_{vx} = \hat{\mathcal{L}}_{vx} + \hat{F}_v^T\hat{V}_{xx}. \tag{3.42h}$$

$$\hat{Q}_{uv} = \hat{\mathcal{L}}_{uv}. \tag{3.42i}$$

Proof. Consider the Taylor series, extend the left side of the equation (3.33)

$$\frac{\partial V(x,t)}{\partial t} = \frac{\partial V(\bar{x} + \hat{\delta}x, t)}{\partial t} \approx \frac{\partial \bar{V}}{\partial t} + \frac{\partial \bar{V}_x^T}{\partial t}\hat{\delta}x + \frac{1}{2}\hat{\delta}x^T\frac{\partial \bar{V}_{xx}^T}{\partial t}\hat{\delta}x. \tag{3.43}$$

We have

$$\frac{d\bar{V}}{dt} = \frac{\partial V}{dt} + \bar{V}_x^T\frac{dx}{dt} = \frac{\partial \bar{V}}{\partial t} + \bar{V}_x^T\bar{F}. \tag{3.44}$$

Thus,

$$\frac{\partial \bar{V}}{\partial t} = \frac{d\bar{V}}{dt} - \bar{V}_x^T\bar{F}, \tag{3.45}$$

similarly

$$\frac{\partial \bar{V}_x}{\partial t} = \frac{d\bar{V}_x}{dt} - \bar{V}_{xx}^T \bar{F}, \tag{3.46}$$

the expansion of the partial time derivative of the Hessian of the value function is given as

$$\frac{\partial \bar{V}_{xx}}{\partial t} = \frac{d\bar{V}_{xx}}{dt} - \sum_{i=1}^{n} \bar{V}_{xxx}^{(i)} \bar{F}^{(i)}, \tag{3.47}$$

where $\bar{V}_{xxx}^{(i)}$ and $\bar{F}^{(i)}$ are Hessian matrix of the i-th element $\bar{V}_x$ and $\bar{F}$, respectively. Hence, by referring to equation (3.44-3.46) the left side of the equation (3.33) is written as

$$\begin{aligned} -\frac{\partial V(x,t)}{\partial t} \approx & -\frac{d\bar{V}}{dt} - \frac{d\bar{V}_x^T}{dt}\hat{\delta}x - \frac{1}{2}\hat{\delta}x^T \frac{d\bar{V}_{xx}^T}{dt}\hat{\delta}x + \bar{V}_x^T \bar{F} + \hat{\delta}x^T \bar{V}_{xx}\bar{F} \\ & + \frac{1}{2}\hat{\delta}x^T \left(\sum_{i=1}^{n} \bar{V}_{xxx}^{(i)} \bar{F}^{(i)} \right) \hat{\delta}x. \end{aligned} \tag{3.48}$$

The expansion of the right hand side of the equation (3.33) can be written as

$$\begin{aligned} \mathcal{L} \approx & \hat{\mathcal{L}} + \hat{\mathcal{L}}_x^T \hat{\delta}x + \hat{\mathcal{L}}_u^T \hat{\delta}u + \hat{\mathcal{L}}_v^T \hat{\delta}v \\ & + \frac{1}{2} \begin{pmatrix} \hat{\delta}x \\ \hat{\delta}u \\ \hat{\delta}v \end{pmatrix}^T \begin{pmatrix} \hat{\mathcal{L}}_{xx} & \hat{\mathcal{L}}_{xu} & \hat{\mathcal{L}}_{xv} \\ \hat{\mathcal{L}}_{ux} & \hat{\mathcal{L}}_{uu} & \hat{\mathcal{L}}_{uv} \\ \hat{\mathcal{L}}_{vx} & \hat{\mathcal{L}}_{vu} & \hat{\mathcal{L}}_{vv} \end{pmatrix} \begin{pmatrix} \hat{\delta}x \\ \hat{\delta}u \\ \hat{\delta}v \end{pmatrix}, \end{aligned} \tag{3.49}$$

The expansion of V_x around $\bar{x}$ is

$$V_x \approx \bar{V}_x + \bar{V}_{xx}\hat{\delta}x + \frac{1}{2}\bar{W}, \tag{3.50}$$

where $\bar{W} = [\bar{W}^i]$ is

$$\bar{W}^i = \hat{\delta}x^T \bar{V}^{(i)}_{xxx} \hat{\delta}x, \qquad i = i, ..., n. \tag{3.51}$$

The first order expansion of the dynamics results in

$$\hat{F}(\hat{x}, \hat{u}, \hat{v}) = \hat{F} + \hat{F}_x \hat{\delta}x + \hat{F}_u \hat{\delta}u + \hat{F}_v \hat{\delta}v. \tag{3.52}$$

The right side of the equation (3.33) yields

$$\begin{aligned}
&\min_{\hat{u}} \max_{\hat{v}} \left\{ \mathcal{L}(\hat{x}, \hat{u}, \hat{v}, t) + \hat{V}_x^T \hat{F}(\hat{x}, \hat{u}, \hat{v}, t) \right\} \approx \left\{ \hat{\mathcal{L}} + \hat{\mathcal{L}}_x^T \hat{\delta}x + \hat{\mathcal{L}}_u^T \hat{\delta}u + \hat{\mathcal{L}}_v^T \hat{\delta}v \right. \\
&+ \frac{1}{2} \begin{pmatrix} \hat{\delta}x \\ \hat{\delta}u \\ \hat{\delta}v \end{pmatrix}^T \begin{pmatrix} \hat{\mathcal{L}}_{xx} & \hat{\mathcal{L}}_{xu} & \hat{\mathcal{L}}_{xv} \\ \hat{\mathcal{L}}_{ux} & \hat{\mathcal{L}}_{uu} & \hat{\mathcal{L}}_{uv} \\ \hat{\mathcal{L}}_{vx} & \hat{\mathcal{L}}_{vu} & \hat{\mathcal{L}}_{vv} \end{pmatrix} \begin{pmatrix} \hat{\delta}x \\ \hat{\delta}u \\ \hat{\delta}v \end{pmatrix} + \hat{V}_x^T \hat{F} + \hat{V}_x^T \hat{F}_x \hat{\delta}x + \hat{V}_x^T \hat{F}_u \hat{\delta}u \\
&\left. + \hat{V}_x^T \hat{F}_v \hat{\delta}v + \hat{\delta}x^T \hat{V}_{xx} \hat{F}_x \hat{\delta}x + \hat{\delta}x^T \hat{V}_{xx} \hat{F}_u \hat{\delta}u + \hat{\delta}x^T \hat{V}_{xx} \hat{F}_v \hat{\delta}v + \hat{\delta}x^T \hat{V}_{xx} \hat{F} + \frac{1}{2} \hat{W}^T \hat{F} \right\},
\end{aligned} \tag{3.53}$$

where

$$\frac{1}{2} \hat{W}^T \hat{F} = \frac{1}{2} \left(\sum_{i=1}^{n} \hat{\delta}x^T \hat{V}^{(i)}_{xxx} \hat{\delta}x \hat{F}^{(i)} \right) = \frac{1}{2} \hat{\delta}x^T \left(\sum_{i=1}^{n} \hat{V}^{(i)}_{xxx} \hat{F}^{(i)} \right) \hat{\delta}x. \tag{3.54}$$

By equating the left hand side and right hand side of the expansion of equation (3.33)

and omitting extra terms, the following equation is obtained

$$
\begin{aligned}
&-\frac{d\hat{V}}{dt}-\hat{\delta}x^T\frac{d\hat{V}_x}{dt}-\frac{1}{2}\hat{\delta}x^T\frac{d\hat{V}_{xx}}{dt}\hat{\delta}x=\min_{\hat{\delta}u}\max_{\hat{\delta}v}\Bigg\{\hat{\mathcal{L}}+\hat{\mathcal{L}}_x^T\hat{\delta}x+\hat{\mathcal{L}}_u^T\hat{\delta}u+\hat{\mathcal{L}}_v^T\hat{\delta}v\\
&+\frac{1}{2}\begin{pmatrix}\hat{\delta}x\\ \hat{\delta}u\\ \hat{\delta}v\end{pmatrix}^T\begin{pmatrix}\hat{\mathcal{L}}_{xx} & \hat{\mathcal{L}}_{xu} & \hat{\mathcal{L}}_{xv}\\ \hat{\mathcal{L}}_{ux} & \hat{\mathcal{L}}_{uu} & \hat{\mathcal{L}}_{uv}\\ \hat{\mathcal{L}}_{vx} & \hat{\mathcal{L}}_{vu} & \hat{\mathcal{L}}_{vv}\end{pmatrix}\begin{pmatrix}\hat{\delta}x\\ \hat{\delta}u\\ \hat{\delta}v\end{pmatrix}+\hat{V}_x^T\hat{F}_x\hat{\delta}x+\hat{V}_x^T\hat{F}_u\hat{\delta}u+\hat{V}_x^T\hat{F}_v\hat{\delta}v\\
&+\hat{\delta}x^T\hat{V}_{xx}\hat{F}_x\hat{\delta}x+\hat{\delta}x^T\hat{V}_{xx}\hat{F}_u\hat{\delta}u+\hat{\delta}x^T\hat{V}_{xx}\hat{F}_v\hat{\delta}v\Bigg\}=\min_{\hat{\delta}u}\max_{\hat{\delta}v}\Bigg\{\hat{\mathcal{L}}+\hat{\delta}x^T\hat{Q}_x\\
&+\hat{\delta}u^T\hat{Q}_u+\hat{\delta}v^T\hat{Q}_v+\frac{1}{2}\hat{\delta}x^T\hat{Q}_{xx}\hat{\delta}x+\frac{1}{2}\hat{\delta}u^T\hat{Q}_{uu}\hat{\delta}u+\frac{1}{2}\hat{\delta}v^T\hat{Q}_{vv}\hat{\delta}v\\
&\hat{\delta}u^T\hat{Q}_{ux}\hat{\delta}x+\hat{\delta}v^T\hat{Q}_{vx}\hat{\delta}x+\hat{\delta}u^T\hat{Q}_{uv}\hat{\delta}v\Bigg\},
\end{aligned}
\tag{3.55}
$$

where the Q functions are given in equation (3.42). Through taking the partial derivative of (3.55) with respect to $\hat{\delta}u$ and $\hat{\delta}v$ and equating them to 0, we obtain the optimal control update $\hat{\delta}u^*$ and $\hat{\delta}v^*$ as

$$\hat{\delta}u^*=-\hat{Q}_{uu}^{-1}(\hat{Q}_{ux}\hat{\delta}x+\hat{Q}_{uv}\hat{\delta}v^*+\hat{Q}_u), \tag{3.56a}$$

$$\hat{\delta}v^*=-\hat{Q}_{vv}^{-1}(\hat{Q}_{vx}\hat{\delta}x+\hat{Q}_{vu}\hat{\delta}u^*+\hat{Q}_v), \tag{3.56b}$$

which is summarized by omitting $\hat{\delta}u^*$ and $\hat{\delta}v^*$ on the right hand side of (3.56) and represented as

$$\hat{\delta}u^*=\hat{I}_u+\hat{K}_u\hat{\delta}x, \tag{3.57a}$$

$$\hat{\delta}v^*=\hat{I}_v+\hat{K}_v\hat{\delta}x, \tag{3.57b}$$

which completes the proof. □

Notice that the functions in equation (3.42) require the value function and its first-

order and second-order state derivatives for evaluation. To this end, we incorporate the control update laws (3.37) into the HJBI (3.33) to derive the update laws of the value function along with its first-order and second-order state derivatives. The result is presented in the following theorem.

Theorem 6. *The optimal update of the value function and its first order, and second order state derivatives are governed by the backward ordinary differential equations as follows*

$$-\frac{d\hat{V}}{dt} = \hat{\mathcal{L}} + \hat{I}_u^T\hat{Q}_u + \hat{I}_v^T\hat{Q}_v + \frac{1}{2}\hat{I}_u\hat{\bar{Q}}_{uu}\hat{I}_u + \hat{I}_u^T\hat{Q}_{uv}\hat{I}_v + \frac{1}{2}\hat{I}_v^T\hat{Q}_{vv}\hat{I}_v, \tag{3.58a}$$

$$-\frac{d\hat{V}_x}{dt} = \hat{Q}_x + \hat{K}_u^T\hat{Q}_u + \hat{K}_v^T\hat{Q}_v + \hat{Q}_{ux}^T\hat{I}_u + \hat{Q}_{vx}^T\hat{I}_v + \hat{K}_u^T\hat{Q}_{uu}\hat{I}_u + \hat{K}_u^T\hat{Q}_{uv}\hat{I}_v + \hat{K}_v^T\hat{Q}_{vu}\hat{I}_u + \hat{K}_v^T\hat{Q}_{vv}\hat{I}_v, \tag{3.58b}$$

$$-\frac{d\hat{V}_{xx}}{dt} = \hat{K}_u^T\hat{Q}_{ux} + \hat{Q}_{ux}^T\hat{K}_u + \hat{K}_v^T\hat{Q}_{vx} + \hat{Q}_{vx}^T\hat{K}_v + \hat{K}_v^T\hat{Q}_{vu}\hat{K}_u + \hat{K}_u^T\hat{Q}_{uv}\hat{K}_v + \hat{K}_u^T\hat{Q}_{uu}\hat{K}_u + \hat{K}_v^T\hat{Q}_{vv}\hat{K}_v + \hat{Q}_{xx}, \tag{3.58c}$$

with the terminal condition

$$\hat{V}(t_f) = \phi(\hat{x}(t_f), t_f). \tag{3.59a}$$

$$\hat{V}_x(t_f) = \phi_x(\hat{x}(t_f), t_f). \tag{3.59b}$$

$$\hat{V}_{xx}(t_f) = \phi_{xx}(\hat{x}(t_f), t_f). \tag{3.59c}$$

Proof. The backward propagation equations (3.58) concerning the value function and its first- and second-order partial derivatives can be derived by substituting the forms of $\hat{\delta}u^*$ and $\hat{\delta}v^*$ expressed in (3.37) into equation (3.33). Then, by equating the coefficients of zeroth-, first-, and second-order expressions of $\hat{\delta}x$ on both the left-hand side and

right-hand side of the equation. In detail we have

$$
\begin{aligned}
&-\frac{d\hat{V}}{dt}-\hat{\delta}x^T\frac{d\hat{V}_x}{dt}-\frac{1}{2}\hat{\delta}x^T\frac{d\hat{V}_{xx}}{dt}\hat{\delta}x^T=\hat{\mathcal{L}}+\hat{\delta}x^T\hat{Q}_x+\hat{\delta}u^{*^T}\hat{Q}_u+\hat{\delta}v^{*^T}\hat{Q}_v\\
&+\hat{\delta}u^{*^T}hatQ_{ux}\hat{\delta}x+\frac{1}{2}\hat{\delta}u^{*^T}\hat{Q}_{uu}\hat{\delta}u^*+\hat{\delta}u^{*^T}\hat{Q}_{uv}\hat{\delta}v^*+\frac{1}{2}\hat{\delta}v^{*^T}\hat{Q}_{vv}\hat{\delta}v^*\\
&+\hat{\delta}v^{*^T}\hat{Q}_{vx}\hat{\delta}x+\frac{1}{2}\hat{\delta}x^T\hat{Q}_{xx}\hat{\delta}x\\
&=\hat{\mathcal{L}}+\hat{\delta}x^T\hat{Q}_x+(\hat{I}_u+\hat{K}_u\hat{\delta}x)^T\hat{Q}_u+(\hat{I}_v+\hat{K}_v\hat{\delta}x)^T\hat{Q}_v+(\hat{I}_u+\hat{K}_u\hat{\delta}x)^T\hat{Q}_{ux}\hat{\delta}x\\
&+(\hat{I}_v+\hat{K}_v\hat{\delta}x)^T\hat{Q}_{vx}\hat{\delta}x\\
&\frac{1}{2}\hat{\delta}x^T\hat{Q}_{xx}\hat{\delta}x+(\hat{I}_u+\hat{K}_u\hat{\delta}x)^T\hat{Q}_{uv}(\hat{I}_v+\hat{K}_v\hat{\delta}x)+\frac{1}{2}(\hat{I}_u+\hat{K}_u\hat{\delta}x)^T\hat{Q}_{uu}(\hat{I}_u+\hat{K}_u\hat{\delta}x)\\
&+\frac{1}{2}(\hat{I}_v+\hat{K}_v\hat{\delta}x)^T\hat{Q}_{vv}(\hat{I}_v+\hat{K}_v\hat{\delta}x)
\end{aligned}
\tag{3.60}
$$

after equating the right-hand side of equation (3.60) as zeroth, first, and second-order expressions of $\hat{\delta}x$, we put the coefficients of $\hat{\delta}x$ in the left-hand side and right-hand side of equation (3.60) and calculate the backward propagation equations with respect to the value function and its first- and second-order partial derivatives as

$$-\frac{d\hat{V}}{dt}=\hat{\mathcal{L}}+\hat{I}_u^T\hat{Q}_u+\hat{I}_v^T\hat{Q}_v+\frac{1}{2}\hat{I}_u\hat{\tilde{Q}}_{uu}\hat{I}_u+\hat{I}_u^T\hat{Q}_{uv}\hat{I}_v+\frac{1}{2}\hat{I}_v^T\hat{Q}_{vv}\hat{I}_v, \tag{3.61a}$$

$$
\begin{aligned}
&-\frac{d\hat{V}_x}{dt}=\hat{Q}_x+\hat{K}_u^T\hat{Q}_u+\hat{K}_v^T\hat{Q}_v+\hat{Q}_{ux}^T\hat{I}_u+\hat{Q}_{vx}^T\hat{I}_v+\hat{K}_u^T\hat{Q}_{uu}\hat{I}_u\\
&+\hat{K}_u^T\hat{Q}_{uv}\hat{I}_v+\hat{K}_v^T\hat{Q}_{vu}\hat{I}_u+\hat{K}_v^T\hat{Q}_{vv}\hat{I}_v,
\end{aligned}
\tag{3.61b}
$$

$$
\begin{aligned}
&-\frac{d\hat{V}_{xx}}{dt}=\hat{K}_u^T\hat{Q}_{ux}+\hat{Q}_{ux}^T\hat{K}_u+\hat{K}_v^T\hat{Q}_{vx}+\hat{Q}_{vx}^T\hat{K}_v+\hat{K}_v^T\hat{Q}_{vu}\hat{K}_u\\
&+\hat{K}_u^T\hat{Q}_{uv}\hat{K}_v+\hat{K}_u^T\hat{Q}_{uu}\hat{K}_u+\hat{K}_v^T\hat{Q}_{vv}\hat{K}_v+\hat{Q}_{xx},
\end{aligned}
\tag{3.61c}
$$

and by having the Taylor series expansion around $x(t_f)$ we can obtain the following

expression with respect to the terminal condition $V(x(t_f), t_f) = \phi(x(t_f), t_f)$

$$\phi(x(t_f), t_f) = \phi(x(t_f) + \hat{\delta}x(t_f), t_f) \approx \phi(x(t_f), t_f) + \hat{\delta}x(t_f)^T \phi_x x(t_f), t_f)$$
$$+ \hat{\delta}x(t_f)^T \phi_{xx}(x(t_f), t_f)\hat{\delta}x(t_f). \tag{3.62}$$

Therefore, the terminal conditions at $t = t_f$ for the backward differential equations (3.58) are

$$\hat{V}(t_f) = \phi(\hat{x}(t_f), t_f). \tag{3.63a}$$

$$\hat{V}_x(t_f) = \phi_x(\hat{x}(t_f), t_f). \tag{3.63b}$$

$$\hat{V}_{xx}(t_f) = \phi_{xx}(\hat{x}(t_f), t_f). \tag{3.63c}$$

Which this complete the proof. □

With the approximation of the dynamics, the HOSM observer-based min-max DDP algorithm is proposed in Algorithm 3.

Remark 6. *In This chapter, we have several parameters that need to be selected and tuned according to different scenarios like dynamics of the system and the cost function. For instance, the parameter γ is a step size multiplier ($0 < \gamma \leq 1$) used to iteratively update the control policies in the optimization process. It is chosen through trial and error, starting with a small value (e.g., $\gamma = 0.01$) to ensure stability and avoiding divergence during updates. A larger γ may speed up convergence but can cause oscillations or instability. The convergence threshold ϵ ensures negligible control updates and provides a balance between efficiency and accuracy. The weighting matrices Q_f, R_u, and R_v were chosen based on system dynamics and the trade-off between control effort and state accuracy, with sensitivity analysis confirming robustness to moderate variations. The multipliers λ and A in the HOSM observer were manually tuned to ensure fast and*

robust convergence, with initial values selected based on system parameters and refined through simulations.

Algorithm 3 Observer-based min-max DDP algorithm

Require: Initial values for x_0, minimizing control u, maximizing control v, terminal time t_f, multiplier γ, large integer N, and small value ϵ.

Ensure: Optimal minimizing control $\hat{u}^*$, optimal maximizing control $\hat{v}^*$, gains $\hat{I}_u$, $\hat{I}_v$, $\hat{K}_u$, and $\hat{K}_v$.

procedure UPDATE CONTROL POLICIES:

Construct the HOSM observer model (3.26) from the output of the dynamics (3.1) forward with x_0.

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

while $|\hat{\delta}u| + |\hat{\delta}v| > \epsilon$ **do**

Compute values of $\hat{V}$, $\hat{V}_x$, and $\hat{V}_{xx}$ at t_f from equation (3.59);

Integrate backward the equation (3.58);

Compute $\hat{I}_u$, $\hat{I}_v$, $\hat{K}_u$, and $\hat{K}_v$ through equations (3.38) to (3.41);

Integrate equation (3.35) by replacing $\hat{\delta}u$ and $\hat{\delta}v$ with $\hat{\delta}u = \hat{I}_u + \hat{K}_u\hat{\delta}x$ and $\hat{\delta}v = \hat{I}_v + \hat{K}_v\hat{\delta}x$ to get $\hat{\delta}x$;

Obtain $\hat{\delta}u = \hat{I}_u + \hat{K}_u\hat{\delta}x$ and $\hat{\delta}v = \hat{I}_v + \hat{K}_v\hat{\delta}x$;

Update $\hat{u} = \hat{u} + \gamma\hat{\delta}u$ and $\hat{v} = \hat{v} + \gamma\hat{\delta}v$;

Set $u^* = \hat{u}$ and $v^* = \hat{v}$;

Apply u^* and v^* to dynamics (3.1) and measure the output of the system.

Construct the HOSM observer model (3.26).

end while

end for

return u^*, v^*, $\hat{I}_u$, $\hat{I}_v$, $\hat{K}_u$, and $\hat{K}_v$;

end procedure

3.6 Numerical examples

In this section, we demonstrate the results of two practical examples to show the effectiveness of the proposed algorithm.

3.6.1 Inverted Pendulum

In the first example, we apply the proposed algorithm to the inverted pendulum [56] with conflicting controls, while u tries to minimize the cost function (3.65), v tries to maximize it. The dynamics of the system is given by

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= mglsin(x_1)/I - bx_2/I + (u-v)/I, \\ y &= x_1, \end{aligned} \tag{3.64}$$

where $m = 1kg$, $l = 0.5m$, $b = 0.1(N.s)/m$, $I = ml^2$. $g = 9.81(kg.m)/s^2$. The initial conditions are selected as $x_0 = [\pi, 0]$, and $t_f = 1$ and the goal is to bring the states from $[\theta, \dot{\theta}] = [\pi, 0]$ to $[\theta, \dot{\theta}] = [0, 0]$. The cost function is defined as

$$J = \frac{1}{2}(x - x_f)^T Q_f (x - x_f) + \int_{t_0}^{t_f} (u^T R_u u - v^T R_v v) dt, \tag{3.65}$$

where initial control policies are $u = v = 0$, $t_f = 1$, and $\gamma = 1$. The weights are chosen as $Q_f = [100, 0; 0, 5]$, $R_u = 0.1$, and $R_v = 1$. The approximate dynamics using HOSM is obtained as

$$\begin{aligned} \dot{\hat{x}}_1 &= \hat{x}_2 + \lambda_1 sign(y_1 - \hat{x}_1), \\ \dot{\hat{x}}_2 &= A_1 sin(y_1) + A_2 \hat{x}_2 + A_3(u+v) + \lambda_2 sign(\lambda_1 sign(y_1 - \hat{x}_1)_{eq}), \\ y &= x_1, \end{aligned} \tag{3.66}$$

which λ_1, λ_2, A_1, A_2, and A_3 are multipliers of the dynamics that can be tuned manually, which are $\lambda_1 = 0.5$, $\lambda_2 = 5.45$, $A_1 = 0.1$, $A_2 = -3$, and $A_3 = 0.4$.

In this example, we just can measure the output of the system which is x_1. Having

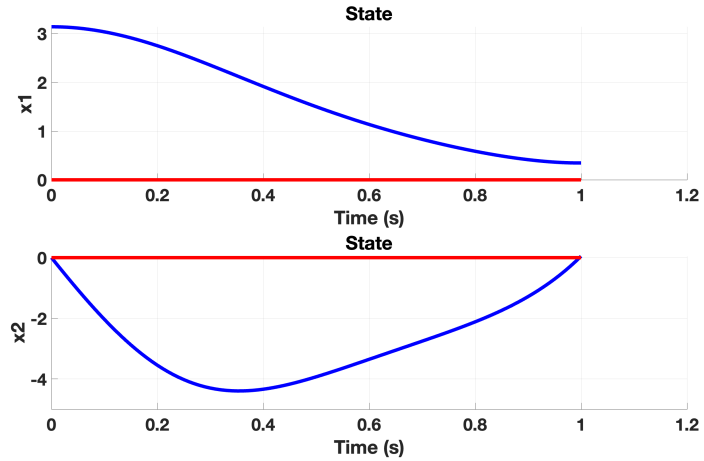


Figure 3.1: State trajectories of double inverted pendulum.

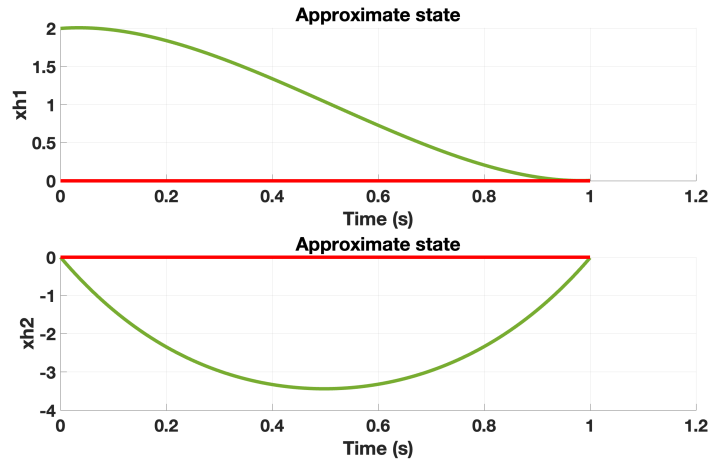


Figure 3.2: Approximate state trajectories of double inverted pendulum.

the output of the system and using the proposed algorithm, we can approximate the dynamics of the system and obtain other states and use them to compute update control laws. Upon comparing Fig. 3.1 which depicts the states trajectory of the inverted pendulum, with Fig. 3.2, illustrating the approximate states trajectory, we observe that the proposed method successfully captures the system dynamics, and the states converge effectively.

The effectiveness of the proposed method in directing the trajectories of the system

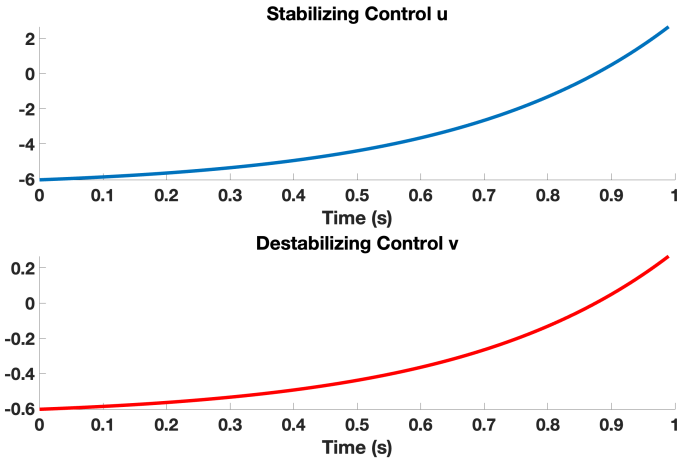


Figure 3.3: Control policies of double inverted pendulum.

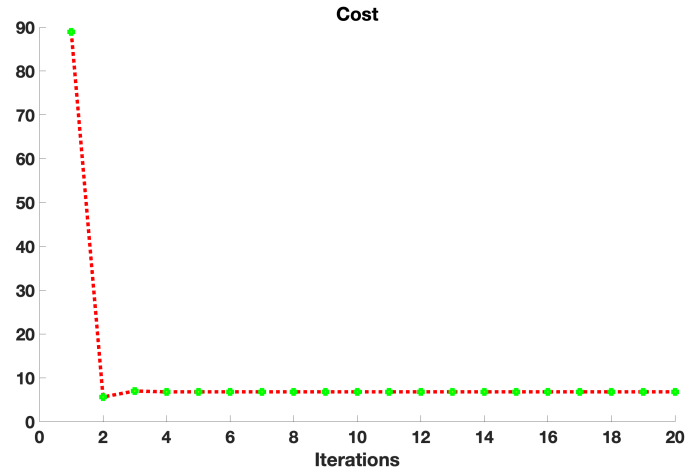


Figure 3.4: Cost per iteration of double inverted pendulum.

towards the desired goal is demonstrated in Fig. 3.3. Besides, the illustration in Fig. 3.4 presents the cost per iteration and provides the evidence of converging the cost towards 0 throughout the span of 20 iterations.

3.6.2 2-DOF Helicopter

In the second example, results of applying the proposed method to the 2-Degree-of-Freedom (2-DOF) Helicopter are illustrated [185]. The states of the system are denoted

as

$$x^T = [\theta, \psi, \dot{\theta}, \dot{\psi}], \tag{3.67}$$

with the control variable

$$u^T = [u_p, u_y]. \tag{3.68}$$

The control policies u_p and u_y are for pitch and yaw as follows

$$\begin{aligned} u_p &= K_{pp}V_p + K_{py}V_y, \\ u_y &= K_{yy}V_y + K_{yp}V_p. \end{aligned} \tag{3.69}$$

The dynamics of the system is given by

$$\begin{aligned} \dot{x}_1 &= x_3, \\ \dot{x}_2 &= x_4, \\ \dot{x}_3 &= b_p u_p + \Psi_1^T(x)\Theta_1, \\ \dot{x}_4 &= b_y u_y + \Psi_2^T(x)\Theta_2, \\ y &= x_1, \end{aligned} \tag{3.70}$$

where

$$\Psi_1 = \begin{pmatrix} -x_3 \\ -sin(x_1) \\ x_4^2 cos(x_1) sin(x_1) \end{pmatrix}, \quad \Psi_2 = \begin{pmatrix} -x_4 \\ -x_2 x_4 cos(x_1) sin(x_1) \end{pmatrix}$$

$$\Theta_1 = \frac{1}{I_p + mI_{cm}^2} \begin{pmatrix} D_{v_P} \\ mgI_{cm} \\ mI_{cm}^2 \end{pmatrix}, \quad \Theta_2 = \frac{I}{I_y} \begin{pmatrix} D_{v_y} \\ 2mI_{cm}^2 \end{pmatrix}, \tag{3.71}$$

and b_p and b_y are constants defined as follows

$$b_p = \frac{1}{I_p + mI_{cm}^2}, \quad b_y = \frac{1}{I_y}. \tag{3.72}$$

The values of the parameters are shown in Table 3.1.

Parameter	Description	value
K_{pp}	Pitch torque	0.204 $N.m/V$
K_{yy}	Yaw torque	0.072 $N.m/V$
K_{py}	Yaw on pitch torque	0.0068 $N.m/V$
K_{yp}	Pitch on yaw torque	0.0219 $N.m/V$
I_p	Total pitch moment of inertia	0.0215 $kg.m^2$
I_y	Total yaw moment of inertia	0.037 $kg.m^2$
D_{v_P}	Pitch viscous damping	0.00071 NN
D_{v_y}	Yaw viscous damping	0.0022 NN
m	Total moving mass	1.075 kg
g	Acceleration due gravity	9.81 m/s^2
I_{cm}	Center of mass length from pitch axis	0.002 m

Table 3.1: Parameter values of 2-DOF helicopter.

The goal is to compute optimal control policies u_p and u_y to bring the states $[\theta, \psi, \dot{\theta}, \dot{\psi}]$ from $[1, 1, 0, 0]$ to $[-1, -2, 0, 0]$. In this illustration, we solely measure the output x_1 of the system. Through utilizing the HOSM algorithm with this output, we

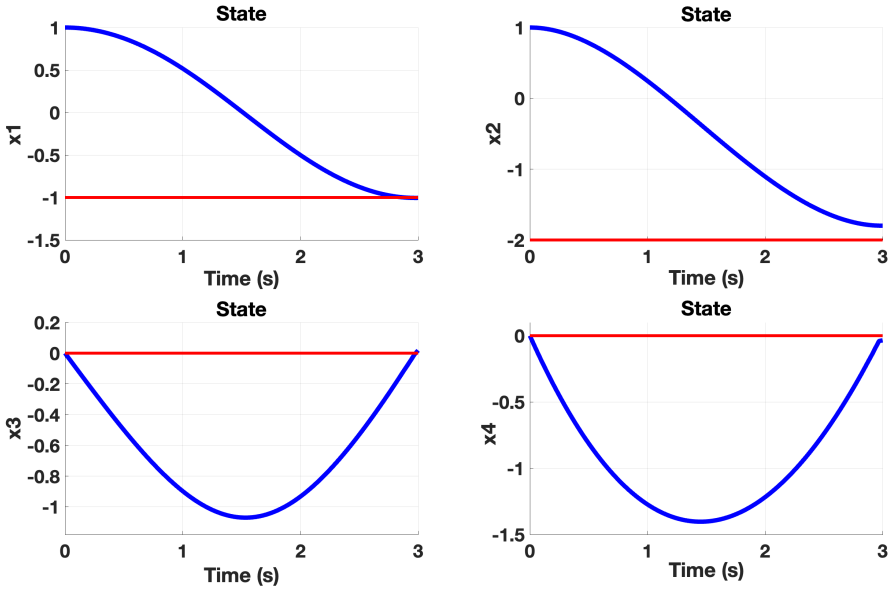


Figure 3.5: State trajectories of 2-DOF helicopter.

can estimate the behavior of the system and derive additional states. The approximate dynamics can be obtained through HOSM as

$$\begin{aligned}
\dot{\hat{x}}_1 &= \hat{x}_3 + \lambda_1 sign(y - \hat{x}_1), \\
\dot{\hat{x}}_2 &= \hat{x}_4 + \lambda_2 sign(\lambda_1 sign(y_1 - \hat{x}_1)), \\
\dot{\hat{x}}_3 &= A_1\hat{x}_3 + A_2 sin(y) + A_3\hat{x}_4^2 cos(y) sin(y) \\
&+ A_4 u_p + \lambda_3 sign(\lambda_2 sign(\lambda_1 sign(y_1 - \hat{x}_1))), \\
\dot{\hat{x}}_4 &= A_5 + \hat{x}_4 + A_6\hat{x}_2\hat{x}_4 cos(y) sin(y) + A_7 u_y \\
&+ \lambda_4 sign(\lambda_3 sign(\lambda_2 sign(\lambda_1 sign(y_1 - \hat{x}_1)))), \\
y &= x_1. \qquad (3.73)
\end{aligned}$$

Where λ_1, λ_2, λ_3, λ_4, A_1, A_2, A_3, A_4, A_5, A_6 and A_7 are multipliers of the dynamics that can be tuned manually, which are $\lambda_1 = 4$, $\lambda_2 = 3$, $\lambda_3 = 5$, $\lambda_4 = 2$, $A_1 = -0.03$, $A_2 = -0.9$, $A_3 = 0.019$, $A_4 = 0.01$, $A_5 = -0.05$, $A_6 = -0.02$ and $A_7 = 0.1$.

The obtained states enable us to formulate updated control strategies. Upon comparing the trajectory of states depicted in Fig. 3.5 (representing the 2-DOF helicopter) with that in Fig. 3.6 (illustrating the approximated trajectory of states), it becomes evident that the proposed approach adeptly captures the dynamics of the system, leading to effective convergence of states.

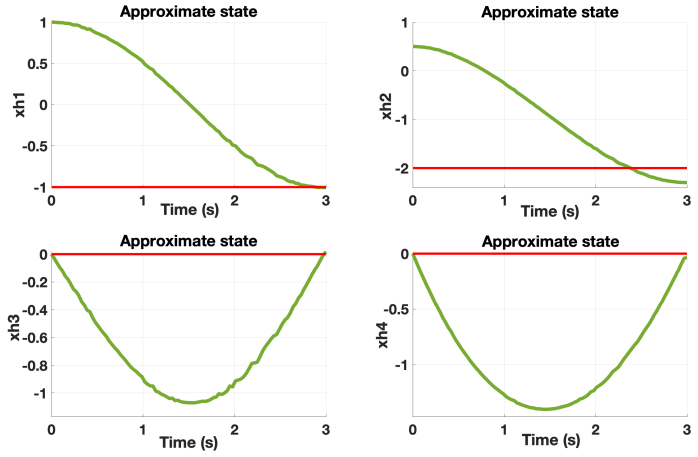


Figure 3.6: Approximate state trajectories of 2-DOF helicopter.

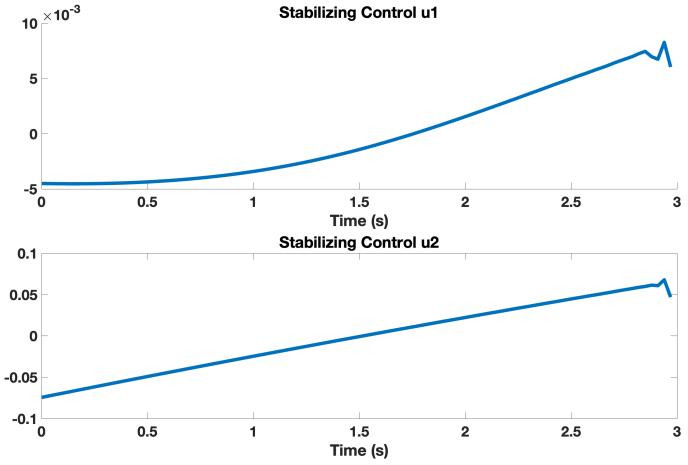


Figure 3.7: Stabilizing control policies of 2-DOF helicopter.

The efficacy of conflicting controls u and v in guiding the system toward the desired objective is showcased in Fig. 3.7 and Fig. 3.8. Additionally, Fig. 3.9 demonstrates

the cost incurred per iteration, revealing a compelling trend where costs consistently diminish towards 0 over the course of 20 iterations.

In comparison with previous works, the proposed method in This chapter achieves superior results in terms of transient response and tracking accuracy for the 2-DOF helicopter system. Unlike the prescribed performance-based approach in [186], which assumes full-state accessibility, our method relies on output feedback control, using a Higher-Order Sliding Mode (HOSM) observer to estimate system states. Despite this additional challenge, our approach achieves faster settling time, reduced overshoot and undershoot, and improved robustness against system uncertainties.

The results of the proposed method in This chapter for 2-DOF helicopter demonstrate faster convergence, reduced overshoot, and improved tracking accuracy compared to the prescribed-time adaptive backstepping control in [187]. While their method ensures a predefined convergence time, it exhibits larger oscillations during transients. Additionally, their approach assumes full-state accessibility, whereas the proposed method operates under the more practical constraint of output feedback. Despite this limitation, the results indicate superior performance in handling system uncertainties, making this approach more effective for the 2-DOF helicopter system.

In [188], a fuzzy PID controller is developed for the 2-DOF helicopter. The results show that the system outputs converge to the target values and reference signals in approximately 10 seconds, which is significantly longer than the convergence time achieved by the proposed method. Due to the finite-time nature of Min-Max DDP, the proposed method achieves convergence in under 3 seconds, with no oscillations, overshoot, or undershoot, demonstrating superior transient performance.

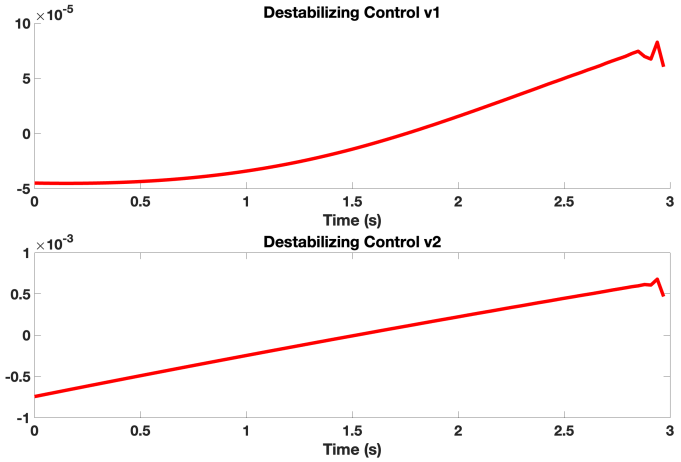


Figure 3.8: Destabilizing control policies of 2-DOF helicopter.

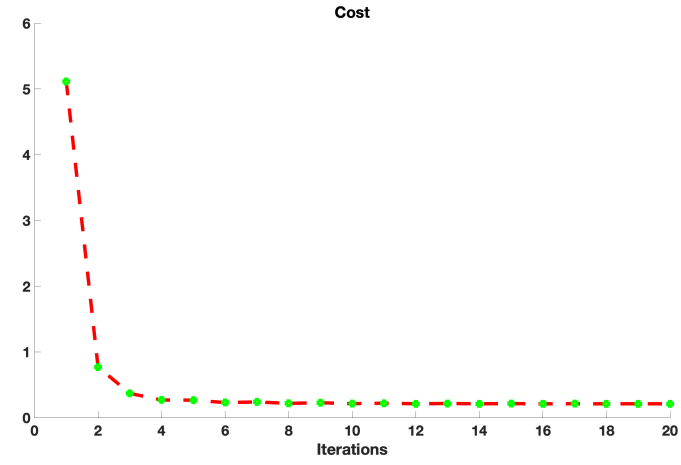


Figure 3.9: Cost per Iteration of 2-DOF helicopter.

Chapter 4

Data-Driven Min-Max Differential Dynamic Programming for Nonlinear Systems

4.1 Introduction

Differential game is an extension of the optimal control concept in control theory that deals with conflicting controls in the context of dynamical systems, which has direct connection to robust control as well as risk-sensitive control [189, 190, 191]. While significant progress has been made in this area through previous research, there are still numerous issues that need to be addressed, particularly when it comes to the trajectory optimization with complex dynamics identification. Min-max differential dynamic programming [8, 192, 56] is an iterative algorithm that solves two-player zero-sum differential games by utilizing the dynamic programming principle and tries to find optimal conflicting control policies by iteratively improving nominal trajectories of states and controls. The idea is based on the original differential dynamic programming (DDP) [4] that deals with optimal control problems. One of the major advantage of the DDP algorithm is its second-order convergence rate under some mild assumptions [193]. Comparison between DDP and other local methods have consistently demonstrated the

superiority of DDP in optimization and robustness [193, 194]. DDP has also achieved notable success in various practical applications, such as surgical systems, autopilots, unmanned aerial robots [193, 195, 196]. Even though DDP was initially derived in continuous time, most previous work in optimal control and differential games, such as the min-max DDP studied in [192], focus on either nonlinear systems operating in discrete time or systems initially represented in continuous time but converted into discrete time versions.

While significant efforts have been made to develop various forms of the DDP approaches and apply them to different systems in the field of engineering, the majority of existing literature primarily focuses on systems with known dynamics. In [197], discrete-time DDP is applied to bio-mechanical models considering state and control multiplicative noise. In [45], the primary emphasis was placed on integrating second-order approximations within a stochastic dynamical system characterized by multiplicative noise affecting both state and control variables, which led to the development of a novel algorithm called Stochastic Differential Dynamic Programming (SDDP). In [198], by using random sampling of states and local trajectories, the control policy and associated value function are optimized globally. The application of DDP to deterministic nonlinear dynamics with control limits is described in [5]. DDP is applied in [39] for trajectory optimization of hybrid systems involving the state-based switching mechanism. DDP for nonlinear systems with nonlinear constraints is studied in [50]. The paper [199] utilized DDP for constrained optimal control by considering a constrained type of Bellman's principle of optimality. Meanwhile, the effectiveness of model-based controls such as DDP heavily relies on the precision and accuracy of the underlying mathematical model. To address the challenge when the dynamical model is not given a prior, we need tools that can obtain a dynamic model that is capable of capturing the complexity and nonlinearity of the system. Machine learning

methods such as neural networks [200, 150] have been widely used to cope with these complex dynamics and solve the problem of system identification. Nevertheless, there are some disadvantages regarding using the Neural Networks (NNs) like NNs, especially deep architectures, require extensive computational resources for training and inference, making them less practical for real-time control applications.[201]. On the other hand, NNs rely on gradient-based optimization, which may result in convergence to suboptimal local minima, leading to reduced control performance [202]. Moreover, NNs, especially with limited training data, are prone to overfitting, resulting in poor generalization to unseen system conditions [203]. Furthermore, NNs performance is highly dependent on the choice of hyperparameters (e.g., learning rate, architecture, activation functions), requiring extensive tuning and large amounts of labeled training data [204]. In this study, we present the development of a data driven Game-Theoretic Differential Dynamic Programming algorithm operating in continuous time. This algorithm is designed to solve a two-player zero-sum differential game in cases where the system dynamics are entirely unknown. We achieve this by employing the Gaussian Process (GP) [205], which mitigates the disadvantages of using NNs.

The GP approximation is a popular framework in probabilistic Artificial Intelligence (AI) including supervised machine learning [206]. It has been extensively applied in regression and classification tasks. GP regression enables making predictions with prior knowledge and provides uncertainty estimations for these predictions [207]. It is based on key principles such as the multivariate normal distribution, kernels, non-parametric models, and joint and conditional probability. In recent years, there has been a growing interest in GP for both model learning and control tasks [9, 62]. Most of the current systems have nonlinear and complex dynamics which cannot be obtained directly and identification techniques are required to approximate such systems. In this research, we have effectively addressed the limitations of previous game theoretic DDP [8] algorithm

where the exact dynamics is needed to obtain the controls. We also provide a set of backward differential equations in This chapter and derive the optimal policies for the two players.

The contribution of the paper is divided into several aspects

1) A min-max zero-sum differential game problem with unknown nonlinear dynamics is solved to obtain the control policies.

2) GP is utilized to approximate the nonlinear dynamics of the system with conflicting controls.

3) The optimal update laws of the control policies of the two players and the corresponding set of backward differential equations are provided.

The remainder of the paper is structured as follows: In Section 4.2, we introduce important notations used throughout This chapter. In Section 4.3, the primary problem formulation including system description and the establishment of the GP is proposed. In 4.4, DDP-based optimal control updates and the backward propagation of the value function approximation are studied. Finally, in Section 4.5, simulation results of the proposed algorithm are demonstrated.

4.2 Notation

In this section, some standard notations that are extensively used throughout This chapter are mentioned. $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{R}^n$, and $\mathbb{R}^{n\times m}$ denote the set of all real numbers, positive real numbers, $n \times 1$ real column vectors, and $n \times m$ real matrices, respectively. $\mathcal{C}^1(\mathbb{R})$ and $\mathcal{C}^2(\mathbb{R})$ represent sets of functions whose first and second-order derivative exist and is continuous with domain $\mathbb{R}$. For any specified functions $\Psi : \Re^n \times \Re \to \Re$

and $W : \Re^n \times \Re \to \Re^n$, the following are defined

$$\Psi_x(x,t) = \begin{bmatrix} \frac{\partial \Psi(x,t)}{\partial x_1} \\ \vdots \\ \frac{\partial \Psi(x,t)}{\partial x_n} \end{bmatrix},$$

$$\Psi_{xx}(x,t) = \begin{bmatrix} \frac{\partial^2 \Psi(x,t)}{\partial^2 x_1}, & \cdots & , \frac{\partial^2 \Psi(x,t)}{\partial x_1 x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi(x,t)}{\partial x_1 \partial x_n}, & \cdots & , \frac{\partial^2 \Psi(x,t)}{\partial^2 x_n} \end{bmatrix},$$

$$W_x(x,t) = \begin{bmatrix} \frac{\partial W_1(x,t)}{\partial x_1}, & \cdots & , \frac{\partial W_1(x,t)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial W_n(x,t)}{\partial x_1}, & \cdots & , \frac{\partial W_n(x,t)}{\partial x_n} \end{bmatrix},$$

that represent partial derivative of Ψ with respect to x, Hessian of Ψ with respect to x, and the Jacobian matrix of W with respect to x, respectively.

4.3 Problem Formulation

In this section, the dynamical model of the system, the cost function, and the value function are introduced. Then, the definition and formulation of the Gaussian Process (GP) are described.

4.3.1 System Description

Consider the nonlinear dynamics of the min-max differential game as follows

$$\frac{dx(t)}{dt} = F(x(t), u(t), v(t)), \quad x(t_0) = x_0 \tag{4.1}$$

where $x(t) \in \Re^n$, $t \in [t_0, t-f]$ is the state and $u \subset \Re^{m_u}$, $v \subset \Re^{m_v}$ are control policies. The dynamics $F(x(t), u(t), v(t))$ is assumed to be unknown and continuous.

The cost function of the min-max differential game is defined as

$$J(x, u, v) = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}(x(t), u(t), v(t))dt, \tag{4.2}$$

where $u(t)$ tries to minimize the cost function while $v(t)$ tries to maximize it. Here, $\mathcal{L}(x(t), u(t), v(t))$ denotes the running cost and $\phi(x(t_f), t_f)$ is the terminal cost.

The value function of the differential game represents the min-max value of the equation (4.2), as expressed in

$$V(x_0, t_0) = \min_u \max_v \left\{ \phi(x(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}(x(t), u(t), v(t))dt \right\}. \tag{4.3}$$

4.3.2 Gaussian Process

The continuous function F in (4.1) that maps (x, u, v) to dx/dt can be considered as an inference of dx/dt given (x, u, v). In this subsection, the Gaussian Process (GP) is introduced to learn the dynamics F. A scalar random variable X is characterized as Gaussian distributed with mean μ and variance σ^2, if its probability density function

(PDF) is defined as follows

$$P_X(x) = \frac{1}{\sqrt{2\pi\sigma}} exp\bigg(- \frac{(x-\mu)^2}{2\sigma^2} \bigg). \tag{4.4}$$

In this context, x is the real argument of the random variable X. The normal distribution of X is denoted by $X \sim \boldsymbol{N}(\mu, \sigma^2)$. The multivariate Gaussian/normal (MVN) [208] distribution is utilized when all the components in a vector are Gaussian distributed. The PDF of an MVN is represented as

$$\boldsymbol{N}\bigg(x|\mu, \sum\bigg) = \frac{1}{(2\pi)^{D/2}|\sum|^{1/2}} exp\bigg[- \frac{1}{2}(x-\mu)^T (\sum)^{-1}(x-\mu)\bigg], \tag{4.5}$$

where D, x, $\mu = E[x] \in \Re^D$, and $\sum = cov[x]$ denote the dimension, variable, mean, and $D \times D$ covariance matrix, respectively.

The GP model characterizes a probability distribution over functions that fit a set of points. This approach enables the computation of the function mean to serve as predictions and the variance to express the level of confidence in those predictions. The standard GP model with regression function can be represented as a multivariate Gaussian as follows

$$P(f|X) = \boldsymbol{N}(f|\mu, K), \tag{4.6}$$

where $X = [x_1, ..., x_n]$ denotes the data points, $f = [f(x_1), ..., f(x_n)]$ denotes the learned model, $\mu = [m(x_1), ..., m(x_n)]$ represents the mean, and K denotes the kernel function where $K_{ij} = k(x_i, x_j)$, and

$$k(x_i, x_j) = exp\bigg(- \frac{w_{ij}(x_i - x_j)^2}{2} \bigg), \tag{4.7}$$

where w_{ij} represents the hyper-parameter of the GP model. The function prediction at a new data point x_* can be then calculated as $f(x_*)$.

The joint distribution of $f(X)$ and $f(x_*)$ are given by

$$\begin{bmatrix} f(X) \\ f(x_*) \end{bmatrix} \sim \boldsymbol{N}\Bigg(\begin{bmatrix} \mu(X) \\ m(x_*) \end{bmatrix}, \begin{bmatrix} K & K_* \\ K_*^T & K_{**} \end{bmatrix} \Bigg), \tag{4.8}$$

where $K = K(X, X), K_* = K(X, x_*)$ and $K_{**} = K(x_*, x_*)$.

Remark 7. *The Gaussian (RBF) kernel was selected due to its smoothness, flexibility, and robust uncertainty quantification, which are crucial for approximating unknown continuous-time system dynamics. Its nonparametric nature allows accurate function approximation without strong structural assumptions. While alternative kernels such as Matérn and polynomial kernels were considered, empirical evaluations showed that the Gaussian kernel provided better accuracy and numerical stability. Kernel selection can be further refined using strategies like empirical validation through cross-validation, Bayesian optimization for hyperparameter tuning, and domain-informed selection based on system characteristics.*

4.4 Data-Driven Min-Max Differential Dynamic Programming

4.4.1 Optimal Control Policy Update

In this section, the optimal control policies updates are derived from the Hamilton-Jacobi-Bellman-Isaacs (HJBI) Partial Differential Equation (PDE) that is satisfied by

the value function of the min-max differential game. The HJBI is given by

$$-\frac{\partial V(x,t)}{\partial t} = \min_{u} \max_{v} \{\mathcal{L}(x(t), u(t), v(t)) + V_x(x,t)^T F(x(t), u(t), v(t))\}, \tag{4.9}$$

with the terminal condition $V(x(t_f), t_f) = \phi(x(t_f), t_f)$.

Given a nominal state and control histories $(\bar{x}, \bar{u}, \bar{v})$, the nonlinear system (4.1) can be written as

$$\frac{dx}{dt} = F(\bar{x} + \delta x, \bar{u} + \delta u, \bar{v} + \delta v), \tag{4.10}$$

where $\delta x = x - \bar{x}$, $\delta u = u - \bar{u}$,and $\delta v = v - \bar{v}$.

The system can be linearized around $(\bar{x}, \bar{u}, \bar{v})$ to obtain

$$\frac{d\delta x}{dt} = \bar{F}_x \delta x + \bar{F}_u \delta u + \bar{F}_v \delta v, \tag{4.11}$$

where

$$\bar{F}_x = F_x(\bar{x}, \bar{u}, \bar{v}).$$

$$\bar{F}_u = F_u(\bar{x}, \bar{u}, \bar{v}).$$

$$\bar{F}_v = F_v(\bar{x}, \bar{u}, \bar{v}).$$

Under the assumption in This chapter that the nonlinear system dynamics is unknown, there is no prior knowledge of $\bar{F}_x, \bar{F}_u, \bar{F}_v$ and they need to be calculated from the approximate dynamics of the system (4.1). As discussed in the previous section, based on (4.8), the joint probability distribution of the observed output and the predicted

function value $F(X^*)$ is given by

$$\begin{bmatrix} F(X) \\ F(X^*) \end{bmatrix} \sim \boldsymbol{N}\Bigg(\mu_{\text{prior}}, \begin{bmatrix} K(X,X) & K(X,X^*) \\ K(X^*,X) & K(X^*,X^*) \end{bmatrix}\Bigg), \tag{4.12}$$

where $F(X)$ is obtained from a collection of data points $X_i = (x_i, u_i, v_i), i \in \{1, 2, ..., N\}$, μ_{prior} is the prior mean of the joint distribution, which is assumed to be 0 when no prior information of the dynamics is provided, and $K(X,X)$ has entries $K(X,X)(i,j) = K(X_i, X_j), i, j \in \{1, 2, ..., N\}$, and $K(X,X^*)$ has row entries $K(X,X^*)(i) = K(X_i, X^*)$, $i \in 1, ..., N, K(X^*,X) = K(X,X^*)^T$. Given the joint probability distribution (4.12), the predictive mean of the function can be specified as

$$E[F(X^*)] = K(X^*,X)(K(X,X))^{-1}F(X) \tag{4.13}$$

The value of $\bar{F}_x, \bar{F}_u, \bar{F}_v$ can then be approximated by the derivative of the GP model. From the property that the derivative of a GP is still a GP, the joint probability distribution of the function observation and the derivative of the predicted function can be calculated as

$$\begin{bmatrix} F(X) \\ \nabla_X F(X_*) \end{bmatrix} \sim \boldsymbol{N}\Bigg(0, \begin{bmatrix} K & \nabla_X K_* \\ \nabla_X K_*^T & \nabla_{XX} K_{**} \end{bmatrix}\Bigg), \tag{4.14}$$

where

$$\nabla_X F(X_*) = \begin{bmatrix} \hat{F}_x(X_*) \\ \hat{F}_u(X_*) \\ \hat{F}_v(X_*), \end{bmatrix} \tag{4.15}$$

and $\hat{F}_x$, $\hat{F}_u$, $\hat{F}_v$ are the approximation of $\bar{F}_x, \bar{F}_u, \bar{F}_v$ at $X_* = (x_*, u_*, v_*)$, respectively. Therefore, the linearized system (4.11) can be approximated by

$$\frac{d\delta\hat{x}}{dt} = \hat{F}_x\delta\hat{x} + \hat{F}_u\delta\hat{u} + \hat{F}_v\delta\hat{v}, \tag{4.16}$$

where all the terms with $\hat{\cdot}$ corresponds to the approximated dynamics learned from GP. The $\delta\hat{u}$ and $\delta\hat{v}$ in equation (4.16) are updated iteratively using the approximate system and the optimal control policy update laws are presented as follows.

Let

$$\frac{d\hat{x}}{dt} = \hat{F}(\bar{x} + \delta\hat{x}, \bar{u} + \delta\hat{u}, \bar{v} + \delta\hat{v}), \tag{4.17}$$

be the learned dynamics, the optimal update laws for $\delta\hat{u}$ and $\delta\hat{v}$ are obtained as

$$\delta\hat{u}^* = \hat{I}_u + \hat{K}_u\delta\hat{x}, \tag{4.18a}$$

$$\delta\hat{v}^* = \hat{I}_v + \hat{K}_v\delta\hat{x}, \tag{4.18b}$$

where

$$\hat{I}_u = -(\hat{Q}_{uu} - \hat{Q}_{uv}\hat{Q}_{vv}^{-1}\hat{Q}_{vu})^{-1}(\hat{Q}_u - \hat{Q}_{uv}\hat{Q}_{vv}^{-1}\hat{Q}_v). \tag{4.19}$$

$$\hat{I}_v = -(\hat{Q}_{vv} - \hat{Q}_{vu}\hat{Q}_{uu}^{-1}\hat{Q}_{uv})^{-1}(\hat{Q}_v - \hat{Q}_{vu}\hat{Q}_{uu}^{-1}\hat{Q}_u). \tag{4.20}$$

$$\hat{K}_u = -(\hat{Q}_{uu} - \hat{Q}_{uv}\hat{Q}_{vv}^{-1}\hat{Q}_{vu})^{-1}(\hat{Q}_{ux} - \hat{Q}_{uv}\hat{Q}_{vv}^{-1}\hat{Q}_{vx}). \tag{4.21}$$

$$\hat{K}_v = -(\hat{Q}_{vv} - \hat{Q}_{vu}\hat{Q}_{uu}^{-1}\hat{Q}_{uv})^{-1}(\hat{Q}_{vx} - \hat{Q}_{vu}\hat{Q}_{uu}^{-1}\hat{Q}_{ux}). \tag{4.22}$$

and

$$\hat{Q}_x = \hat{F}_x^T\hat{V}_x + \hat{\mathcal{L}}_x. \tag{4.23a}$$

$$\hat{Q}_u = \hat{F}_u^T\hat{V}_x + \hat{\mathcal{L}}_u. \tag{4.23b}$$

$$\hat{Q}_v = \hat{F}_v^T\hat{V}_x + \hat{\mathcal{L}}_v. \tag{4.23c}$$

$$\hat{Q}_{xx} = \hat{\mathcal{L}}_{xx} + \hat{V}_{xx}\hat{F}_x + \hat{F}_x^T\hat{V}_{xx}. \tag{4.23d}$$

$$\hat{Q}_{uu} = \hat{\mathcal{L}}_{uu}. \tag{4.23e}$$

$$\hat{Q}_{vv} = \hat{\mathcal{L}}_{vv}. \tag{4.23f}$$

$$\hat{Q}_{ux} = \hat{\mathcal{L}}_{ux} + \hat{F}_u^T\hat{V}_{xx}. \tag{4.23g}$$

$$\hat{Q}_{vx} = \hat{\mathcal{L}}_{vx} + \hat{F}_v^T\hat{V}_{xx}. \tag{4.23h}$$

$$\hat{Q}_{uv} = \hat{\mathcal{L}}_{uv}. \tag{4.23i}$$

$$\hat{Q}_{vu} = \hat{Q}_{uv}^T. \tag{4.23j}$$

The parameter $0 < \gamma \leq 1$ is added to the gains to make sure the convergence is obtained as follows

$$\delta\hat{u}^* = \gamma\hat{I}_u + \hat{K}_u\delta\hat{x}, \tag{4.24a}$$

$$\delta\hat{v}^* = \gamma\hat{I}_v + \hat{K}_v\delta\hat{x}. \tag{4.24b}$$

To derive the optimal update laws 4.18, consider the left hand side of the equation (4.9) as follow

$$\frac{\partial V(x,t)}{\partial t} = \frac{\partial V(\bar{x}+\delta\hat{x},t)}{\partial t} \approx \frac{\partial \bar{V}}{\partial t} + \frac{\partial \bar{V}_x^T}{\partial t}\delta\hat{x} + \frac{1}{2}\delta\hat{x}^T\frac{\partial \bar{V}_{xx}^T}{\partial t}\delta\hat{x}. \tag{4.25}$$

Also, we have

$$\frac{d\bar{V}}{dt} = \frac{\partial V}{dt} + \bar{V}_x^T \frac{dx}{dt} = \frac{\partial \bar{V}}{\partial t} + \bar{V}_x^T \bar{F}. \tag{4.26}$$

Hence,

$$\frac{\partial \bar{V}}{\partial t} = \frac{d\bar{V}}{dt} - \bar{V}_x^T \bar{F}, \tag{4.27}$$

and similarly

$$\frac{\partial \bar{V}_x}{\partial t} = \frac{d\bar{V}_x}{dt} - \bar{V}_{xx}^T \bar{F}, \tag{4.28}$$

and the expansion of the partial time derivative of the Hessian of the value function yields

$$\frac{\partial \bar{V}_{xx}}{\partial t} = \frac{d\bar{V}_{xx}}{dt} - \sum_{i=1}^{n} \bar{V}_{xxx}^{(i)} \bar{F}^{(i)}, \tag{4.29}$$

where $\bar{V}_{xxx}^{(i)}$ and $\bar{F}^{(i)}$ are Hessian matrix of the i-th element $\bar{V}_x$ and $\bar{F}$, respectively. Hence, by referring to equation (4.26-4.28) the left side of the equation (4.9) becomes

$$\begin{aligned} -\frac{\partial V(x,t)}{\partial t} \approx & -\frac{d\bar{V}}{dt} - \frac{d\bar{V}_x^T}{dt}\delta\hat{x} - \frac{1}{2}\delta\hat{x}^T \frac{d\bar{V}_{xx}^T}{dt}\delta\hat{x} + \bar{V}_x^T \bar{F} + \delta\hat{x}^T \bar{V}_{xx} \bar{F} \\ & + \frac{1}{2}\delta\hat{x}^T \left(\sum_{i=1}^{n} \bar{V}_{xxx}^{(i)} \bar{F}^{(i)} \right) \delta\hat{x}. \end{aligned} \tag{4.30}$$

The partial expansion of the right-hand side of equation (4.9) results in

$$\mathcal{L} \approx \hat{\mathcal{L}} + \hat{\mathcal{L}}_x^T \delta\hat{x} + \hat{\mathcal{L}}_u^T \delta\hat{u} + \hat{\mathcal{L}}_v^T \delta\hat{v} + \frac{1}{2} \begin{bmatrix} \delta\hat{x} \\ \delta\hat{u} \\ \delta\hat{v} \end{bmatrix}^T \begin{bmatrix} \hat{\mathcal{L}}_{xx} & \hat{\mathcal{L}}_{xu} & \hat{\mathcal{L}}_{xv} \\ \hat{\mathcal{L}}_{ux} & \hat{\mathcal{L}}_{uu} & \hat{\mathcal{L}}_{uv} \\ \hat{\mathcal{L}}_{vx} & \hat{\mathcal{L}}_{vu} & \hat{\mathcal{L}}_{vv} \end{bmatrix} \begin{bmatrix} \delta\hat{x} \\ \delta\hat{u} \\ \delta\hat{v} \end{bmatrix}, \tag{4.31}$$

The expansion of V_x around $\bar{x}$ yields

$$V_x \approx \bar{V}_x + \bar{V}_{xx}\delta\hat{x} + \frac{1}{2}\bar{W}, \tag{4.32}$$

where $\bar{W} = [\bar{W}^i]$ is such that

$$\bar{W}^i = \delta\hat{x}^T \bar{V}_{xxx}^{(i)} \delta\hat{x}, \qquad i = i, ..., n. \tag{4.33}$$

The first order expansion of the approximate dynamics of the system results in

$$\hat{F}(\hat{x}, \hat{u}, \hat{v}) = \hat{F} + \hat{F}_x \delta\hat{x} + \hat{F}_u \delta\hat{u} + \hat{F}_v \delta\hat{v}. \tag{4.34}$$

Hence, the right side of the equation (4.9) becomes

$$\min_{\hat{u}} \max_{\hat{v}} \left\{ \mathcal{L}(\hat{x},\hat{u},\hat{v},t) + \hat{V}_x^T \hat{F}(\hat{x},\hat{u},\hat{v},t) \right\} \approx \left\{ \hat{\mathcal{L}} + \hat{\mathcal{L}}_x^T \delta\hat{x} + \hat{\mathcal{L}}_u^T \delta\hat{u} + \hat{\mathcal{L}}_v^T \delta\hat{v} \right.$$
$$+ \frac{1}{2} \begin{bmatrix} \delta\hat{x} \\ \delta\hat{u} \\ \delta\hat{v} \end{bmatrix}^T \begin{bmatrix} \hat{\mathcal{L}}_{xx} & \hat{\mathcal{L}}_{xu} & \hat{\mathcal{L}}_{xv} \\ \hat{\mathcal{L}}_{ux} & \hat{\mathcal{L}}_{uu} & \hat{\mathcal{L}}_{uv} \\ \hat{\mathcal{L}}_{vx} & \hat{\mathcal{L}}_{vu} & \hat{\mathcal{L}}_{vv} \end{bmatrix} \begin{bmatrix} \delta\hat{x} \\ \delta\hat{u} \\ \delta\hat{v} \end{bmatrix} + \hat{V}_x^T \hat{F} + \hat{V}_x^T \hat{F}_x \delta\hat{x} + \hat{V}_x^T \hat{F}_u \delta\hat{u}$$
$$+ \hat{V}_x^T \hat{F}_v \delta\hat{v} + \delta\hat{x}^T \hat{V}_{xx} \hat{F}_x \delta\hat{x} + \delta\hat{x}^T \hat{V}_{xx} \hat{F}_u \delta\hat{u} + \delta\hat{x}^T \hat{V}_{xx} \hat{F}_v \delta\hat{v}$$
$$\left. + \delta\hat{x}^T \hat{V}_{xx} \hat{F} + \frac{1}{2} \hat{W}^T \hat{F} \right\}, \tag{4.35}$$

and notice that

$$\frac{1}{2} \hat{W}^T \hat{F} = \frac{1}{2} \left(\sum_{i=1}^{n} \delta\hat{x}^T \hat{V}_{xxx}^{(i)} \delta\hat{x} \hat{F}^{(i)} \right) = \frac{1}{2} \delta\hat{x}^T \left(\sum_{i=1}^{n} \hat{V}_{xxx}^{(i)} \hat{F}^{(i)} \right) \delta\hat{x}. \tag{4.36}$$

By equating the left hand side and right hand side of the expansion of equation (4.9) and cancelling extra terms, we obtain

$$- \frac{d\hat{V}}{dt} - \delta\hat{x}^T \frac{d\hat{V}_x}{dt} - \frac{1}{2} \delta\hat{x}^T \frac{d\hat{V}_{xx}}{dt} \delta\hat{x} = \min_{\delta\hat{u}} \max_{\delta\hat{v}} \left\{ \hat{\mathcal{L}} + \hat{\mathcal{L}}_x^T \delta\hat{x} + \hat{\mathcal{L}}_u^T \delta\hat{u} + \hat{\mathcal{L}}_v^T \delta\hat{v} \right.$$
$$+ \frac{1}{2} \begin{bmatrix} \delta\hat{x} \\ \delta\hat{u} \\ \delta\hat{v} \end{bmatrix}^T \begin{bmatrix} \hat{\mathcal{L}}_{xx} & \hat{\mathcal{L}}_{xu} & \hat{\mathcal{L}}_{xv} \\ \hat{\mathcal{L}}_{ux} & \hat{\mathcal{L}}_{uu} & \hat{\mathcal{L}}_{uv} \\ \hat{\mathcal{L}}_{vx} & \hat{\mathcal{L}}_{vu} & \hat{\mathcal{L}}_{vv} \end{bmatrix} \begin{bmatrix} \delta\hat{x} \\ \delta\hat{u} \\ \delta\hat{v} \end{bmatrix} + \hat{V}_x^T \hat{F}_x \delta\hat{x} + \hat{V}_x^T \hat{F}_u \delta\hat{u}$$
$$\left. + \hat{V}_x^T \hat{F}_v \delta\hat{v} + \delta\hat{x}^T \hat{V}_{xx} \hat{F}_x \delta\hat{x} + \delta\hat{x}^T \hat{V}_{xx} \hat{F}_u \delta\hat{u} + \delta\hat{x}^T \hat{V}_{xx} \hat{F}_v \delta\hat{v} \right\}$$
$$= \min_{\delta\hat{u}} \max_{\delta\hat{v}} \left\{ \hat{\mathcal{L}} + \delta\hat{x}^T \hat{Q}_x + \delta\hat{u}^T \hat{Q}_u + \delta\hat{v}^T \hat{Q}_v + \frac{1}{2} \delta\hat{x}^T \hat{Q}_{xx} \delta\hat{x} + \frac{1}{2} \delta\hat{u}^T \hat{Q}_{uu} \delta\hat{u} \right.$$
$$\left. + \frac{1}{2} \delta\hat{v}^T \hat{Q}_{vv} \delta\hat{v} + \delta\hat{u}^T \hat{Q}_{ux} \delta\hat{x} + \delta\hat{v}^T \hat{Q}_{vx} \delta\hat{x} + \delta\hat{u}^T \hat{Q}_{uv} \delta\hat{v} \right\}, \tag{4.37}$$

where the Q functions are given in equation (4.23).

We take partial derivative of (4.37) with respect to $\delta\hat{u}$ and $\delta\hat{v}$ and set them to 0 to obtain the the optimal control update $\delta\hat{u}^*$ and $\delta\hat{v}^*$ as

$$\delta\hat{u}^* = -\hat{Q}_{uu}^{-1}(\hat{Q}_{ux}\delta\hat{x} + \hat{Q}_{uv}\delta\hat{v}^* + \hat{Q}_u), \tag{4.38a}$$

$$\delta\hat{v}^* = -\hat{Q}_{vv}^{-1}(\hat{Q}_{vx}\delta\hat{x} + \hat{Q}_{vu}\delta\hat{u}^* + \hat{Q}_v), \tag{4.38b}$$

which can be summarized by eliminating $\delta\hat{u}^*$ and $\delta\hat{v}^*$ on the right hand side of (4.38) and rewritten as

$$\delta\hat{u}^* = \hat{I}_u + \hat{K}_u\delta\hat{x}, \tag{4.39a}$$

$$\delta\hat{v}^* = \hat{I}_v + \hat{K}_v\delta\hat{x}, \tag{4.39b}$$

where $\hat{I}_u$, $\hat{I}_v$, $\hat{K}_u$, $\hat{K}_v$ follow the form in equations (4.19) through (4.22), which completes the proof.

4.4.2 Backward Propagation

The next step is to find the update law of the value function and its first- and second-order partial derivatives. We present the result as follows

Proposition 2. *The optimal update of the value function and its first order, and second order state derivatives are governed by the backward ordinary differential equations as*

follows

$$-\frac{d\hat{V}}{dt} = \hat{\mathcal{L}} + \hat{I}_u^T\hat{Q}_u + \hat{I}_v^T\hat{Q}_v + \frac{1}{2}\hat{I}_u\hat{\tilde{Q}}_{uu}\hat{I}_u + \hat{I}_u^T\hat{Q}_{uv}\hat{I}_v + \frac{1}{2}\hat{I}_v^T\hat{Q}_{vv}\hat{I}_v, \tag{4.40a}$$

$$-\frac{d\hat{V}_x}{dt} = \hat{Q}_x + \hat{K}_u^T\hat{Q}_u + \hat{K}_v^T\hat{Q}_v + \hat{Q}_{ux}^T\hat{I}_u + \hat{Q}_{vx}^T\hat{I}_v + \hat{K}_u^T\hat{Q}_{uu}\hat{I}_u + \hat{K}_u^T\hat{Q}_{uv}\hat{I}_v$$
$$+ \hat{K}_v^T\hat{Q}_{vu}\hat{I}_u + \hat{K}_v^T\hat{Q}_{vv}\hat{I}_v, \tag{4.40b}$$

$$-\frac{d\hat{V}_{xx}}{dt} = \hat{K}_u^T\hat{Q}_{ux} + \hat{Q}_{ux}^T\hat{K}_u + \hat{K}_v^T\hat{Q}_{vx} + \hat{Q}_{vx}^T\hat{K}_v + \hat{K}_v^T\hat{Q}_{vu}\hat{K}_u + \hat{K}_u^T\hat{Q}_{uv}\hat{K}_v$$
$$+ \hat{K}_u^T\hat{Q}_{uu}\hat{K}_u + \hat{K}_v^T\hat{Q}_{vv}\hat{K}_v + \hat{Q}_{xx}, \tag{4.40c}$$

with the given terminal condition

$$\hat{V}(t_f) = \phi(\hat{x}(t_f), t_f). \tag{4.41a}$$

$$\hat{V}_x(t_f) = \phi_x(\hat{x}(t_f), t_f). \tag{4.41b}$$

$$\hat{V}_{xx}(t_f) = \phi_{xx}(\hat{x}(t_f), t_f). \tag{4.41c}$$

Proof. The backward propagation equations (4.40) with respect to the value function and its first- and second-order partial derivatives can be obtained by plugging the form of $\delta\hat{u}^*$ and $\delta\hat{v}^*$ expressed in (4.18) to equation (4.37), and then equating the coefficients of zeroth-, first-, and second-order expressions of $\delta\hat{x}$ in the left-hand side and right-hand side of the equation. A detailed proof with similar steps can be found in [8]. □

The data-driven min-max DDP algorithm using the GP is presented in Algorithm 4.

Remark 8. *The proposed data-driven Min-Max Differential Dynamic Programming (DDP) algorithm is applicable to nonlinear dynamical systems where the underlying dynamics are unknown and the control problem involves adversarial interactions. Due to existence of the second player which is a maximizer, the method is particularly useful*

Algorithm 4 Data-driven min-max DDP algorithm

Require: Initial values for x_0, minimizing control u, maximizing control v, terminal time t_f, multiplier γ, large integer N, and small value ϵ.

Ensure: Optimal minimizing control $\hat{u}^*$, optimal maximizing control $\hat{v}^*$, gains $\hat{I}_u$, $\hat{I}_v$, $\hat{K}_u$, and $\hat{K}_v$.

procedure UPDATE CONTROL POLICIES:

Collect the input-output data by applying random input data to equation (4.1) forward with x_0.

Construct the Gaussian Process model of the dynamics from the collected input-output data.

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

while $|\delta\hat{u}| + |\delta\hat{v}| > \epsilon$ **do**

Compute values of $\hat{V}$, $\hat{V}_x$, and $\hat{V}_{xx}$ at t_f from equation (4.41);

Integrate backward the equation (4.40);

Compute $\hat{I}_u$, $\hat{I}_v$, $\hat{K}_u$, and $\hat{K}_v$ through equations (4.19) to (4.22);

Integrate equation (4.16) by replacing $\delta\hat{u}$ and $\delta\hat{v}$ with $\delta\hat{u} = \hat{I}_u + \hat{K}_u\delta\hat{x}$ and $\delta\hat{v} = \hat{I}_v + \hat{K}_v\delta\hat{x}$ to get $\delta\hat{x}$;

Obtain $\delta\hat{u} = \hat{I}_u + \hat{K}_u\delta\hat{x}$ and $\delta\hat{v} = \hat{I}_v + \hat{K}_v\delta\hat{x}$;

Update $\hat{u} = \hat{u} + \gamma\delta\hat{u}$ and $\hat{v} = \hat{v} + \gamma\delta\hat{v}$;

Set $u^* = \hat{u}$ and $v^* = \hat{v}$;

end while

end for

return u^*, v^*, $\hat{I}_u$, $\hat{I}_v$, $\hat{K}_u$, and $\hat{K}_v$;

end procedure

in continuous-time process where there is a strong uncertainty or disturbance. Since the algorithm employs GP regression to approximate the unknown system dynamics, it is well-suited for systems where direct modeling is impractical due to complexity, unmodeled dynamics, or external disturbances. This approach is most effective in dynamical systems characterized by smooth and continuous dynamics with moderate noise levels, where GP-based approximations can provide reliable uncertainty estimates. It is particularly relevant for robotic systems, aerospace applications, biomechanical systems, and defense-related multi-agent interactions, where robust and adaptive control strategies are necessary. The method can be applied to highly nonlinear systems, including underactuated and constrained control problems, where traditional model-based DDP methods may struggle due to inaccurate system identification. Additionally, it extends to multi-agent pursuit-evasion games, optimal guidance and navigation, flight control systems, and robotic manipulation tasks, where two-player game-theoretic strategies are needed. Moreover, the algorithm is best suited for real-time and iterative learning scenarios, where control policies can be refined based on newly observed data. The proposed approach is particularly beneficial when handling structured uncertainties in the system dynamics, where Gaussian Process models can capture the underlying trends and provide probabilistic estimates of model confidence.

4.5 Numerical Examples

In this section, three practical examples are provided to show the effectiveness of the proposed algorithm. In these examples, as mentioned in Algorithm 6, we collect data by running the system with random input controls u and v, starting from an initial condition x_0, over a time horizon of 101 time steps. During this process, we record the input data $[x, u, v]$ and the output data $\frac{dx}{dt}$, which represents the trajectory of the

system. To enhance the robustness of our learned dynamics, we repeat this process 10 times in a for-loop, thereby collecting 10 trajectories and corresponding 10 sets of input-output data of size (10×101). After collecting this dataset, we use Gaussian Process (GP) regression to approximate the unknown dynamics of the system. The GP model learns a mapping from $[x, u, v]$ to $\frac{dx}{dt}$, capturing both the system dynamics and the associated uncertainty. Additionally, we introduce a small amount of noise in the data and use the mean function of the GP as the estimated system dynamics. This allows the method to remain robust against measurement variability while providing a smooth approximation of the underlying system behavior. Finally, the Gaussian models of the dynamics are utilized to generate optimal control laws.

4.5.1 Inverted Pendulum

In the first example, the algorithm is applied to the inverted pendulum with conflicting control laws. The dynamics of the system is given by

$$I\ddot{\theta} + b\dot{\theta} - mglsin\theta = u + v, \tag{4.42}$$

where θ is the angular position of the inverted pendulum, $\dot{\theta}$ is the angular velocity of the inverted pendulum, $m = 1kg$, $l = 0.5m$, $b = 0.1(N.s)/m$, $I = ml^2$. $g = 9.81(kg.m)/s^2$. For this example, the initial condition is selected as $x_0 = [\pi, 0]$, and $t_f = 1$. The goal is to bring the states of the system from $x = [\theta, \dot{\theta}] = [\pi, 0]$ to $[\theta, \dot{\theta}] = [0, 0]$. The cost function is given by

$$J = \int_{t_0}^{t_f} (x^T Q x + u^T R_u u - v^T R_v v) dt, \tag{4.43}$$

For this example, the initial control policies u and v are set as random variables drawn from a standard normal distribution, and $\gamma = 0.6$. The weights are chosen as $Q = [1, 0; 0, 0.1]$, $R_u = 1$, and $R_v = 0.1$.

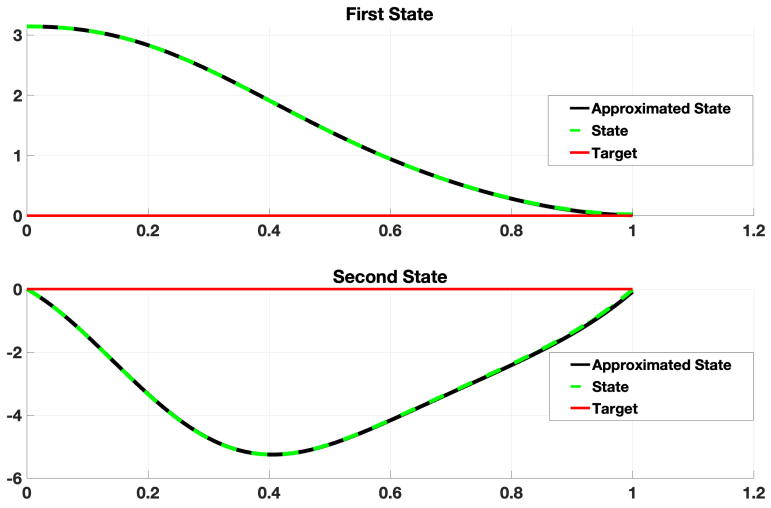


Figure 4.1: Comparison of States and Approximated States pendulum.

In Fig. 4.1, we present the state trajectories and their approximations for the inverted pendulum subject to the optimal controls obtained from the proposed algorithm in a single plot and demonstrate the effectiveness of GP in approximating the dynamics and the proposed method in stabilizing both the nonlinear and approximated dynamics. It can be seen that the control u manage to drive the states to the desired location despite the conflicting control v, which shows the effectiveness of the method.

Fig. 4.2 presents the optimal control policies of u and v. The chattering on the controls in this example are due to the initial choice of control policies, which are drawn from a standard normal distribution. In Fig. 4.3, the cost per iteration is illustrated. As it can be seen, the cost converges towards 0 in about 50 iterations.

In Fig. 4.4, we present the state and control policy results of implementing the GT-DDP algorithm with exact dynamics on the inverted pendulum example to demonstrate

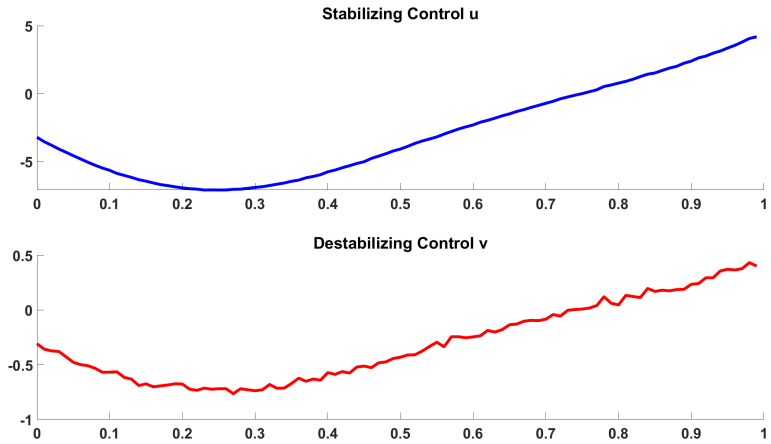


Figure 4.2: Control Policies of inverted pendulum.

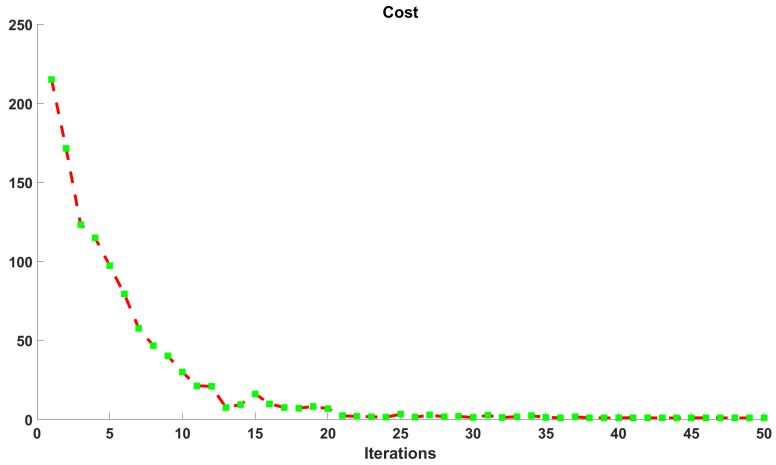


Figure 4.3: Cost per Iteration of inverted pendulum.

that our results closely match the case where the exact dynamics of the system are available. This shows that the proposed algorithm in This chapter outperforms the GT-DDP algorithm, as we assume no access to the system dynamics, making the scenario more realistic.

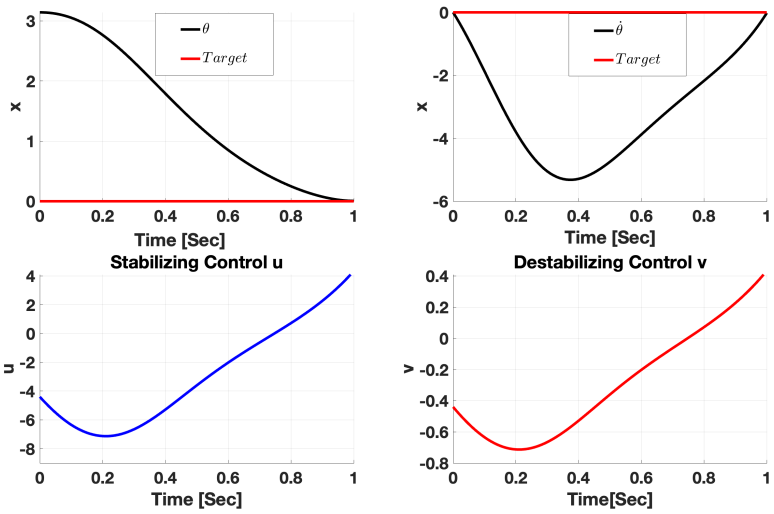


Figure 4.4: States and control policies of inverted pendulum under GT-DDP case

4.5.2 2-DOF Helicopter

In the second example, the proposed algorithm is applied to the state-space model of the 2-degree-of-freedom (2-DOF) helicopter [209]. The state-space matrices are

$$\dot{x}(t) = Ax(t) + B(u(t) + v(t)). \tag{4.44}$$

where

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{B_p}{J_{eqp}+mI_{cm}^2} & 0 \\ 0 & 0 & 0 & -\frac{B_y}{J_{eqy}+mI_{cm}^2} \end{bmatrix}. \tag{4.45}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{K_{pp}(u_p+v_p)}{J_{eqp}+mI_{cm}^2} & \frac{K_{py}(u_y+v_y)}{J_{eqp}+mI_{cm}^2} \\ \frac{K_{yp}(u_p+v_p)}{J_{eqy}+mI_{cm}^2} & \frac{K_{yy}(u_y+v_y)}{J_{eqy}+mI_{cm}^2} \end{bmatrix}, \tag{4.46}$$

The state and controls are given by $x = [x, y, \theta, \psi]^T$, $u = [u_p, u_y]^T$, and $v = [v_p, v_y]^T$, where x and y are positions of the helicopter, θ is the pitch angle and ψ is the yaw angle. u and v are control signals applied to pitch and yaw motors, respectively. The goal is to bring the helicopter from initial conditions $x_0 = [1,\ 1,\ 0,\ 0]^T$ to $[0,\ 0,\ 0,\ 0]^T$ and the corresponding cost function is defined as

$$J = \frac{1}{2}(x - x_f)^T Q_f (x - x_f) + \int_{t_0}^{t_f} ((x - x_f)^T Q (x - x_f) + u^T R_u u - v^T R_v v) dt, \tag{4.47}$$

For this example, the initial control policies are set as $u = v = 0$, $t_f = 5$, and $\gamma = 0.01$. The weights are chosen as $Q_f = [5, 0, 0, 0; 0, 5, 0, 0; 0, 0, 5, 0; 0, 0, 0, 5]$, $Q = [10, 0, 0, 0; 0, 10, 0, 0; 0, 0, 10, 0; 0, 0, 0, 10]$, $R_u = [1, 0; 0, 1]$, and $R_v = [0.1, 0; 0, 0.1]$. The values of the parameters in (4.46) are shown in Table 4.1.

In Fig. 4.5, we depict the trajectories of the true and approximated states of the 2-DOF helicopter within the same plot in 5 seconds and demonstrate how GP accurately captures the system dynamics and how the proposed approach ensures stability for both the original and approximated models. It is noticeable that both the dynamics

Parameter	name	value
K_{pp}	Pitch torque	0.204 $N.m/V$
K_{yy}	Yaw torque	0.072 $N.m/V$
K_{py}	Yaw on pitch torque	0.0068 $N.m/V$
K_{yp}	Pitch on yaw torque	0.0219 $N.m/V$
J_{eq_p}	Total pitch moment of inertia	0.0384 $kg.m^2$
J_{eq_y}	Total yaw moment of inertia	0.0432 $kg.m^2$
B_p	Pitch viscous damping	0.800 NN
B_y	Yaw viscous damping	0.318 NN
m	Total moving mass	1.3872 kg
I_{cm}	Center of mass length from pitch axis	0.186 m

Table 4.1: Parameter values of linearized helicopter model.

trajectories and GP model dynamics trajectories are similar and converge to their goal. Fig. 4.6 represents the optimal control trajectories and Fig. 4.7 shows the cost per iteration that illustrates the convergence of the cost.

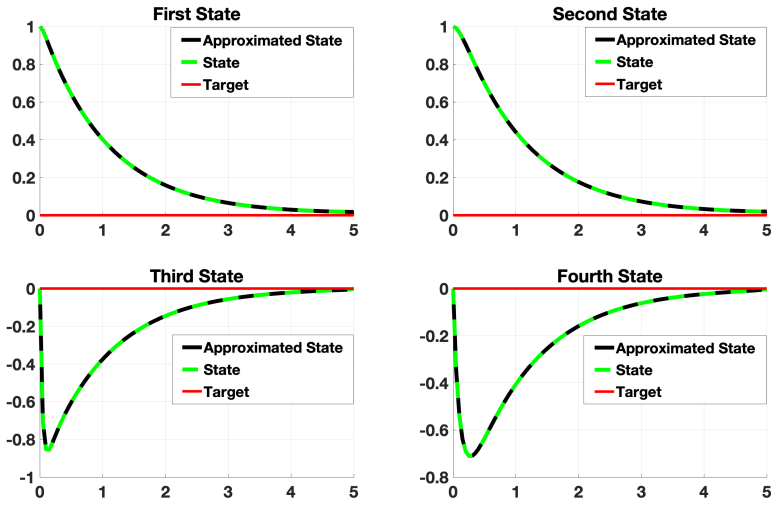


Figure 4.5: Comparison of States and Approximated States of 2-DOF helicopter.

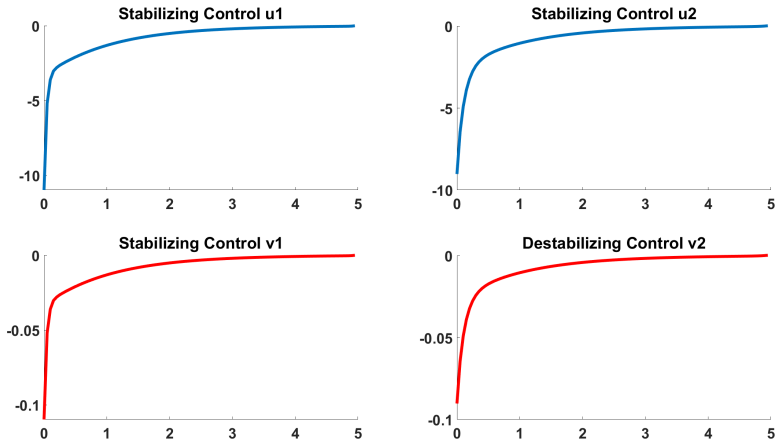


Figure 4.6: Control policies of 2-DOF helicopter.

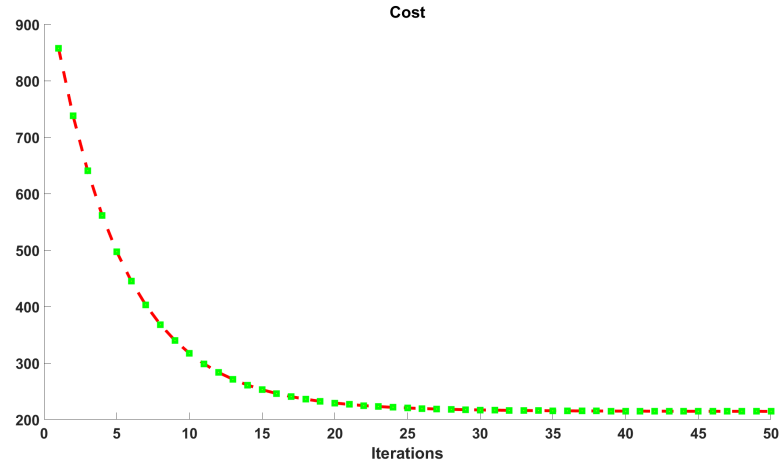


Figure 4.7: Cost per Iteration of 2-DOF helicopter.

4.5.3 Vertical Flight Regulation

In the third example, the algorithm is applied to the nonlinear dynamics model of the vertical flight regulation described in [210]. The dynamics of the system is defined as

$$\begin{aligned}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= a_0 + a_1 x_2 + a_2 x_2^2 + (a_3 + a_4 x_4 - \sqrt{a_5 + a_6 x_4}) x_3^2, \\
\dot{x}_3 &= a_7 + a_8 x_3 + [a_9 sin(x_4) + a_{10}] x_3^2 + u_1 + v_1, \\
\dot{x}_4 &= x_5, \\
\dot{x}_5 &= a_{11} + a_{12} x_4 + a_{13} x_3^2 sin(x_4) + a_{14} x_5 + u_2 + v_2,
\end{aligned} \tag{4.48}$$

where the values are given to be $a_0 = -17.67 m/s^2, a_1 = a_2 = -0.1 s^{-l}, a_3 = 5.31 \times 10^{-4}, a_4 = 1.5364 \times 10^{-2}, a_5 = 2.82 \times 10^{-7}, a_6 = 1.632 \times 10^{-5}, a_7 = -13.92 s^{-2}, a_8 = -0.7 s^{-1}, a_9 = a_{10} = -0.0028, a_{11} = 434.88 s^{-2}, a_{12} = -800 s^{-2}, a_{13} = -0.1$ and $a_{14} = -65 s^{-1}$. The state $x = [h; \dot{h}; \omega; \theta; \dot{\theta}]$ where h denotes the height above ground (metres), ω denotes the rotational speed of the rotor blades (rad/s) and θ denotes the collective pitch angle of rotor blades (rad). The goal is to bring the helicopter from the height $h = 4$ with the rotational speed of the rotor blades $\omega = 200$ to height $h = 6$ and rotational speed of the rotor blades $\omega = 250$.

The corresponding cost function is defined as

$$J = \frac{1}{2}(x - x_f)^T Q_f (x - x_f) + \int_{t_0}^{t_f} (u^T R_u u - v^T R_v v) dt, \tag{4.49}$$

For this example, the initial control policies are set as $u = v = 0$, $t_f = 1$, and $\gamma = 0.01$. The desired goal state $x_f = [6; 0; 300; 0; 0]$. The weights are chosen as $Q_f = [1000, 0, 0, 0, 0; 0, 1000, 0, 0, 0; 0, 0, 50, 0, 0; 0, 0, 0, 50, 0; 0, 0, 0, 0, 50]$, $R_u = [1, 0; 0, 1]$, and

$R_v = [0.01, 0; 0, 0.01]$.

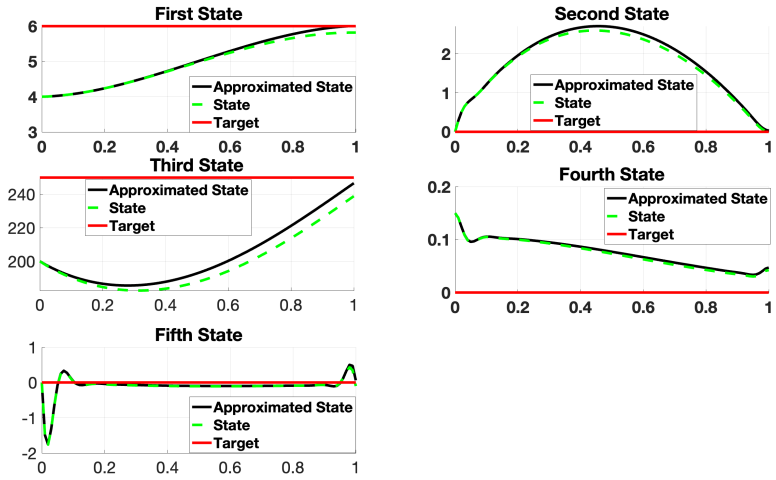


Figure 4.8: Comparison of States and Approximated States of vertical flight regulation.

Fig. 4.8 illustrates the trajectories of both the actual and approximated states of the vertical flight regulation in a single plot in $t = 1$ and highlights the ability of GP to approximate the system dynamics and the effectiveness of the proposed method in stabilizing both the nonlinear and estimated dynamics.

The trajectories of the optimal controls are shown in Fig. 4.9, and the cost per iteration is illustrated in Fig. 4.10. It can be seen that the cost of the system decreases and converges during the iterations.

Remark 9. *While the examples used in the paper involve systems with known dynamics, the key distinction is that our approach does not rely on explicit model knowledge. Instead, it learns the dynamics purely from input-output data using Gaussian Process (GP) regression, which allows handling completely unknown systems effectively. Although classical approaches like Game-Theoretic Differential Dynamic Programming (GT-DDP) require prior knowledge of the system dynamics, our method enables trajectory optimization without requiring explicit analytical models, making it applicable*

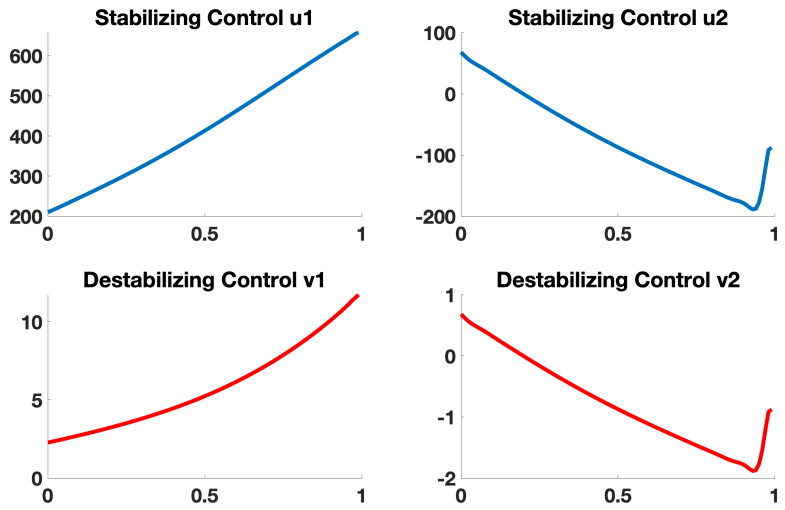


Figure 4.9: Control Policies of vertical flight regulation.

to cases where system identification is impractical or costly. The proposed algorithm is particularly beneficial for systems with unmodeled disturbances, parametric uncertainties, or time-varying dynamics, where conventional GT-DDP would struggle due to modeling errors. Furthermore, when approximating the system via input-output data, we introduced a small noise in the data and used the mean function of the GP model as the learned dynamics of the system. This accounts for uncertainties in real-world measurements while maintaining a reliable approximation of the system.

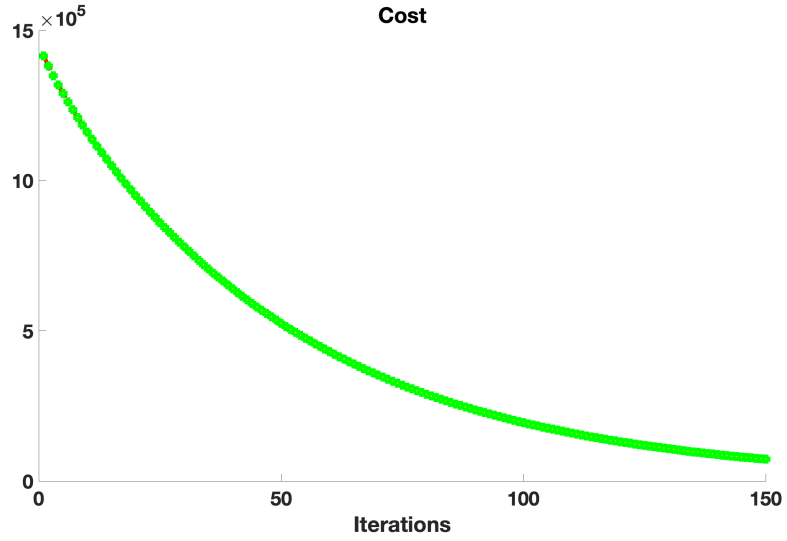


Figure 4.10: Cost per Iteration of vertical flight regulation.

Chapter 5

Data-Driven Stochastic Game Theoretic Differential Dynamic Programming

5.1 Introduction

Stochastic systems, characterized by inherent randomness and uncertainty, are prevalent in various industrial applications—such as financial modeling, robotics, chemical processes, and autonomous vehicles—and offer more realistic representations of real-world phenomena compared to deterministic models [211, 212, 213]. These systems are governed by dynamics that are not entirely deterministic and often influenced by random external noise which makes them more realistic representations of real-world phenomena compared to purely deterministic models. Modeling and controlling stochastic systems is essential for robust decision-making under uncertainty. However, this comes with challenges such as accurately estimating dynamics and managing the computational demands of stochastic control problems. While stochastic models offer deeper insights, their inherent uncertainty requires advanced techniques and this requires the both powerful and complex tools in control engineering. Overcoming these challenges is a key to enhancing the reliability and efficiency of systems in uncertain

environments [214].

Due to existence of the uncertainties in dynamics of the nonlinear systems specifically autonomous systems, Differential Dynamic Programming (DDP) has been utilized widely due to its applicability and strength. For instance, [215] presents a flexible final-time stochastic differential dynamic programming (SDDP) approach for trajectory optimization under stochastic disturbances. [216] Proposes a learning-based and ensemble-based model predictive control (MPC) framework for autonomous reactor control. Mechanism models and neural networks are used as predictors, with ensemble learning methods improving prediction accuracy. In [217], the problem of 2D vehicle path following under adversarial disturbances by computing a maximal invariant set through dynamic programming is proposed. Paper [218] presents a Variable Splitting-based Differential Dynamic Programming (VS-DDP) method for autonomous vehicle trajectory planning. By employing the variable splitting technique, VS-DDP effectively handles constraints, significantly enhancing the capability of differential dynamic programming. Paper [219] focuses on a path-following control strategy for autonomous vehicles under adversarial disturbances by employing the concept of weakly invariant sets. Over the years, some methods have been proposed to address the control and optimization of stochastic systems [220, 221, 8]. These methods include Stochastic Dynamic Programming (SDP) [222], Model Predictive Control (MPC) [223], and Stochastic Optimal Control (SOC) [224], which have been extensively studied and applied in various domains. While various methods have been developed for stochastic systems, many of these approaches face notable limitations. These methods often suffer from high computational complexity, specifically when dealing with high-dimensional systems or long time horizons. Additionally, they may lack robustness when dealing with significant uncertainties or disturbances, as they typically optimize for expected performance. These limitations can result in unstable performance in real-world ap-

plications, where uncertainties are inherent and unpredictable. One of the reliable methods in control theory for dealing with uncertainties in stochastic systems is the Game Theoretic Differential Dynamic Programming (GT-DDP) approach [8], which utilizes the differential dynamic programming (DDP) technique to optimize control policies where one controller attempts to minimize the cost function, while the other one attempts to maximize it. GT-DDP addresses some of the aforementioned challenges by incorporating a robust, game-theoretic approach into the control design. This approach has been utilized in some works such as the iterative algorithm for differential game problems based on DDP [225], control constrained GT-DDP [226], and the Stochastic GT-DDP (SGT-DDP) [13].

One restriction of the traditional DDP-based methods is the requirement of an exact mathematical model of the system to compute the optimal control policies [26]. This reliance on a precise model limits its applicability in many real-world scenarios, where the true dynamics of the system may be unknown. In such cases, obtaining an exact model can be challenging due to factors such as complex model. As a result, the need for precise system models poses a significant constraint. To overcome this limitation, there is a growing need for reliable data-driven techniques that can approximate the dynamics directly from data [227]. Machine learning methods have gained a lot of attention in recent years for modeling unknown systems by leveraging historical input-output data [228]. Among the various machine learning techniques, Gaussian Process Regression (GPR) has emerged as a powerful tool for approximating the drift components of the stochastic systems [229]. GPR provide flexible frameworks that not only estimate the drift dynamics but also quantify the uncertainty associated with these estimates. In contrast, there has been much less work on the approximation of the diffusion component of the stochastic systems [230]. The binning method [203] has been utilized in [231] to extract the diffusion coefficient of the system. By integrating the data-driven

approaches with the SGT-DDP framework, we can develop a reliable control strategy that effectively handles uncertainties and unknowns in dynamic environments.

Hence, our approach extends the standard SGT-DDP framework by integrating data-driven model identification through Gaussian Process Regression (GPR) and binning methods. This allows SGT-DDP to be used with unknown or partially known dynamics, as we approximate both drift and diffusion dynamics from data before implementing DDP optimization. In addition, we consider the effects of control policies in the diffusion dynamics which makes the algorithm practical to real-world systems. Besides, this is the first study to combine the binning method, Savitzky-Golay filter [232], and GPR to model diffusion dynamics in stochastic systems. This combination systematically isolates noise trends, smoothing them before the final GPR estimation. Furthermore, although the GPR itself treats control inputs as standard data points, the conflicting nature of the controls becomes relevant in the SGT-DDP framework, where two players have opposing objectives

The contribution of the paper is divided into several aspects:

1) SGT-DDP problem with unknown nonlinear dynamics is solved to obtain the control policies.

2) GPR is utilized to approximate the drift dynamics of the stochastic nonlinear system with conflicting controls.

3) To the best knowledge of the authors, the binning method, Savitzky-Golay filter, and GPR methods are combined for the first time to approximate the diffusion dynamics of the stochastic nonlinear system.

4) The optimal update laws of the control policies of the two players and the corresponding set of backward differential equations for the value function are provided.

The rest of the paper is organized as follows. In Section 5.2, the problem formulation which includes the description of the system and its approximations is provided.

Section 5.3 discusses the design of optimal control policy. In Section 5.4, backward propagation of the value function is proposed. Finally, in Section 5.5, three numerical examples with results are demonstrated.

For the convenience of the readers, some useful notations and prerequisites are mentioned here. $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{R}^n$, and $\mathbb{R}^{n\times m}$ present the set of real numbers, positive real numbers, $n \times 1$ real column vectors, and $n \times m$ real matrices, respectively. $\mathcal{C}^1(\mathbb{R})$ and $\mathcal{C}^2(\mathbb{R})$ illustrate the sets of functions whose first and second-order derivative exist and is continuous with domain $\mathbb{R}$. $tr(A)$ and A^T represent the trace and transpose of a matrix A, respectively.

5.2 Problem Formulation

In this section, the description of the system and the approximations of the drift and diffusion dynamics through the GPR and binning methods are introduced.

5.2.1 System description

Consider the dynamics of the stochastic differential game

$$dx = F(x(t), u(t), v(t))dt + G(x(t), u(t), v(t))dw, \quad x(t_0) = x_0, \quad t \in [t_0, t_f], \tag{5.1}$$

subject to the cost function

$$J(x(t), u(t), v(t)) = \mathbf{E}\left[\phi(x(t_f)) + \int_{t_0}^{t_f} \mathcal{L}(x(t), u(t), v(t))dt\right], \tag{5.2}$$

where $\mathbf{E}[\cdot]$ represents the expectation, $\mathcal{L}$ is the running cost, ϕ is the terminal cost, $x(t) \in \Re^n$ is the state of the system, $u(t) \in \Re^m$ and $v(t) \in \Re^q$ are the control policies. dw is an increment of a m-dimensional Wiener process (standard Brownian motion)

where $G(x(t), u(t), v(t)) \in \Re^{n\times m}$ is given to scale dw and match the dimension of $F(x(t), u(t), v(t)) \in \Re^n$. The term dw satisfies $dw \sim N(0, I_{m\times m}dt)$. In the equation (5.1), the drift dynamics F and the diffusion dynamics G are assumed to be unknown.

The value of the game is defined as

$$\begin{aligned} V(x_0, t_0) &= \min_u \max_v J(x(t), u(t), v(t)) \\ &= \max_v \min_u J(x(t), u(t), v(t)), \end{aligned} \tag{5.3}$$

where u attempts to minimize the cost function while v attempts to maximize it. Henceforth, we omit the time dependence t from the equation terms for simplicity of representation.

Remark 10. *The proposed framework requires explicit knowledge or accurate approximation of both drift and diffusion dynamics to solve the underlying stochastic differential game. Unlike data-driven methods, our approach relies on local expansions of the drift and diffusion around nominal trajectories, essential for game-theoretic formulation and precise backward computation of the cost-to-go function using the Bellman/Isaacs principle. In This chapter, we approximate the dynamics using GPR and a binning method. These approximations enable the application of min-max DDP in data-rich but uncertain environments, ensuring robustness, structured control updates, and theoretical guarantees of local optimality.*

5.2.2 Approximation of drift dynamics

In this subsection, the GPR is introduced to learn the dynamics $F(x, u, v)$ which is defined as the drift dynamic. The drift dynamics represent the deterministic behavior or trend of the states and are estimated as the mean function of the GPR model. By using input-output data of the system, the GPR captures the underlying patterns and

relationships, and provides a probabilistic estimation of the drift dynamics. This mean function serves as the best estimate of the drift dynamics and allows the SGT-DDP framework to incorporate this approximation for designing control policies. A random variable $x \in \Re^D$ follows a multivariate Gaussian/normal (MVN) [208] distribution if its probability density function (PDF) is defined as

$$\boldsymbol{N}\left(x|\mu,\Sigma\right) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right], \tag{5.4}$$

where $\mu = \mathbf{E}[x]$, and $\Sigma = \text{cov}[x]$ denote the mean and covariance matrix, respectively.

The GPR model characterizes a probability distribution over functions that fit a set of data points. This approach enables the computation of the function mean to serve as predictions and the variance to express the level of confidence in those predictions. The GPR model with regression function can be represented by a multivariate Gaussian as $P(f|X) = \boldsymbol{N}(f|\mu, K)$ where $X = [x_1, ..., x_n]^T$ denotes the data points, $f = [f(x_1), ..., f(x_n)]^T$ denotes the learned model, $\mu = [m(x_1), ..., m(x_n)]^T$ represents the mean, and K denotes the kernel function where $K_{ij} = k(x_i, x_j)$, and

$$k(x_i, x_j) = \exp\left(-\frac{w_{ij}(x_i - x_j)^2}{2}\right), \tag{5.5}$$

where w_{ij} represents the hyper-parameter of the GPR model. The function prediction at a new data point x_* can be then calculated as $f(x_*)$.

The joint distribution of $f(X)$ and $f(x_*)$ are given by

$$\begin{bmatrix} f(X) \\ f(x_*) \end{bmatrix} \sim \boldsymbol{N}\left(\begin{bmatrix} \mu(X) \\ m(x_*) \end{bmatrix}, \begin{bmatrix} K & K_* \\ K_*^T & K_{**} \end{bmatrix}\right), \tag{5.6}$$

where $K = K(X, X), K_* = K(X, x_*)$ and $K_{**} = K(x_*, x_*)$.

5.2.3 Approximation of diffusion dynamics

In this subsection, we propose to combine the binning method [233] with GPR to approximate the diffusion dynamics of the stochastic system. Diffusion dynamics are critical in capturing the random fluctuations around the mean trajectory of the system, which are influenced by noise and uncertainties. Accurately modeling of these dynamics is essential for designing the control strategies that takes the diffusion into consideration. The binning method provides a data-driven approach to estimate the diffusion by analyzing the variability of noise across different states. To approximate the diffusion dynamics of a stochastic system, we first define the number of bins N. Second, we need to obtain the noise component $\eta = G(x, u, v)dw$. To this end, we isolate the underlying stochastic behavior through the Savitzky-Golay filter, which smooths the noisy output data while preserving the important trends. The noise component $\eta = G(x, u, v)dw$ is extracted by subtracting the smoothed output from the original noisy output. Finally, by using the binning method, we obtain the estimation of $G(x, u, v)$ through the noise component $\eta = G(x, u, v)dw$. At the end, we use the estimation of $G(x, u, v)$ obtained by binning method as the output, and state and control policies (x, u, v) as the input to achieve the GPR model of the $G(x, u, v)$.

To this goal, we choose the z as the selected vector of state or controller that has the best correlation with the vector of noise $\eta = G(x, u, v)dw$.

Then, we calculate the bin edges, e_i, which partitions the range of the selected component z into N equal intervals

$$e_i = min(z) + \frac{i}{N}\bigg(max(z) - min(z)\bigg), \quad i = 0, 1, ..., N. \tag{5.7}$$

Next, the bin centers, c_i, are computed as the midpoint of each bin interval

$$c_i = \frac{e_i + e_{i+1}}{2}, \quad i = 0, 1, ..., N. \tag{5.8}$$

For each bin i, identify the data points that fall within the range of bin and calculate the variance of the noise data within that bin to estimate the local diffusion coefficient. Thus, define the bin indicator function, $I_i(z)$ as follows

$$I_i(z) = \begin{cases} 1 & \textbf{if } e_i \leq z < e_{i+1}, \\ 0 & \textbf{otherwise.} \end{cases} \tag{5.9}$$

Compute the variance of the noise $\eta = G(x, u, v)dw$ within each bin to estimate the local diffusion coefficient as

$$\sigma_i^2 = \frac{\sum_{j=1}^{M} \eta_j^2 . I_i(z_i)}{\sum_{j=1}^{M} I_i(z_i)}, \tag{5.10}$$

where M is the total number of data points, and η_j represents the noise data at the j^{th} point.

The diffusion coefficient $G(z)$ for each bin i is estimated as

$$G_{est}(c_i) = \sqrt{\frac{\sigma_i^2}{\Delta t}}, \quad i = 0, 1, ..., N. \tag{5.11}$$

where Δt is the time step. To ensure a smooth representation of the estimated diffusion dynamics across bins, a polynomial fitting is applied. We can calculate the polynomial values at the bin centers and refine it using the Gaussian smoothing to produce a final smooth approximation of the diffusion dynamics $G_{\textbf{smooth}}$. Finally, by using the

data of G_{smooth} as the output, and state and control policies (x, u, v) as the input we can obtain the GPR model of $G(x, u, v)$. The proposed approach to approximate the diffusion dynamics of the stochastic system is presented in Algorithm.

In This chapter, to enhance the quality of the approximated model, several several steps are done to enhance the quality of the approximated method including using Savitzky-Golay filter to record pure data, computing the mean function of GPR as the best estimate of the drift dynamics, recording a seperate data-set to validate the approximated method, and applying the method to three practical dynamics.

Algorithm 5 Diffusion Dynamics Approximation Method

Require: Noise component η and number of bins N.
Ensure: Approximation of $G(x, u, v)$.

Step 1 Choose the z as the selected component (state or controller) that has the best correlation with noise η.

Step 2 Calculate the bin edges using $e_i = \min(z) + \frac{i}{N}(\max(z) - \min(z))$ where $i = 0, 1, ..., N$.

Step 3 Computing the bin centers $c_i = \frac{e_i + e_{i+1}}{2}$ as the midpoint of each bin interval.

Step 4 For each bin i, identify the data points that fall within the range of bin as

$$I_i(z) = \begin{cases} 1 & \textbf{if } e_i \leq z < e_{i+1}, \\ 0 & \textbf{otherwise.} \end{cases}$$

Step 5 Calculate the variance of the noise data within that bin to estimate the local diffusion coefficient forward with eqation $\sigma_i^2 = \frac{\Sigma_{j=1}^{M} \eta_j^2 . I_i(z_i)}{\Sigma_{j=1}^{M} I_i(z_i)}$ where M is the total number of data points, and η_j represents the noise data at the j^{th} point.

Step 6 Obtain the diffusion coefficient $G_{\textbf{est}}(c_i) = \sqrt{\frac{\sigma_i^2}{\Delta t}}$ for each bin i where Δt is the time step.

Step 7 Apply the polynomial fitting to ensure a smooth representation of the estimated diffusion dynamics across bins.

Step 8 Calculate the polynomial values at the bin centers and refine it using the Gaussian smoothing to produce a final smooth approximation of the diffusion dynamics $G_{\textbf{smooth}}$.

Step 8 Use data of G_{smooth} as the output, and state and control policies (x, u, v) as the input to obtain the GPR model of $G(x, u, v)$.

return GPR model of $G(x, u, v)$.

5.3 Optimal Control Policy

After obtaining the GPR model of the drift and diffusion dynamics, the optimal control policies are obtained in this section. To this end, consider the Bellman/Isaac's principle [234]

$$V(x_t, t) = \min_u \max_v \mathbf{E}\Big[\int_t^{t+dt} \mathcal{L}(x, u, v)dt + V(x_{t+dt}, t + dt) \mid x_t \Big], \tag{5.12}$$

where t and $t + dt$ represent the values where the subscripts t and $t + dt$ represent the values of the variable at times t and $t + dt$, respectively. To compute the control policies, we expand both sides of equation (5.12) using the Taylor series around the nominal mean trajectories of state and controls $(\bar{x}, \bar{u}, \bar{v})$, where $\delta x = x - \bar{x}$, $\delta u = u - \bar{u}$, $\delta v = v - \bar{v}$.

The linearized dynamics (5.1) around $(\bar{x}, \bar{u}, \bar{v})$ holds

$$\begin{aligned} d\delta x_t = (\nabla_x f \delta x_t + \nabla_u f \delta u_t + \nabla_v f \delta v_t)dt + (G(\bar{x}, \bar{u}, \bar{v}) \\ + \nabla_x G(\delta x_t) + \nabla_u G(\delta u_t) + \nabla_v G(\delta v_t))dw, \end{aligned} \tag{5.13}$$

where $\nabla_x G(\delta x_t) = [\nabla_x G^{(1)} \delta x_t, ..., \nabla_x G^{(m)} \delta x_t]$ and $G^{(j)}$ represents the j^{th} column vector of G, $j = 1, ..., m$ and this is true for both $\nabla_u G(\delta u_t)$ and $\nabla_v G(\delta v_t)$. The expression for δx_{t+dt} in terms of δx_t is then obtained as

$$\begin{aligned} \delta x_{t+dt} = \delta x_t + (\nabla_x f \delta x_t + \nabla_u f \delta u_t + \nabla_v f \delta v_t)dt + (G(\bar{x}, \bar{u}, \bar{v}) \\ + \nabla_x G(\delta x_t) + \nabla_u G(\delta u_t) + \nabla_v G(\delta v_t))dw, \end{aligned} \tag{5.14}$$

and the update control laws δu_t and δv_t are computed from the following theorem, iteratively.

Theorem 7. *Let δx_t follow the continuous time nonlinear stochastic dynamics (5.13), the update control laws δu_t and δv_t are computed as*

$$\delta u_t^* = I_u + K_u \delta x_t, \tag{5.15a}$$

$$\delta v_t^* = I_v + K_v \delta x_t, \tag{5.15b}$$

where

$$I_u = -(P_{uu} - P_{uv}P_{vv}^{-1}P_{vu})^{-1}(P_u - P_{uv}P_{vv}^{-1}P_v), \tag{5.16a}$$

$$I_v = -(P_{vv} - P_{vu}P_{uu}^{-1}P_{uv})^{-1}(P_v - P_{vu}P_{uu}^{-1}P_u), \tag{5.16b}$$

$$K_u = -(P_{uu} - P_{uv}P_{vv}^{-1}P_{vu})^{-1}(P_{ux} - P_{uv}P_{vv}^{-1}P_{vx}), \tag{5.16c}$$

$$K_v = -(P_{vv} - P_{vu}P_{uu}^{-1}P_{uv})^{-1}(P_{vx} - P_{vu}P_{uu}^{-1}P_{ux}). \tag{5.16d}$$

and

$$P_0 = \mathcal{L} + \frac{1}{2} tr(\nabla_{xx} V_{t+dt} G G^T), \tag{5.17a}$$

$$P_x = \nabla_x \mathcal{L} + \nabla_x V_{t+dt} \nabla_x f + \sum_{j=1}^{m} G^{(j)^T} \nabla_{xx} V_{t+dt} \nabla_x G^{(j)}, \tag{5.17b}$$

$$P_u = \nabla_u \mathcal{L} + \nabla_x V_{t+dt} \nabla_u f + G^T \nabla_{xx} V_{t+dt} \nabla_u G, \tag{5.17c}$$

$$P_v = \nabla_v \mathcal{L} + \nabla_x V_{t+dt} \nabla_v f + G^T \nabla_{xx} V_{t+dt} \nabla_v G, \tag{5.17d}$$

$$P_{xx} = \nabla_{xx} \mathcal{L} + \nabla_x f^T \nabla_{xx} V_{t+dt} \nabla_x f + 2 \nabla_{xx} V_{t+dt} \nabla_x f \nabla_x G^T \nabla_{xx} V_{t+dt} \nabla_x G + \sum_{j=1}^{m} \nabla_x G^{(j)^T} \nabla_{xx} V_{t+dt} \nabla_x G^{(j)}, \tag{5.17e}$$

$$P_{uu} = \nabla_{uu} \mathcal{L} + \nabla_u f^T \nabla_{xx} V_{t+dt} \nabla_u f + \nabla_u G^T \nabla_{xx} V_{t+dt} \nabla_u G, \tag{5.17f}$$

$$P_{vv} = \nabla_{vv} \mathcal{L} + \nabla_v f^T \nabla_{xx} V_{t+dt} \nabla_v f + \nabla_v G^T \nabla_{xx} V_{t+dt} \nabla_v G, \tag{5.17g}$$

$$P_{ux} = \nabla_{ux} \mathcal{L} + \nabla_u f^T \nabla_{xx} V_{t+dt} + \nabla_u f^T \nabla_{xx} V_{t+dt} \nabla_x f + \nabla_u G^T \nabla_{xx} V_{t+dt} \nabla_x G, \tag{5.17h}$$

$$P_{vx} = \nabla_{vx} \mathcal{L} + \nabla_v f^T \nabla_{xx} V_{t+dt} + \nabla_v f^T \nabla_{xx} V_{t+dt} \nabla_x f + \nabla_v G^T \nabla_{xx} V_{t+dt} \nabla_x G, \tag{5.17i}$$

$$P_{uv} = \nabla_{uv} \mathcal{L} + \nabla_u f^T \nabla_{xx} V_{t+dt} \nabla_v f + \nabla_u G^T \nabla_{xx} V_{t+dt} \nabla_v G, \tag{5.17j}$$

$$P_{xu} = P_{ux}^T, \quad P_{xv} = P_{vx}^T, \quad P_{vu} = P_{uv}^T. \tag{5.17k}$$

The parameter $0 < \epsilon_i \leq 1$ *is added to the gains to make sure the convergence is obtained as follows*

$$\delta u_t^* = \epsilon I_u + K_u \delta x, \tag{5.18a}$$

$$\delta v_t^* = \epsilon I_v + K_v \delta x. \tag{5.18b}$$

In Differential Dynamic Programming (DDP), the step size for control updates is adjusted using a line-search procedure on the feedforward control component. This involves scaling the control update by a factor $\epsilon \in [0, 1]$ and evaluating multiple candi-

date solutions through forward passes under nonlinear dynamics. The best solution is chosen based on cost reproduction or a balance of cost and stability criteria. This process prevents overshooting into poor state regions while maintaining sufficiently large steps for faster convergence. Adjusting ϵ dynamically optimizes the balance between exploration and caution, improving DDP performance.

Proof. Extend the left-hand side of the equation (5.12) around the nominal state $\bar{x}$, one obtain

$$\begin{aligned} V(x_t, t) &= V(\bar{x}_t + \delta x_t, t) \\ &\approx V_t + \nabla_x V_t \delta x_t + \frac{1}{2} \delta x_t^T \nabla_{xx} V_t \delta x_t, \end{aligned} \tag{5.19}$$

where $V_t = V(\bar{x}_t, t)$. The first term of the right-hand side of the equation (5.12) can be approximated by

$$\begin{aligned} & E\left[\int_t^{t+dt} \mathcal{L}(x, u, v) dt \mid x_t \right] \\ & \approx \mathcal{L}(\bar{x}_t + \delta x_t, \bar{u}_t + \delta u_t, \bar{v}_t + \delta v_t) dt, \end{aligned} \tag{5.20}$$

which can be expanded as

$$\begin{aligned} & \mathcal{L} dt + (\nabla_x \mathcal{L} \delta x_t + \nabla_u \mathcal{L} \delta u_t + \nabla_v \mathcal{L} \delta v_t) dt \\ & + \frac{1}{2} \begin{bmatrix} \delta x_t \\ \delta u_t \\ \delta v_t \end{bmatrix}^T \begin{bmatrix} \nabla_{xx} \mathcal{L} & \nabla_{xu} \mathcal{L} & \nabla_{xv} \mathcal{L} \\ \nabla_{ux} \mathcal{L} & \nabla_{uu} \mathcal{L} & \nabla_{uv} \mathcal{L} \\ \nabla_{vx} \mathcal{L} & \nabla_{vu} \mathcal{L} & \nabla_{vv} \mathcal{L} \end{bmatrix} \begin{bmatrix} \delta x_t \\ \delta u_t \\ \delta v_t \end{bmatrix} dt, \end{aligned} \tag{5.21}$$

where $\mathcal{L} = \mathcal{L}(\bar{x}_t, \bar{u}_t, \bar{v}_t)$. By expanding the second term of the right-hand side of the

equation (5.12), one obtain

$$E[V_{t+dt} + \nabla_x V_{t+dt}\delta x_{t+dt} + \frac{1}{2}\delta x_{t+dt}^T \nabla_{xx} V_{t+dt}\delta x_{t+dt} \mid x_t]. \tag{5.22}$$

By referring to equation (5.13), equation (5.22) can be written as

$$\begin{aligned}
&V_{t+dt} + \nabla_x V_{t+dt}[\delta x_t + (\nabla_x f\delta x_t + \nabla_u f\delta u_t + \nabla_v f\delta v_t)dt] \\
&+ \frac{1}{2}[\delta x_t + (\nabla_x f\delta x_t + \nabla_u f\delta u_t + \nabla_v f\delta v_t)dt]^T \nabla_{xx} V_{t+dt} \\
&[\delta x_t + (\nabla_x f\delta x_t + \nabla_u f\delta u_t + \nabla_v f\delta v_t)dt] + \frac{1}{2}tr(\nabla_{xx} V_{t+dt} \\
&(G + \nabla_x G(\delta x_t) + \nabla_u G(\delta u_t) + \nabla_v G(\delta v_t))(G + \nabla_x G(\delta x_t) \\
&+ \nabla_u G(\delta u_t) + \nabla_v G(\delta v_t))^T)dt,
\end{aligned} \tag{5.23}$$

where G stands for $G(\bar{x}, \bar{u}, \bar{v})$ and $tr(.)$ represents the trace of a matrix. By combining (5.21) and (5.23), the right-hand side of the (5.12) can be written as

$$\begin{aligned}
&\mathbf{E}\Big[\int_t^{t+dt} \mathcal{L}(x,u,v)dt + V(x_{t+dt}, t+dt) \mid x_t\Big] \\
&= V_{t+dt} + P_0 dt + \nabla_x V_{t+dt}\delta x_t \\
&+ (P_x\delta x_t + P_u\delta u_t + P_v\delta v_t)dt + \frac{1}{2}\delta x_t^T \nabla_{xx} V_{t+dt}\delta x_t \\
&+ \frac{1}{2}\begin{bmatrix}\delta x_t \\ \delta u_t \\ \delta v_t\end{bmatrix}^T \begin{bmatrix}P_{xx} & P_{xu} & P_{xv} \\ P_{ux} & P_{uu} & P_{uv} \\ P_{vx} & P_{vu} & P_{vv}\end{bmatrix} \begin{bmatrix}\delta x_t \\ \delta u_t \\ \delta v_t\end{bmatrix} dt,
\end{aligned} \tag{5.24}$$

where the P functions are defined in (5.17). To obtain the optimal control laws δu_t^* and δv_t^*, we compute the derivative of (5.24) with respect to δu_t and δv_t and set them

equal to zero to get

$$\delta u_t^* = -P_{uu}^{-1}(P_{ux}\delta x_t + P_{uv}\delta v_t + P_u), \tag{5.25}$$

$$\delta v_t^* = -P_{vv}^{-1}(P_{vx}\delta x_t + P_{vu}\delta u_t + P_v). \tag{5.26}$$

We further solve for δu_t^* and δv_t^* from the equations (5.25) and (5.26) to eliminate δu_t and δv_t on the right-hand side to obtain

$$\delta u_t^* = I_u + K_u\delta x, \tag{5.27a}$$

$$\delta v_t^* = I_v + K_v\delta x, \tag{5.27b}$$

where

$$I_u = -(P_{uu} - P_{uv}P_{vv}^{-1}P_{vu})^{-1}(P_u - P_{uv}P_{vv}^{-1}P_v), \tag{5.28}$$

$$I_v = -(P_{vv} - P_{vu}P_{uu}^{-1}P_{uv})^{-1}(P_v - P_{vu}P_{uu}^{-1}P_u), \tag{5.29}$$

$$K_u = -(P_{uu} - P_{uv}P_{vv}^{-1}P_{vu})^{-1}(P_{ux} - P_{uv}P_{vv}^{-1}P_{vx}), \tag{5.30}$$

$$K_v = -(P_{vv} - P_{vu}P_{uu}^{-1}P_{uv})^{-1}(P_{vx} - P_{vu}P_{uu}^{-1}P_{ux}), \tag{5.31}$$

which completes the proof. □

5.4 Backward Propagation of the Value Function

To obtain the backward propagation of the value function, the corresponding backward ordinary differential equations for the value function and its first order and second order partial derivatives with respect to x are computed. The results are presented in the following theorem.

Theorem 8. *The backward ordinary differential equations for the value function and its first order and second order partial derivatives with respect to x_i are computed as*

$$
\begin{aligned}
-\frac{dV}{dt} &= P_0 + I_u^T P_u + I_v^T P_v + \frac{1}{2} I_u P_{uu} I_u + I_u^T P_{uv} I_v + \frac{1}{2} I_v^T P_v I_v, \\
-\frac{d(\nabla_x V)}{dt} &= P_x + K_u^T P_u + K_v^T P_v + P_{ux}^T I_u + P_{vx}^T I_w + K_u^T P_{uu} I_u \\
&+ K_u^T P_{uv} I_v + K_v^T P_{vu} I_u + K_v^T P_{vv} I_v, \\
-\frac{d(\nabla_{xx} V)}{dt} &= P_{xx} + 2K_u^T P_{ux} + 2K_v^T P_{vx} + 2K_v^T P_{vu} K_u + K_u^T P_{uu} K_u + K_v^T P_{vv} K_v,
\end{aligned}
\tag{5.32}
$$

subject to the terminal conditions

$$V(t_f) = \phi(\bar{x}(t_f), t_f), \tag{5.33a}$$

$$\nabla_x V(t_f) = \nabla_x \phi(\bar{x}(t_f), t_f), \tag{5.33b}$$

$$\nabla_{xx} V(t_f) = \nabla_{xx} \phi(\bar{x}(t_f), t_f). \tag{5.33c}$$

where

$$P_0 = \mathcal{L} + \frac{1}{2} tr(\nabla_{xx} V_{t+dt} G G^T), \tag{5.34a}$$

$$P_x = \nabla_x \mathcal{L} + \nabla_x V_{t+dt} \nabla_x f + \sum_{j=1}^{m} G^{(j)^T} \nabla_{xx} V_{t+dt} \nabla_x G^{(j)}, \tag{5.34b}$$

$$P_u = \nabla_u \mathcal{L} + \nabla_x V_{t+dt} \nabla_u f + G^T \nabla_{xx} V_{t+dt} \nabla_u G, \tag{5.34c}$$

$$P_v = \nabla_v \mathcal{L} + \nabla_x V_{t+dt} \nabla_v f + G^T \nabla_{xx} V_{t+dt} \nabla_v G, \tag{5.34d}$$

$$\begin{aligned} P_{xx} = {} & \nabla_{xx} \mathcal{L} + \nabla_x f^T \nabla_{xx} V_{t+dt} \nabla_x f + 2 \nabla_{xx} V_{t+dt} \nabla_x f \nabla_x G^T \nabla_{xx} V_{t+dt} \nabla_x G \\ & + \sum_{j=1}^{m} \nabla_x G^{(j)^T} \nabla_{xx} V_{t+dt} \nabla_x G^{(j)}, \end{aligned} \tag{5.34e}$$

$$P_{uu} = \nabla_{uu} \mathcal{L} + \nabla_u f^T \nabla_{xx} V_{t+dt} \nabla_u f + \nabla_u G^T \nabla_{xx} V_{t+dt} \nabla_u G, \tag{5.34f}$$

$$P_{vv} = \nabla_{vv} \mathcal{L} + \nabla_v f^T \nabla_{xx} V_{t+dt} \nabla_v f + \nabla_v G^T \nabla_{xx} V_{t+dt} \nabla_v G, \tag{5.34g}$$

$$P_{ux} = \nabla_{ux} \mathcal{L} + \nabla_u f^T \nabla_{xx} V_{t+dt} + \nabla_u f^T \nabla_{xx} V_{t+dt} \nabla_x f + \nabla_u G^T \nabla_{xx} V_{t+dt} \nabla_x G, \tag{5.34h}$$

$$P_{vx} = \nabla_{vx} \mathcal{L} + \nabla_v f^T \nabla_{xx} V_{t+dt} + \nabla_v f^T \nabla_{xx} V_{t+dt} \nabla_x f + \nabla_v G^T \nabla_{xx} V_{t+dt} \nabla_x G, \tag{5.34i}$$

$$P_{uv} = \nabla_{uv} \mathcal{L} + \nabla_u f^T \nabla_{xx} V_{t+dt} \nabla_v f + \nabla_u G^T \nabla_{xx} V_{t+dt} \nabla_v G, \tag{5.34j}$$

$$P_{xu} = P_{ux}^T, \quad P_{xv} = P_{vx}^T, \quad P_{vu} = P_{uv}^T. \tag{5.34k}$$

Proof. To derive the update law for the value function and its first and second-order partial derivatives, we substitute the optimal control policies (5.25) and (5.26) into the

expansion of equation (5.12) to obtain

$$
\begin{aligned}
& V_t + \nabla_x V_t \delta x_t + \frac{1}{2} \delta x_t^T \nabla_{xx} V_t \delta x_t \\
& = V_{t+dt} + P_0 dt + \nabla_x V_{t+dt} \delta x_t \\
& + P_x dt \delta x_t + P_u dt \delta u_t + P_v dt \delta v_t + \frac{1}{2} \delta x_t^T \nabla_{xx} V_{t+dt} \delta x_t \\
& + \frac{1}{2} \begin{bmatrix} \delta x_t \\ \delta u_t \\ \delta v_t \end{bmatrix}^T \begin{bmatrix} P_{xx} & P_{xu} & P_{xv} \\ P_{ux} & P_{uu} & P_{uv} \\ P_{vx} & P_{vu} & P_{vv} \end{bmatrix} \begin{bmatrix} \delta x_t \\ \delta u_t \\ \delta v_t \end{bmatrix} dt,
\end{aligned} \tag{5.35}
$$

after organizing the terms on the right-hand side of equation (5.35) into zeroth-order, first-order, and second-order expressions of δx_t, we equate the coefficients on both sides of equation (5.35). With some mathematical manipulations, we obtain

$$
\begin{aligned}
& -\frac{dV_t}{dt} = P_0 + I_u^T P_u + I_v^T P_v + \frac{1}{2} I_u P_{uu} I_u + I_u^T P_{uv} I_v + \frac{1}{2} I_v^T P_v I_v, \\
& -\frac{d\nabla_x V_t}{dt} = P_x + K_u^T P_u + K_v^T P_v + P_{ux}^T I_u + P_{vx}^T I_w + K_u^T P_{uu} I_u + K_u^T P_{uv} I_v \\
& + K_v^T P_{vu} I_u + K_v^T P_{vv} I_v, \\
& -\frac{d\nabla_{xx} V_t}{dt} = P_{xx} + 2K_u^T P_{ux} + 2K_v^T P_{vx} + 2K_v^T P_{vu} K_u + K_u^T P_{uu} K_u + K_v^T P_{vv} K_v.
\end{aligned} \tag{5.36}
$$

Finally, we have $V(x(t_f), t_f) = \phi(x(t_f))$. By taking the expansion around $\bar{x}(t_f)$ we have

$$
\begin{aligned}
& \phi(x(t_f)) = \phi(x(t_f) + \delta x_t(t_f)) \approx \phi(x(t_f)) \\
& + \nabla_x \phi x(t_f)) \delta x_t(t_f)^T + \delta x(t_f)^T \nabla_{xx} \phi(x(t_f)) \delta x_t(t_f).
\end{aligned} \tag{5.37}
$$

Hence, the boundary conditions at $t = t_f$ for the backward differential equations are

represented by (5.33), which completes the proof. □

Having established a method to compute the value function along with its first and second-order partial derivatives with respect to the state through backward propagation, the data-driven SGT-DDP algorithm is presented in Algorithm 6.

Algorithm 6 Data-Driven SGT-DDP

Require: Initial values for x_0, minimizing control u, maximizing control v, terminal time t_f, multiplier ϵ.

Ensure: Optimal minimizing control u^*, optimal maximizing control v^*, gains I_u, I_v, K_u, and K_v.

procedure UPDATE CONTROL($x_0, \bar{u}, \bar{v}, t_f, \epsilon$)

Collect the input-output data by applying random input data to equation (5.1) forward with x_0 to obtain the GPR model of drift dynamics.

Computing noise η using Savitzky-Golay filter to obtain the GPR of $G(x, u, v)$ from Algorithm 5.

for i from 1 to N, **do**

Obtain the initial mean state $\bar{x}$ by integrating the approximated drift part of the controlled dynamics forward with $x_0, \bar{u}$, and $\bar{v}$;

Calculate $V, \nabla_x V, \nabla_{xx} V$ at terminal time with respect to (5.33);

Integrate the equation (5.32) backward in the time;

Calculate gains I_u, I_v, K_u, K_v with respect to (5.16);

Obtain $\delta x_t(t)$ through $\delta x_{t+dt} = \delta x_t + (\nabla_x f \delta x_t + \nabla_u f \delta u_t + \nabla_v f \delta v_t)dt$ while putting δu_t and δv_t in $(I_u + K_u \delta x_t)$ and $(I_v + K_v \delta x_t)$, respectively;

Calculate δu_t and δv_t according to (5.15);

Update control policies $\bar{u} = \bar{u} + \delta u_t$ and $\bar{v} = \bar{v} + \delta v_t$;

Set $u^* = \bar{u}$ and $v^* = \bar{v}$;

end for

return $x^*, u^*, v^*, I_u, I_v, K_u, K_v$.

end procedure

Remark 11. *The proposed approach separates computational tasks into offline and online phases to ensure real-time feasibility. During the offline phase, computationally intensive processes, such as Gaussian Process Regression (GPR) training for drift dynamics and binning-based estimation for diffusion dynamics, are performed using pre-collected input-output data. Since these tasks involve significant computation, they are carried out offline to avoid burdening the real-time controller. Once the models*

are fully trained and validated, they are stored for subsequent use in real-time control. In the real-time phase, the control strategy leverages the pre-trained models for both drift and diffusion dynamics. Instead of requiring complex retraining or recalibration during operation, the real-time implementation involves only the evaluation of pre-trained models, which reduces computational demands significantly. This evaluation process primarily consists of simple function calls, ensuring low latency and efficient execution. As a result, the approach can handle real-time control tasks effectively, especially in systems with moderate sampling rates. By decoupling the offline training from real-time operation, this method ensures both accuracy and computational efficiency in dynamic environments.

Remark 12. *The solvability of the proposed zero-sum stochastic differential game is guaranteed by leveraging established results from continuous-time, two-player game theory [235]. Specifically, the existence of a well-defined value function satisfying the Hamilton-Jacobi-Isaacs (HJI) partial differential equation is ensured under certain regularity conditions. These include smoothness and Lipschitz continuity of the drift and diffusion dynamics in both state and control variables, as approximated by Gaussian Process Regression (GPR) models, and differentiability of the running and terminal cost functions. To facilitate solvability, we make some key assumptions as follows: (1) the GPR-based approximations of the drift and diffusion terms are assumed to be sufficiently smooth, ensuring the continuity and bounded partial derivatives required for accurate local linearization in the backward pass of the Stochastic Game Theoretic Differential Dynamic Programming (SGT-DDP) algorithm; (2) both control inputs and costs are bounded, ensuring the problem remains well-posed and that the Bellman/Isaacs principle can be applied; (3) any modeling errors introduced by the data-driven approach are considered negligible within the operating region as long as the offline training data sufficiently covers this region and the GPR models maintain local continuity and dif-*

ferentiability. Given these conditions, the proposed method admits a locally solvable formulation, where the iterative backward pass converges to a pair of feedback policies that constitute a local saddle-point solution. These clarifications highlight the robustness and well-posedness of the approach, ensuring that it provides a reliable solution framework for zero-sum stochastic differential games.

5.5 Numerical Examples

In this section, we apply the proposed algorithm to three practical examples, the inverted pendulum [236], the cart pole problem [237], and the unicycle problem [50].

5.5.1 Inverted Pendulum

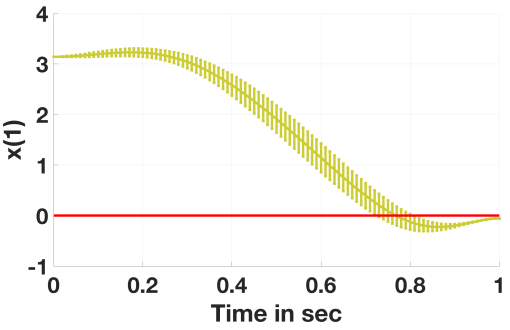
The dynamics of the inverted pendulum is given by

$$dx = \begin{bmatrix} x(2) \\ (mgl/I)sinx(1) - (b/I)x(2) + (1/I)(u+v) \end{bmatrix} dt + \begin{bmatrix} 0 \\ a(x(1)^2)(u+v) \end{bmatrix} dw. \tag{5.38}$$

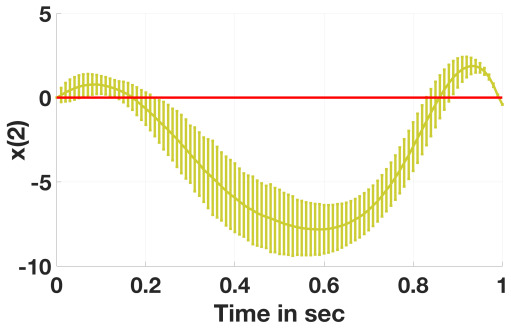
where $m = 1kg$, $l = 0.5m$, $b = 0.1(N.s)/m$, $I = ml^2$, $g = 9.81(kg.m)/s^2$, and $a = 1$. The goal is to bring the states $[x(1), x(2)]$ from $[\pi, 0]$ to $[0, 0]$. The cost function is defined as

$$J = \frac{1}{2}x^T Q_f x + \frac{1}{2}\int_{t_0}^{t_f} (u^T R_u u - v^T R_v v)dt, \tag{5.39}$$

where the parameters and initial conditions are set as $Q_f = [50, 0; 0, 1]$, $R_u = 0.01$, $R_v = 10$, $\bar{u} \equiv 0$, $\bar{v} \equiv 0$, $\epsilon = 0.1$, and $t_f = 1$.



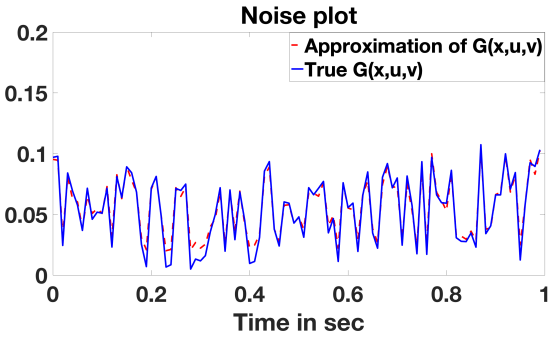
(a) Mean and standard deviation of 1000 trajectories of x(1).



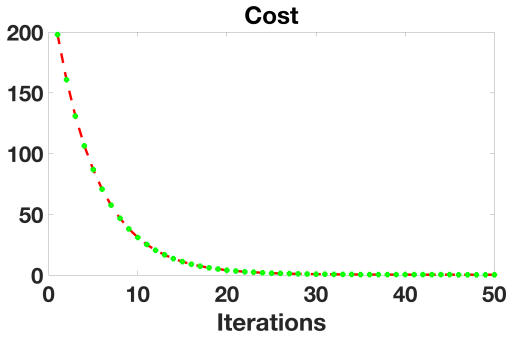
(b) Mean and standard deviation of 1000 trajectories of x(2).

Figure 5.1: Mean and standard deviation of 1000 trajectories of inverted pendulum

Fig. 5.1a and 5.1b show the evolution of the mean and standard deviation for 1000 simulated trajectories in the inverted pendulum system using the data-driven SGT-DDP method. Fig. 5.1a depicts the angle with the mean trajectory reaching the upright position, while Fig. 5.1b shows the angular velocity stabilizing toward zero. Fig. 5.2a compares the true diffusion dynamics of the system with the approximation



(a) G(x,u,v) and approximation.



(b) Cost per iteration.

Figure 5.2: Noise and cost per iteration of inverted pendulum

using the binning method and GPR. The close match validates the effectiveness of method in capturing the dynamics, which is essential for designing control policies. Fig. 5.2b illustrates the convergence of the SGT-DDP algorithm and shows a monotonically decreasing cost function over iterations. This confirms the successful optimization of

control inputs to minimize system cost.

5.5.2 Cart Pole Dynamics

In the second example, we consider the cart pole problem, where the dynamics of the system is given by

$$dx = F(x, u, v)dt + G(x, u, v)dw, \tag{5.40}$$

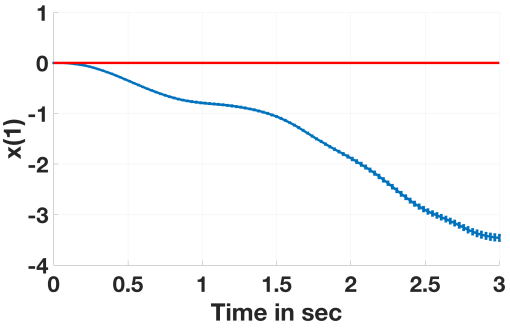
where

$$F = \begin{bmatrix} x(2) \\ \frac{m\sin(x(3))(-lx(4)^2+g\cos(x(3))+u+v}{M+m\sin^2 x(3)} \\ x(4) \\ \frac{(M+m)g\sin(x(3))+\cos(x(3))(-mlx(4)^2\sin(x(3))+u+v)}{l(M+m\sin^2(x(3)))} \end{bmatrix} \tag{5.41}$$

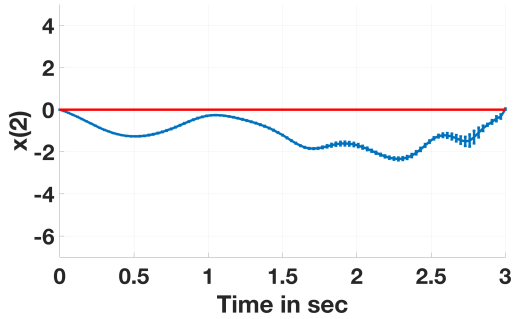
$$G = \begin{bmatrix} x(4)^2 + \sin(x(4))^2 \\ 0 \\ x(2)u^2 \\ 0 \end{bmatrix}, \tag{5.42}$$

and $l = 0.5$, $M = 10$, $g = 9.81$,and $m = 1$. The goal is to bring the states $[x, \dot{x}, \theta, \dot{\theta}] = [x(1), x(2), x(3), x(4)]$, where x represents the displacement of the cart and θ stands for the angle of the pole, from $[0, 0, \pi, 0]$ to $[0, 0, 0, 0]$. Here, $x(1)$ which represents the final position of the cart is not specified. The cost function is given same as previous example. We set parameters and initial conditions as $Q_f = [0, 0, 0, 0; 0, 500, 0, 0; 0, 0, 500, 0; 0, 0, 0, 10]$, $R_u = 0.01$, $R_v = 10$, $\bar{u} \equiv 0$, $\bar{v} \equiv 0$,

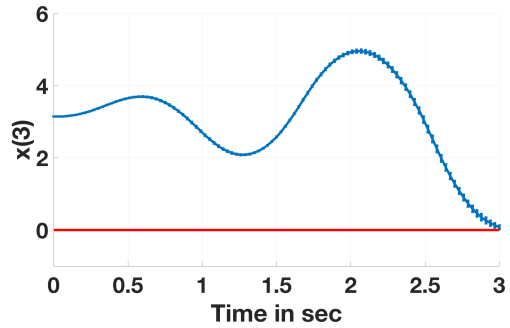
$\epsilon = 0.9$, and $t_f = 3$.



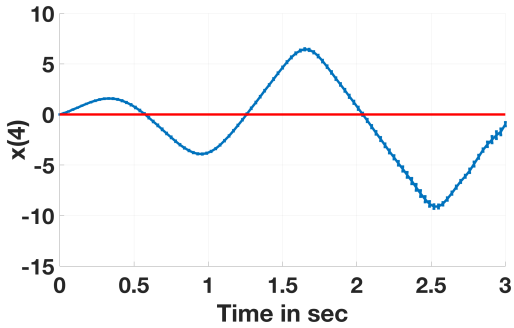
(a) Mean and standard deviation of 1000 trajectories of x(1).



(b) Mean and standard deviation of 1000 trajectories of x(2).



(c) Mean and standard deviation of 1000 trajectories of x(3).



(d) Mean and standard deviation of 1000 trajectories of x(4).

Figure 5.3: Mean and standard deviation of 1000 trajectories of cart pole.

Fig. 5.3a through Fig. 5.3d illustrate the performance of the proposed SGT-DDP method in controlling the cart pole by displaying the mean and standard deviation of 1000 simulated trajectories for the state variables which are cart's position and velocity, and pole's angle and angular velocity. These figures collectively showcase the capability of the method to stabilize the cart pole. Note that Fig. 5.3a illustrates the position of the cart and we do not enforce it to converge to zero.

Fig. 5.5 compares the actual diffusion dynamics with their approximations obtained through the binning method and GPR in the cart pole. The close match between the true and estimated values for $G(1)$ and $G(3)$ validates the accuracy of the method

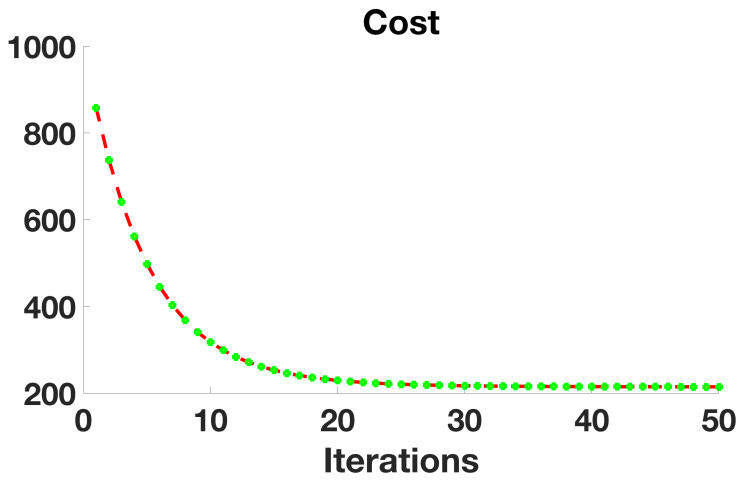
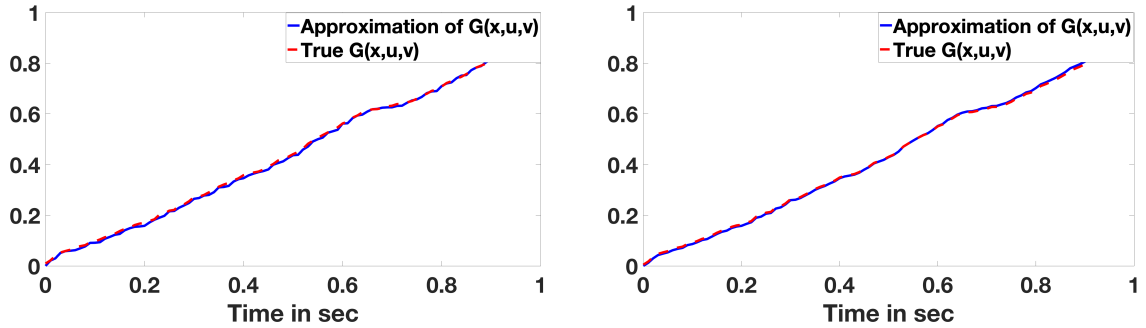


Figure 5.4: Cost per iteration of cart pole.



(a) Plots of G(1) and approximation of G(1). (b) Plots of G(3) and approximation of G(3).

Figure 5.5: Plots of G in cart pole model and approximation of G.

in modeling diffusion dynamics. Meanwhile, Fig. 5.4 displays the convergence of the cost function, with a steady decrease over iterations and confirms the success of the SGT-DDP method in optimizing control inputs and minimizing system cost.

5.5.3 Unicycle Robot

The third example considers a unicycle robot [50] where the state vector $x = [x, y, \theta, \nu]$ includes $2D$ position, the orientation, and the forward velocity of the vehicle. The control vector $u = [u^\theta, u^\nu]$ includes variable u^θ that changes the steering angle and u^ν that changes the forward velocity, and assuming that the noise only enters the velocity

channel. The dynamics of the system is given by

$$dx = F(x, u, v)dt + G(x, u, v)dw, \tag{5.43}$$

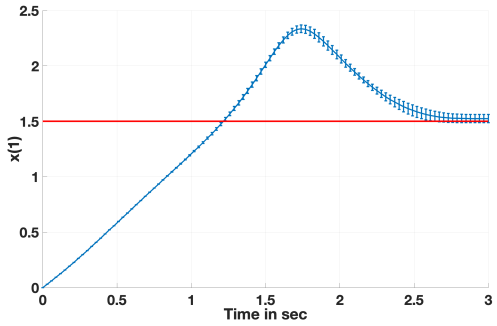
where

$$F = \begin{bmatrix} \nu cos(\theta) \\ \nu sin(\theta) \\ \nu(u^{\theta} + v^{\theta}) \\ (u^{\nu} + v^{\nu}) \end{bmatrix}, \tag{5.44}$$

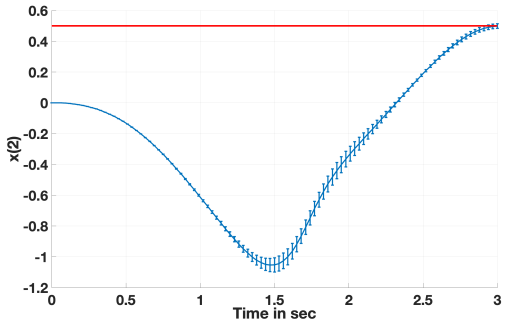
$$G = \begin{bmatrix} 0 \\ 0 \\ y^2 \\ 0 \end{bmatrix}. \tag{5.45}$$

The cost function is given the same as in (5.39). We set parameters and initial conditions as $Q_f = [50, 0, 0, 0; 0, 500, 0, 0; 0, 0, 500, 0; 0, 0, 0, 50]$, $R_u = [0.1, 0, 0, 0.1]$, $R_v = [10, 0; 0, 10]$, $\bar{u} \equiv 0$, $\bar{v} \equiv 0$, $\epsilon = 0.1$, and $t_f = 3$. In this example, the goal is to bring the states $[x, y, \theta, \nu]$ from $[0, 0, 0, 1]$ to $[1.5, 0.5, \pi/2, 0]$.

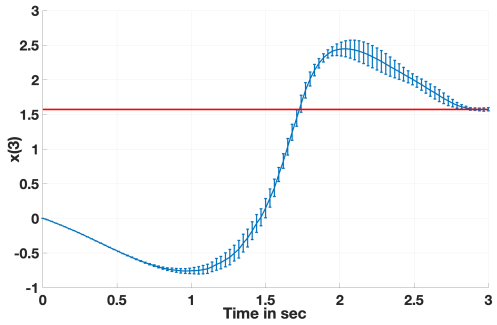
Fig. 5.6 illustrates the performance of the proposed SGT-DDP algorithm for the unicycle robot system over 1000 simulated trajectories. The subfigures (5.6a through 5.6d) display the evolution of the mean and standard deviation for the state variables $x(1), x(2), x(3)$, and $x(4)$, which correspond to the $2D$ position, orientation, and forward velocity of the robot, respectively. These results confirm that the algorithm successfully guides the robot from its initial state $[0, 0, 0, 1]$ to its target state



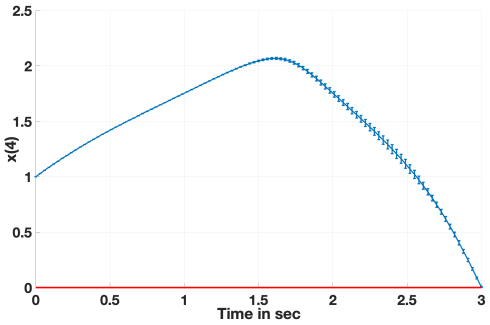
(a) Mean and standard deviation of 1000 trajectories of x(1).



(b) Mean and standard deviation of 1000 trajectories of x(2).



(c) Mean and standard deviation of 1000 trajectories of x(3).



(d) Mean and standard deviation of 1000 trajectories of x(4).

Figure 5.6: Mean and standard deviation of 1000 trajectories of unicycle robot.

$[1.5, 0.5, pi/2, 0]$, achieving stability in all states over time.

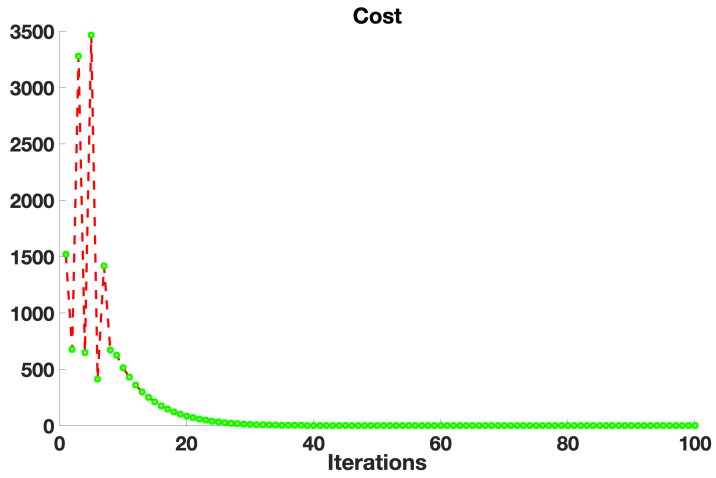


Figure 5.7: Cost per iteration of unicycle robot.

Fig. 5.7 shows the convergence of the cost function during the iterations of the SGT-DDP algorithm. The monotonic convergence in the cost over 100 iterations indicates that the algorithm effectively optimizes the control inputs to achieve the desired goal while minimizing the cost function.

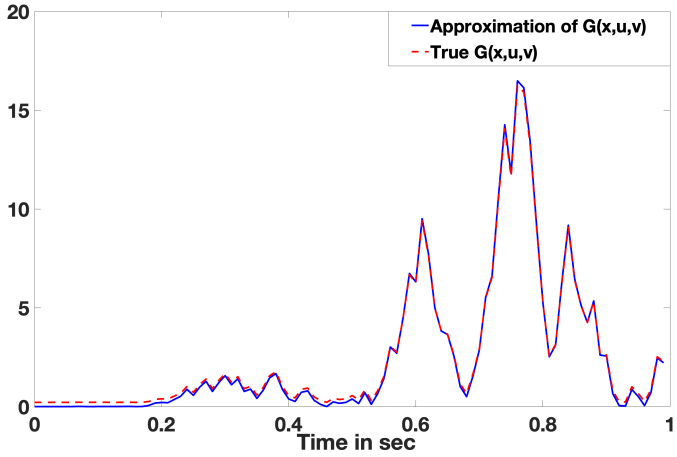


Figure 5.8: Plots of G in unicycle model and approximation of G.

Fig. 5.8 compares the actual diffusion dynamics of the unicycle robot system with the approximations obtained through the binning method and GPR. The close agree-

ment between the true and estimated values validates the accuracy of the proposed method in modeling diffusion dynamics. Notably, this study includes the diffusion dynamics in the control framework, unlike [50], where noise dynamics were absent. This addition enhances the robustness of the proposed SGT-DDP algorithm under stochastic environments.

Chapter 6

Distributed Min-Max Differential Dynamic Programming for Large-Scale Systems with Mismatched Interconnections

6.1 Introduction

Large-scale systems represent complex and intricate networks or structures comprising numerous interconnected components, often exhibiting a high degree of interdependence and complexity [238]. These systems pervade various domains, including engineering, economics, and biology [239, 240]. Characterized by a multitude of interacting variables or subsystems, large-scale systems pose significant challenges in terms of analysis, optimization, and control [241, 242]. Managing such systems requires sophisticated methods that not only address the intricacies of their dynamics but also account for the potential uncertainties and constraints [243]. Developing effective control strategies for these systems is a pivotal pursuit, with the aim to enhance their efficiency, stability, and performance across a wide spectrum of real-world applications. Large-scale systems find applications across diverse fields, influencing critical domains

such as transportation networks, power grids, environmental modeling, and healthcare systems. In transportation networks, the work by [244] discusses employing the Parallel Internet of Vehicles (PIV) using an artificial system, a computational experiment, and parallel execution intelligent approach to monitor large-scale logistics transport vehicles. Similarly, within the power storage problems, the research conducted by [245] examines large-scale renewable energy storage systems, particularly focusing on the potential to store energy in the form of hydrogen as a green and promising alternative. Furthermore, in healthcare systems, the study by [246] exemplifies a large-scale model for healthcare organizations, addressing the crucial aspects of cost of care and quality of care within the US healthcare system. The inherent advantage of large-scale systems lies in their capacity to model complex interactions between numerous variables, providing a clear view of complex phenomena. Their strengths encompass the ability to capture emergent behaviors, facilitate comprehensive optimization, and enable the development of robust control strategies that can effectively manage uncertainties and disturbances.

In the realm of large-scale systems, various control strategies have been developed and implemented to manage their complexity. Classic control approaches such as Proportional-Integral-Derivative (PID) controllers and Linear Quadratic Regulator (LQR) techniques have been extensively utilized [247, 248, 249]. While effective in certain cases, these methods often encounter limitations when applied to large-scale systems. For instance, PID controllers lack robustness when dealing with highly interconnected subsystems. On the other hand, LQR method is primarily effective for linear systems and may not perform desirable results under uncertainties, nonlinearities, and interconnections. Modern control methodologies have emerged as alternatives to traditional methods [250]. Model Predictive Control (MPC) stands as a notable advancement, offering a predictive framework that considers system dynamics, constraints, and

objectives [133]. For instance, [251] proposes MPC to enhance the power quality of a large-scale power plant connected to the grid through Paralleled Voltage Source Inverters (PVSIs). In [252], the fuzzy MPC is considered for a fuzzy large-scale system with hierarchical optimization scheme. Although MPC methods consider interconnections, it cannot handle the complex interconnection between subsystems effectively and can face challenges in computational complexity. In [253], distributed strategy based on distributed dynamic programming algorithm is proposed to optimally allocate the total power demand among different generation units. In [254], the MPC is designed for a large-scale system with time-varying delay which the interconnections are obtained through the Kalman filter approach. However, the approach only works for linear dynamics. Besides MPC methods, the robust control strategies have gained traction due to their ability to address uncertainties and disturbances inherent in large-scale systems [255]. The Min-Max DDP for a type of continuous nonlinear systems in proposed in [225]. But it is not specified whether the proposed method is applicable for nonlinear multi-agent systems or not. In paper [256], the problem of managing the power flow among different components of a microgrid is considered while ensures that the demands of all loads are satisfied. However, the mismatch interconnection which is appeared in many applications is not considered here. A two level robust optimal control is proposed in [257] to stabilize a large-scale system where control inputs are obtained by solving regularized least squares problems with bounded data uncertainties. In [258], a continuous time dynamic programming is developed for networked system including multi-agent Markov decision processes. However, there is no discussion of how the algorithm scales with the number of agents or the size of state/action spaces.

Although there have been a lot of modern control methodologies such as MPC, fuzzy control, robust control, PID, and LQR for large-scale systems and they represent significant advancements in managing large-scale systems, they are not without

limitations [259, 260, 261, 262]. Challenges such as computational complexity, conservative designs, and stability issues, emphasizing the need for more advanced control strategies. In contrast to traditional approaches, the employment of optimal control strategies, such as Min-Max DDP, presents a promising avenue for addressing the shortcomings encountered by conventional methods in managing large-scale systems [8]. Min-Max DDP is an advanced control technique used to solve optimal control and differential game problems, especially those affected by uncertainties, disturbances, or adversarial conditions. It is particularly adept at handling scenarios where the system might face varying conditions or environments, making it suitable for applications in robotics and adversarial scenarios. The Min-Max DDP for both continuous and discrete time nonlinear dynamics is derived in [8]. An extension of Min-Max DDP to deal with stochastic dynamics is proposed in [13]. The Min-Max DDP framework for output feedback control systems with observer is studied in [263]. The DDP framework for dynamics including time-varying delay in states is studied in [264]. Although the Min-Max DDP has been used in many works including stochastic dynamics, output feedback systems, continuous and discrete time systems, time-varying delay systems, due to its capability of solving complex problems, to the best knowledge of the authors, it has not been utilized for large-scale dynamics with several subsystems and interconnections.

To this end, the proposed paper addresses the problem of applying a distributed control algorithm for large-scale systems by applying Min-Max DDP on subsystems of a large-scale system with mismatched interconnections which refers to discrepancies between the assumed or modeled interactions among subsystems and their actual dynamics and arise from modeling errors, parameter variations, or unaccounted external disturbances.

The contribution of the paper is divided as follows.

1) The large-scale model is decomposed into several distributed models, each with two players, where a distributed Min-Max problem is considered.

2) The optimal update laws of the control policies of the two players and the corresponding set of backward differential equations are provided.

3) The convergence of the cost function and control update laws to their optimal values are guaranteed under some mild assumptions.

4) The state of the overall closed-loop system is proved to converge in the concept of Lyapunov function.

The rest of the paper is organized as follows. In Section 6.2, some notations that are used in the paper are presented. Section 6.3 provides the problem description, including the initial formulations, assumptions, and definitions of the study. In Section 6.4, the distributed control methodology is proposed for the large-scale system. Finally, in Section 6.5, numerical examples and their results are presented.

6.2 Notation

Some required notations that are used in this study are presented in this section. $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{R}^n$, and $\mathbb{R}^{n\times m}$ shows the set of all real numbers, positive real numbers, $n \times 1$ real column vectors, and $n \times m$ real matrices, respectively. $\mathcal{C}^1(\mathbb{R})$ and $\mathcal{C}^2(\mathbb{R})$ represent sets of functions whose first and second-order derivative exist and is continuous with domain $\mathbb{R}$. I_n is the identity matrix with the dimension $n \times n$. For $A \in \mathbb{R}^{n\times m}$, A^T illustrates the transpose of A, and $\|A\| = \sqrt{tr(A^T A)}$ represents the Frobenius-norm, and $tr(.)$ corresponds to the trace. For any specified functions $\psi : \Re^n \times \Re \to \Re$,

$\Psi : \Re^n \times \Re^m \times \Re \to \Re$ and $\Omega : \Re^n \times \Re \to \Re^n$, the following are given

$$\psi_x(x,t) = \begin{bmatrix} \frac{\partial \psi(x,t)}{\partial x_1} \\ \vdots \\ \frac{\partial \psi(x,t)}{\partial x_n} \end{bmatrix},$$

$$\psi_{xx}(x,t) = \begin{bmatrix} \frac{\partial^2 \psi(x,t)}{\partial^2 x_1}, & \cdots & , \frac{\partial^2 \psi(x,t)}{\partial x_1 x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \psi(x,t)}{\partial x_1 \partial x_n}, & \cdots & , \frac{\partial^2 \psi(x,t)}{\partial^2 x_n} \end{bmatrix},$$

$$\Psi_{xu}(x,u,t) = \begin{bmatrix} \frac{\partial^2 \Psi(x,u,t)}{\partial x_1 \partial u_1}, & \cdots & , \frac{\partial^2 \Psi(x,u,t)}{\partial x_1 \partial u_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi(x,u,t)}{\partial x_n \partial u_1}, & \cdots & , \frac{\partial^2 \Psi(x,u,t)}{\partial x_n \partial u_n} \end{bmatrix},$$

$$\Omega_x(x,t) = \begin{bmatrix} \frac{\partial \Omega(x,t)}{\partial x_1}, & \cdots & , \frac{\partial \Omega_1(x,t)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \Omega_n(x,t)}{\partial x_1}, & \cdots & , \frac{\partial \Omega_n(x,t)}{\partial x_n} \end{bmatrix},$$

which represents the partial derivative of ψ with respect to x, Hessian of ψ with respect to x, Hessian of Ψ with respect to x and u, and the Jacobian matrix of Ω with respect to x, respectively.

6.3 Problem Description

Consider the continuous time nonlinear large-scale system as follows

$$\frac{dx_i(t)}{dt} = f_i(x_i(t)) + g_i(x_i(t))u_i(t) + \Delta E_i(x(t)),\ x_i(t_0) = x_{i0}, \quad i = 1, 2, ..., N, \quad (6.1)$$

where N is the number of subsystems, $x_i(t) \in \mathbb{R}^n$ is the state of the i^{th} subsystem, $u_i(t) \in \mathbb{R}^m$ is the control of the i^{th} subsystem, $f_i(x_i(t)) \in \mathbb{R}^n$ and $g_i(x_i(t)) \in \mathbb{R}^{n\times m}$ are the internal dynamics and input matrix. $\Delta E_i(x(t)) \in \mathbb{R}^n$ is the uncertain interconnection between subsystems. Standard assumptions are provided as follows to facilitate the analyses of the properties of the proposed distributed control policies for the large-scale system (6.1).

Assumption 6. *The internal dynamic $f_i(x_i)$ and input matrix $g_i(x_i)$ are assumed to be Lipschitz continuous. Simultaneously, $f_i(0) = 0$, i.e., $x_i = 0$ is the equilibrium point of the i^{th} subsystem (6.1) when the corresponding input $u_i(t)$ and mismatched interconnection $\Delta E_i(x(t))$ are equal to zero for all $t \geq 0$.*

Assumption 7. *The uncertain interconnection $\Delta E_i(x)$ satisfies the following mismatched condition*

$$\Delta E_i(x) = k_i(x_i)\Psi_i(x), \quad (6.2)$$

where $k_i(x_i) \in \mathbb{R}^{n_i\times I_i}$ is an unknown smooth function and $\Psi_i(x) \in \mathbb{R}^{I_i}$ is the uncertain function and the uncertain function $\Psi_i(x)$ is bounded as

$$\|\Psi_i(x)\| \leq \sum_{s=1}^{N} a_{is}\alpha_{is}(\|x_s\|), \quad (6.3)$$

where $\alpha_{is}(\cdot), s = 1, 2, ...N$, are class κ functions [265] and $a_{is}, s = 1, 2, ..., N$, are non-negative constants.

Let

$$\alpha_s(\|x_s\|) = \max\{\alpha_{1s}(\|x_s\|), \alpha_{2s}(\|x_s\|), ..., \alpha_{Ns}(\|x_s\|)\}, \tag{6.4}$$

then, (6.3) can be replaced by

$$\|\Psi_i(x_i)\| \leq \sum_{s=1}^{N} b_{is}\alpha_s(\|x_s\|), \tag{6.5}$$

where $b_{is} \geq a_{is}\alpha_{is}(\|x_s\|)/\alpha_s(\|x_s\|), s = 1, 2, ..., N$ are positive constant.

Assumption 8. *There exists class κ functions $\beta_l(.), l = 1, 2, ..., N$, such that*

$$\|g_i^+(x_i)\Delta E_i(x(t))\| \leq \sum_{l=1}^{N} c_{il}\beta_l(\|x_l\|), \tag{6.6}$$

where $g_i^+(x_i)$ represents the Moore-Penrose pseudo-inverse of $g_i(x_i)$ and $c_{il}, l = 1, 2, ..., N$ are positive constants.

The object of This chapter is to find distributed control policies for the nonlinear large-scale system (6.1) with several subsystems and uncertain mismatched interconnections. To reach this goal, we first transform the dynamical model of the large-scale system (6.1) to a distributed form. Then, a decentralized optimal control problem is considered for each subsystem. As the uncertain interconnection $\Delta E_i(x)$ is unavailable, we solve these optimal control problems through the Min-Max DDP algorithm which does not require the information of the interconnections $\Delta E_i(x)$.

Remark 13. *Assumptions 7 and 8 are introduced to ensure a structured and bounded representation of the uncertain interconnections among subsystems. Assumption 7 ex-*

presses the interconnection as a product of a smooth subsystem-dependent function and a bounded uncertain function, using class-κ bounds to ensure that the interaction terms remain tractable. Assumption 8 utilizes the pseudo-inverse of the input matrix to quantify the influence of mismatched components in a form that supports decomposition into a two-player differential game. These assumptions are standard in distributed control literature and are essential to guarantee stability and convergence within the proposed Min-Max DDP framework. Nonetheless, relaxing these assumptions to accommodate more general interconnection models is a meaningful topic for future research.

6.4 Distributed Control Method

In this section, we first transform the nonlinear large-scale system (6.1) into a distributed form of nonlinear system with two conflicting controls where one is trying to stabilize the system by minimizing the cost function, while the other one is trying to destabilize the system by maximizing the cost function. Then, the decentralized optimal control policies for each subsystem are applied to the large-scale system.

6.4.1 Distributed Formulation

For the i^{th} subsystem, $\Delta E_i(x)$ can be written as

$$\Delta E_i(x) = g_i(x_i)g_i^{+}(x_i)\Delta E_i(x) + (I_{ni} - g_i(x_i)g_i^{+}(x_i))\Delta E_i(x). \tag{6.7}$$

The first term at the right-hand side of (6.7) is the matched component of $g_i(x_i)$ and the second term at the right-hand side of (6.7) is the mismatched component of $g_i(x_i)$.

Hence, equation (6.1) can be written as

$$\frac{dx_i}{dt} = f_i(x_i) + g_i(x_i)u_i + g_i(x_i)g_i^+(x_i)\Delta E_i(x) \\ + (I_{ni} - g_i(x_i)g_i^+(x_i))\Delta E_i(x). \tag{6.8}$$

First, considering the matched component $g_i(x_i)g_i^+(x_i)\Delta E_i(x)$, this term lies in the range of $g_i(x_i)$, meaning it is aligned with the input matrix $g_i(x_i)$. Thus, we can write

$$g_i(x_i)(u_i + g_i^+(x_i)\Delta E_i(x)) = g_i(x_i)u_i^{new}. \tag{6.9}$$

Next, we focus on the mismatched component of the dynamics. The mismatched component which is given by $(I_{ni} - g_i(x_i g_i^+(x_i))\Delta E_i(x)$, cannot be directly controlled by $u_i(t)$, where the term $(I_{ni} - g_i(x_i)g_i^+(x_i))$ projects $\Delta E_i(x)$ into the null space of $g_i(x_i)$, meaning this part cannot be directly controlled by $g_i(x_i)u_i$. Thus, we want to use the Assumption (7) to handle this component that is $E_i(x) = k_i(x_i)\Psi_i(x)$. We have

$$(I_{ni} - g_i(x_i)g_i^+(x_i))\Delta E_i(x) = (I_{ni} - g_i(x_i)g_i^+(x_i))k_i(x_i)\Psi_i(x). \tag{6.10}$$

To handle the mismatched component, we design h_i as a control input that can play the role of uncertainty and approximate the effect of $\Psi_i(x)$, and the term $(I_{ni} - g_i(x_i)g_i^+(x_i))k_i(x_i) = q_i(x_i)$ as the second input matrix.

Consider (6.7) through (6.10) and dynamics (6.1), the i^{th} subsystem can be further

recast as

$$\frac{dx_i}{dt} = f_i(x_i) + g_i(x_i)u_i^{new} + q_i(x_i)h_i, \tag{6.11}$$

where

$$q_i(x_i) = (I_{ni} - g_i(x_i)g_i^{+}(x_i))k_i(x_i), \tag{6.12}$$

and $u_i^{new} \in \mathbb{R}^m$ and $h_i(t) \in \mathbb{R}^m$ are the control inputs. The i^{th} subsystem (6.11) represents the form of a system with two players. In this study, since the interconnection between subsystems $\Delta E_i(x$ is assumed to be uncertain, we consider the scenario where the control h_i can be considered as the intelligent disturbance that plays against the control u_i. A differential game formulation between the two players can be utilized to find the controls, which can be solved through the Min-Max differential dynamic programming (DDP) approach [8]. The Min-Max DDP is a method used in the field of dynamic optimization to deal with problems involving conflicting objectives between two decision-makers in a dynamic environment. In this framework, the first player u_i aims to to stabilize the system by minimizing a cost function, while the second player h_i attempts to destabilize the system by maximizing the same cost function.

Remark 14. *In contrast to previous approach in [266] where an adaptive dynamic programming (ADP) algorithm is utilized for large-scale system, our methodology presents a notable departure in both modeling and controller strategies. Specifically, the prior work transforms the system into a non-zero sum game, allocating both controllers to minimize a quadratic cost, which indicates that the mismatched interconnection is cooperating with the control u_i. But their formulation is not aligned with the uncertainty of the interconnection term $\Delta E_i(x)$. Instead, our novel framework formulates the problem*

as a zero-sum game with a general form of cost, where u_i serves as a stabilizer, while h_i serves as a destabilizer, which is a good representation of the mismatched interconnection and the u_i we obtain can cope with the uncertain mismatched component.

The differential game problem of the i^{th} subsystem is cast as follows. The dynamics is described as

$$\frac{dx_i}{dt} = F_i(x_i, u_i, h_i),\ x_i(t_0) = x_{i0}, \quad t \in [t_0, t_f], \tag{6.13}$$

where

$$F_i(x_i, u_i, h_i) = f_i(x_i) + g_i(x_i)u_i + q_i(x_i)h_i. \tag{6.14}$$

The corresponding local cost function is introduced as

$$J_i(x_i, u_i, h_i) = \Gamma_i(x_i(t_f), t_f) + \int_{t_0}^{t_f} r_i(x_i(t), u_i(t), h_i(t))dt, \tag{6.15}$$

where $r_i(x_i(t), u_i(t), h_i(t))$ represents the running cost and $\Gamma_i(x_i(t_f), t_f)$ denotes the terminal cost for each subsystem. The Min-Max value of the cost function (6.15) is expressed as

$$V_i(x_{i0}, t_0) = \min_{u_i} \max_{h_i} \left\{ \Gamma_i(x_i(t_f), t_f) + \int_{t_0}^{t_f} r_i(x_i(t), u_i(t), h_i(t))dt \right\}. \tag{6.16}$$

6.4.2 Optimal Control Policy

In this section, the local optimal control update laws for each subsystem is obtained by solving the Hamilton-Jacobi-Bellman-Isaacs (HJBI) Partial Differential Equation (PDE) that is satisfied by the value function of the Min-Max differential game. The

local HJBI equation for the i^{th} subsystem is provided as

$$-\frac{\partial V_i(x_i,t)}{\partial t} = \min_{u_i}\max_{h_i}\{r_i(x_i(t),u_i(t),h_i(t)) + V_{ix_i}(x_i,t)^T F_i(x_i(t),u_i(t),h_i(t))\}, \tag{6.17}$$

with the terminal condition $V_i(x_i(t_f),t_f) = \Gamma_i(x_i(t_f),t_f)$. Henceforth, we omit the time dependence t from the equation terms for simplicity of representation.

Let the state and control histories be represented as $(\bar{x}_i, \bar{u}_i, \bar{h}_i)$, the local nonlinear dynamics (6.13) can be expressed as

$$\frac{dx_i}{dt} = F_i(\bar{x}_i + \delta x_i, \bar{u}_i + \delta u_i, \bar{h}_i + \delta h_i), \tag{6.18}$$

where

$$\delta x_i = x_i - \bar{x}_i, \tag{6.19a}$$

$$\delta u_i = u_i - \bar{u}_i, \tag{6.19b}$$

$$\delta h_i = h_i - \bar{h}_i, \tag{6.19c}$$

and the linearized dynamics around $(\bar{x}_i, \bar{u}_i, \bar{h}_i)$ for each subsystem is obtained as

$$\frac{d\delta_{x_i}}{dt} = \bar{F}_{i_{x_i}}\delta x_i + \bar{F}_{i_{u_i}}\delta u_i + \bar{F}_{i_{h_i}}\delta h_i, \tag{6.20}$$

where $\bar{F}_{i_{x_i}}$, $\bar{F}_{i_{u_i}}$, and $\bar{F}_{i_{h_i}}$ stands for $\bar{F}_{i_{x_i}}(\bar{x}_i, \bar{u}_i, \bar{h}_i)$, $\bar{F}_{i_{u_i}}(\bar{x}_i, \bar{u}_i, \bar{h}_i)$, and $\bar{F}_{i_{h_i}}(\bar{x}_i, \bar{u}_i, \bar{h}_i)$. The optimal control update laws δu_i and δh_i in (6.18) are obtained iteratively through the following theorem.

Theorem 9. *Let*

$$\frac{dx_i}{dt} = F_i(\bar{x}_i + \delta x_i, \bar{u}_i + \delta u_i, \bar{h}_i + \delta h_i), \tag{6.21}$$

be the nonlinear dynamics model of the i^{th} subsystem subject to the cost given by (6.15), the optimal control update laws δu_i and δh_i for each i^{th} subsystem are calculated iteratively as

$$\delta u_i^* = l_{u_i} + K_{u_i}\delta x_i, \tag{6.22a}$$

$$\delta h_i^* = l_{h_i} + K_{h_i}\delta x_i, \tag{6.22b}$$

where

$$l_{u_i} = -(\bar{P}_{i_{u_iu_i}} - \bar{P}_{i_{u_ih_i}}\bar{P}^{-1}_{i_{h_ih_i}}\bar{P}_{i_{h_iu_i}})^{-1}(\bar{P}_{i_{u_i}} - \bar{P}_{i_{u_ih_i}}\bar{P}^{-1}_{i_{h_ih_i}}\bar{P}_{i_{h_i}}), \tag{6.23a}$$

$$l_{h_i} = -(\bar{P}_{i_{h_ih_i}} - \bar{P}_{i_{h_iu_i}}\bar{P}^{-1}_{i_{u_iu_i}}\bar{P}_{i_{u_ih_i}})^{-1}(\bar{P}_{i_{h_i}} - \bar{P}_{i_{h_iu_i}}\bar{P}^{-1}_{i_{u_iu_i}}\bar{P}_{i_{u_i}}), \tag{6.23b}$$

$$K_{u_i} = -(\bar{P}_{i_{u_iu_i}} - \bar{P}_{i_{u_ih_i}}\bar{P}^{-1}_{i_{h_ih_i}}\bar{P}_{i_{h_iu_i}})^{-1}(\bar{P}_{i_{u_ix_i}} - \bar{P}_{i_{u_ih_i}}\bar{P}^{-1}_{i_{h_ih_i}}\bar{P}_{i_{h_ix_i}}), \tag{6.23c}$$

$$K_{h_i} = -(\bar{P}_{i_{h_ih_i}} - \bar{P}_{i_{h_iu_i}}\bar{P}^{-1}_{i_{u_iu_i}}\bar{P}_{i_{u_ih_i}})^{-1}(\bar{P}_{i_{h_ix_i}} - \bar{P}_{i_{h_iu_i}}\bar{P}^{-1}_{i_{u_iu_i}}\bar{P}_{i_{u_ix_i}}), \tag{6.23d}$$

and

$$\bar{P}_{i_{x_i}} = \bar{F}_{i_x}^T \bar{V}_{i_{x_i}} + \bar{r}_{i_{x_i}}, \tag{6.24a}$$

$$\bar{P}_{i_{u_i}} = \bar{F}_{i_{u_i}}^T \bar{V}_{i_{x_i}} + \bar{r}_{i_{u_i}}, \tag{6.24b}$$

$$\bar{P}_{i_{h_i}} = \bar{F}_{i_{h_i}}^T \bar{V}_{i_{x_i}} + \bar{r}_{i_{h_i}}, \tag{6.24c}$$

$$\bar{P}_{i_{x_i x_i}} = \bar{r}_{i_{x_i x_i}} + \bar{V}_{i_{x_i x_i}} \bar{F}_{i_{x_i}} + \bar{F}_{i_{x_i}}^T \bar{V}_{i_{x_i x_i}}, \tag{6.24d}$$

$$\bar{P}_{i_{u_i u_i}} = \bar{r}_{i_{u_i u_i}}, \tag{6.24e}$$

$$\bar{P}_{i_{h_i h_i}} = \bar{r}_{i_{h_i h_i}}, \tag{6.24f}$$

$$\bar{P}_{i_{u_i x_i}} = \bar{r}_{i_{u_i x_i}} + \bar{F}_{i_{u_i}}^T \bar{V}_{i_{x_i x_i}}, \tag{6.24g}$$

$$\bar{P}_{i_{h_i x_i}} = \bar{r}_{i_{h_i x_i}} + \bar{F}_{i_{h_i}}^T \bar{V}_{i_{x_i x_i}}, \tag{6.24h}$$

$$\bar{P}_{i_{u_i h_i}} = \bar{r}_{i_{u_i h_i}}. \tag{6.24i}$$

In control update laws, K_{u_i} and l_{u_i} represent the state-feedback gain and feedforward term for the stabilizing control u_i, while K_{h_i} and l_{h_i} represent the state-feedback gain and feedforward term for the adversarial disturbance input h_i, which models the worst-case influence of mismatched interconnections in the Min-Max game formulation. This structure ensures that the control can robustly stabilize the subsystem under worst-case interconnection effects.

Proof. Consider the Taylor series, extend the left side of the (6.17)

$$\frac{\partial V_i(x_i, t)}{\partial t} = \frac{\partial V_i(\bar{x}_i + \delta x_i, t)}{\partial t} \approx \frac{\partial \bar{V}_i}{\partial t} + \frac{\partial \bar{V}_{i_{x_i}}^T}{\partial t} \delta x_i + \frac{1}{2} \delta x_i^T \frac{\partial \bar{V}_{i_{x_i x_i}}^T}{\partial t} \delta x_i, \tag{6.25}$$

we have

$$\frac{\partial \bar{V}_i}{\partial t} = \frac{d\bar{V}_i}{dt} - \bar{V}_{i_{x_i}}^T \bar{F}_i, \tag{6.26a}$$

$$\frac{\partial \bar{V}_{i_{x_i}}}{\partial t} = \frac{d\bar{V}_{i_{x_i}}}{dt} - \bar{V}_{i_{x_i x_i}}^T \bar{F}_i, \tag{6.26b}$$

$$\frac{\partial \bar{V}_{i_{x_i x_i}}}{\partial t} = \frac{d\bar{V}_{i_{x_i x_i}}}{dt} - \sum_{p=1}^{n} \bar{V}_{i_{x_i x_i x_i}}^{(p)} \bar{F}_i^{(p)}, \tag{6.26c}$$

the left hand side of (6.17) becomes

$$\begin{aligned} -\frac{\partial V_i(x_i,t)}{\partial t} \approx & -\frac{d\bar{V}_i}{dt} - \frac{d\bar{V}_{i_{x_i}}^T}{dt}\delta x_i - \frac{1}{2}\delta x_i^T \frac{d\bar{V}_{i_{x_i x_i}}^T}{dt}\delta x_i \\ & + \bar{V}_{i_{x_i}}^T \bar{F}_i + \delta x^T \bar{V}_{i_{x_i x_i}} \bar{F}_i + \frac{1}{2}\delta x_i^T \left(\sum_{p=1}^{n} \bar{V}_{i_{x_i x_i x_i}}^{(p)} \bar{F}_i^{(p)} \right) \delta x_i. \end{aligned} \tag{6.27}$$

The extension of the right hand side of (6.17) by Taylor series proposes

$$\bar{r}_i \approx \bar{r}_i + \bar{r}_{i_{x_i}}^T \delta x_i + \bar{r}_{i_{u_i}}^T \delta u_i + \bar{r}_{i_{h_i}}^T \delta h_i + \frac{1}{2} \begin{bmatrix} \delta x_i \\ \delta u_i \\ \delta h_i \end{bmatrix}^T \begin{bmatrix} \bar{r}_{i_{x_i x_i}} & \bar{r}_{i_{x_i u_i}} & \bar{r}_{i_{x_i h_i}} \\ \bar{r}_{i_{u_i x_i}} & \bar{r}_{i_{u_i u_i}} & \bar{r}_{i_{u_i h_i}} \\ \bar{r}_{i_{h_i x_i}} & \bar{r}_{i_{h_i u_i}} & \bar{r}_{i_{h_i h_i}} \end{bmatrix} \begin{bmatrix} \delta x_i \\ \delta u_i \\ \delta h_i \end{bmatrix}. \tag{6.28}$$

The extension of $V_{i_{x_i}}$ around $\bar{x}_i$ yields

$$V_{ix_i} \approx \bar{V}_{i_{x_i}} + \bar{V}_{i_{x_i x_i}} \bar{\delta} x_i + \frac{1}{2}\bar{W}_i, \tag{6.29}$$

where $\bar{W}_i = [\bar{W}_i^p]$ is such that

$$\bar{W}_i^p = \delta x_i^T \bar{V}_{i_{x_i x_i x_i}}^{(p)} \delta x_i, \qquad p = 1, ..., n. \tag{6.30}$$

The expansion of the dynamics of the system is obtained as

$$\bar{F}_i(x_i, u_i, h_i) = \bar{F}_i + \bar{F}_{i_{x_i}}\delta x_i + \bar{F}_{i_{u_i}}\delta u_i + \bar{F}_{i_{h_i}}\delta h_i. \tag{6.31}$$

The right hand side of (6.17) can be

$$\begin{aligned}
&\Bigg\{\bar{r}_i + \bar{r}^T_{i_{x_i}}\delta x_i + \bar{r}^T_{i_{u_i}}\delta u_i + \bar{r}^T_{i_{h_i}}\delta h_i + \frac{1}{2}\begin{bmatrix}\delta x_i\\ \delta u_i\\ \delta h_i\end{bmatrix}^T \begin{bmatrix}\bar{r}_{i_{x_ix_i}} & \bar{r}_{i_{x_iu_i}} & \bar{r}_{i_{x_ih_i}}\\ \bar{r}_{i_{u_ix_i}} & \bar{r}_{i_{u_iu_i}} & \bar{r}_{i_{u_ih_i}}\\ \bar{r}_{i_{h_ix_i}} & \bar{r}_{i_{h_iu_i}} & \bar{r}_{i_{h_ih_i}}\end{bmatrix}\begin{bmatrix}\delta x_i\\ \delta u_i\\ \delta h_i\end{bmatrix}\\
&+ \bar{V}^T_{i_{x_i}}\bar{F}_i + \bar{V}^T_{i_{x_i}}\bar{F}_{i_{x_i}}\delta x_i + \bar{V}^T_{i_{x_i}}\bar{F}_{i_{u_i}}\delta u_i + \bar{V}^T_{i_{x_i}}\bar{F}_{i_{h_i}}\delta h_i + \delta x_i^T\bar{V}_{i_{x_ix_i}}\bar{F}_{i_{x_i}}\delta x_i\\
&+ \delta x_i^T\bar{V}_{i_{x_ix_i}}\bar{F}_{i_{u_i}}\delta u_i + \delta x_i^T\bar{V}_{i_{x_ix_i}}\bar{F}_{i_{h_i}}\delta h_i + \delta x_i^T\bar{V}_{i_{x_ix_i}}\bar{F}_i + \frac{1}{2}\bar{W}_i^T\bar{F}_i\Bigg\},
\end{aligned} \tag{6.32}$$

next, the term $\frac{1}{2}\bar{W}_i^T\bar{F}_i$ can be written as follows:

$$\frac{1}{2}\bar{W}_i^T\bar{F}_i = \frac{1}{2}\delta x_i^T\Bigg(\sum_{p=1}^{n}\bar{V}^{(p)}_{i_{x_ix_ix_i}}\bar{F}_i^{(p)}\Bigg)\delta x_i. \tag{6.33}$$

By equating the left hand side and right hand side of the expansion of (6.17) and doing mathematical operations we have

$$\begin{aligned}
&-\frac{d\bar{V}_i}{dt} - \delta x_i^T\frac{d\bar{V}_{i_{x_i}}}{dt} - \frac{1}{2}\delta x_i^T\frac{d\bar{V}_{i_{x_ix_i}}}{dt}\delta x_i = \min_{\delta u_i}\max_{\bar{\delta} h_i}\Bigg\{\bar{r}_i + \delta x_i^T\bar{P}_{i_{x_i}} + \delta u_i^T\bar{P}_{i_{u_i}} + \delta h_i^T\bar{P}_{i_{h_i}}\\
&+ \frac{1}{2}\delta x_i^T\bar{P}_{i_{x_ix_i}}\delta x_i + \frac{1}{2}\delta u_i^T\bar{P}_{i_{u_iu_i}}\delta u_i + \frac{1}{2}\delta h_i^T\bar{P}_{i_{h_ih_i}}\delta h_i + \delta u_i^T\bar{P}_{i_{u_ix_i}}\delta x_i\\
&+ \delta h_i^T\bar{P}_{i_{h_ix_i}}\delta x_i + \delta u_i^T\bar{P}_{i_{u_ih_i}}\delta h_i\Bigg\}.
\end{aligned} \tag{6.34}$$

By taking partial derivative of (6.34) with respect to δu_i and δh_i and make them equal

to 0, the optimal control update δu_i^* and δh_i^* for each subsystem are calculated as

$$\delta u_i^* = -\bar{P}_{i_{u_i u_i}}^{-1}(\bar{P}_{i_{u_i x_i}}\delta x_i + \bar{P}_{i_{u_i h_i}}\delta h_i^* + \bar{P}_{i_{u_i}}), \tag{6.35a}$$

$$\delta h_i^* = -\bar{P}_{i_{h_i h_i}}^{-1}(\bar{P}_{i_{h_i x_i}}\delta x_i + \bar{P}_{i_{h_i u_i}}\delta u_i^* + \bar{P}_{i_{h_i}}), \tag{6.35b}$$

if we eliminate δu_i^* and δh_i^* from the right hand side of (6.35)

$$\delta u_i^* = l_{u_i} + K_{u_i}\delta x_i, \tag{6.36a}$$

$$\delta h_i^* = l_{h_i} + K_{h_i}\delta x_i. \tag{6.36b}$$

This, completes the proof. □

Remark 15. *To make sure that the convergence is satisfied, the parameter* $0 < \gamma_i \leq 1$ *is added to the feedforward gains, which can be obtained through a line search. Hence, (6.22) is written as*

$$\delta u_i^* = \gamma_i l_{u_i} + K_{u_i}\delta x_i, \tag{6.37a}$$

$$\delta h_i^* = \gamma_i l_{h_i} + K_{h_i}\delta x_i. \tag{6.37b}$$

Notice that the time history of $V_{i_{x_i}}$ and $V_{i_{x_i x_i}}$ are required to calculate the terms in (6.24). To this end, we substitute the control update laws (6.22) into (6.17) to obtain the update laws of the value function and its first order and second order state derivatives, and the results are presented in the following theorem.

Theorem 10. *The backward ordinary differential equations for the value function and*

its first order and second order partial derivatives with respect to x_i are computed as

$$
\begin{aligned}
-\frac{d\bar{V}_i}{dt} &= \bar{r}_i + l_{u_i}^T \bar{P}_{i_{u_i}} + l_{h_i}^T \bar{P}_{i_{h_i}} + \frac{1}{2} l_{u_i} \bar{P}_{i_{u_i u_i}} l_{u_i} \\
&+ l_{u_i}^T \bar{P}_{i_{u_i h_i}} l_{h_i} + \frac{1}{2} l_{h_i}^T \bar{P}_{i_{h_i h_i}} l_{h_i}, \qquad (6.38a) \\
-\frac{d\bar{V}_{i_{x_i}}}{dt} &= \bar{P}_{i_{x_i}} + K_{u_i}^T \bar{P}_{i_{u_i}} + K_{h_i}^T \bar{P}_{i_{h_i}} + \bar{P}_{i_{u_i x_i}}^T l_{u_i} \\
&+ \bar{P}_{i_{h_i x_i}}^T l_{h_i} + K_{u_i}^T \bar{P}_{i_{u_i u_i}} l_{u_i} + K_{u_i}^T \bar{P}_{i_{u_i h_i}} l_{h_i} \\
&+ K_{h_i}^T \bar{P}_{i_{h_i u_i}} l_{u_i} + K_{h_i}^T \bar{P}_{i_{h_i h_i}} l_{h_i}, \qquad (6.38b) \\
-\frac{d\bar{V}_{i_{x_i x_i}}}{dt} &= K_{u_i}^T \bar{P}_{i_{u_i x_i}} + \bar{P}_{i_{u_i x_i}}^T K_{u_i} + K_{h_i}^T \bar{P}_{i_{h_i x_i}} \\
&+ \bar{P}_{i_{h_i x_i}}^T K_{h_i} + K_{h_i}^T \bar{P}_{i_{h_i u_i}} K_{u_i} + K_{u_i}^T \bar{P}_{i_{u_i h_i}} K_{h_i} \\
&+ K_{u_i}^T \bar{P}_{i_{u_i u_i}} K_{u_i} + K_{h_i}^T \bar{P}_{i_{h_i h_i}} K_{h_i} + \bar{P}_{i_{x_i x_i}}, \qquad (6.38c)
\end{aligned}
$$

and the terminal conditions are given by

$$
\begin{aligned}
\bar{V}_i(t_f) &= \Gamma_i(\bar{x}_i(t_f), t_f), \qquad (6.39a) \\
\bar{V}_{i_{x_i}}(t_f) &= \Gamma_{i x_i}(\bar{x}_i(t_f), t_f), \qquad (6.39b) \\
\bar{V}_{i_{x_i x_i}}(t_f) &= \Gamma_{i x_i x_i}(\bar{x}_i(t_f), t_f). \qquad (6.39c)
\end{aligned}
$$

Proof. To obtain the update law, its first- and second-order partial derivatives of the

value function, put the update control laws (6.22) to (6.17). We have

$$
\begin{aligned}
&-\frac{dV_i}{dt}-\delta x_i^T\frac{dV_{i_{x_i}}}{dt}-\frac{1}{2}\delta x_i^T\frac{dV_{i_{x_ix_i}}}{dt}\delta x_i = r_i+\delta x_i^T P_{i_{x_i}}\\
&+\delta u_i^{*^T} P_{i_{u_i}}+\delta h_i^{*^T} P_{i_{h_i}}+\delta u_i^{*^T} P_{i_{u_ix_i}}\delta x_i+\frac{1}{2}\delta u_i^{*^T} P_{i_{u_iu_i}}\delta u_i^*\\
&+\delta u_i^{*^T} P_{i_{u_ih_i}}\delta h_i^*+\frac{1}{2}\delta h_i^{*^T} P_{i_{h_ih_i}}\delta h_i^*+\delta h_i^{*^T} P_{i_{h_ix_i}}\delta x_i\\
&+\frac{1}{2}\delta x_i^T P_{i_{x_ix_i}}\delta x_i^T = r_i+\delta x_i^T P_{i_{x_i}}+(l_{u_i}+K_{u_i}\delta x_i)^T P_{i_{u_i}}\\
&+(l_{h_i}+K_{h_i}\delta x_i)^T P_{i_{h_i}}+(l_{u_i}+K_{u_i}\delta x_i)^T P_{i_{u_ix_i}}\delta x_i\\
&+(l_{h_i}+K_{h_i}\delta x_i)^T P_{i_{h_ix_i}}\delta x_i+\frac{1}{2}\delta x_i^T P_{i_{x_ix_i}}\delta x_i\\
&+(l_{u_i}+K_{u_i}\delta x_i)^T P_{i_{u_ih_i}}(l_{h_i}+K_{h_i}\delta x_i)\\
&+\frac{1}{2}(l_{u_i}+K_{u_i}\delta x_i)^T P_{i_{u_iu_i}}(l_{u_i}+K_{u_i}\delta x_i)\\
&+\frac{1}{2}(l_{h_i}+K_{h_i}\delta x_i)^T P_{i_{h_ih_i}}(l_{h_i}+K_{h_i}\delta x_i).
\end{aligned}
\tag{6.40}
$$

Next, we collect all terms in the right-hand side of the (6.40) as zeroth-,first-, and second-order expressions of δx_i. Now, put the coefficients of δx_i in the left-hand side and right-hand side of the (6.40) and compute the backward propagation equations with respect to the value function and its first- and second-order partial derivatives as

$$
\begin{aligned}
-\frac{dV_i}{dt} &= r_i + \epsilon_i l_{u_i}^T P_{i_{u_i}} + \epsilon_i l_{h_i}^T P_{i_{h_i}} + \frac{1}{2}\epsilon_i^2 l_{u_i} P_{i_{u_i u_i}} l_{u_i} \\
&+ \epsilon_i^2 l_{u_i}^T P_{i_{u_i h_i}} l_{h_i} + \frac{1}{2}\epsilon_i^2 l_{h_i}^T P_{i_{h_i h_i}} l_{h_i}, \qquad (6.41a) \\
-\frac{dV_{i_{x_i}}}{dt} &= P_{i_{x_i}} + K_{u_i}^T P_{i_{u_i}} + K_{h_i}^T P_{i_{h_i}} + P_{i_{u_i x_i}}^T l_{u_i} \\
&+ P_{i_{h_i x_i}}^T l_{h_i} + K_{u_i}^T P_{i_{u_i u_i}} l_{u_i} + K_{u_i}^T P_{i_{u_i h_i}} l_{h_i} \\
&+ K_{h_i}^T P_{i_{h_i u_i}} l_{u_i} + K_{h_i}^T P_{i_{h_i h_i}} l_{h_i}, \qquad (6.41b) \\
-\frac{dV_{i_{x_i x_i}}}{dt} &= K_{u_i}^T P_{i_{u_i x_i}} + P_{i_{u_i x_i}}^T K_{u_i} + K_{h_i}^T P_{i_{h_i x_i}} \\
&+ P_{i_{h_i x_i}}^T K_{h_i} + K_{h_i}^T P_{i_{h_i u_i}} K_{u_i} + K_{u_i}^T P_{i_{u_i h_i}} K_{h_i} \\
&+ K_{u_i}^T P_{i_{u_i u_i}} K_{u_i} + K_{h_i}^T P_{i_{h_i h_i}} K_{h_i} + P_{i_{x_i x_i}}. \qquad (6.41c)
\end{aligned}
$$

At the terminal time, we have the terminal condition $V_i(x_i(t_f), t_f) = \Gamma_i(x_i(t_f), t_f)$. If the Taylor series is used to expand the around $x_i(t_f)$, we obtain the terms as follow

$$
\begin{aligned}
&\Gamma_i(x_i(t_f), t_f) = \Gamma_i(x_i(t_f) + \delta x_i(t_f), t_f) \approx \Gamma_i(x_i(t_f), t_f) + \delta x_i(t_f)^T \Gamma_{x_i}(x_i(t_f), t_f) \\
&+ \delta x_i(t_f)^T \Gamma_{x_i x_i}(x_i(t_f), t_f) \delta x_i(t_f). \qquad (6.42)
\end{aligned}
$$

Henceforth, the terminal conditions at t_f for the backward differential equations are obtain as

$$
\begin{aligned}
V_i(t_f) &= \Gamma_i(x_i(t_f), t_f), \qquad (6.43a) \\
V_{i_{x_i}}(t_f) &= \Gamma_{i x_i}(x_i(t_f), t_f), \qquad (6.43b) \\
V_{i_{x_i x_i}}(t_f) &= \Gamma_{i x_i x_i}(x_i(t_f), t_f). \qquad (6.43c)
\end{aligned}
$$

And this, completes the proof. □

In the next section, the convergence of the Min-Max DDP is discussed. The following Assumption (9) is introduced to facilitate subsequent analyses

Assumption 9. *It is assumed that the running cost $\bar{r}_i(x_i(t), u_i(t), h_i(t))$ does not contain a cross term that includes both u_i and h_i. That is,*

$$\bar{P}_{i_{u_i h_i}} = \bar{P}_{i_{h_i u_i}} = 0. \tag{6.44}$$

This assumption is valid in many applications of differential games between two entities such as pursuit evasion of two vehicles, where the controls affect the dynamics separately and no cross terms of the controls would show up in the cost. The following theorem summarizes the convergence of the Min-Max DDP algorithm

Theorem 11. *If Assumption 9 holds and the following condition is satisfied*

$$\bar{P}^T_{i_{u_i}} \bar{P}^{-1}_{i_{u_i u_i}} \bar{P}_{i_{u_i}} + \bar{P}^T_{i_{h_i}} \bar{P}^{-1}_{i_{h_i h_i}} \bar{P}_{i_{h_i}} > 0, \quad t \in [t_0, t_f], \tag{6.45}$$

then, the total cost J_i and the control laws u_i and h_i converge to their optimal solutions at the end of the iteration.

Proof. First, we discretize the time interval $[t_0, t_f]$ into $t_0 < t_1 < ... < t_{N-1} < t_N = t_f$, and $dt = t_{k+1} - t_k$, for time step $k = 0, ..., N-1$. The gains will be

$$l_{u_i k} = -P^{-1}_{i_{u_i u_i} k} P_{i_{u_i} k}, \tag{6.46a}$$

$$l_{h_i k} = -P^{-1}_{i_{h_i h_i} k} P_{i_{h_i} k}, \tag{6.46b}$$

$$K_{u_i k} = -P^{-1}_{i_{u_i u_i} k} P_{i_{u_i x_i} k}, \tag{6.46c}$$

$$K_{h_i k} = -P^{-1}_{i_{h_i h_i} k} P_{i_{h_i x_i} k}, \tag{6.46d}$$

we obtain the following equation by using (6.46) into the first equation of (6.38) in discrete time

$$V_{ik} - V_{ik+1} = r_{ik}dt + (\frac{1}{2}\epsilon_i^2 - \epsilon_i)\eta_{ik}dt, \quad V_{iN} = \Gamma_i(x_{iN}), \tag{6.47}$$

where $\eta_{ik} = P^T_{i_{u_i}k} P^{-1}_{i_{u_i u_i}k} P_{i_{u_i}k} + P^T_{i_{h_i}k} P^{-1}_{i_{h_i h_i}k} P_{i_{h_i}k}$. The total cost at iteration j with respect to the nominal state and control trajectories $(\bar{x}_i, \bar{u}_i, \bar{h}_i)$ is shown as

$$J_i^{(j)} = \Gamma_i(\bar{x}_{iN}) + \sum_{k=0}^{N-1} r_{ik}(\bar{x}_{ik}, \bar{u}_{ik}, \bar{h}_{ik})dt. \tag{6.48}$$

The total cost $J_i^{(j+1)}$ at iteration $(j+1)$ after control update is equal to V_{i0}, so that

$$\begin{aligned} J_i^{(j+1)} &= V_{i0} = V_{i1} + r_{i0}(\bar{x}_{i0}, \bar{u}_{i0}, \bar{h}_{i0})dt + (\frac{1}{2}\epsilon_i^2 - \epsilon_i)\eta_0 dt \\ &= J_i^{(j)} + (\frac{1}{2}\epsilon_i^2 - \epsilon_i) \sum_{k=0}^{N-1} \eta_{ik} dt, \end{aligned} \tag{6.49}$$

as $0 < \epsilon_i \leq 1$, we can write $\frac{1}{2}\epsilon_i^2 - \epsilon_i \leq -\frac{1}{2}$. The cost deviation in the discrete time zone at iteration j is written as

$$\begin{aligned} \Delta J_i^{(j)} &= J_i^{(j+1)} - J_i^{(j)} = (\frac{1}{2}\epsilon_i^2 - \epsilon_i) \sum_{k=0}^{N-1} \eta_{ik} dt \leq -\frac{1}{2}\epsilon_i \sum_{k=0}^{N-1} \eta_{ik} dt \\ &= -\frac{1}{2}\epsilon_i \sum_{k=0}^{N-1} (P^T_{i_{u_i}k} P^{-1}_{i_{u_i u_i}k} P_{i_{u_i}k} + P^T_{i_{h_i}k} P^{-1}_{i_{h_i h_i}k} P_{i_{h_i}k}) < 0, \end{aligned} \tag{6.50}$$

which means

$$\lim_{j \to \infty} \Delta J_i^{(j)} = 0, \tag{6.51}$$

and therefore

$$\lim_{j\to\infty} J_i^{(j)} = J_i(u_i^*, h_i^*), \tag{6.52}$$

where u_i^* and h_i^* are optimal minimizing and maximizing controls, respectively. Here, it is proved that in the Min-Max DDP algorithm, u_i and h_i converge to their optimal value and the total cost J_i decreases monotonically during the iteration under the theorem (11). This completes the proof. □

Remark 16. *From the Theorem 11, it is shown that the total cost J_i and control laws u_i and h_i converge to their optimal value at the end of the iteration. In (6.45), $P_{i_{u_i}}^T P_{i_{u_i u_i}}^{-1} P_{i_{u_i}}$ and $P_{i_{h_i}}^T P_{i_{h_i h_i}}^{-1} P_{i_{h_i}}$ indicate the dependence of the dynamics and running cost on u_i and h_i, respectively. For illustration purpose, consider a quadratic running cost $r_i(x_i, u_i, h_i) = q_i(x_i) + u_i^T R_{u_i} u_i - h_i^T R_{h_i} h_i$, where R_{u_i} and R_{h_i} are positive definite, then $P_{i_{u_i u_i}}^{-1} = R_{u_i}^{-1}$ is positive definite and $P_{i_{h_i h_i}}^{-1} = -R_{h_i}^{-1}$ is negative definite. Hence, $P_{i_{u_i}}^T P_{i_{u_i u_i}}^{-1} P_{i_{u_i}} \geq 0$ and $P_{i_{h_i}}^T P_{i_{h_i h_i}}^{-1} P_{i_{h_i}} \leq 0$. Thus, (6.45) indicates that the control authority of u_i is more than that of h_i. This implies that u_i can bring the states to their desired value despite the best effort of h_i, which makes (6.45) a reasonable condition.*

The distributed Min-Max DDP algorithm for the large-scale system is presented in Algorithm 7. We introduce the following assumption to analyze the stability of the proposed distributed Min-Max DDP algorithm.

Assumption 10. *For the quadratic form of the local cost function $J_i(x_i, u_i, h_i)$, consider the running cost $r_i(x_i, u_i, h_i) = q_i(x_i) + u_i^T R_{u_i} u_i - h_i^T R_{h_i} h_i$, where R_{u_i} and R_{h_i} are positive definite. It is assumed that $u_i^{*T} R_{u_i} u_i^* \geq h_i^{*T} R_{h_i} h_i^*$ where u_i^* and h_i^* represent the optimal controls, and $q_i(x_i) \geq c_i \parallel x_i \parallel^2$.*

Theorem 12. *Consider N auxiliary subsystems and the associated value functions given by (6.13) and (6.15), respectively. The objective is to stabilize the system. If*

Assumption (9) and Assumption (10) hold, then for any $\varepsilon_i > 0$, there exists a large enough t_f, such that for the closed-loop system under optimal controls, $\|x_i(t_f)\| < \varepsilon_i$, $i = 1, \ldots, N$.

Proof. Consider the following Lyapunov function candidate for the large-scale system as

$$V(x) = \sum_{i=1}^{N} V_i^*(x_i, t), \tag{6.53}$$

where $V_i^*(x_i, t)$ is the optimal value function defined in (6.16). For each subsystem, $V_i^*(x_i, t) > 0 \ \forall x_i \neq 0$ and $V_i^*(x_i, t) = 0 \ \forall x_i = 0$. Thus, $V(x)$ is positive semi-definite. Differentiating the Lyapunov function candidate with respect to the time variable t, we obtain

$$\dot{V}(x) = \sum_{i=1}^{N} (\frac{\partial V_i^*}{\partial x_i} \frac{dx_i}{dt} + \frac{\partial V_i^*}{\partial t}). \tag{6.54}$$

From the local HJBI (6.17) for subsystem i^{th}, we have

$$-\frac{\partial V_i}{\partial t} = r_i^* + V_{ix_i}^T F_i^*, \tag{6.55}$$

thus,

$$-\sum_{i=1}^{N} r_i^* = \sum_{i=1}^{N} (\frac{\partial V_i}{\partial t} + \frac{\partial V_i}{\partial x_i} \frac{dx_i}{dt}) = \dot{V}(x). \tag{6.56}$$

Applying optimal controls and considering the Assumption 10, we obtain the following

equation

$$\sum_{i=1}^{N} r_i(x_i, u_i, h_i) = \sum_{i=1}^{N} (q_i(x_i) + u_i^{*T} R_{u_i} u_i^* - h_i^{*T} R_{h_i} h_i^*) \geq \sum_{i=1}^{N} c_i \parallel x_i \parallel^2 . \tag{6.57}$$

Therefore,

$$\dot{V}(x) \leq -\sum_{i=1}^{N} c_i \parallel x_i \parallel^2, \quad t_f \geq t_0. \tag{6.58}$$

Integrating both sides of the (6.58) over the time interval $[t_0, \infty)$, it follows

$$\int_{t_0}^{\infty} \parallel x_i(t) \parallel^2 dt \leq \frac{1}{c_i} \int_{t_0}^{\infty} -\dot{V}(x(t)) dt. \tag{6.59}$$

The right-hand side of (6.59) is finite because the Lyapunov function $V(x)$ is bounded from below (non-negative) and decreases monotonically over time dute to (6.58), meaning the total decrease, represented by the integral of $-\dot{V}(X)$, is limited by its initial value $V(x(t_0))$. Thus, the integral converges to a finite value. Since the right-hand side of the (6.59) is finite, using Barbalat's lemma [267], we receive

$$\lim_{t \to \infty} \parallel x_i(t) \parallel = 0, \quad i = 1, 2, ..., N. \tag{6.60}$$

This implies that for any $\varepsilon_i > 0$, there exists a t_f, such that

$$\parallel x_i(t_f) \parallel < \varepsilon_i, \tag{6.61}$$

which completes the proof. □

Remark 17. *The advantage of this Lyapunov function lies in its natural compatibility with the distributed Min-Max DDP structure. Since the DDP framework is used*

to generate the optimal control for each subsystem, the corresponding value function inherently captures the stability and performance characteristics of that subsystem under the optimal policy. Summing these value functions allows us to evaluate the global stability of the overall large-scale system while respecting the decentralized nature of the control design. While there might be other possible candidates for the Lyapunov function, the considered Lyapunov function is a suitable and justified choice since: 1) It is positive definite and zero at the equilibrium point. 2) Its derivative is negative semi-definite due to Assumption 10 and HJBI equation (6.17). 3) It directly links to the cost function.

Remark 18. *Theorem 12 ensures that the state $x_i(t)$ converges to zero as the terminal time t_f increases. This result is established through a Lyapunov-based argument and Barbalat's Lemma, which guarantee asymptotic convergence. However, the analysis does not provide an explicit rate of decay for $||x(t_f)||$. It only guarantees that the final state norm becomes arbitrarily small if t_f is sufficiently large.*

Remark 19. *The assumptions presented in this chapter are essential for ensuring the theoretical soundness and practical tractability of the proposed distributed Min-Max DDP framework. Assumption 1, which imposes smoothness on the system dynamics and cost functions, is required to carry out second-order Taylor expansions during the backward pass and ensure convergence of the algorithm. Assumption 2 introduces structural sparsity in the dynamics and cost coupling between subsystems, which enables distributed implementation by reducing computational complexity and allowing for localized updates. Assumption 3 ensures that the local subsystem interactions are well-posed and stable under nominal conditions. Assumption 4 guarantees the existence of a local saddle-point solution in each Min-Max differential game, which is critical to ensure that the inner optimization steps in each subsystem yield valid optimality conditions. Finally, Assumption 5 supports the construction of a global Lyapunov function*

from the local value functions, which is used to analyze the stability of the closed-loop system under the distributed policy. These assumptions are common in the literature on distributed optimization and game-theoretic control, and together they form the basis for the derivation and convergence of the proposed method. Similar structural and stability-related assumptions are also found in related works on distributed control of interconnected systems. For instance, the references [268] and [269] make comparable assumptions regarding the boundedness of coupling terms, admissibility of delay structures, and subsystem interaction topologies to ensure convergence and finite-time performance. Although our work focuses on continuous-time nonlinear dynamics rather than fractional-order systems, the overall modeling philosophy and use of structured assumptions are consistent across these frameworks.

Remark 20. *In the proposed distributed Min-Max DDP framework, each subsystem solves its own local two-player differential game based on its individual cost function and interconnections. For a system composed of N subsystems, the computational cost grows with the number of agents, as each subsystem performs a backward Riccati pass and forward simulation independently. While this increases the overall computational burden for large-scale systems, the structure of the algorithm is inherently parallelizable. Each local optimization problem is decoupled from the others during the backward pass and forward rollout, enabling simultaneous computation across subsystems. This makes the framework highly suitable for implementation on parallel computing platforms, such as GPUs or multi-core CPUs. As a result, despite the increase in complexity with system size, the method remains computationally tractable and scalable to large networks by leveraging modern parallel hardware.*

Algorithm 7 Distributed Min-Max Differential Dynamic Programming

Require: Transforming large-scale system (6.1) to (6.11), for each subsystem, initial value of x_{i0}, initial trajectories of u_i and h_i, terminal time t_f, multiplier γ_i, large integer N, and small value ϵ.

Ensure: Optimal minimizing control u_i^*, gains l_{u_i}, K_{u_i}.

procedure UPDATE CONTROL POLICIES:

Transform the large-scale system (6.1) to large-scale system (6.11) with two conflicting controls, obtain minimizing control law u_i and maximizing control law h_i for each subsystem, applying u_i to large-scale system (6.1).

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

while $|\delta u_i| + |\delta h_i| > \epsilon$ **do**

Compute values of V_i, V_{x_i}, and $V_{x_i x_i}$ at t_f from (6.39);

Integrate (6.38) backward in the time;

Compute l_{u_i}, l_{h_i}, K_{u_i}, and K_{h_i} through (6.23);

Integrate (6.20) by replacing δu_i and δh_i with $\delta u_i = l_{u_i} + K_{u_i}\delta x_i$ and $\delta h_i = l_{h_i} + K_{h_i}\delta x_i$ to get δx_i;

Obtain $\delta u_i = \gamma_i l_{u_i} + K_{u_i}\delta x_i$ and $\delta h_i = \gamma_i l_{h_i} + K_{h_i}\delta x_i$;

Update $u_i = u_i + \gamma_i \delta u_i$ and $h_i = h_i + \gamma_i \delta h_i$;

Set $u_i^* = u_i$ and $h_i^* = h_i$;

Apply the u_i^* to system (6.1);

end while

end for

return u_i^*, I_{u_i}, and K_{u_i};

end procedure

6.5 Numerical Examples

In this section, three practical large-scale systems are studied to show the effectiveness of the proposed framework. In the application of the algorithm, we first transform the dynamics of the large-scale system with mismatched interconnection to dynamic model with two conflicting controls in the form of Min-Max DDP. Then, we solve the proposed algorithm for two players model. After obtaining the control policies u_i and h_i, we apply the control update law u_i to the original large-scale system with mismatched interconnection.

6.5.1 Double Inverted Pendulum

In the first example, the proposed method is applied to the double inverted pendulum with two subsystems described as [270]

$$\begin{cases} \dot{x}_{11} = x_{12}, \\ \dot{x}_{12} = \left(\frac{M_1 gr}{J_1} - \frac{Kr^2}{4J_1}\right) sin(x_{11}) + \frac{Kr}{2J_1}(I-b) + \frac{Kr^2}{4J_1} sin(x_{21}) + \frac{u_1}{J_1}, \end{cases}$$
$$\begin{cases} \dot{x}_{21} = x_{22}, \\ \dot{x}_{22} = \left(\frac{M_2 gr}{J_2} - \frac{Kr^2}{4J_2}\right) sin(x_{21}) + \frac{Kr}{2J_2}(I-b) + \frac{Kr^2}{4J_2} sin(x_{11}) + \frac{u_2}{J_2}, \end{cases} \tag{6.62}$$

where $M_1 = 0.2kg$ and $M_2 = 0.25kg$ are the pendulum end masses, $J_1 = 0.5kg$ and $J_2 = 0.625kg$ are the moments of inertia, $K = 10N/s$ is the spring constant of the connecting spring, $g = 9.81m/s^2$ is the gravitational acceleration, $r = 0.5m$ is the pendulum length, $I = 0.5m$ is the natural length of the spring, and $b = 0.4m$ represents the distance between the pendulum hinges, and assume that the inequality $b < I$ holds. The goal is to bring the states of the system from the initial condition

$[x_{11}, x_{12}, x_{21}, x_{22}] = [\pi, 0, \pi, 0]$ to the terminal condition $[x_{11}, x_{12}, x_{21}, x_{22}] = [0, 0, 0, 0]$, and $t_f = 1$. Observing the expressions of the interconnections given in (6.62), we let $\alpha_1(x_1) = \beta_1(x_1) = \|x_1\|$ and $\alpha_2(x_2) = \beta_2(x_2) = \|x_2\|$. To satisfy Assumption 7, the parameters are designed as follows: $b_{11} = 1, b_{12} = 1, b_{21} = 0.5, b_{22} = 0.5$. Owing to dynamics $g_i(x_i(t))$, $q_i(x_i(t))$ and $k_i(x_i(t))$, we derive that $\|g_i^+(x_i)q_i^+(x_i)q_i(x)\| = 0$. Thus, we choose parameters $c_{il} = 0(i.l = 1, 2)$ to satisfy Assumption 8. For this example, the cost function is

$$J_i = \int_{t_0}^{t_f} (x_i^T Q_i x_i + u_i^T R_{u_i} u_i - h_i^T R_{h_i} h_i) dt, \tag{6.63}$$

and initial control policies are set as $u_i = h_i = 0$, and $\gamma_1 = \gamma_2 = 0.1$. The weights are chosen as $Q_1 = Q_2 = [1, 0; 0, 0.1]$, $R_{u_1} = R_{u_2} = 0.014$, and $R_{h_1} = 0.57$ and $R_{h_2} = 0.40$. The weighting matrices Q_i, R_{u_i}, and R_{h_i} in all examples are selected through simulation-based tuning. Specifically, they are adjusted empirically to balance state regulation and control effort for each subsystem, while also ensuring numerical stability and meaningful interaction across agents.

Fig. 6.1 presents the state trajectories of the double inverted pendulum for both subsystems. As it is demonstrated, the stabilizing update control laws u_1 and u_2 manage to bring the system states to the desired goal despite the uncertain interconnections to the controllers.

The controls $u_i, i = 1, 2$ applied to the large-scale system are illustrated in Fig. 6.2. As shown in Fig. 6.4, the cost per iteration for each subsystem converge towards the optimal cost in 50 iterations.

Fig. 6.3 shows 50 trajectories of the feedback gains K_{u_1} and K_{u_2} over 1 second for the double inverted pendulum, with each trajectory corresponding to one iteration of the Min-Max DDP. The figure illustrates how the gains evolve across iterations

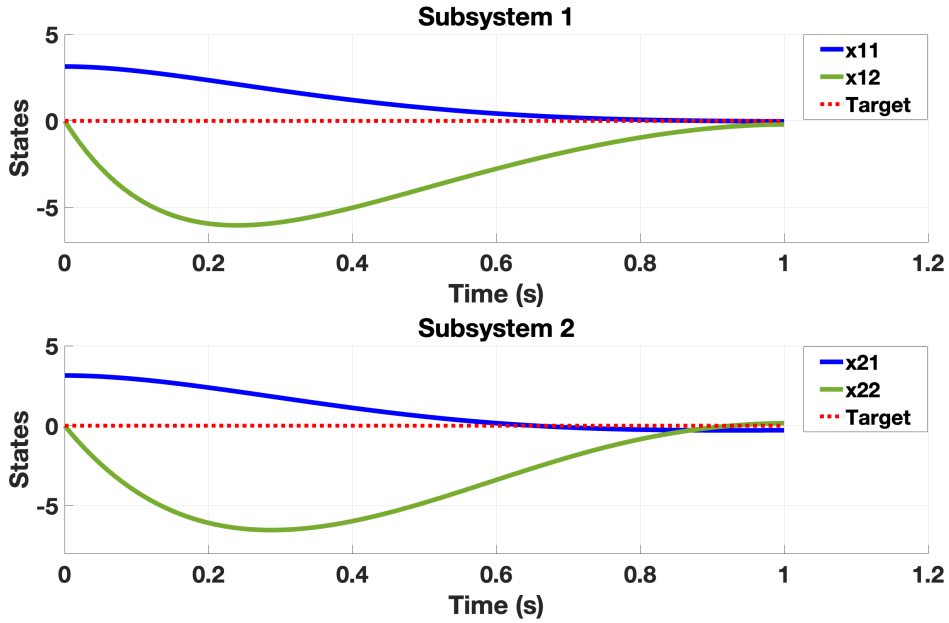


Figure 6.1: State trajectories of double inverted pendulum.

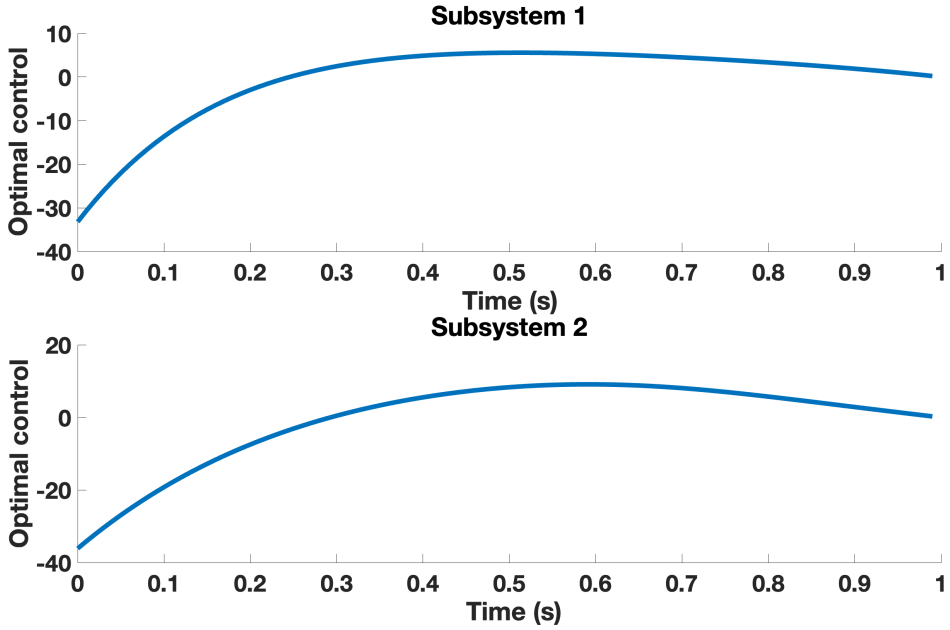


Figure 6.2: Control policies of double inverted pendulum.

and gradually align, indicating that the computed feedback laws are converging and becoming consistent for control implementation.

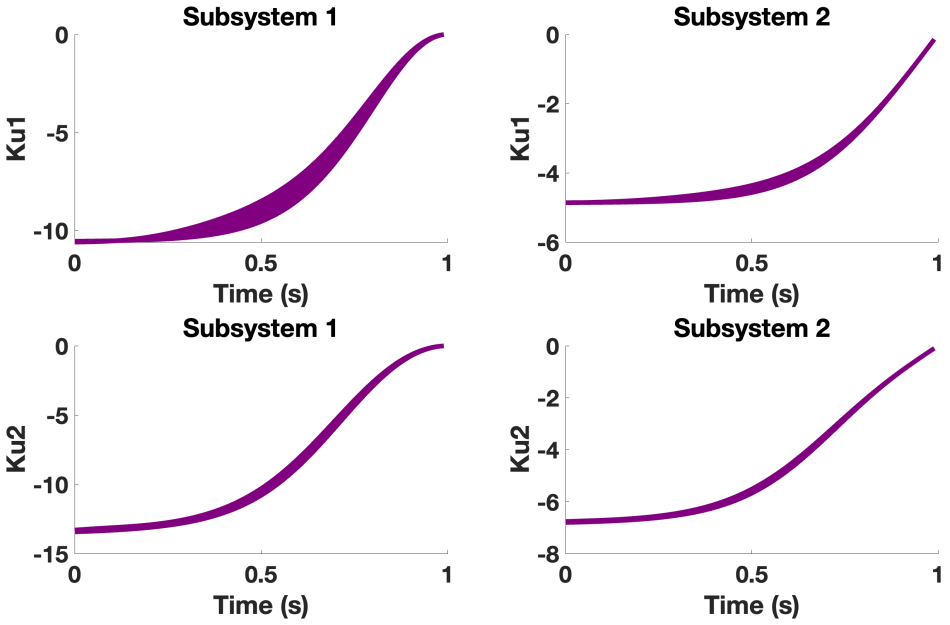


Figure 6.3: Feedback gains for double inverted pendulum.

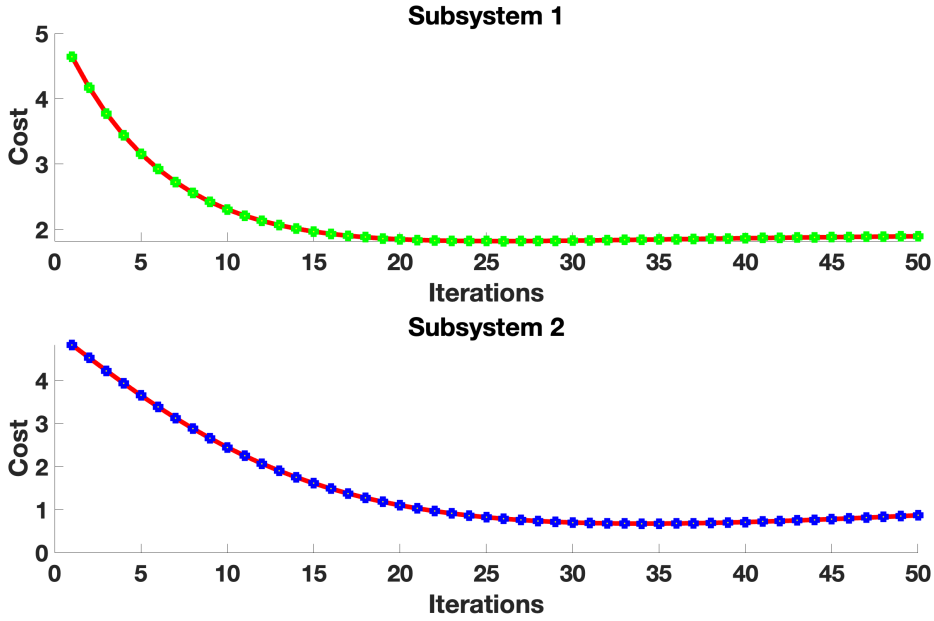


Figure 6.4: Cost per Iteration of double inverted pendulum.

6.5.2 Mass Spring Damper

In the second example, simulation results of the large-scale mass–spring–damper system [271] are presented. The dynamics of the system is given by

$$\begin{cases} M_1\ddot{y}_1 = u_1 - f_{k1}(x) - f_{B1}(x) + f_{k2}(x) + f_{B2}(x) - f_{C1}(x) + f_{C2}(x), \\ M_2\ddot{y}_2 = u_2 - f_{k2}(x) - f_{B1}(x) - f_{C2}(x), \end{cases}$$

where $x = [y_1, \dot{y}_1, y_2, \dot{y}_2]^T$ are the states of the system. $f_{k1}(x) = K_{10}y_1 + \Delta K_1 y_1^3$ and $f_{k2}(x) = K_{20}(y_2 - y_1) + \Delta K_2(y_2 - y_1)^3$ are the spring forces. $f_{B1}(x) = 2\dot{y}_1 + 0.2\dot{y}_1^2$ and $f_{B2}(x) = 2.2(\dot{y}_2 - \dot{y}_1) + 0.15(\dot{y}_2 - \dot{y}_1)^2$ shows the friction forces. $f_{C1}(x) = 0.02sgn(\dot{y}_1)$ and $f_{C2}(x) = 0.02sgn(\dot{y}_2 - \dot{y}_1)$ are the coulomb friction forces. The parameters are provided as $M_1 = 0.25kg$, $M_2 = 0.2kg$, $K_{10} = 1N/m$, and $K_{20} = 2N/m$. The perturbations are given as $\Delta K_1 = 0.1$ and $\Delta K_2 = 0.12$. The goal is to bring the states of the system from $[x_{11}, x_{12}, x_{21}, x_{22}] = [0.5, -2, -1, 2]$ to $[x_{11}, x_{12}, x_{21}, x_{22}] = [0, 0, 0, 0]$, and $t_f = 1$. To satisfy Assumptions 7 and 8, we choose the parameters exactly same as the previous example. For this example, the cost function is

$$J_i = \int_{t_0}^{t_f} (x_i^T Q_i x_i + u_i^T R_{u_i} u_i - h_i^T R_{h_i} h_i) dt, \tag{6.64}$$

and initial control policies are set as $u_i = h_i = 0$, and $\gamma_1 = \gamma_2 = 0.1$. The weights are chosen as $Q_1 = Q_2 = [1, 0; 0, 0.1]$, $R_{u_1} = R_{u_2} = 0.001$, and $R_{h_1} = R_{h_2} = 1$.

Fig. 6.6 presents 100 trajectories of the feedback gains over 1 second for the mass–spring–damper system, with each trajectory representing one iteration of the Min-Max DDP. The gradual alignment of the gain trajectories across iterations demonstrates the convergence of the feedback gains, showing the method reliably computes consistent control policies for the system.

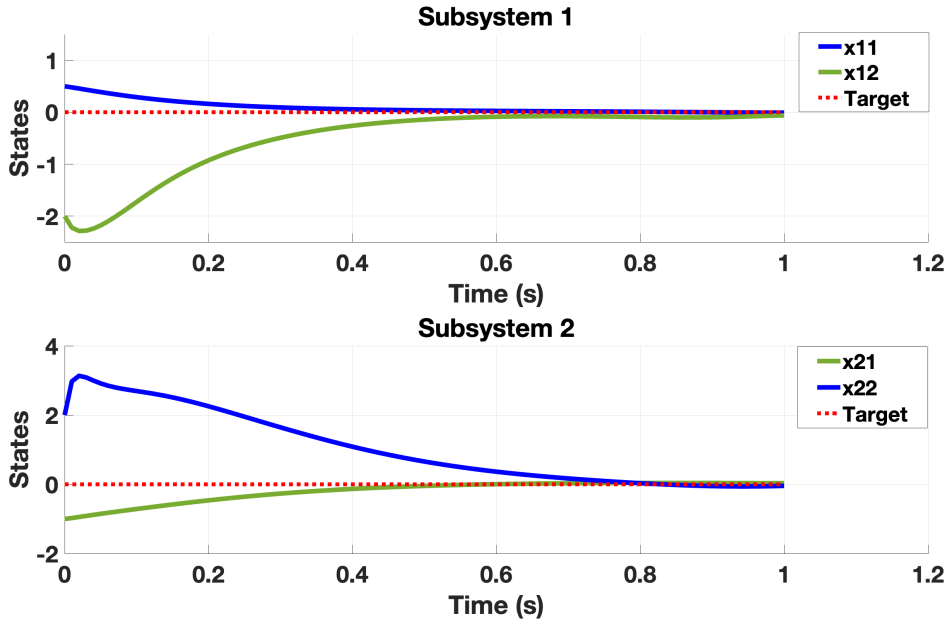


Figure 6.5: State trajectories of mass–spring–damper.

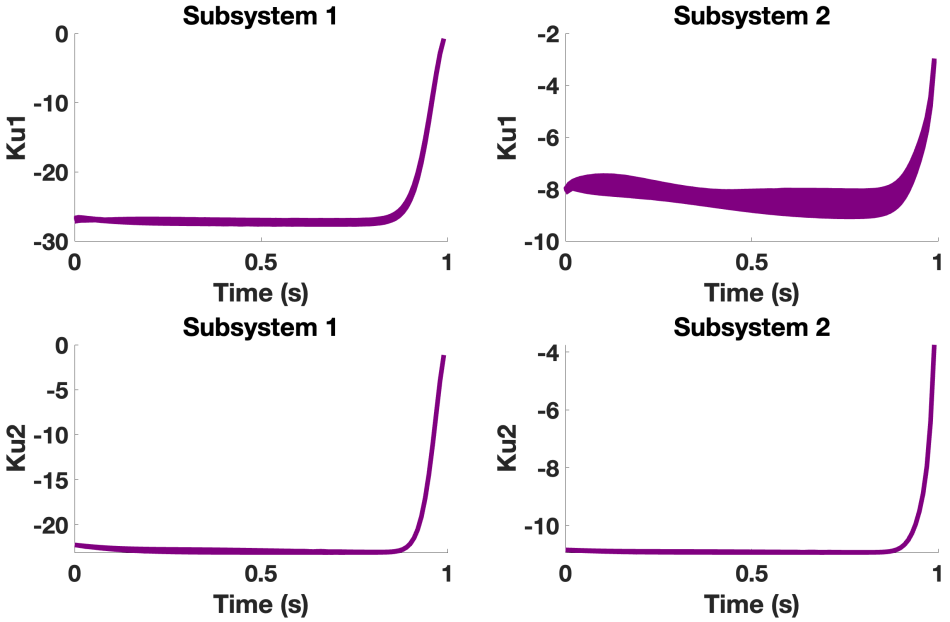


Figure 6.6: Feedback gains for mass-spring-damper.

In Fig. 6.5, the state trajectories of the mass–spring–damper for both subsystems are depicted. The system is effectively guided towards the desired goal through the

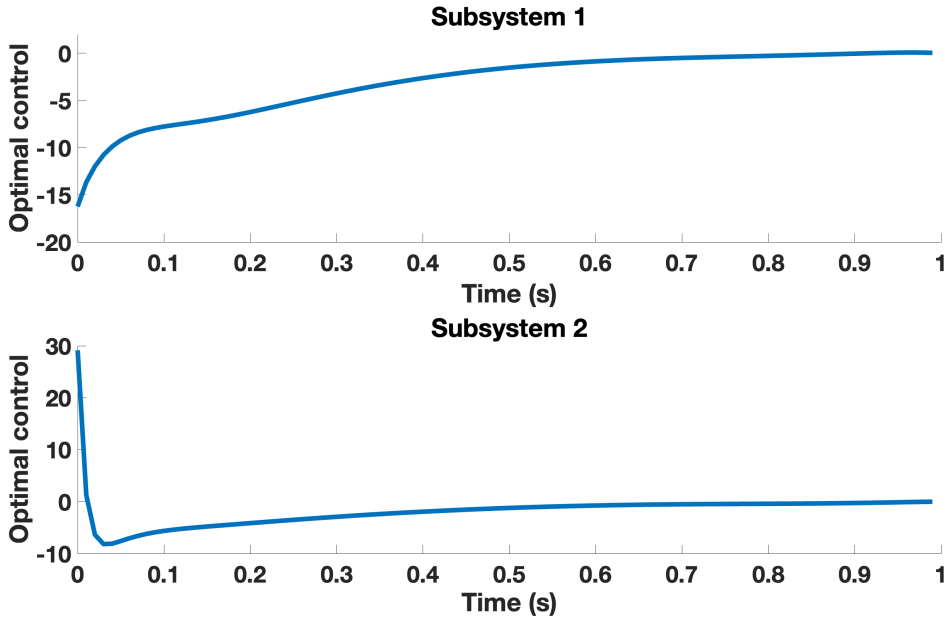


Figure 6.7: Control policies of mass–spring–damper.

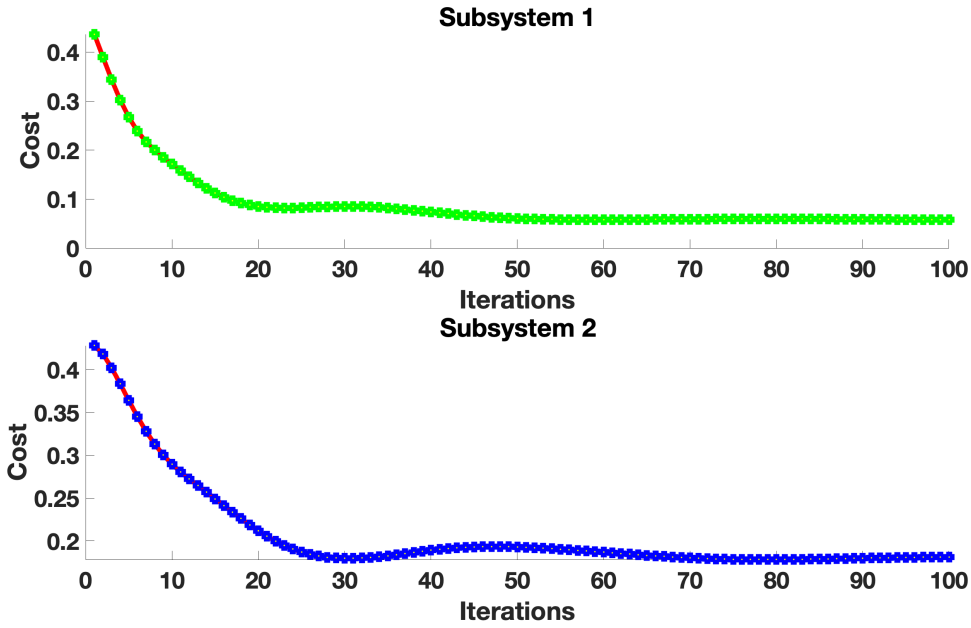


Figure 6.8: Cost per Iteration of double mass–spring–damper.

implementation of stabilizing update control laws u_1 and u_2. The control trajectories for u_1 and u_2 are illustrated in Fig. 6.7. As depicted in Fig. 6.8, the cost per iteration

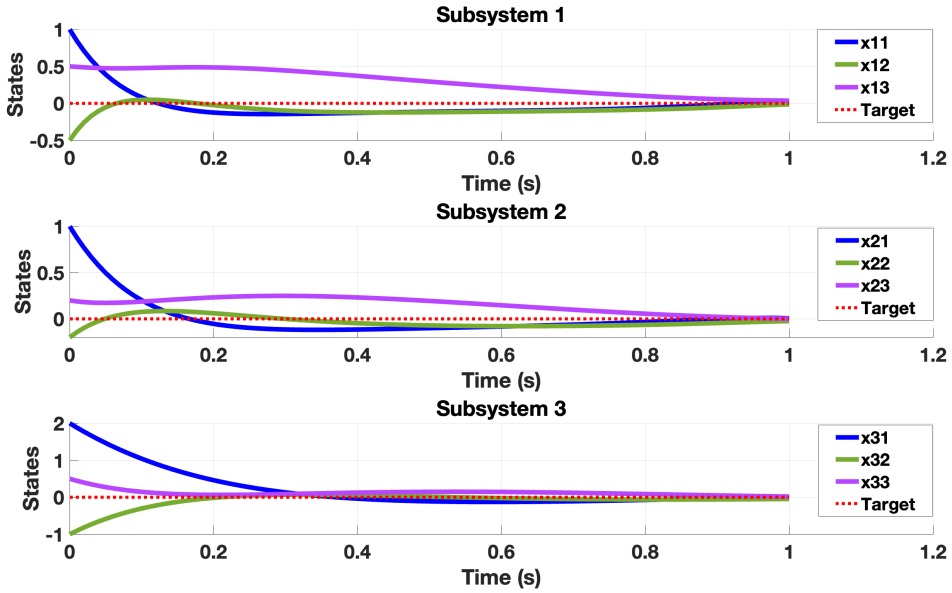


Figure 6.9: State trajectories of power system.

for each subsystem converges towards the optimal cost over the course of 100 iterations.

6.5.3 Power System Dynamics

In the third example, we consider a realistic large-scale power system with three interconnected subsystems [272]. The state-space model of each power station dynamics is given as follows

$$\begin{aligned}
\frac{d(\Delta v_i)}{dt} &= -\frac{1}{T_{g_i}}\Delta v_i + \frac{1}{R_{g_i}T_{g_i}}\Delta f_{G_i} + \frac{1}{T_{g_i}}u_i, \\
\frac{d(\Delta P_{m_i})}{dt} &= \frac{K_{t_i}}{T_{t_i}}\Delta v_i - \frac{1}{T_{t_i}}\Delta P_{m_i}, \\
\frac{d(\Delta f_{G_i})}{dt} &= \frac{K_{p_i}}{T_{p_i}}\Delta P_{m_i} - \frac{\Delta f_{G_i}}{T_{p_i}} - \frac{K_{p_i}}{T_{p_i}}\Delta P_{G_i},
\end{aligned} \tag{6.65}$$

Fig. 6.10 displays 50 trajectories of the feedback gains over 1 second for the power

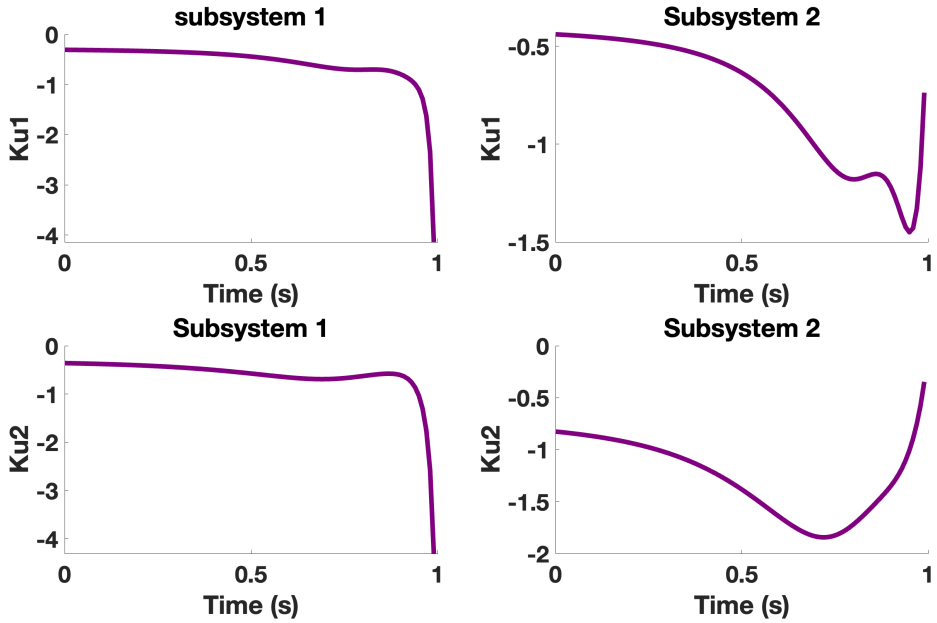


Figure 6.10: Feedback gains for power system.

system example, with each trajectory representing one iteration during the Min-Max DDP. The convergence of these gain profiles across iterations confirms that the proposed method consistently computes stable feedback gains for use in controlling large-scale interconnected power systems.

where $\Delta v_i \in \mathbb{R}, \Delta P_{m_i} \in \mathbb{R}$, and $\Delta f_{G_i} \in \mathbb{R}$, $i = 1, 2, 3$ are incremental change in governor value position, generator output, and frequency deviation, respectively. $u_i \in \mathbb{R}$ is the control input and $\Delta P_{G_i} \in \mathbb{R}$ is the incremental change in the electrical power and is defined as

$$\Delta P_{G_i} = \zeta_i \sin(\Delta P_{m_i} \Delta f_{G_i}), \tag{6.66}$$

where $\zeta_1 = \sum_{i=1}^{3} e_{1i} \Delta v_i$,$\zeta_2 = \sum_{i=1}^{3} e_{2i} \Delta P_{m_i}$, and $\zeta_3 = \sum_{i=1}^{3} e_{3i} \Delta f_{G_i}$. $e_{is}, i, s = 1, 2, 3$, are chosen as $e_{1i} \in [-0.5, 0.5]$, $e_{2i} \in [-0.5, 0.5]$, and $e_{3i} \in [-0.25, 0.25]$. The goal is to bring the states from $[x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}]$

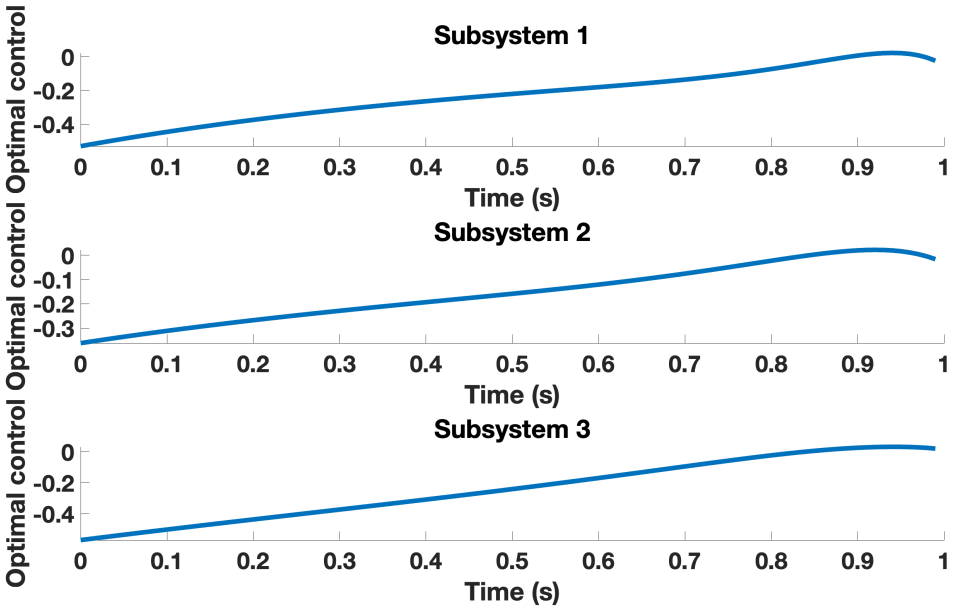


Figure 6.11: Control policies of power system.

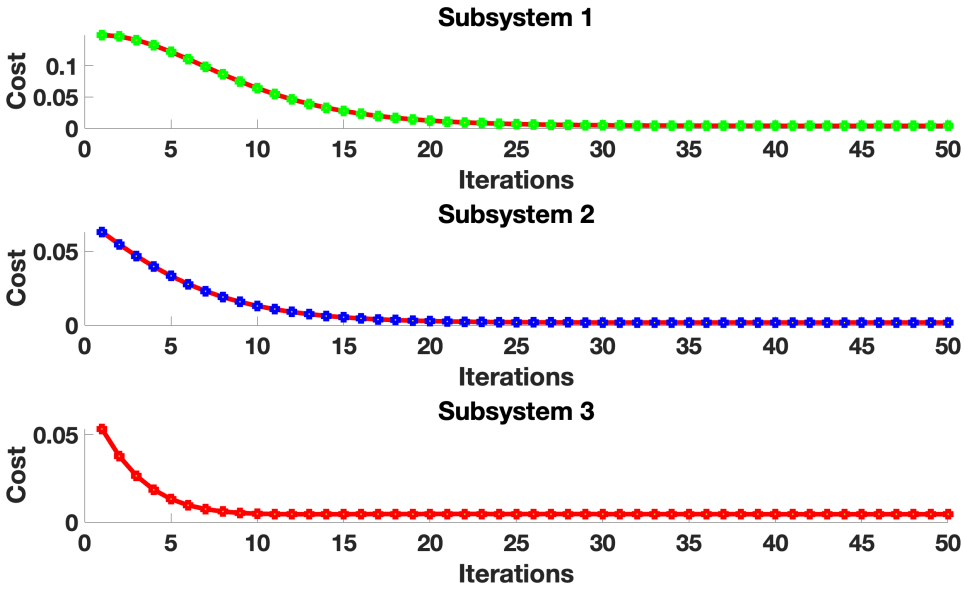


Figure 6.12: Cost per iteration of the three subsystems of power system.

$= [1, -0.5, 0.5, 1, -0.2, 0.2, 2, -1, 0.5]$ to $[0, 0, 0, 0, 0, 0, 0, 0, 0]$ in $t_f = 1$. Thus, to satisfy Assumptions 7 and 8, we design the parameters as follows: $b_{11}, b_{1l} = 0.5(l = 2, 3), b_{21} =$

$0.5, b_{2l} = 0.5(l = 2, 3), b_{31} = 0.5, b_{3l} = 0.25(l = 2, 3)$ and $c_{is} = 0(i, s = 1, 2, 3)$. The cost function is defined as

$$J_i = \int_{t_0}^{t_f} (x_i^T Q_i x_i + u_i^T R_{u_i} u_i - h_i^T R_{h_i} h_i) dt, \tag{6.67}$$

and the initial control policies are set as $u_i = h_i = 0$, and $\gamma_1 = \gamma_2 = \gamma_3 = 0.1$. The weights are chosen as $Q_1 = Q_2 = [1, 0, 0; 0, 0.1, 0; 0, 0, 0.1]$, $R_{u_1} = R_{u_2} = R_{u_3} = 0.1$, and $R_{h_1} = R_{h_2} = R_{h_3} = 5$.

It can be observed in Fig. 6.9 that the states of the three subsystems of the power plant converge to 0 in 1 second. This shows the the proposed method is effective and can be applied to real plants.

In Fig. 6.11, the trajectories of stabilizing controllers $u_i, i = 1, 2, 3$ are illustrated and Fig. 6.12 presents the trajectories of cost per iteration for the three subsystems. Again, the costs for all subsystems converge to the optimal costs.

To benchmark our proposed method, we compare it with three representative approaches in the literature:

Unlike [48], who focused on consensus-based coordination using alternating direction method of multipliers (ADMM) for fully cooperative agents, our method introduces a Min-Max game-theoretic structure that directly addresses robustness to disturbances. While their approach excels in scalability and decentralized computation, it lacks robustness guarantees and does not consider adversarial inputs. Our formulation incorporates mismatched interconnections and disturbance rejection by solving local zero-sum games. Furthermore, we provide Lyapunov-based stability analysis, which is not included in their work.

Paper [273] address robustness in multi-agent systems using tightened constraints and polyhedral over-approximations in a distributed MPC framework, their approach

is tailored for obstacle and collision avoidance in specific robot formations. Our method instead formulates the distributed control problem as a Min-Max differential game, offering robustness to generic disturbances across nonlinear subsystems with mismatched interconnections. Unlike their discrete-time MPC, our approach provides continuous-time convergence analysis and Lyapunov-based guarantees, making it more broadly applicable to large-scale uncertain systems beyond collision avoidance tasks.

[274] proposes a robust distributed MPC framework for frequency and voltage regulation in smart grids using a bilevel structure and dADMM optimization, tailored for deterministic and stochastic uncertainties in grid operations. In contrast, our method addresses robustness through a Min-Max DDP formulation that inherently considers worst adversarial disturbances. Moreover, while their two-stage architecture is problem-specific and designed for secondary control in islanded smart grids, our approach generalizes to arbitrary large-scale nonlinear subsystems with mismatched interconnections.

Chapter 7

Chance Constraint Game-Theoretic Differential Dynamic Programming for Safe Trajectory Optimization Under Uncertainties

7.1 Introduction

Optimal control has been a foundational framework for designing decision-making policies in complex dynamical systems [275]. Classical and modern techniques such as Model Predictive Control (MPC) [194], Dynamic Programming (DP) [276], and Adaptive Dynamic Programming (ADP) [236] have provided powerful tools to generate control inputs that optimize system performance over time. These approaches aim to minimize a given cost function, subject to the dynamics of the system and input limitations, and are widely used across robotics, aerospace, and automation [277, 278, 279]. More recently, these ideas have also influenced reinforcement learning, where control policies are learned through repeated interaction with the environment [280]. Regardless of the specific methodology, the core objective has been always synthesizing optimal control strategies that drive the system toward desired outcomes efficiently

and reliably.

Despite the success of optimal control methods in theory and simulation, their application to real-world systems introduces several challenges. One of the most significant challenges is the need to ensure reliable operation under practical constraints and imperfect conditions. In real environments, systems are often subject to modeling inaccuracies, external disturbances, and unforeseen interactions. These factors can lead to deviations from planned trajectories and, in many cases, violations of critical physical or operational limits. As a result, the ability to incorporate safety considerations such as avoiding collisions, respecting actuator bounds, or staying within prescribed regions of operation has become an essential requirement in the design of optimal controllers [281].

Safety in control systems can be understood across different levels, ranging from soft constraints that reflect performance preferences to hard constraints that must never be violated to ensure system integrity [282]. In real-world applications such as autonomous vehicles, aerial robotics, and medical devices, hard safety requirements are often critical, as failure to satisfy them can lead to collisions, instability, or physical damage [283]. This has led to the development of a wide range of control strategies that explicitly account for safety during both planning and execution. Model Predictive Control (MPC) enforces safety through hard or soft constraints over a finite horizon, allowing predictive adjustments before constraint violations occur [284]. Control Barrier Functions (CBFs) offer a formal tool for maintaining set invariance, ensuring that system trajectories remain within a safe region at all times [285]. Reachability analysis helps determine which initial states can be safely controlled to a goal, and robust control techniques are often employed to guarantee constraint satisfaction under worst-case disturbances [286]. These methods demonstrate that safety is no longer a secondary consideration but a primary component of control system design. As control algorithms

are increasingly deployed in complex and uncertain environments, the ability to maintain safety despite modeling errors, environmental changes, or unexpected disturbances has become a defining challenge and a key driver of modern control theory.

Min-Max Differential Dynamic Programming (Min-Max DDP) is a powerful trajectory optimization method designed for robust control in adversarial or uncertain environments [8]. It extends the classical Differential Dynamic Programming (DDP) framework [50] by formulating the control problem as a dynamic game between two controllers which first attempts to minimize the cost function and then tries to maximize it, resulting in control policies that are inherently robust to worst-case perturbations. By leveraging second-order expansions of the value function, Min-Max DDP efficiently computes feedback policies that adapt to both system nonlinearities and the influence of external disturbances. This makes it well suited for applications where uncertainty cannot be ignored or merely treated as random noise. Given its robustness properties and feedback structure, Min-Max DDP presents a natural foundation for integrating safety into trajectory planning. Hence, these make this approach a good candidate for designing the optimal control for nonlinear dynamics.

As mentioned earlier, safety is a critical requirement in real-world control systems, particularly when operating under uncertainty or in environments with physical constraints. While Min-Max DDP provides robustness to disturbances, it does not inherently guarantee constraint satisfaction, especially under stochastic dynamics. To address this, several works have extended DDP-based methods to incorporate safety considerations. For instance, [287] developed the chance constraint framework for DDP method to make the optimal control safe and robust to uncertainties, especially in the stochastic dynamics. The main application in this paper is to guide a quadrotor nonlinear system to a target point while avoiding collision with obstacles. The paper [288] proposed a method that embeds the barrier function in the dynamics of the system

to reach the goal safely under constraints. In paper [289], the system reaches the target point avoiding multiple static and dynamics no-entry regions by considering the optimal avoidance control problem. In paper [17], the barrier function embedded in the dynamics of the uncertain systems and the optimal control is derived using the Min-Max DDP in a higher dimensional state space to clarify the safety of optimized trajectory.

While these approaches mark significant progress toward integrating safety into DDP frameworks, they each carry limitations that motivate further development. The chance-constrained DDP accounts for uncertainty using probabilistic margins but does not model disturbances strategically, and therefore lacks robustness against intelligent or adversarial perturbations. Barrier-function-based methods embed safety constraints into the system dynamics, but this embedding can increase system dimensionality and complicate the analysis and convergence of the optimizer, especially in high-dimensional systems. Moreover, these methods often rely on deterministic formulations or assume known and bounded uncertainties, limiting their applicability in stochastic environments with unknown or unstructured disturbances. The optimal avoidance strategy in [289] considers dynamic no-entry regions but does not fully exploit feedback structures for uncertainty reduction nor handle safety in a probabilistic sense. Overall, most existing approaches either sacrifice performance for conservatism, ignore feedback in uncertainty propagation, or are not formulated in a min-max structure capable of modeling intelligent disturbance interactions explicitly. Hence, This chapter proposes the Chance-Constraint Game-Theoretic Differential Dynamic Programming (CC-GT-DDP) for a nonlinear stochastic dynamics for safe trajectory optimization and navigation under uncertainty. We also introduce a dual reformulation of the tightened joint chance constraint and provide a feasibility/near-optimality guarantee, showing that the risk-to-go decreases monotonically with the dual multiplier and giving explicit

error bounds.

The remainder of the paper is organized as follows. In Section 7.2, some important notations are provided. In Section 7.3, the problem is formulated and analyzed, which includes the system description, optimal control design, constraint tightening, a dual reformulation with a risk-to-go used by our algorithm, and a feasibility/near-optimality guarantee for the resulting controller. Finally, in Section 7.4, numerical examples and results are illustrated.

7.2 Notation

This section represents the essential notation employed throughout the paper. $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{R}^n$, and $\mathbb{R}^{n\times m}$ presents the set of all real numbers, positive real numbers, $n \times 1$ real column vectors, and $n \times m$ real matrices, respectively. $\mathcal{C}^1(\mathbb{R})$ and $\mathcal{C}^2(\mathbb{R})$ represent sets of functions whose first- and second-order derivative exist and is continuous with domain $\mathbb{R}$. For any specified function $\psi : \Re^n \times \Re \to \Re$, $\Psi : \Re^n \times \Re^m \times \Re \to \Re$ and $\Omega : \Re^n \times \Re \to \Re^n$, the following are given

$$\psi_{xx}(x,t) = \begin{pmatrix} \frac{\partial^2\psi(x,t)}{\partial^2 x_1}, & \cdots & , \frac{\partial^2\psi(x,t)}{\partial x_1 x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2\psi(x,t)}{\partial x_1 \partial x_n}, & \cdots & , \frac{\partial^2\psi(x,t)}{\partial^2 x_n} \end{pmatrix},$$

$$\Psi_{xu}(x,u,t) = \begin{pmatrix} \frac{\partial^2\Psi(x,u,t)}{\partial x_1 \partial u_1}, & \cdots & , \frac{\partial^2\Psi(x,u,t)}{\partial x_1 \partial u_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2\Psi(x,u,t)}{\partial x_n \partial u_1}, & \cdots & , \frac{\partial^2\Psi(x,u,t)}{\partial x_n \partial u_n} \end{pmatrix},$$

$$\Omega_x(x,t) = \begin{pmatrix} \frac{\partial \Omega_1(x,t)}{\partial x_1}, & \dots & , \frac{\partial \Omega_1(x,t)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \Omega_n(x,t)}{\partial x_1}, & \dots & , \frac{\partial \Omega_n(x,t)}{\partial x_n} \end{pmatrix},$$

where ψ_{xx}, Ψ_{xu}, and Ω_x denote the Hessian of ψ with respect to x, the Hessian of Ψ with respect to x and u, and the Jacobian matrix of Ω with respect to x, respectively.

7.3 Problem Formulation and Analysis

7.3.1 System Description

We consider a nonlinear discrete-time dynamical system of the form

$$x_{k+1} = F(x_k, u_k, v_k) + \omega_k, \tag{7.1}$$

where for a given initial condition x_0, target state x_{target}, and a time horizon N, the aim is to find input trajectories $\{u_o, ..., u_{N-1}\}$ and $\{v_o, ..., v_{N-1}\}$ that minimizes and maximizes the expectation of an objective function

$$J(x_0, u_{0:N-1}, v_{0:N-1}) = \sum_{k=1}^{N-1} \mathcal{L}(x_k, u_k, v_k) + \phi(x_N), \tag{7.2}$$

where $x_k \in \Re^n$ is the state of the system, $u_k \in U \subset \Re^{m_1}$ and $v_k \in V \subset \Re^{m_2}$ are the minimizer and maximizer control policies at the time step k. $\mathcal{L}$ is the running cost, ϕ is the terminal cost, ω_k is assumed to be spatially uncorrelated independent and identically distributed noise, drawn from a zero mean Gaussian distribution with a known covariance matrix, $\omega_k \sim N(0, \Sigma_k^\omega)$. In Section 7.3.4 we introduce a Lagrangian

relaxation J^{λ} that adds a penalty on tightened-set violations to enforce the chance constraint during synthesis; unless stated otherwise, "cost" refers to equation (7.2).

Considering the input limitations and external constraints on the states, the problem of CC-GT-DDP can be formulated as

$$\min_{u_{0:N-1}} \max_{v_{0:N-1}} \mathbf{E}\Big[\sum_{k=1}^{N-1} \mathcal{L}(x_k, u_k, v_k) + \phi(x_N)\Big], \tag{7.3}$$

$$\begin{aligned} \text{subject to } & x_{k+1} = F(x_k, u_k, v_k) + \omega_k, \\ & u_{min} \leq u_k \leq u_{max}, \\ & v_{min} \leq v_k \leq v_{max}, \\ & Pr\Big[g(x_k, u_k.v_k) \leq 0\Big] > \beta_k, \\ & x_0 = \bar{x}, \end{aligned} \tag{7.4}$$

for all $k = 0, ..., N-1$ where g is a vector of c constraints in the form of differentiable functions representing the deterministic state constraints and $\beta_k = [\beta_{1,k}, ..., \beta_{c,k}]$ is a vector of minimum satisfaction probabilities for these constraints for time step k. The constraint is referred to an individual or joint chance constraint if $c = 1$ or $c > 1$, respectively.

By following the chance-constrained formulation, future state trajectories become uncertain due to the stochastic nature of the system dynamics. As a result, safety constraints must be satisfied with a desired probability. To address this within an optimization framework, we transform the chance constraints into deterministic constraints through constraint tightening, where each original constraint is adjusted by an uncertainty margin based on the propagated covariance of the states. However, direct tightening based on open-loop uncertainty propagation can lead to excessive conservatism, severely restricting feasible trajectories and potentially making the op-

timization problem infeasible. This leads to unnecessarily limited feasible trajectories and may even make the optimization problem infeasible. To avoid this, we incorporate the effect of the feedback control policy into the uncertainty prediction. By using the feedback gains computed in the backward pass, we are able to better capture how the control actions reduce uncertainty over time, allowing for tighter, less conservative constraint margins while still maintaining safety.

Moreover, our approach adopts the game-theoretic formulation to explicitly model the presence of disturbances. In this framework, the maximizer player acts as an intelligent adversary that perturbs the system dynamics within allowed stochastic bounds, while the minimizer player seeks to optimize the nominal objective. This interpretation enhances the robustness of the policy against uncertainty rather than reacting to random noise. Hence, the system is proactively optimized to withstand deliberate and structured disturbances that attempt to degrade performance. As a result, the trajectories generated by our Chance-Constraint Game-Theoretic Differential Dynamic Programming (CC-GT-DDP) are safe, robust, and resilient under uncertainty.

The overall method proceeds in three main steps: 1) The backward pass computes local approximations of the cost-to-go functions. 2) The forward pass updates the trajectories while respecting the tightened safety constraints. 3) The constraint tightening step refines the constraints based on the feedback-stabilized uncertainty propagation. These steps are repeated until convergence, resulting in trajectories that satisfy safety requirements with high probability and are robust against stochastic disturbances.

The next subsections describe each component of the method in detail, beginning with the optimal control design, and following with the backward pass, which handles the Riccati-like updates in the presence of tightened constraints and adversarial disturbances.

7.3.2 Optimal Control Policy

In this section, the aim is to develop the local optimal control update laws for high-dimensional discrete-time systems using discrete DDP in a min-max framework [8]. In this chapter, we are interested in robustifying the trajectory optimization problem in safety-critical environments. In particular, we are to develop a safe and robust optimal control policy through the use of chance constraint tightening in the framework of differential games using Min-Max DDP.

We adopt a zero-sum formulation at each stage: the controller chooses u_k while the adversary (disturbance policy) chooses v_k, and the dynamic programming recursion proceeds through the stage action-value $V_k(x_k, u_k, v_k) = L(x_k, u_k, v_k) + V_{k+1}(x_{k+1})$. To ensure that the min–max Bellman recursion is well posed, we state the following standard condition.

Assumption 11 (Isaacs condition)**.** *For each state and stage, the stage game admits measurable selectors and*

$$\min_{u_k} \max_{v_k} V_k(x_k, u_k, v_k) = \max_{v_k} \min_{u_k} V_k(x_k, u_k, v_k), \tag{7.5}$$

so the dynamic programming principle holds for the min–max recursion, where V_k denotes the stage action-value.

Remark 21. *Assumption 11 guarantees the local saddle structure used by the min–max DDP backward pass. The quadratic model of V_k admits a unique minimizer in u_k and a maximizer in v_k in a neighborhood of the nominal trajectory, so the feedback and feedforward gains are well-defined.*

Consider a nominal trajectory of the chance constraint and the state and control

histories $(\bar{x}_k, \bar{u}_k, \bar{v}_k)$, and the HJBI equation

$$V(x_k, t_k) = \min_{u_k} \max_{v_k} \{\mathcal{L}(x_k, u_k, v_k, t_k) + V(x_{k+1}, t_{k+1})\}, \tag{7.6}$$

with the terminal condition $V(x_N) = \phi(x_N)$. In the discrete-time approach, the problem is first discretized along the given time interval $[t_0, t_f]$. Let the time discretization be $0 = t_0 < t_1 < ... < t_M = t_f$, and let $x(k) = x(t_k)$, $u(k) = u(t_k)$ and $v(k) = v(t_k)$. Starting from $x_{k+1} = x_k + F(x_k, u_k, v_k)\delta t$, where $\delta t = t_{k+1} - t_k$, the linearized dynamics model (7.1) around the state and control histories $(\bar{x}_k, \bar{u}_k, \bar{v}_k)$ in discrete time can be written as

$$\delta x_{k+1} = \bar{A}(t_k)\delta x_k + \bar{B}_u(t_k)\delta u_k + \bar{B}_v(t_k)\delta v_k, \tag{7.7}$$

where $\bar{A}(t_k) = I + F_x(\bar{x}_k, \bar{u}_k, \bar{v}_k)\delta t$, $\bar{B}_u(t_k) = F_u(\bar{x}_k, \bar{u}_k, \bar{v}_k)\delta t$ and $\bar{B}_v(t_k) = F_v(\bar{x}_k, \bar{u}_k, \bar{v}_k)\delta t$, and

$$\delta x_k = x_k - \bar{x}_k, \tag{7.8a}$$

$$\delta u_k = u_k - \bar{u}_k, \tag{7.8b}$$

$$\delta v_k = v_k - \bar{v}_k, \tag{7.8c}$$

The algorithm consists of iterative backward passes along the value function and its derivatives along the states of the system resulting from expanding the HJBI equation (7.6) around the nominal trajectories of the state and controls and forward passes along the dynamics (7.1). Using the standard Min-Max DDP formulation [8], action-value

function (7.6) can be approximated as a quadratic function as

$$\begin{aligned} & V(x+\delta x, u+\delta u, v+\delta v) \\ & \approx V + V_x^T \delta x + V_u^T \delta u + V_v^T \delta v + \frac{1}{2}(\delta x^T V_{xx} \delta x + \delta u^T V_{uu} \delta u \\ & + \delta v^T V_{vv} \delta v) + \delta x^T V_{xu} \delta u + \delta x^T V_{xv} \delta v + \delta u^T V_{uv} \delta v, \end{aligned} \tag{7.9}$$

where δx, δu, and δv are the deviations about the nominal action state (x, u, v). The derivatives of V function are then obtained as

$$\bar{Q}_x = \bar{A}(t_k)^T \bar{V}_x(t_{k+1}) + \bar{\mathcal{L}}_x(t_k). \tag{7.10a}$$

$$\bar{Q}_u = \bar{B}_u(t_k)^T \bar{V}_x(t_{k+1}) + \bar{\mathcal{L}}_u(t_k) \tag{7.10b}$$

$$\bar{Q}_v = \bar{B}_v(t_k)^T \bar{V}_x(t_{k+1}) + \bar{\mathcal{L}}_v(t_k). \tag{7.10c}$$

$$\bar{Q}_{xx} = \bar{\mathcal{L}}_{xx}(t_k) + \bar{A}(t_k)^T \bar{V}_{xx}(t_{k+1}) \bar{A}(t_k). \tag{7.10d}$$

$$\bar{Q}_{uu} = \bar{\mathcal{L}}_{uu}(t_k) + \bar{B}_u(t_k)^T \bar{V}_{xx}(t_{k+1}) \bar{B}_u(t_k). \tag{7.10e}$$

$$\bar{Q}_{vv} = \bar{\mathcal{L}}_{vv}(t_k) + \bar{B}_v(t_k)^T \bar{V}_{xx}(t_{k+1}) \bar{B}_v(t_k). \tag{7.10f}$$

$$\bar{Q}_{xu} = \bar{\mathcal{L}}_{xu}(t_k) + \bar{A}(t_k)^T \bar{V}_{xx}(t_{k+1}) \bar{B}_u(t_k). \tag{7.10g}$$

$$\bar{Q}_{xv} = \bar{\mathcal{L}}_{xv}(t_k) + \bar{A}(t_k)^T \bar{V}_{xx}(t_{k+1}) \bar{B}_v(t_k). \tag{7.10h}$$

$$\bar{Q}_{uv} = \bar{\mathcal{L}}_{uv}(t_k) + \bar{B}_u(t_k) \bar{V}_{xx}(t_{k+1}) \bar{B}_v(t_k). \tag{7.10i}$$

$$\bar{Q}_{vu} = \bar{Q}_{uv}^T, \quad \bar{Q}_{ux} = \bar{Q}_{xu}^T, \quad \bar{Q}_{vx} = \bar{Q}_{xv}^T. \tag{7.10j}$$

Following the derivations in [8], the optimal controls are then computed as

$$\delta u_k^* = I_u + K_u \delta x_k, \tag{7.11a}$$

$$\delta v_k^* = I_v + K_v \delta x_k, \tag{7.11b}$$

where, at time instant k,

$$I_u = -(\bar{Q}_{uu} - \bar{Q}_{uv}\bar{Q}_{vv}^{-1}\bar{Q}_{vu})^{-1}(\bar{Q}_u - \bar{Q}_{uv}\bar{Q}_{vv}^{-1}\bar{Q}_v), \tag{7.12a}$$

$$I_v = -(\bar{Q}_{vv} - \bar{Q}_{vu}\bar{Q}_{uu}^{-1}\bar{Q}_{uv})^{-1}(\bar{Q}_v - \bar{Q}_{vu}\bar{Q}_{uu}^{-1}\bar{Q}_u), \tag{7.12b}$$

$$K_u = -(\bar{Q}_{uu} - \bar{Q}_{uv}\bar{Q}_{vv}^{-1}\bar{Q}_{vu})^{-1}(\bar{Q}_{ux} - \bar{Q}_{uv}\bar{Q}_{vv}^{-1}\bar{Q}_{vx}), \tag{7.12c}$$

$$K_v = -(\bar{Q}_{vv} - \bar{Q}_{vu}\bar{Q}_{uu}^{-1}\bar{Q}_{uv})^{-1}(\bar{Q}_{vx} - \bar{Q}_{vu}\bar{Q}_{uu}^{-1}\bar{Q}_{ux}). \tag{7.12d}$$

The parameter $0 < \epsilon \leq 1$ is added to the gains as follows to ensure that the convergence is obtained

$$\delta u_k^* = \epsilon I_u + K_u \delta x_k, \tag{7.13a}$$

$$\delta v_k^* = \epsilon I_v + K_v \delta x_k. \tag{7.13b}$$

For unconstrained case, minimizing the cost function (7.2) with respect to δu_k and maximizing it with respect to δv_k, results in equation (7.13). Under state and/or input constraints, the optimal control problem results in

$$\begin{aligned} &\min_{\delta u} \max_{\delta v} V(x + \delta x, u + \delta u, v + \delta v) \\ &\text{subject to } \ \tilde{g}(x + \delta x, u + \delta u, v + \delta v) \leq 0. \end{aligned} \tag{7.14}$$

where the constraint vector $\tilde{g}$ includes the deterministic state constraints obtained in the constraint tightening step including input limitations. In order to incorporate the

constraints in the estimation of the value function (7.6), the first-order approximations of the constraints are obtained as

$$\tilde{g}(x+\delta x, u+\delta u, v+\delta v) \approx \tilde{g}(x,u,v) + \tilde{g}_u(x,u,v)\delta u + \tilde{g}_v(x,u,v)\delta v + \tilde{g}_x(x,u,v)\delta x \tag{7.15}$$

which is written as

$$\tilde{g}(x+\delta x, u+\delta u, v+\delta v) \approx \tilde{g}(x,u,v) + C\delta u + D\delta v - E\delta x. \tag{7.16}$$

The paper [50] proposed to check the set of active constraints during the iterations to ensure that the analytical approximation of the value function around the nominal trajectories still yields a good approximation. To achieve this, all active constraints are included in a set that contains all equality constraints that are always active and inequality constraints for which the value is greater than $-\zeta(\tilde{g}_i > -\zeta)$ where ζ is a small constant for numerical stability. Hence, if a constraint is active, then at the nominal point

$$\tilde{g}(x,u,v) = 0. \tag{7.17}$$

The action-value function (7.14) is then converted to

$$\begin{aligned} &\min_{\delta u}\max_{\delta v} \Big(V_u^T\delta u + V_v^T\delta v + \frac{1}{2}(\delta u^T V_{uu}\delta u) + \frac{1}{2}(\delta v^T V_{vv}\delta v) \\ &+ \delta x^T V_{xu}\delta u + \delta x^T V_{xv}\delta v + \delta u^T V_{uv}\delta v \Big), \\ &\text{subject to } \; C\delta u + D\delta v = E\delta x. \end{aligned} \tag{7.18}$$

Equation (7.18) solves the minimization of V with respect to δu and maximization of

V with respect to δv for a fixed δx (since we optimize locally for the control deviation only). Thus, constants such as V and terms involving only δx are irrelevant for optimization, because they do not affect the solution to δu and δv. Hence, these terms are removed from the min-max operation because they do not affect the optimal values of δu and δv.

Let the Lagrangian equation be

$$L(\delta u, \delta v, \lambda) = V_u^T \delta u + V_v^T \delta v + \frac{1}{2}(\delta u^T V_{uu} \delta u) + \frac{1}{2}(\delta v^T V_{vv} \delta v) + \delta x^T V_{xu} \delta u + \delta x^T V_{xv} \delta v + \delta u^T V_{uv} \delta v + \lambda^T (C\delta u + D\delta v - E\delta x). \tag{7.19}$$

An analytical solution to this problem can be found through the KKT condition [290] and the gradient with respect to δu, δv, and λ can be found by solving the following equations

$$V_{uu}\delta u + V_{ux}\delta x + V_u + V_{uv}\delta v + C^T\lambda = 0, \tag{7.20a}$$

$$V_{vv}\delta v + V_{vx}\delta x + V_v + V_{vu}^T\delta u + D^T\lambda = 0, \tag{7.20b}$$

$$C\delta u + D\delta v - E\delta x = 0, \tag{7.20c}$$

where λ is a vector of Lagrangian multipliers. By solving (7.20) with pruned matrices $\tilde{C}$, $\tilde{D}$, and $\tilde{E}$ that ensure the nonexistence of negative λ values (check [50]), locally optimal input deviations can be further expressed as a function of state deviation.

Theorem 13. *For the constrained min–max DDP problem in equation (7.18) with active constraint set $\tilde{g}(x, u, v) \leq 0$ approximated as in equation (7.16), the locally*

optimal control deviations δu_k^* and δv_k^* can be expressed as

$$\delta u_k^* = I_u + K_u \delta x, \tag{7.21a}$$

$$\delta v_k^* = I_v + K_v \delta x, \tag{7.21b}$$

where the control parameters I_u, I_v, K_u, and K_v are now calculated as

$$I_u = -A(V_{ux} - V_{uv}V_{vv}^{-1}V_{vx} - V_{uv}V_{vv}^{-1}D^T\Lambda_{\delta x} + C^T\Lambda_{\delta x}), \tag{7.22a}$$

$$I_v = -B(V_{vx} - V_{vu}^TV_{uu}^{-1}V_{ux} - V_{vu}^TV_{uu}^{-1}C^T\Lambda_{\delta x} + D^T\Lambda_{\delta x}), \tag{7.22b}$$

$$K_u = -A(V_u - V_{uv}V_{vv}^{-1}V_v - V_{uv}V_{vv}^{-1}D^T\Lambda_c + C^T\Lambda_c), \tag{7.22c}$$

$$K_v = -B(V_v - V_{vu}^TV_{uu}^{-1}V_u - V_{vu}^TV_{uu}^{-1}C^T\Lambda_c + D^T\Lambda_c), \tag{7.22d}$$

and

$$A := \left(I + V_{uu}^{-1}V_{uv}V_{vv}^{-1}V_{vu}^T\right)^{-1}V_{uu}^{-1}, \tag{7.23a}$$

$$B := \left(I + V_{vv}^{-1}V_{vu}^TV_{uu}^{-1}V_{uv}\right)^{-1}V_{vv}^{-1}, \tag{7.23b}$$

$$\begin{aligned}\Lambda_{\delta x} &= [CA(V_{uv}V_{vv}^{-1}D^T - C^T) + DB(V_{vu}^TV_{uu}^{-1}C^T - D^T)]^{-1} \\ &\cdot [CA(V_{ux} - V_{uv}V_{vv}^{-1}V_{vx}) + DB(V_{vx} - V_{vu}^TV_{uu}^{-1}V_{ux}) + E],\end{aligned} \tag{7.23c}$$

$$\begin{aligned}\Lambda_c &= [CA(V_{uv}V_{vv}^{-1}D^T - C^T) + DB(V_{vu}^TV_{uu}^{-1}C^T - D^T)]^{-1} \\ &\cdot [CA(V_u - V_{uv}V_{vv}^{-1}V_v) + DB(V_v - V_{vu}^TV_{uu}^{-1}V_u)].\end{aligned} \tag{7.23d}$$

Proof. We rewrite the equation (7.20) as

$$\delta u = -V_{uu}^{-1}\left(V_{ux}\delta x + V_u + V_{uv}\delta v + C^T\lambda\right), \tag{7.24a}$$

$$\delta v = -V_{vv}^{-1}\left(V_{vx}\delta x + V_v + V_{vu}^T\delta u + D^T\lambda\right), \tag{7.24b}$$

$$C\delta u + D\delta v - E\delta x = 0. \tag{7.24c}$$

Substituting δu expression in (7.24a) into (7.24b), and δv expression in (7.24b) into (7.24a), equation (7.24) can be written as

$$\delta u = -A\left(V_{ux}\delta x + V_u - V_{uv}V_{vv}^{-1}V_{vx}\delta x - V_{uv}V_{vv}^{-1}V_v - V_{uv}V_{vv}^{-1}D^T\lambda + C^T\lambda\right) \tag{7.25}$$

$$\delta v = -B\left(V_{vx}\delta x + V_v - V_{vu}^TV_{uu}^{-1}V_{ux}\delta x - V_{vu}^TV_{uu}^{-1}V_u - V_{vu}^TV_{uu}^{-1}C^T\lambda + D^T\lambda\right) \tag{7.26}$$

$$C\delta u + D\delta v - E\delta x = 0, \tag{7.27}$$

plugging equation (7.25) and (7.26) into (7.27) results λ as

$$\lambda = \Lambda_{\delta x}\,\delta x + \Lambda_c. \tag{7.28}$$

By plugging (7.28) into (7.25) and (7.26) we obtain

$$\delta u_k^* = I_u + K_u\delta x, \tag{7.29a}$$

$$\delta v_k^* = I_v + K_v\delta x, \tag{7.29b}$$

where

$$K_u = -A(V_{ux} - V_{uv}V_{vv}^{-1}V_{vx} - V_{uv}V_{vv}^{-1}D^T\Lambda_{\delta x} + C^T\Lambda_{\delta x}), \tag{7.30a}$$

$$K_v = -B(V_{vx} - V_{vu}^TV_{uu}^{-1}V_{ux} - V_{vu}^TV_{uu}^{-1}C^T\Lambda_{\delta x} + D^T\Lambda_{\delta x}), \tag{7.30b}$$

$$I_u = -A(V_u - V_{uv}V_{vv}^{-1}V_v - V_{uv}V_{vv}^{-1}D^T\Lambda_c + C^T\Lambda_c), \tag{7.30c}$$

$$I_v = -B(V_v - V_{vu}^TV_{uu}^{-1}V_u - V_{vu}^TV_{uu}^{-1}C^T\Lambda_c + D^T\Lambda_c), \tag{7.30d}$$

which completes the proof. □

The time history of V_x and V_{xx} are required to calculate the terms in (7.10). To this end, we substitute the control update laws (7.11) into (7.6) to obtain the update laws of the value function and its first order and second order state derivatives,

$$\bar{V}(t_k) = \bar{V}(t_{k+1}) + \bar{\mathcal{L}} + l_u^T\bar{Q}_u + l_v^T\bar{Q}_v + \frac{1}{2}l_u\bar{Q}_{uu}l_u + l_u^T\bar{Q}_{uv}l_v + \frac{1}{2}l_v^T\bar{Q}_{vv}l_v, \tag{7.31a}$$

$$\bar{V}_x(t_k) = \bar{Q}_x + K_u^T\bar{Q}_u + K_v^T\bar{Q}_v + \bar{Q}_{ux}^Tl_u + \bar{Q}_{vx}^Tl_v + K_u^T\bar{Q}_{uu}l_u + K_u^T\bar{Q}_{uv}l_v$$
$$+ K_v^T\bar{Q}_{vu}l_u + K_v^T\bar{Q}_{vv}l_v, \tag{7.31b}$$

$$\bar{V}_{xx}(t_k) = K_u^T\bar{Q}_{ux} + \bar{Q}_{ux}^TK_u + K_v^T\bar{Q}_{vx} + \bar{Q}_{vx}^TK_v + K_v^T\bar{Q}_{vu}K_u + K_u^T\bar{Q}_{uv}K_v$$
$$+ K_u^T\bar{Q}_{uu}K_u + K_v^T\bar{Q}_{vv}K_v + \bar{Q}_{xx}, \tag{7.31c}$$

Since the optimal cost-to-go is computed backward in time, boundary conditions need to be given at $t = t_f$. These boundary conditions are given by

$$\bar{V}(t_f) = \phi(\bar{x}(t_f), t_f), \tag{7.32a}$$

$$\bar{V}_x(t_f) = \phi_x(\bar{x}(t_f), t_f), \tag{7.32b}$$

$$\bar{V}_{xx}(t_f) = \phi_{xx}(\bar{x}(t_f), t_f). \tag{7.32c}$$

For later use in the risk-aware controller, we also define a λ-penalized stage cost that penalizes tightened-set violations:

$$L_k^{\lambda}(x_k, u_k, v_k) := L(x_k, u_k, v_k) \; + \; \lambda \, I_k^{\beta}(x_k). \tag{7.33}$$

where L_k^{λ} is the penalized stage cost used when solving the inner min–max DDP for a fixed λ, $I_k^{\beta}(x_k)$ is the tightened-set violation indicator from Section 7.2. In the backward pass, we differentiate the penalized stage cost, where the non-differentiable indicator I_k^{β} is replaced by a smooth upper bound $\bar{I}_k^{\beta}$; the true I_k^{β} is still used when evaluating the risk during the forward rollouts.

Remark 22. *The additive term $\lambda I_k^{\beta}(x_k)$ is independent of (u_k, v_k), hence it does not change the local optimality conditions for the control/disturbance gains; it shifts the value function V_k by a state-dependent offset. If a smooth surrogate $\bar{I}_k^{\beta}$ is preferred for differentiability in the backward pass, we use it only in derivatives and evaluate risk with the true I_k^{β} during rollouts.*

As the next step after the backward pass to update the value function, the forward pass to update the state trajectory is described here. To ensure that the constraints are satisfied during the forward pass, we pose the forward pass as a quadratic programming (QP) problem

$$\begin{aligned} \min_{\delta u} \max_{\delta v} \Big(& V_u^T \delta u + V_v^T \delta v + \frac{1}{2}(\delta u^T V_{uu} \delta u) + \frac{1}{2}(\delta v^T V_{vv} \delta v) \\ & + \delta x^T V_{xu} \delta u + \delta x^T V_{xv} \delta v + \delta u^T V_{uv} \delta v \Big), \end{aligned} \tag{7.34}$$

$$\begin{aligned} \text{subject to } & \tilde{g}(x + \delta x, u + \delta u, v + \delta v) \leq 0, \\ & u_{min} \leq u + \delta u \leq u_{max}, \\ & v_{min} \leq v + \delta v \leq v_{max}. \end{aligned} \tag{7.35}$$

The solution to this problem represents the optimal deviations in the control inputs, δu and δv, while ensuring the constraints are satisfied.

The state is then updated by forward integration using the optimal input deviations, δu^* and δv^*, as follows

$$x_{k+1} = f(x_k, u_k + \delta u^*, v_k + \delta v^*). \tag{7.36}$$

If we choose the right λ, then the expected number of violations will not exceed the prescribed joint risk budget Δ. To enforce this in practice, we close the forward pass with a Monte Carlo risk estimate and a bracketed update of the dual multiplier λ:

$$\hat{r} := \frac{1}{M} \sum_{m=1}^{M} \sum_{k=1}^{N} I_k^{\beta}\big(x_k^{(m)}\big), \tag{7.37}$$

$$\lambda := \tfrac{1}{2}\big(\lambda_L + \lambda_U\big), \tag{7.38}$$

$$(\lambda_L, \lambda_U) \leftarrow \begin{cases} (\lambda, \lambda_U), & \text{if } \hat{r} > \Delta, \\ (\lambda_L, \lambda), & \text{if } \hat{r} \le \Delta, \end{cases} \tag{7.39}$$

$$\text{stop when} \quad |\hat{r} - \Delta| \le \text{tol} \quad \text{and} \quad \lambda_U - \lambda_L \le \text{tol}_\lambda. \tag{7.40}$$

where x_k^m is the $m - th$ Monte Carlo rollout of the closed loop under the current policy pair, $\hat{r}$ estimates the expected number of tightened-set violations, $[\lambda_L, \lambda_U]$ is the working bracket for the dual search, and *tol* and tol_λ are the stopping tolerances.

Remark 23. *The update rules* (7.38)–(7.39) *implement a standard bisection search on the risk residual* $\hat{r} - \Delta$. *Intuitively, if the estimated risk* $\hat{r}$ *is larger than the budget* Δ, *then* λ *is too small and the lower bracket is increased; if* $\hat{r}$ *is smaller than* Δ, *then* λ *is too large and the upper bracket is decreased. Empirically, the mapping* $\lambda \mapsto \hat{r}$ *is*

monotone nonincreasing in all experiments, so each update shrinks the bracket until the stopping criterion (7.40) *is satisfied. The tolerances* tol *and* tol_λ *control when the search stops, and they reflect a trade-off: smaller tolerances give higher accuracy but require more Monte Carlo samples and runtime.*

7.3.3 Constraint Tightening

This section explains how the chance constraints formulated in (7.3) are transformed into the deterministic ones which are required for backward and forward passes. In order to obtain the constraints in a deterministic form, we tighten the constraints using the propagated uncertainty through prediction, which can be decreased efficiently once a feedback policy is employed in the optimization.

Nonlinear transition dynamics $F(\cdot)$ in (7.1) can be approximated to first order by

$$x_{k+1} \approx x_k + F_x \delta x + F_u \delta u + F_v \delta v, \tag{7.41}$$

employing the locally optimal control from (7.21) results in

$$\begin{aligned} x_{k+1} &\approx x_k + F_x \delta x + F_u K_u \delta x + F_v K_v \delta x + I_u + I_v \\ &= x_k + (F_x + F_u K_u + F_v K_v)\delta x + I_u + I_v. \end{aligned} \tag{7.42}$$

Closed-loop uncertainty propagation through prediction for the system in (7.1) then can be approximated as

$$\Sigma^x_{k+1} = (F_x + F_u K_u + F_v K_v)^T \Sigma^x_k (F_x + F_u K_u + F_v K_v) + \Sigma^w_{k+1}, \tag{7.43}$$

where Σ^x_k denotes the covariance matrix of x_k. Since constrained Min-Max DDP performs a local optimization at each step through prediction, general nonlinear constraint

can be locally interpreted as a half-space constraint for that particular step. By utilizing the relationship between probabilistic and robust invariant sets presented in [291] for linear time-invariant systems, the effect of the propagated uncertainty could be approximately taken into account in constraint definitions for each step as follows

$$\tilde{g}(x, u, v, \Sigma^x) \triangleq g(x, u, v) + \Phi^{-1}(\beta)\sqrt{\tilde{g}_x^T \Sigma^x \tilde{g}_x} \tag{7.44}$$

where Φ^{-1} is the quantile function of the standard normal distribution. Moreover, the β value adjusts the amount of uncertainty to be considered, which proportionally affects the safety limits. Although different β values can be assigned for different constraints at different time steps, we did not find it beneficial for safety-required tasks and selected all the β values the same for simplicity. Once all the chance-constraints are reformulated, the problem formulation of (7.3) can be written in a deterministic form as

$$\begin{aligned}
&\min_{u_k} \max_{v_k} \; J(x, U, V), \\
&\text{subject to } x_{k+1} = F(x_k, u_k, v_k), \\
&\qquad u_{min} \leq u_k \leq u_{max}, \\
&\qquad v_{min} \leq v_k \leq v_{max}, \\
&\qquad \textstyle\sum_{k+1}^x = h(x_k, u_k, v_k, K_u, K_v, \sum_k^x) \\
&\qquad \tilde{g}(x_k, u_k, v_k, \textstyle\sum_k^x) \leq 0, \\
&\qquad x_0 = \bar{x},
\end{aligned} \tag{7.45}$$

for all $k = 1, ..., N-1$, where $h(\cdot)$ denotes the right-hand side of equation (7.43).

7.3.4 Dual reformulation and risk-to-go

The safety requirement in our problem is introduced by the joint chance constraint in (7.4), which demands that the entire closed-loop trajectory remains inside the admissible state set with high probability. Formally, this means that the probability of any violation across the horizon is bounded by a prescribed budget Δ. In other words, instead of enforcing safety at each step deterministically, the joint chance constraint ensures that the system may tolerate uncertainty and disturbances, but the overall probability of leaving the safe set over the entire horizon remains below the risk budget. This global probabilistic condition captures the notion of "safety with high confidence," but it is not directly compatible with the additive structure of dynamic programming, and thus requires reformulation. The required parameters for formalizing the joint chance constraint, including the violation indicator, the tightened safe sets, the risk budget, and the dual multiplier, are introduced as follows

$$
\begin{aligned}
&\lambda \in \mathbb{R}_{\geq 0}, \quad \Delta \in [0, N], \quad \beta \in (0,1), \quad I_k^\beta(x) := \mathbf{1}\{x \notin \mathcal{X}_k^\beta\}, \\
&r_0^\beta(u_k, v_k) := \mathbb{E}\left[\sum_{k=1}^{N} I_k^\beta(x_k) \,\middle|\, x_0 \right], \\
&J^\lambda(x_0) := \min_{u_k} \max_{v_k} \mathbb{E}\left[\phi(x_N) + \sum_{k=0}^{N-1} L(x_k, u_k, v_k) + \lambda \sum_{k=1}^{N} I_k^\beta(x_k) \right], \\
&q(\lambda) := J^\lambda(x_0) - \lambda\, \Delta, \\
&h^\lambda(x_0) := \min_{u_k} \max_{v_k} \mathbb{E}\left[\phi(x_N) + \sum_{k=0}^{N-1} L(x_k, u_k, v_k) \right].
\end{aligned} \tag{7.46}
$$

where λ is the dual (Lagrange) multiplier of the tightened chance constraint ($\lambda \to$ the optimizer prefers safety more), Δ is the joint risk budget (with $\Delta = N\delta$ if a per-stage probability δ is used), β is the tightening/confidence parameter used to build shrunken safe sets, $\mathcal{X}_k^\beta$ is the tightened safe set at stage k, $I_k^\beta(x)$ is the violation indicator of

being outside, $r_0^\beta(u, v)$ is the risk-to-go or expected count of violations for a given policy pair, $J^\lambda(x_0)$ is the λ-penalized objective, $q(\lambda)$ is the dual function, and $h^\lambda(x_0)$ is the unpenalized min–max value attained by the optimal policy pair. The search interval $[\lambda_L, \lambda_U]$ and tolerances $\texttt{tol}$, $\texttt{tol}_\lambda$ are used for the outer dual iteration to stop the $\lambda-$tuning when we essentially hit the target risk.

Remark 24 (Dual search parameters)**.** *We denote by $[\lambda_L, \lambda_U]$ the working bracket for the outer dual search, and by $\texttt{tol}$, $\texttt{tol}_\lambda$ the stopping tolerances for $|\,\hat{r} - \Delta\,|$ (risk residual) and for the bracket width, respectively.*

To handle the joint chance constraint, we introduce a dual penalty on tightened-set violations and define the penalized objective

$$J^\lambda(x_0) := \min_{u_k} \max_{v_k} \mathbb{E}\left[\phi(x_N) + \sum_{k=1}^{N-1} L(x_k, u_k, v_k) + \lambda \sum_{k=1}^{N} I_k^\beta(x_k) \,\middle|\, x_0\right], \tag{7.47}$$

together with the dual function

$$q(\lambda) := J^\lambda(x_0) - \lambda\,\Delta, \tag{7.48}$$

and the risk-to-go

$$r_0^\beta(u_k, v_k) := \mathbb{E}\left[\sum_{k=1}^{N} I_k^\beta(x_k) \,\middle|\, x_0\right]. \tag{7.49}$$

The stage cost L and terminal cost ϕ in J^λ are exactly those of (7.2); the additional term $\lambda \sum_{k=1}^{N} I_k^\beta(x_k)$ is used only during the dual search. Final performance is always evaluated with (7.2). Here $I_k^\beta(x_k) = \mathbf{1}\{x_k \notin \mathcal{X}_k^\beta\}$ is the tightened-set violation indicator from Section 7.2, $\lambda \in \mathbb{R}_{\geq 0}$ is the dual multiplier, and $\Delta \in [0, N]$ is the joint risk budget. For any fixed λ, the inner min–max problem in (7.47) is solved by the existing DDP

recursion using the penalized stage cost L_k^λ defined in (7.33); we denote the resulting policy pair by $(u_k^\lambda, v_k^\lambda)$.

7.3.5 Feasibility and near-optimality via duality

In this subsection we establish a dual certificate for the chance-constrained min–max problem by analyzing the penalized objective (7.47), the dual function (7.48), and the risk-to-go (7.49). The results below rely on the min–max DDP recursion introduced in Section 7.3.2 and the tightened sets constructed in Section 7.3.3.

Assumption 12 (Standing assumptions for the dual analysis)**.** *(i) The horizon is finite ($N < \infty$) and the process noise is independent and identically distributed with finite second moments. (ii) The maps F, L, and ϕ are measurable and bounded below. (iii) The min–max dynamic programming principle holds by Assumption 11. (iv) For every $\lambda \geq 0$, the penalized problem* (7.47) *admits an optimal policy pair $(u_k^\lambda, v_k^\lambda)$.*

Lemma 1 (Concavity and subgradient)**.** *The dual function*

$$q(\lambda) := J^\lambda(x_0) - \lambda\Delta, \tag{7.50}$$

with

$$J^\lambda(x_0) = \min_{u_k} \max_{v_k} \mathbb{E}\Big[\phi(x_N) + \sum_{k=0}^{N-1} L(x_k, u_k, v_k) + \lambda \sum_{k=1}^{N} I_k^\beta(x_k) \,\Big|\, x_0\Big], \tag{7.51}$$

is concave in $\lambda \geq 0$. Moreover, the subgradient of q at λ is

$$r_0^\beta(u_k^\lambda, v_k^\lambda) - \Delta, \tag{7.52}$$

where $(u_k^\lambda, v_k^\lambda)$ is an optimal policy pair for J^λ.

Proof. For any fixed policies (u_k, v_k), define

$$f_{u_k,v_k}(\lambda) = \mathbb{E}\Big[\phi + \sum_{k=0}^{N-1} L + \lambda \sum_{k=1}^{N} I_k^\beta\Big] - \lambda\Delta, \tag{7.53}$$

which is affine in λ.

Taking $\min_{u_k} \max_{v_k} f_{u_k,v_k}(\lambda)$ yields a pointwise infimum of affine functions, which is concave. Thus $q(\lambda)$ is concave. By Danskin's theorem, if $(u_k^\lambda, v_k^\lambda)$ is an optimal policy pair at λ, the subgradient of q is

$$\frac{\partial}{\partial\lambda}\Big(\mathbb{E}[\phi + \sum L] + \lambda\,\mathbb{E}[\sum I_k^\beta] - \lambda\Delta\Big) = r_0^\beta(u_k^\lambda, v_k^\lambda) - \Delta. \tag{7.54}$$

□

Lemma 2 (KKT condition at maximizer)**.** *If $\lambda^\star \in \arg\max_{\lambda\geq 0} q(\lambda)$, then*

$$\begin{aligned} \lambda^\star > 0 \;&\Rightarrow\; r_0^\beta(u_k^{\lambda^\star}, v_k^{\lambda^\star}) = \Delta, \\ \lambda^\star = 0 \;&\Rightarrow\; r_0^\beta(u_k^0, v_k^0) \leq \Delta. \end{aligned} \tag{7.55}$$

Proof. Optimality requires $0 \in \partial q(\lambda^\star)$. From Lemma 1,

$$\partial q(\lambda^\star) = \{\, r_0^\beta(u_k^{\lambda^\star}, v_k^{\lambda^\star}) - \Delta \,\}. \tag{7.56}$$

Thus: if $\lambda^\star > 0$, complementary slackness forces $r_0^\beta(u_k^{\lambda^\star}, v_k^{\lambda^\star}) - \Delta = 0$. If $\lambda^\star = 0$, then concavity requires the subgradient ≤ 0, hence $r_0^\beta(u_k^0, v_k^0) - \Delta \leq 0$. □

Lemma 3 (Monotonicity of risk)**.** *The mapping $\lambda \mapsto r_0^\beta(u_k^\lambda, v_k^\lambda)$ is nonincreasing.*

Proof. Let $\lambda_1 < \lambda_2$, by definition,

$$J^{\lambda_i}(x_0) = \min_{u_k} \max_{v_k} \mathbb{E}\Big[\phi + \sum L + \lambda_i \sum I_k^\beta\Big], \tag{7.57}$$

optimality gives

$$J^{\lambda_2}(x_0) \leq \mathbb{E}\Big[\phi + \sum L + \lambda_2 \sum I_k^\beta \,\Big|\, (u_k^{\lambda_1}, v_k^{\lambda_1})\Big]. \tag{7.58}$$

Subtracting the same expression with λ_1 yields

$$J^{\lambda_2}(x_0) - J^{\lambda_1}(x_0) \leq (\lambda_2 - \lambda_1) r_0^\beta(u_k^{\lambda_1}, v_k^{\lambda_1}). \tag{7.59}$$

Similarly, we have

$$J^{\lambda_1}(x_0) - J^{\lambda_2}(x_0) \leq (\lambda_1 - \lambda_2) r_0^\beta(u_k^{\lambda_2}, v_k^{\lambda_2}). \tag{7.60}$$

Adding (7.59) to (7.60) yields

$$(\lambda_2 - \lambda_1)\Big(r_0^\beta(u_k^{\lambda_2}, v_k^{\lambda_2}) - r_0^\beta(u_k^{\lambda_1}, v_k^{\lambda_1})\Big) \leq 0. \tag{7.61}$$

Since $\lambda_2 - \lambda_1 > 0$, the difference must be ≤ 0. Thus risk is nonincreasing. □

Lemma 4 (Feasibility via bracketing)**.** *Suppose $\lambda_L \leq \lambda^\star \leq \lambda_U$, where $\lambda^\star$ maximizes q. If*

$$r_0^\beta(u_k^{\lambda_L}, v_k^{\lambda_L}) \geq \Delta, \quad r_0^\beta(u_k^{\lambda_U}, v_k^{\lambda_U}) \leq \Delta, \tag{7.62}$$

then $(u_k^{\lambda_U}, v_k^{\lambda_U})$ *satisfies*

$$\mathbb{E}\Big[\sum_{k=1}^{N} I_k^{\beta}(x_k)\Big] \leq \Delta. \tag{7.63}$$

Proof. By Lemma 3,

$$r_0^{\beta}(u_k^{\lambda_L}, v_k^{\lambda_L}) \geq r_0^{\beta}(u_k^{\lambda^\star}, v_k^{\lambda^\star}) \geq r_0^{\beta}(u_k^{\lambda_U}, v_k^{\lambda_U}), \tag{7.64}$$

thus

$$r_0^{\beta}(u_k^{\lambda_U}, v_k^{\lambda_U}) \leq \Delta. \tag{7.65}$$

Therefore the policy at λ_U is feasible. □

The chance constraint introduces a global probabilistic requirement that is not naturally compatible with the additive structure of dynamic programming. By introducing the dual multiplier λ, we obtain a penalized problem that can be solved using min–max DDP. The following theorem formalizes the guarantees of this approach, which ensures that the proposed dual update mechanism both enforces the prescribed risk budget and bounds the performance loss compared to the best achievable constrained solution.

Theorem 14 (Dual and primal suboptimality bounds)**.** *Under Assumption 12, define the residual risk*

$$\varepsilon_r := \Delta - r_0^{\beta}(u_k^{\lambda_U}, v_k^{\lambda_U}) \;\geq\; 0, \tag{7.66}$$

and set

$$\varepsilon_d := (\lambda_U - \lambda_L)\, \varepsilon_r. \tag{7.67}$$

Then the following hold:

$$0 \;\le\; q(\lambda^\star) - q(\lambda_U) \;\le\; \varepsilon_d, \tag{7.68}$$

$$\begin{aligned} 0 \;\le\; h^{\lambda_U}(x_0) - h^\star(x_0) \;\le\; \min\Big\{ & -\lambda_U\big(r_0^\beta(u_k^{\lambda_U}, v_k^{\lambda_U}) - \Delta\big),\; h^{\lambda_U}(x_0) - h^{\lambda_L}(x_0) \\ & - \lambda_L\big(r_0^\beta(u_k^{\lambda_L}, v_k^{\lambda_L}) - \Delta\big)\Big\} \;=:\; \varepsilon_p. \end{aligned} \tag{7.69}$$

Proof. From Lemma 1, $q(\lambda)$ is concave and its subgradient at λ is

$$g(\lambda) = r_0^\beta(u_k^\lambda, v_k^\lambda) - \Delta = -\varepsilon_r. \tag{7.70}$$

Consider λ_U with subgradient $g(\lambda_U) = -\varepsilon_r$. Concavity of q gives, for all $\lambda \in [\lambda_L, \lambda_U]$,

$$q(\lambda) \;\le\; q(\lambda_U) + g(\lambda_U)(\lambda - \lambda_U), \tag{7.71}$$

plugging in $\lambda = \lambda^\star$ and $g(\lambda_U) = -\varepsilon_r$, we get

$$q(\lambda^\star) \;\le\; q(\lambda_U) + \varepsilon_r(\lambda_U - \lambda^\star), \tag{7.72}$$

since $\lambda^\star \ge \lambda_L$, we bound

$$q(\lambda^\star) - q(\lambda_U) \;\le\; \varepsilon_r(\lambda_U - \lambda_L) \;=\; \varepsilon_d. \tag{7.73}$$

Clearly $q(\lambda^\star) \ge q(\lambda_U)$ by optimality of $\lambda^\star$, so inequality (7.68) follows.

By definition of q,

$$q(\lambda) \;=\; h^{\lambda}(x_0) + \lambda\big(r_0^{\beta}(u_k^{\lambda}, v_k^{\lambda}) - \Delta\big). \tag{7.74}$$

This identity will be used to relate primal and dual values.

Since $q(\lambda^{\star}) \geq q(\lambda_U)$,

$$h^{\lambda^{\star}}(x_0) + \lambda^{\star}\big(r_0^{\beta}(u_k^{\lambda^{\star}}, v_k^{\lambda^{\star}}) - \Delta\big) \;\geq\; h^{\lambda_U}(x_0) + \lambda_U\big(r_0^{\beta}(u_k^{\lambda_U}, v_k^{\lambda_U}) - \Delta\big). \tag{7.75}$$

By Lemma 2, at $\lambda^{\star}$ the slack satisfies $r_0^{\beta}(u_k^{\lambda^{\star}}, v_k^{\lambda^{\star}}) - \Delta \leq 0$. Hence

$$h^{\lambda^{\star}}(x_0) \;\geq\; h^{\lambda_U}(x_0) + \lambda_U\big(r_0^{\beta}(u_k^{\lambda_U}, v_k^{\lambda_U}) - \Delta\big). \tag{7.76}$$

The inequality (7.76) can be rearranged as

$$h^{\lambda_U}(x_0) - h^{\lambda^{\star}}(x_0) \;\leq\; -\lambda_U\big(r_0^{\beta}(u_k^{\lambda_U}, v_k^{\lambda_U}) - \Delta\big). \tag{7.77}$$

Similarly, comparing $q(\lambda^{\star}) \geq q(\lambda_L)$ yields

$$h^{\lambda^{\star}}(x_0) \;\geq\; h^{\lambda_L}(x_0) + \lambda_L\big(r_0^{\beta}(u_k^{\lambda_L}, v_k^{\lambda_L}) - \Delta\big). \tag{7.78}$$

Subtracting (7.78) from $h^{\lambda_U}(x_0)$ on both sides gives

$$h^{\lambda_U}(x_0) - h^{\lambda^{\star}}(x_0) \;\leq\; h^{\lambda_U}(x_0) - h^{\lambda_L}(x_0) - \lambda_L\big(r_0^{\beta}(u_k^{\lambda_L}, v_k^{\lambda_L}) - \Delta\big). \tag{7.79}$$

Since $h^{\star}(x_0) \leq h^{\lambda^{\star}}(x_0)$ by feasibility of $\lambda^{\star}$, combining the two bounds yields inequality (7.69). □

In this chapter, we use a diminishing trust-region for infeasible solutions and ensure

the numerical stability of matrix inversions by adding a regularization term (Please refer to [199] for regularization). The Chance Constraint Game-Theoretic Differential Dynamic Programming (CC-GT-DDP) algorithm is presented in Algorithm 8.

7.3.6 CC-GT-DDP Algorithm in a Nutshell

Here we describe the Chance-Constraint Game-Theoretic Differential Dynamic Programming (CC-GT-DDP) algorithm in its entirety with emphasis on its practical implementation in trajectory optimization under uncertainty. As with other DDP-based methods, initialization remains a critical step. A feasible initialization can be obtained using a relaxed or unconstrained DDP rollout, which generates a nominal trajectory of states and controls for both players. This trajectory, although suboptimal, provides a consistent starting point for the min–max iterations of CC-GT-DDP.

After initialization, the algorithm proceeds within a model predictive control framework, where the horizon is decreased as the system approaches the target state. The number of iterations for backward and forward passes, denoted n_1, is a hyper-parameter that depends on system dimension, horizon length, and number of constraints. While convergence-based stopping criteria could be employed, in practice a fixed iteration count offers predictable runtime.

A key challenge arises from the interplay between uncertainty propagation and feedback gain computation. Specifically, the closed-loop covariance update requires knowledge of the feedback gains, while the feedback gains themselves are computed from the backward pass constrained by tightened chance constraints that depend on the covariance. To resolve this chicken-and-egg problem, we initially neglect stochastic disturbances when performing the backward/forward passes and enforce constraints deterministically. Afterward, we refine the constraints by applying chance-constrained tightening using the covariance update. This ensures that safety requirements are

Algorithm 8 Chance Constraint Game-Theoretic DDP

Require: N, $\mathbf{x}_0$, $\mathbf{x}_{\text{goal}}$, $\mathbf{x}_1, ..., \mathbf{x}_N$, $\mathbf{u}_0, ..., \mathbf{u}_{N-1}$,$\mathbf{v}_0, ..., \mathbf{v}_{N-1}$. Δ, β, M, tol, tol_λ given;
$\lambda_L \leftarrow 0$; Initialize λ_U by starting from a small positive value and repeatedly increasing it with short rollouts until the estimated risk $\hat{r} \leq \Delta$; $\lambda \leftarrow \frac{1}{2}(\lambda_L + \lambda_U)$;
while robot is not close to $\mathbf{x}_{\text{goal}}$ **do**
  for iter $= 0, ..., n_1$ **do**
    function Backward Pass:
    $V_x(\mathbf{x}_N) \leftarrow \ell_x^f(\mathbf{x}_N)$ and $V_{xx}(\mathbf{x}_N) \leftarrow \ell_{xx}^f(\mathbf{x}_N)$;
    for $k = N-1, ..., 0$ **do**
      Calculate derivatives of Q and V(Eq. 7.10 and Eq. 7.31), and save;
    end for
    function Forward Pass:
    $\mathbf{x} \leftarrow \mathbf{x}_0$;
    for $k = 1, ..., N-1$ **do**
      $\delta\mathbf{x} \leftarrow \mathbf{x} - \mathbf{x}_k$ and solve QP for $\delta\mathbf{u}$ and $\delta\mathbf{v}$ (Eq. 7.34);
      if feasible solution **then**
        $\mathbf{x}_{k+1} \leftarrow F(\mathbf{x}, \mathbf{u}_k + \delta\mathbf{u}, \mathbf{v}_k + \delta\mathbf{v})$ and $\mathbf{x} \leftarrow \mathbf{x}_{k+1}$;
        else
        Regularization;
        break;
      end if
    end for
    if iter mod $n_2 == 0$ **then**
      function Constraint Tightening:
      Calculate uncertainty (Eq. 7.43); and update constraints (Eq. 7.44);
    end if
    Dual update:
    $\hat{r} \leftarrow \frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{N} I_k^\beta(x_k^{(m)})$;
    if $\hat{r} > \Delta$ **then**
      $\lambda_L \leftarrow \lambda$;
      else
      $\lambda_U \leftarrow \lambda$;
    end if
    $\lambda \leftarrow \frac{1}{2}(\lambda_L + \lambda_U)$;
    if $|\hat{r} - \Delta| \leq \text{tol}$ **and** $(\lambda_U - \lambda_L) \leq \text{tol}_\lambda$ **then**
      break;
    end if
  end for
  Apply $\mathbf{u}_0$ and $\mathbf{v}_0$ to robot;
  $\mathbf{x}_0 \leftarrow$ measurement; $\mathbf{x}_1, ..., \mathbf{x}_{N-1} \leftarrow \mathbf{x}_2, ..., \mathbf{x}_N$; $\mathbf{u}_0, ..., \mathbf{u}_{N-2} \leftarrow \mathbf{u}_1, ..., \mathbf{u}_{N-1}$; $\mathbf{v}_0, ..., \mathbf{v}_{N-2} \leftarrow \mathbf{v}_1, ..., \mathbf{v}_{N-1}$;
  $N \leftarrow N - 1$;
end while

incorporated without stalling the optimization loop.

In practice, frequent updates of the tightened constraints can destabilize the recursion, since large shifts in feasible regions disrupt the quadratic approximation of the action-value function. To balance accuracy and stability, we employ a slower update schedule: constraint tightening is carried out every n_2 iterations (with $n_2 = n_1/2$ serving as a practical rule of thumb). This allows the optimizer to settle on consistent feedback gains before updating the constraints and re-optimizing under the refined feasible set. Both n_1 and n_2 may be tuned empirically, with larger horizons and more nonlinear dynamics requiring higher values.

Another essential feature of CC-GT-DDP is the dual update mechanism for enforcing the joint chance constraint. The dual multiplier λ penalizes tightened-set violations, and is updated via a bisection scheme based on Monte Carlo risk estimates. Intuitively, if the estimated risk of violation exceeds the prescribed budget, λ is increased; otherwise it is decreased. This guarantees convergence to a feasible policy while maintaining near-optimality bounds. The search interval $[\lambda_L, \lambda_U]$ and tolerances tol, tol_λ control the precision of this update and can be adjusted to balance runtime and accuracy.

Regularization techniques are also employed to guarantee numerical stability of matrix inversions in the Riccati-like backward pass, especially when constraints render certain directions ill-conditioned. As in standard DDP practice, we adopt a diminishing trust region for infeasible rollouts and apply diagonal regularization to the Hessians. This ensures smooth convergence even when disturbances push the system close to the constraint boundaries.

Overall, the CC-GT-DDP algorithm alternates between backward passes, forward passes, constraint tightening, and dual updates until either the iteration limit is reached or convergence criteria are satisfied. At each receding-horizon step, the first control/disturbance pair is applied to the system, the state is re-measured, and the horizon is

shifted. This iterative loop continues until the system reaches the goal region. Through this process, CC-GT-DDP produces policies that are not only optimal in the min–max sense, but also satisfy probabilistic safety guarantees under stochastic disturbances.

7.4 Numerical Examples

In this section we apply the CC-GT-DDP algorithm to a complicated and realistic example, namely, a quadrotor system in a constraint environment in four scenarios to validate the effectiveness of the proposed algorithm. The dynamic model of the quadrotor includes 12 states: 3 for position $(x, y, z)^T$, 3 for the Euler angles $(\phi, \theta, \psi)^T$, 3 for the velocity $(\dot{x}, \dot{y}, \dot{z})^T$, and 3 for the body angular rates $(p, q, r)^T$. The corresponding dynamics of the quadrotor with conflicting controls is given as follows:

$$\frac{dx}{dt} = f(x) + G(u + v), \tag{7.80}$$

where the complete dynamics of the system can be found in [292].

Remark 25. *In this work, stochastic disturbances are modeled as zero-mean Gaussian noise with known covariance, which is a standard and widely adopted assumption in control, robotics, and aerospace applications. Gaussian noise provides a practical and tractable means of capturing aggregate effects of sensor noise, unmodeled dynamics, environmental perturbations, and modeling inaccuracies, and is commonly used in both simulation-based validation and real-world experimental testing. This assumption enables analytical covariance propagation and systematic chance-constraint tightening, which are central to the proposed framework. In addition to stochastic disturbances, the proposed CC-GT-DDP explicitly accounts for structured and worst-case uncertainties through its game-theoretic Min–Max formulation. In this setting, the second player*

acts as an adversarial disturbance that actively seeks to degrade system performance and destabilize the trajectory, thereby representing intelligent or worst-case uncertainty beyond purely random noise. This dual treatment allows the framework to simultaneously capture random variability through Gaussian noise and adversarial effects through the disturbance player, providing a more realistic and robust representation of uncertainty in safety-critical systems.

7.4.1 Quadcopter Navigation Using CC-GT-DDP

In this scenario, the proposed Chance-Constrained Game-Theoretic Differential Dynamic Programming (GT-DDP) algorithm is applied to a quadcopter system navigating through a static environment with predefined obstacles. The objective is to demonstrate the algorithm's capability in generating safe and optimal trajectories under stochastic disturbances while accounting for game-theoretic interactions between two opposing agents u and v. The stochasticity is explicitly accounted for by performing a Monte Carlo simulation loop, where multiple trajectory rollouts are generated under sampled Gaussian process noise. The following cost function is defined for this scenario

$$\frac{1}{2}\int_0^{t_f}[(x-x_f)^T Q(x-x_f)] + u^T R_u u - v^T R_v v] + \frac{1}{2}(x-x_f)^T Q_f(x-x_f), \qquad (7.81)$$

7.4.1.0.1 Simulation parameters and implementation details For this scenario, we use a discrete-time step of $\Delta t = 0.02$ s and a prediction horizon of length $N = 50$, resulting in a total simulation duration of $T = 1.0$ s. The maximum number of backward–forward DDP iterations is set to 80. The quadrotor state dimension is $n_x = 12$ and the control dimension is $n_u = 4$. The running-state weighting matrix is chosen as $Q\text{diag}\big(10, 10, 10,\ 1, 1, 1,\ 0.1, 0.1, 0.1,\ 0.1, 0.1, 0.1\big)$, and the terminal cost

matrix is $Q_f = \mathrm{diag}\big(10^4, 10^4, 10^4,\ 10^3, 10^3, 10^3,\ 1, 1, 1,\ 1, 1, 1\big)$. The control cost matrices for the minimizer and maximizer are $R_u = 0.01\, I_4$ and $R_v = 10\, I_4$. The chance constraint confidence level is set to $\beta = 0.96$. The process disturbance is modeled as zero-mean Gaussian noise, $w_k \sim \mathcal{N}(0, \Sigma_w)$, with $\Sigma_w = 10^{-2} I_{12}$ used for covariance propagation. For Monte Carlo evaluation, $M = 100$ closed-loop rollouts are generated, with additive Gaussian noise of standard deviation $\sigma = 0.01$ applied to each state component to evaluate safety and robustness. Unless explicitly stated, all parameters remain identical across scenarios, with only the noise intensity or initial conditions varied to represent different uncertainty levels. In this scenario, the proposed CC-GT-DDP algorithm is applied to guide the quadcopter to a target position $[2.0, 2.0, 2.0]$ while safely avoiding predefined spherical obstacles under stochastic disturbances. Obstacle avoidance is enforced using a chance-constrained tightening strategy based on the closed-loop covariance propagation. The cost function penalizes deviation from the target, control effort, and adversarial disturbances using a min-max formulation. The algorithm is assessed using the following metrics:

Safety Rate: percentage of rollouts with no collisions.

Reachability Rate: percentage reaching within a threshold of the target.

Success Rate: percentage satisfying both safety and reachability.

Root-Mean-Square Deviation (RMSD): average root-mean-square deviation to the target.

Total State Variance: overall variance across all states.

Results show that the algorithm generates smooth, feasible, and safe trajectories. Both nominal and Monte Carlo mean trajectories remain collision-free, validating the effectiveness of CC-GT-DDP in achieving robust and safe navigation under uncertainty.

Fig. 7.1 illustrates the nominal and Monte Carlo-simulated trajectories of a quad-

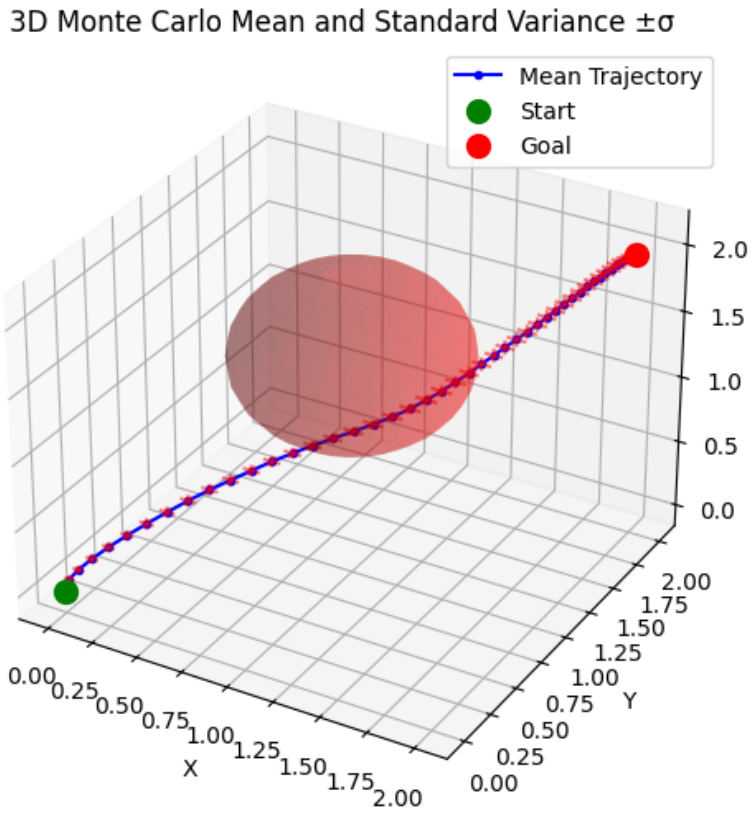


Figure 7.1: CC-GT-DDP Trajectories of 3D Quadcopter in Constrained Environment

copter navigating a 3D environment using the proposed Chance-Constrained Game-Theoretic Differential Dynamic Programming (CC-GT-DDP) algorithm. The quadcopter starts from an initial position and successfully reaches the target location while safely avoiding spherical obstacles under stochastic disturbances. The smoothness of the nominal trajectory demonstrates the optimality of the control policy, while the tight clustering of Monte Carlo rollouts around the nominal path highlights the robustness and safety of the method under uncertainty. The absence of any collisions in the rollouts validates the effectiveness of constraint tightening and closed-loop covariance propagation embedded within the CC-GT-DDP framework

Fig. 7.2 presents the time evolution of the quadcopter's velocity components along the x, y, and z axes during execution of the CC-GT-DDP control strategy. The plotted

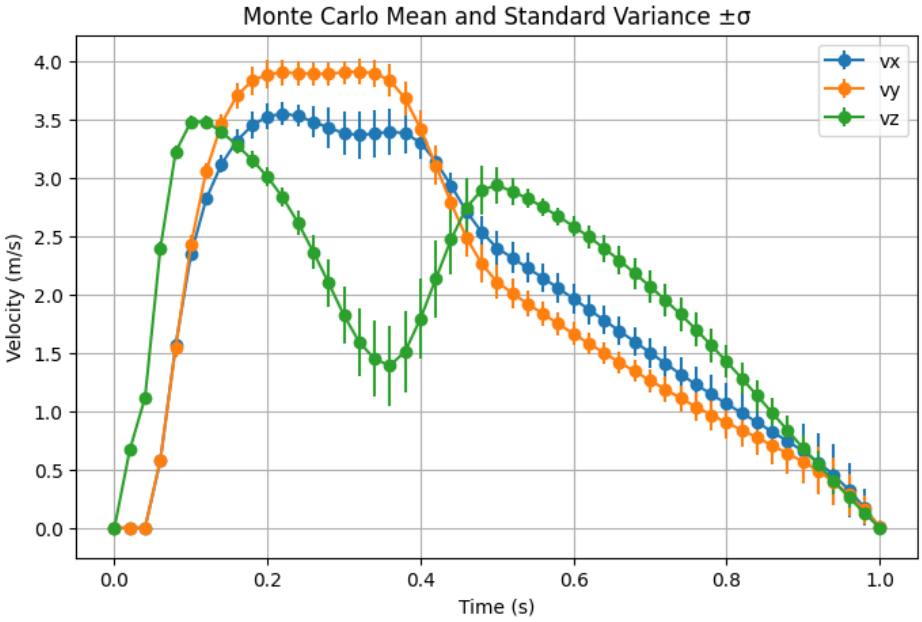


Figure 7.2: Trajectories of Velocity for Quadcopter.

velocities exhibit smooth transitions with no abrupt spikes, indicating stable control behavior even in the presence of disturbances. The convergence of velocity magnitudes over time confirms that the quadcopter achieves steady motion toward the target while respecting chance constraints. This result further validates the reliability of the optimized control policy in ensuring both safety and performance throughout the flight.

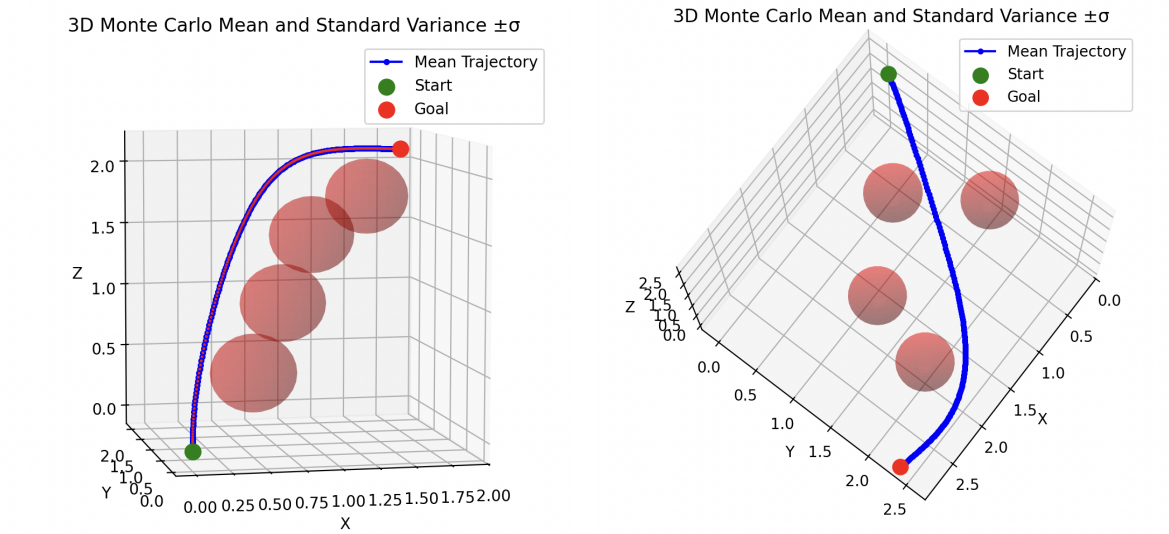
7.4.2 Quadcopter Navigation Using CC-MPC-GT-DDP

In this scenario, the CC-GT-DDP is embedded in a Model Predictive Control (MPC) framework and the resulting Chance-Constrained MPC–Game-Theoretic DDP (CC-MPC-GT-DDP) algorithm is applied to the quadcopter system.

7.4.2.0.1 Simulation parameters and implementation details In this scenario, a receding-horizon Min–Max DDP framework is implemented within a model predictive control (MPC) scheme. The single-shot DDP prediction horizon is set to

$N = 80$, with a discrete-time step of $\Delta t = 0.01$ s. At each MPC update, $N_{\text{apply}} = 10$ control inputs are applied before re-solving the DDP problem, over a total control budget of 200 steps. The maximum number of backward–forward DDP iterations per MPC update is 80. The quadrotor state dimension is $n_x = 12$ and the control dimension is $n_u = 4$. The running-state cost matrix is selected as $Q = \mathrm{diag}\big(1, 1, 1,\ 0.1, 0.1, 0.1,\ 0.1, 0.1, 0.1,\ 0.1, 0.1, 0.1\big)$, and the terminal cost matrix is $Q_f = \mathrm{diag}\big(5, 5, 5,\ 1, 1, 1,\ 1, 1, 1,\ 1, 1, 1\big)$. The control cost matrices for the minimizer and maximizer are chosen as $R_u = 0.01\, I_4$ and $R_v = 10\, I_4$. The process disturbance is modeled as zero-mean Gaussian noise, $w_k \sim \mathcal{N}(0, \Sigma_w)$, with covariance $\Sigma_w = 10^{-2} I_{12}$ used for covariance propagation within the chance-constrained formulation. During closed-loop execution, additional Gaussian noise is injected into the dynamics to emulate real process disturbances. The goal-reaching tolerance is set to 0.1 m. Monte Carlo evaluation is performed over 20 independent closed-loop rollouts. Unless otherwise stated, all remaining parameters are identical across scenarios, with this case representing a higher-uncertainty receding-horizon setting.

Fig. 7.3 showcase trajectories of quadcopter in a constrained 3D environment using the Chance-Constrained Model Predictive Control combined with Game-Theoretic DDP (CC-MPC-GT-DDP) algorithm with different placement of obstacles, start, and target point. Fig. 7.3a is the 3D plot of navigation from $(0, 0, 0)$ to $(2, 2, 2)$ and Fig. 7.3b is the 3D plot of navigation from $(0, 0, 1.75)$ to $(2, 2, 1)$. They illustrate how the quadcopter effectively maneuvers around obstacles while progressing toward the target location. The visualizations emphasize the safety margin maintained around obstacles, resulting from the constraint tightening and feedback-driven uncertainty propagation. By applying the first ten control inputs from each MPC iteration and re-planning with updated measurements, the algorithm balances long-term optimization with real-time robustness, as evidenced by the feasibility and safety of all visualized trajectories.



(a) Trajectory view for the first scenario

(b) Trajectory view for the second scenario

Figure 7.3: CC-MPC-GT-DDP Trajectories of 3D Quadcopter in Constrained Environment for Two Scenarios.

Table 7.1: Comparison of Chance Constraint GT-DDP with Embedded Barrier Function (EBF) for Quadrotor Example under the Non-Receding Horizon Scenario

Noise Level	Moderate		High	
Algorithm	**Proposed**	**EBF**	**Proposed**	**EBF**
Safety (%)	91	**92.5**	80.5	**81.7**
Reachability (%)	**92**	86.7	**85**	71.0
Success (%)	**91.1**	86.4	**75**	66.3
RMSD	**0.66**	0.72	**0.7**	0.75
Total State Variance	231	**230**	435.4	**430**

Table 7.2: Comparison of Chance Constraint GT-DDP with Embedded Barrier Function (EBF) for Quadrotor Example under the Receding Horizon Scenario

Noise Level	Moderate		High	
Algorithm	**Proposed**	**EBF**	**Proposed**	**EBF**
Safety (%)	**99.5**	98.6	**95.5**	94.3
Reachability (%)	**98.9**	98.1	**92.2**	86.5
Success (%)	**98.8**	96.7	**89.2**	81.6
RMSD	**0.45**	0.655	**0.8**	0.852
Total State Variance	**210**	228.2	**410.1**	421.8

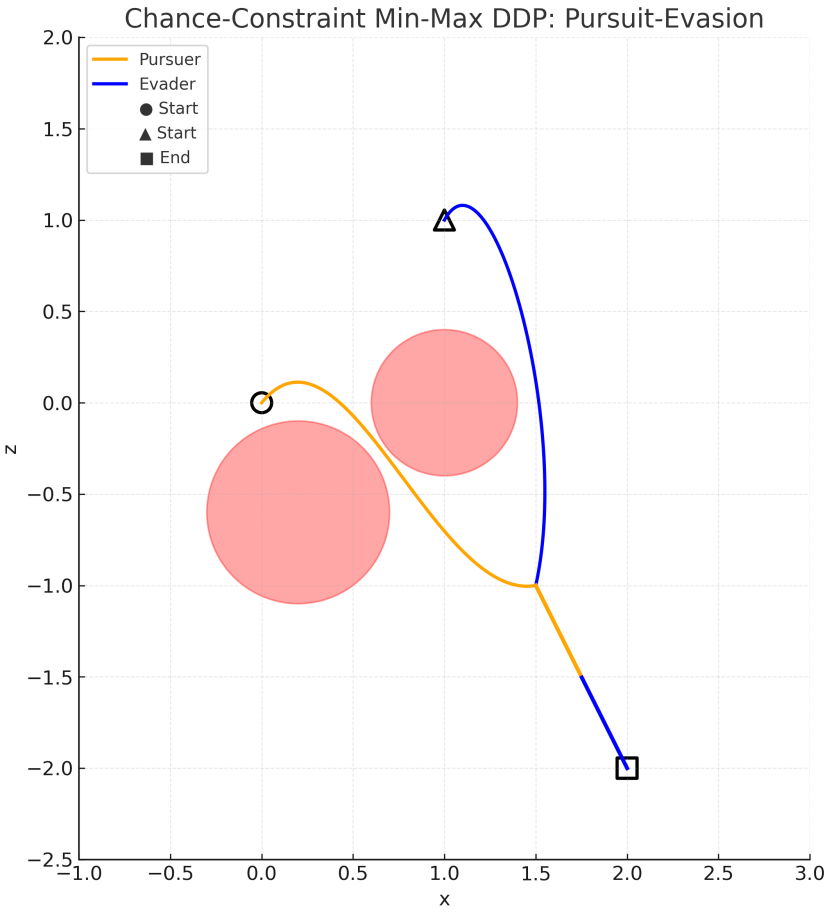


Figure 7.4: Pursuit-Evasion Trajectories.

Table 7.1 and Table 7.2 present a side-by-side comparison of the CC-GT-DDP formulations against an Embedded Barrier Function (EBF) baseline in two scenarios receding horizon and non-receding horizon under both moderate and high stochastic noise levels for a quadrotor navigation task. Under the non-receding horizon CC-GT-DDP scheme, the proposed method maintains a safety rate of 91% versus 92.5%for EBF at moderate noise, while substantially boosting reachability (92% vs. 86.7%) and overall success (91.1% vs. 86.4%), and reducing path deviation (RMSD $0.66m$ vs. $0.72m$) with almost identical total state variance (231 vs. 230). Even under high noise, it preserves superior reachability (85% vs. 71.0%) and success (75% vs. 66.3%) with modest

variance increase. Results in the non-receding horizon scenario show that although the CC-GT-DDP approach generally produces better outcomes, in two parameters—Safety and Total State Variance—the EBF method yields higher-quality results. When embedded in an MPC framework, CC-MPC-GT-DDP further amplifies these gains. At moderate noise it achieves 99.5% safety (vs. 98.6%), 98.9% reachability (vs. 98.1%), and an RMSD of only $0.45m$ (vs. $0.655m$), all while cutting total variance to 210 (vs. 228.2); similar trends hold at high noise (95.5% vs. 94.3% safety, 92.2% vs. 86.5% reachability, RMSD $0.8m$ vs. $0.852m$). These results demonstrate that incorporating chance-constraint tightening and game-theoretic disturbance modeling yields consistently more reliable, accurate, and robust quadrotor trajectories compared to the embedded barrier function framework in receding horizon scenario.

7.4.3 Safe Pursuit-Evasion Game

In this scenario, the Chance-Constrained Min-Max Differential Dynamic Programming (CC-Min-Max DDP) algorithm is applied to a two-dimensional quadcopter pursuit–evasion system. Each quadcopter is modeled with six states: two positions $(x, z)^T$, two velocities $(\dot{x}, \dot{z})^T$, the pitch angle θ, and the angular rate ω. The corresponding nonlinear dynamics of a single quadcopter are given by

$$\begin{aligned}
\dot{x} &= v_x, \\
\dot{z} &= v_z, \\
\dot{v}_x &= \tfrac{F}{m} \sin\theta, \\
\dot{v}_z &= \tfrac{F}{m} \cos\theta - g, \\
\dot{\theta} &= \omega, \\
\dot{\omega} &= \tfrac{M}{I},
\end{aligned} \tag{7.82}$$

where F and M denote the thrust and pitching moment, m is the mass, I is the rotational inertia, and g is gravity. The control inputs of the system are the thrust F and the pitching moment M, which directly govern the translational and rotational dynamics, respectively.

Since the system consists of both a pursuer and an evader, the full state vector is obtained by stacking their individual states, resulting in 12 total states. The combined dynamics with conflicting control inputs u (pursuer) and v (evader) can be expressed compactly as

$$\dot{\mathbf{x}} = f(\mathbf{x}) + G(u + v), \tag{7.83}$$

where $\mathbf{x} = [x_p^T, x_e^T]^T$, and x_p, x_e denote the pursuer and evader states, respectively.

The performance index for this pursuit–evasion game is defined as

$$\begin{aligned} J(u, v) = &\frac{1}{2} \int_0^{t_f} \Big[(x_p - x_e)^T Q (x_p - x_e) + u^T R_u u - v^T R_v v \Big] \, dt \\ &+ \frac{1}{2} (x_p(t_f) - x_e(t_f))^T Q_f (x_p(t_f) - x_e(t_f)), \end{aligned} \tag{7.84}$$

where the matrices $Q, Q_f \succeq 0$ penalize the separation between the agents, while R_u and R_v weight the effort of the pursuer and evader. Obstacles are incorporated as chance constraints, ensuring that safety is maintained with high probability under uncertainty. The pursuer as the first player u aims to minimize the distance to the evader, while the evader as the second player v seeks to maximize this distance and escape. Both agents are modeled using nonlinear quadcopter dynamics and interact through opposing objectives in the Game-Theoretic DDP framework. Chance constraints are incorporated to ensure safe maneuvering under stochastic disturbances, allowing both players to generate robust strategies in a competitive and uncertain environment.

7.4.3.0.1 Simulation parameters and implementation details This scenario considers a two-dimensional pursuit–evasion problem solved using chance-constrained Min–Max Differential Dynamic Programming. The system state consists of a pursuer and an evader, each modeled by a six-dimensional planar quadrotor dynamics, resulting in a total state dimension of $n_x = 12$. The control dimension for both players is $n_u = 2$. The discrete-time step is set to $\Delta t = 0.02$ s, and the planning horizon length is $N = 35$. The maximum number of backward–forward DDP iterations is fixed at 80. The running-state cost matrix for the pursuit–evasion objective is chosen as $Q = \mathrm{diag}\big(100, 100, 1, 1, 0.1, 0.1\big)$, and the terminal cost matrix is $Q_f = \mathrm{diag}\big(10^5, 10^5, 100, 100, 1, 1\big)$. The control cost matrices for the minimizer (pursuer) and maximizer (evader) are $R_u = \mathrm{diag}(0.01, 0.01)$ and $R_v = \mathrm{diag}(10, 10)$. Chance constraints are enforced through probabilistic obstacle avoidance, with confidence level $\beta = 0.95$. The process disturbance is modeled as zero-mean Gaussian noise, $w_k \sim \mathcal{N}(0, \Sigma_w)$, where the disturbance covariance is set to $\Sigma_w = 10^{-2} I_{12}$. A quadratic penalty with weight $\alpha = 10^3$ is applied to constraint violations to ensure smooth incorporation of safety constraints within the DDP backward pass. Two circular obstacles are placed in the environment, each defined by its center position and radius. Unless otherwise stated, all parameters remain fixed throughout the simulation.

Fig. 7.4 shows the trajectories of the pursuer and evader in a dynamic pursuit–evasion scenario under the CC-GT-DDP framework. Both agents are modeled as quadcopters with identical nonlinear dynamics and operate in a constrained environment with two spherical obstacles. The evader (blue trajectory) attempts to maintain separation and maximize the distance to the pursuer, while the pursuer (orange trajectory) dynamically adapts its strategy to minimize the distance to the evader over time. The pursuer successfully intercepts the evader by safely navigating around both obstacles using feedback-based control and constraint tightening. The smooth and safe

convergence of the trajectories illustrates the ability of CC-GT-DDP to handle adversarial interactions, enforce probabilistic safety, and achieve effective pursuit despite the presence of uncertainty and environmental constraints.

7.4.4 3D Safe Pursuit–Evasion Game

In this scenario, the Chance-Constrained Min–Max Differential Dynamic Programming (CC–Min–Max DDP) algorithm is applied to a three-dimensional pursuit–evasion problem involving quadcopters. Each agent is modeled as a 12-state quadcopter system. The system dynamics is identical to that used in Scenarios 7.4.1 and stage cost functions is

$$J(u,v) = \frac{1}{2}\int_0^{t_f} \Big[(x_p - x_e)^T Q (x_p - x_e) + u^T R_u u - v^T R_v v\Big]\, dt \tag{7.85}$$
$$+ \frac{1}{2}(x_p(t_f) - x_e(t_f))^T Q_f (x_p(t_f) - x_e(t_f)),$$

while the overall objective and safety requirements follow the formulation introduced in Scenario 7.4.3. The goal is to generate safe pursuit and evasion trajectories under stochastic disturbances while satisfying probabilistic collision-avoidance constraints.

7.4.4.0.1 Simulation parameters and implementation details This scenario considers a two-dimensional pursuit–evasion problem solved using chance-constrained Min–Max Differential Dynamic Programming. The system state consists of a pursuer and an evader, each modeled by a six-dimensional planar quadrotor dynamics, resulting in a total state dimension of $n_x = 24$. The control dimension for both players is $n_u = 2$. The discrete-time step is set to $\Delta t = 0.05$ s, and the planning horizon length is $N = 100$. The maximum number of backward–forward DDP iterations is fixed at 100. The running-state cost matrix for the pursuit–evasion objective is chosen as

$Q = \text{diag}\big(100, 100, 100, 1, 1, 1, 0.1, 0.1.0.1, 0.1, 0.1, 0.1\big)$, and the terminal cost matrix is $Q_f = \text{diag}\big(10^4, 10^4, 10^3, 10, 10, 10, 1, 1, 1, 0.1, 0.1, 0.1\big)$. The control cost matrices for the minimizer (pursuer) and maximizer (evader) are $R_u = 0.05\, I_4$ and $R_v = 5\, I_4$. Chance constraints are enforced through probabilistic obstacle avoidance, with confidence level $\beta = 0.95$. The process disturbance is modeled as zero-mean Gaussian noise, $w_k \sim \mathcal{N}(0, \Sigma_w)$, where the disturbance covariance is set to $\Sigma_w = 10^{-2} I_{12}$. A quadratic penalty with weight $\alpha = 10^4$ is applied to constraint violations to ensure smooth incorporation of safety constraints within the DDP backward pass. The initial positions for pursuer is $x_0 = [1.5, 1.5, 1.0]$ and for evader is $x_0 = [7.0, 6.2, 0.5]$. Two circular obstacles are placed in the environment, each defined by its center position and radius. Unless otherwise stated, all parameters remain fixed throughout the simulation.

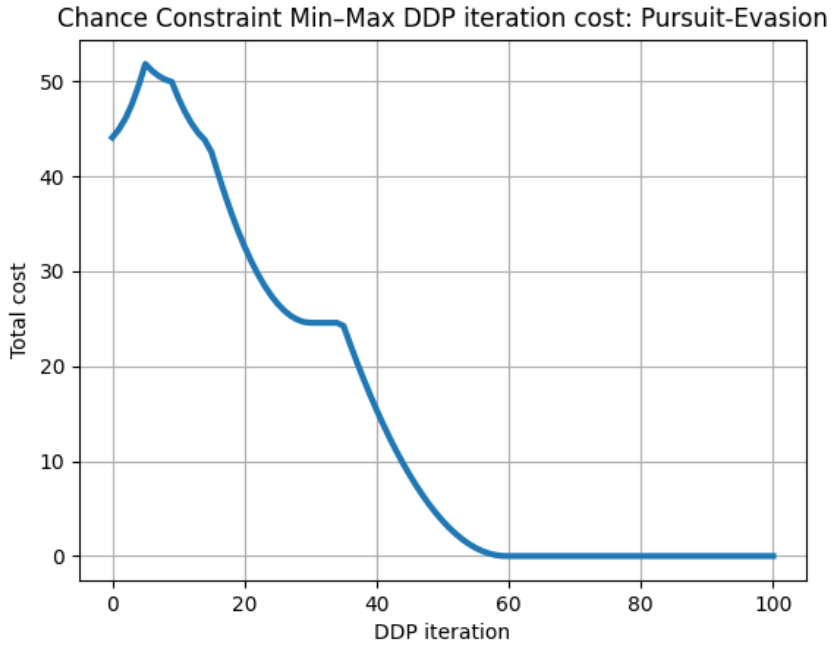


Figure 7.5: Cost Per Iteration of Pursuit-Evasion Game.

Fig. 7.5 shows the overall cost function in the pursuit-evasion game over the course of 100 iterations. The objective of pursuer is to minimize cost while the objective of

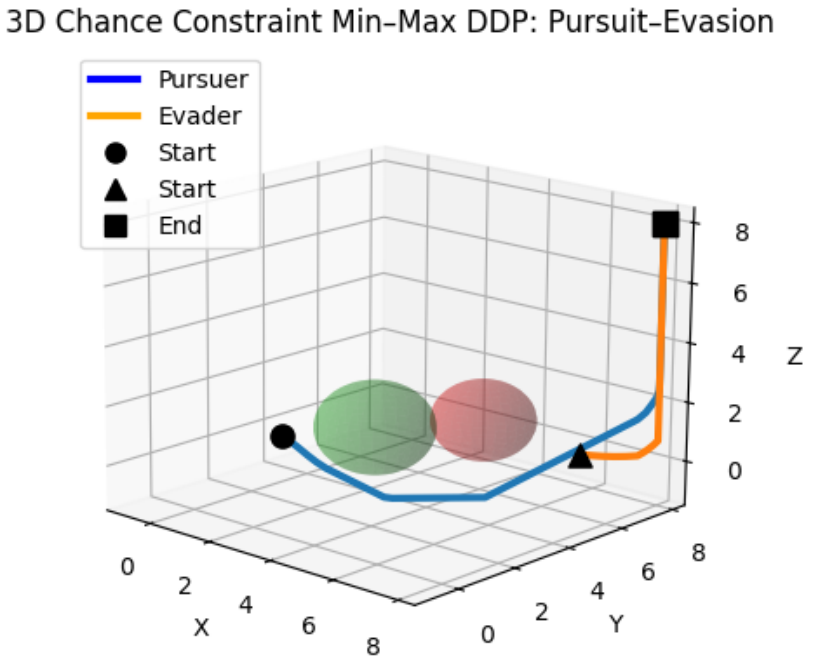


Figure 7.6: 3D Pursuit-Evasion Trajectories.

evader is to maximizing it. The monotonic convergence in the cost indicates that the algorithm effectively optimizes the control inputs to achieve the desired goal which is catching the evader by pursuer. Fig. 7.6 visualizes the resulting 3D trajectories of the pursuer and evader in the presence of obstacles. The evader adapts its motion to maintain separation, while the pursuer dynamically adjusts its path to intercept the evader without violating probabilistic collision-avoidance constraints. The absence of collisions and the smooth obstacle-circumventing paths demonstrate how feedback-based uncertainty propagation and chance-constraint tightening enable safe, robust, and strategically consistent behavior throughout the pursuit–evasion process.

Chapter 8

Continuous Time Differential Dynamic Programming for Nonzero-Sum Differential Games

8.1 Introduction

Optimal control has gained significant attention across diverse fields, including robotics, aerospace, automotive systems, and energy management, driven by the need for improved performance, efficiency, and reliability [293, 294, 295, 296]. Optimal control design for nonlinear systems aims to find a control strategy that minimizes a performance criterion while ensuring stability. Nonlinear dynamics requires advanced methods beyond traditional linear approaches, as they introduce challenges such as uncertainty and computational complexity [297]. Several algorithms, such as Differential Dynamic Programming (DDP) [4], Adaptive Dynamic Programming (ADP) [298], Model Predictive Control (MPC) [260], and Reinforcement Learning (RL) [299], have been developed to address these challenges and design optimal control strategies for nonlinear systems. These methods provide powerful tools to handle the complexities of nonlinear behavior while achieving objectives such as minimizing energy consumption or tracking error. As a result, they play a crucial role in enhancing the performance and robustness of

control systems in applications such as robotics, aerospace, and automotive systems.

An important challenge in the optimal control of nonlinear systems arises in multi-agent scenarios [300], where multiple controllers interact and influence the behavior of the whole system. These interactions are often modeled using differential games [163], which can be classified as zero-sum or nonzero-sum differential games [301, 302]. In zero-sum games, the gain of one player comes at the exact loss of the other player, which makes these frameworks ideal for adversarial environments where robustness to uncertainty and disturbance is critical. On the other hand, the objectives of the agents in many practical systems are not always completely opposing, which motivates the use of nonzero-sum approaches, where each agent pursues its individual goal, but their interactions are not purely opposed. Nonzero-sum algorithms are essential for real-world applications such as decentralized network systems [303] or multi-agent robotics [304], where agents must balance competition and cooperation, such as in multi-vehicle coordination or decentralized energy networks.

Due to the flexibility and realism offered by nonzero-sum frameworks, they have been increasingly adopted by researchers in combination with advanced control methods. Algorithms such as Adaptive Dynamic Programming (ADP) [305] and Reinforcement Learning (RL) [306] have been integrated with nonzero-sum strategies to handle complex interactions in multi-agent systems. These methods enable agents to optimize their objectives while adapting to changing environments and the strategies of other players. The ability of nonzero-sum frameworks to accommodate both cooperative and competitive dynamics makes them particularly useful for applications like multi-robot systems, autonomous driving, and distributed energy management [307, 308]. ADP has been utilized recently in many works as a practical tool to design optimal control for nonzero-sum games. A sliding mode control approach is combined with ADP in [301]. ADP is also utilized to design optimal control for nonlinear systems with input

constraints [309]. A data-driven ADP is designed for nonlinear multi-player unknown dynamics in [310]. An event-triggered robust optimal control is proposed through ADP for mismatched dynamics with uncertainty [311]. Other methods such as MPC and RL have also been used to design controller for multi-agent systems. The inverse optimal control technique is applied to deal with nonzero-sum MPC games with nonlinear dynamics [312]. An RL-based nonzero-sum game is studied for transmission systems [313]. An RL scheme is developed to learn the nash equilibrium solution of the multi-agent nonzero-sum games [314]. Some other works use data-based approaches and apply RL to design optimal control for nonzero-sum multi-agent games [306, 315, 316].

While previous works have applied ADP, MPC, and RL to design optimal control for nonzero-sum games, these methods come with certain limitations. ADP and RL often require extensive data for training and can struggle with convergence, especially in real-time applications with continuous dynamics. MPC, on the other hand, offers real-time feasibility but can become computationally expensive for complex multi-agent systems with long time horizons. In contrast, Differential Dynamic Programming (DDP) provides a more efficient framework for handling nonlinear dynamics by leveraging second-order approximations, making it well-suited for systems that require precise control. The ability of DDP to compute feedback control policies directly from system dynamics offers significant advantages in terms of performance and computational efficiency, especially in scenarios with tight real-time constraints. Additionally, DDP provides better scalability for multi-agent interactions, making it an attractive choice for designing optimal control in nonzero-sum games in comparison with ADP and MPC. However, like other methods, DDP also faces challenges, such as local convergence issues, but its overall efficiency and structure-driven approach make it a compelling alternative for multi-agent optimal control problems. Therefore, in This chapter, we design the optimal control algorithm for nonzero-sum differential games problem through the DDP

framework. The contribution of the paper is to solve the nonzero-sum differential game problem and obtain the control policies. Furthermore, the optimal update laws of the control policies of the multi-players and the corresponding set of backward differential equations are provided. In particular, the effect of the nonzero-sum formulation in the feed-forward and feedback parts of the optimal control policies is investigated.

The rest of the paper is organized as follows. In Section 8.2, some important notations are provided. In Section 8.3, the nonzero-sum differential game problem is formulated, and the backward differential equations and optimal control policies are derived. Finally, in Section 8.4, numerical examples and results are illustrated.

8.2 Notation

This section presents the essential notations employed throughout the paper. $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{R}^n$, and $\mathbb{R}^{n\times m}$ presents the set of all real numbers, positive real numbers, $n \times 1$ real column vectors, and $n \times m$ real matrices, respectively. $\mathcal{C}^1(\mathbb{R})$ and $\mathcal{C}^2(\mathbb{R})$ represent sets of functions whose first and second-order derivative exist and is continuous with domain $\mathbb{R}$. For any specified functions $\psi : \Re^n \times \Re \to \Re$, $\Psi : \Re^n \times \Re^m \times \Re \to \Re$ and $\Omega : \Re^n \times \Re \to \Re^n$, the following are given

$$\psi_{xx}(x,t) = \begin{pmatrix} \frac{\partial^2 \psi(x,t)}{\partial^2 x_1}, & \cdots & , \frac{\partial^2 \psi(x,t)}{\partial x_1 x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \psi(x,t)}{\partial x_1 \partial x_n}, & \cdots & , \frac{\partial^2 \psi(x,t)}{\partial^2 x_n} \end{pmatrix},$$

$$\Psi_{xu}(x,u,t) = \begin{pmatrix} \frac{\partial^2 \Psi(x,u,t)}{\partial x_1 \partial u_1}, & \cdots & , \frac{\partial^2 \Psi(x,u,t)}{\partial x_1 \partial u_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Psi(x,u,t)}{\partial x_n \partial u_1}, & \cdots & , \frac{\partial^2 \Psi(x,u,t)}{\partial x_n \partial u_n} \end{pmatrix},$$

$$\Omega_x(x,t) = \begin{pmatrix} \frac{\partial \Omega_1(x,t)}{\partial x_1}, & \cdots & , \frac{\partial \Omega_1(x,t)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \Omega_n(x,t)}{\partial x_1}, & \cdots & , \frac{\partial \Omega_n(x,t)}{\partial x_n} \end{pmatrix},$$

where ψ_{xx}, Ψ_{xu}, and Ω_x denote the Hessian of ψ with respect to x, Hessian of Ψ with respect to x and u, and the Jacobian matrix of Ω with respect to x, respectively. In This chapter, let i denote the player's number. Hence, $\bar{\mathcal{Q}}_{i,x}$, $\bar{\mathcal{Q}}_{i,xx}$ and $\bar{\mathcal{Q}}_{i,u_k u_p}$ represent Jacobian matrix of $\bar{\mathcal{Q}}$ with respect to x, Hessian of $\bar{\mathcal{Q}}$ with respect to x and Hessian of $\bar{\mathcal{Q}}$ with respect to x and u, respectively, for player number i.

For a given vector $a \in \mathbb{R}^n$ or a matrix $A \in \mathbb{R}^{n \times m}$, $a[i:j]$ refers to rows i through j (inclusive) of a, and $A[i:j]$ refers to rows i through j (inclusive) of A.

8.3 Nonzero-Sum Differential Dynamic Programming

8.3.1 Problem description

Consider the nonzero-sum differential game problem involving N agents with dynamics

$$\frac{dx(t)}{dt} = F\left(x(t), u_1(t), u_2(t), \ldots, u_N(t)\right), \quad x(t_0) = x_0, \tag{8.1}$$

subject to the cost function of the player i as

$$J_i(x_0, t_0) = \phi_i(x(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}_i(x(t), u_1(t), u_2(t), \ldots, u_N(t))dt, \tag{8.2}$$

where $x(t) \in \Re^n$ is the state of the system, $u_i(t) \in \Re^{m_{u_i}}$ is the control input, $\mathcal{L}_i$ is the running cost, and ϕ_i is the terminal cost of player $i \in \mathcal{N} \triangleq \{1, 2, ..., N\}$. The value of the cost function (8.2) for each player i is expressed as

$$V_i(x_0, t_0; u_{-i}) = \min_{u_i} \left\{ \phi_i(x(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}_i(x(t), u_i(t), u_{-i}(t))dt \right\}. \tag{8.3}$$

Here, $u_{-i} = \{u_1, ..., u_{i-1}, u_{i+1}, ...u_N\}$ denotes the control inputs of all players other than player i, which are treated as fixed admissible strategies when computing the best-response of player i in the Nash equilibrium sense.

8.3.2 Optimal control policy

It is known that the value function satisfies the Hamilton–Jacobi–Bellman (HJB) partial differential equation [8]. Specifically, the value of the game for player i in (8.3) follows the equation

$$-\frac{\partial V_i(x, t; u_{-i})}{\partial t} = \min_{u_i} \left\{ \mathcal{L}_i(x, u_i, u_{-i}) + V_{i,x}(x, t)^T F(x, u_i, u_{-i}) \right\}, \tag{8.4}$$

with the terminal condition $V_i(x(t_f), t_f; u_{-i}) = \phi_i(x(t_f), t_f)$.

The above Hamilton–Jacobi–Bellman equation is evaluated conditionally on u_{-i}, and the optimization is carried out only with respect to u_i. We omit the time dependence t from the equation terms for simplicity of representation. Given the nominal state and control histories $(\bar{x}, \bar{u}_1, \bar{u}_2, ..., \bar{u}_N)$, the local nonlinear dynamics (8.1) can be expressed

as

$$\frac{dx}{dt} = F(\bar{x} + \delta x, \bar{u}_1 + \delta u_1, \bar{u}_2 + \delta u_2, \ldots, \bar{u}_N + \delta u_N), \tag{8.5}$$

where

$$\delta x = x - \bar{x}, \tag{8.6a}$$

$$\delta u_i = u_i - \bar{u}_i, \quad i = 1, 2, ..., N. \tag{8.6b}$$

The linearized dynamics around $(\bar{x}, \bar{u}_1, \bar{u}_2, ..., \bar{u}_N)$ is obtained as

$$\frac{d\delta x}{dt} = \bar{F}_x \delta x + \bar{F}_{u_1} \delta u_1 + \bar{F}_{u_2} \delta u_2 + ... + \bar{F}_{u_N} \delta u_N, \tag{8.7}$$

where $\bar{F}_x$, $\bar{F}_{u_1}$,
$\bar{F}_{u_2}$,...,$\bar{F}_{u_N}$ stands for $F_x(\bar{x}, \bar{u}_1, \bar{u}_2, ..., \bar{u}_N)$, $F_{u_1}(\bar{x}, \bar{u}_1, \bar{u}_2, ..., \bar{u}_N)$, $F_{u_2}(\bar{x}, \bar{u}_1, \bar{u}_2, ..., \bar{u}_N)$,
$\ldots$,
$F_{u_N}(\bar{x}, \bar{u}_1, \bar{u}_2, ..., \bar{u}_N)$, respectively. The optimal control update laws are obtained iteratively through the following theorem.

Theorem 15. *If*

$$\frac{d\delta x}{dt} = \bar{F}_x \delta x + \bar{F}_{u_1} \delta u_1 + \bar{F}_{u_2} \delta u_2 + \cdots + \bar{F}_{u_N} \delta u_N, \tag{8.8}$$

be the linearized dynamics around the nominal trajectories subject to the cost function (8.2) and the global control update law is given by

$$\delta u = K_u \delta x + I_u, \tag{8.9}$$

with global feedback and feedforward gain matrices as

$$K_u = -Q_{uu}^{-1} Q_{ux}, \quad I_u = -Q_{uu}^{-1} Q_u, \tag{8.10}$$

where

$$Q_{ux} = \begin{pmatrix} Q_{1,u_1 x} \\ Q_{2,u_2 x} \\ \vdots \\ Q_{i,u_N x} \end{pmatrix}, \quad Q_u = \begin{pmatrix} Q_{1,u_1} \\ Q_{2,u_2} \\ \vdots \\ Q_{i,u_N} \end{pmatrix}, \tag{8.11}$$

$$Q_{uu} = \begin{pmatrix} Q_{1,u_1 u_1} & Q_{1,u_1 u_2} & \cdots & Q_{1,u_1 u_N} \\ Q_{2,u_2 u_1} & Q_{2,u_2 u_2} & \cdots & Q_{2,u_2 u_N} \\ \vdots & \vdots & \ddots & \vdots \\ Q_{i,u_N u_1} & Q_{i,u_N u_2} & \cdots & Q_{i,u_N u_N} \end{pmatrix},$$

and

$$\bar{\mathcal{Q}}_{i,u_k} = \bar{F}_{u_k}^T \bar{V}_{i,x} + \bar{\mathcal{L}}_{i,u_k}, \tag{8.12a}$$

$$\bar{\mathcal{Q}}_{i,u_k u_k} = \bar{\mathcal{L}}_{i,u_k u_k}, \tag{8.12b}$$

$$\bar{\mathcal{Q}}_{i,u_k x} = \bar{F}_{u_k}^T \bar{V}_{i,xx} + \bar{\mathcal{L}}_{i,u_k x}. \tag{8.12c}$$

Then, the optimal control update law for each player $i \in \{1, \ldots, N\}$ is given by

$$\delta u_i = K_{u_i} \delta x + I_{u_i}. \tag{8.13}$$

Here $Q_{uu} \in \mathbb{R}^{M \times M}$, $Q_u \in \mathbb{R}^M$, and $Q_{ux} \in \mathbb{R}^{M \times n}$, with $M = \sum_{i=1}^N m_{u_i}$ being the total dimension of the combined control vector. Individual gains K_{u_i} and I_{u_i} for each player

are given as

$$I_{u_i} = I_u[s_i + 1 : s_i + m_{u_i}], \quad K_{u_i} = K_u[s_i + 1 : s_i + m_{u_i}], \tag{8.14}$$

where $s_i = \sum_{j=1}^{i-1} m_{u_j}, i = 2, \ldots, n,$ *and* $s_1 = 0.$

$$\delta u_i = K_{u_i} \delta x + \epsilon_i I_{u_i}. \tag{8.15}$$

The step-size parameter $\epsilon_i \in (0, 1]$ *is introduced to enhance numerical stability and empirically facilitate convergence of the iterative updates. As in classical DDP methods, the resulting solution corresponds to a locally optimal feedback Nash equilibrium, with convergence properties that are inherently local due to the reliance on second-order Taylor expansions around nominal trajectories. Formal global convergence or stability guarantees are beyond the scope of this work.*

Remark 26. *The proposed nonzero-sum DDP formulation assumes that the block matrix* Q_{uu} *is locally invertible in order to compute the coupled feedback and feedforward gains. In general nonzero-sum differential games,* Q_{uu} *may be indefinite due to the interaction between multiple players with distinct objectives. In practice, invertibility can be ensured locally by appropriate choice of running cost weights or by applying standard regularization techniques commonly used in DDP, such as Levenberg–Marquardt regularization, to improve numerical conditioning*

Proof. To derive the control update law presented in Theorem 15, we perform a local quadratic expansion of the cost function and a first-order linearization of the system dynamics. The resulting player-wise optimality condition leads to the following expression. If we extend the left-hand side of the equation (8.4) with respect to the Taylor

series, we obtain

$$\frac{\partial V_i(x,t)}{\partial t} = \frac{\partial V_i(\bar{x}+\delta x,t)}{\partial t} \approx \frac{\partial \bar{V}_i}{\partial t} + \frac{\partial \bar{V}_{i,x}^T}{\partial t}\delta x + \frac{1}{2}\delta x^T \frac{\partial \bar{V}_{i,xx}}{\partial t}\delta x, \tag{8.16}$$

we have

$$\frac{d\bar{V}_i}{dt} = \frac{\partial \bar{V}_i}{\partial t} + \bar{V}_{i,x}^T \frac{dx}{dt} = \frac{\partial \bar{V}_i}{\partial t} + \bar{V}_{i,x}^T \bar{F}, \tag{8.17a}$$

$$\frac{\partial \bar{V}_i}{\partial t} = \frac{d\bar{V}_i}{dt} - \bar{V}_{i,x}^T \bar{F}, \tag{8.17b}$$

$$\frac{\partial \bar{V}_{i,x}}{\partial t} = \frac{d\bar{V}_{i,x}}{dt} - \bar{V}_{i,xx} \bar{F}, \tag{8.17c}$$

$$\frac{\partial \bar{V}_{i,xx}}{\partial t} = \frac{d\bar{V}_{i,xx}}{dt} - \sum_{j=1}^{n} \bar{V}_{i,xxx}^{(j)} \bar{F}^{(j)}, \tag{8.17d}$$

where $\bar{V}_{i,xxx}^{(j)}$ represents the Hessian matrix of the j-th element of $\bar{V}_{i,x}$ and $\bar{F}^{(j)}$ shows the j-th element of $\bar{F}$. The left hand side of (8.4) becomes

$$\begin{aligned} -\frac{\partial V_i(x,t)}{\partial t} \approx & -\frac{d\bar{V}_i}{dt} - \frac{d\bar{V}_{i,x}^T}{dt}\delta x - \frac{1}{2}\delta x^T \frac{d\bar{V}_{i,xx}}{dt}\delta x \\ & + \bar{V}_{i,x}^T \bar{F} + \delta x^T \bar{V}_{i,xx} \bar{F} + \frac{1}{2}\delta x^T \left(\sum_{j=1}^{n} \bar{V}_{i,xxx}^{(j)} \bar{F}^{(j)} \right) \delta x. \end{aligned} \tag{8.18}$$

The extension of the right hand side of (8.4) via Taylor series obtains

$$\begin{aligned}
&\mathcal{L}_i(x, u_1, \ldots, u_N, t) = \mathcal{L}_i(\bar{x} + \delta x, \bar{u}_1 + \delta u_1, \ldots, \bar{u}_N + \delta u_N, t) \\
&\approx \bar{\mathcal{L}}_i + \bar{\mathcal{L}}_{i,x}^T \delta x + \bar{\mathcal{L}}_{i,u_1}^T \delta u_1 + \bar{\mathcal{L}}_{i,u_2}^T \delta u_2 + ... + \bar{\mathcal{L}}_{i,u_N}^T \delta u_N \\
&+ \frac{1}{2} \begin{pmatrix} \delta x \\ \delta u_1 \\ \delta u_2 \\ \vdots \\ \delta u_N \end{pmatrix}^T \begin{pmatrix} \bar{\mathcal{L}}_{i,xx} & \bar{\mathcal{L}}_{i,xu_1} & \bar{\mathcal{L}}_{i,xu_2} & \cdots & \bar{\mathcal{L}}_{i,xu_N} \\ \bar{\mathcal{L}}_{i,u_1x} & \bar{\mathcal{L}}_{i,u_1u_1} & \bar{\mathcal{L}}_{i,u_1u_2} & \cdots & \bar{\mathcal{L}}_{i,u_1u_N} \\ \bar{\mathcal{L}}_{i,u_2x} & \bar{\mathcal{L}}_{i,u_2u_1} & \bar{\mathcal{L}}_{i,u_2u_2} & \cdots & \bar{\mathcal{L}}_{i,u_2u_N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \bar{\mathcal{L}}_{i,u_Nx} & \bar{\mathcal{L}}_{i,u_Nu_1} & \bar{\mathcal{L}}_{i,u_Nu_2} & \cdots & \bar{\mathcal{L}}_{i,u_Nu_N} \end{pmatrix} \begin{pmatrix} \delta x \\ \delta u_1 \\ \delta u_2 \\ \vdots \\ \delta u_N \end{pmatrix},
\end{aligned} \tag{8.19}$$

by expanding $V_{i,x}$ around $\bar{x}$ we have

$$V_{i,x}(x, t) = V_{i,x}(\bar{x} + \delta x, t) \approx \bar{V}_{i,x} + \bar{V}_{i,xx}\delta x + \frac{1}{2}\bar{W}_i, \tag{8.20}$$

where

$$\bar{W}_i^{(j)} = \delta x^T \bar{V}_{i,xxx}^{(j)} \delta x. \tag{8.21}$$

The same expansion of the dynamics system proposes

$$\begin{aligned}
F(x, u_1, \ldots, u_N, t) &= F(\bar{x} + \delta x, \bar{u}_1 + \delta u_1, \ldots, \bar{u}_N + \delta u_N, t) \\
&\approx \bar{F} + \bar{F}_x \delta x + \bar{F}_{u_1} \delta u_1 + \bar{F}_{u_2} \delta u_2 + ... + \bar{F}_{u_N} \delta u_N.
\end{aligned} \tag{8.22}$$

The right hand side of (8.4) can be written

$$\min_{\delta u_i}\Bigg\{\bar{\mathcal{L}}_i+\bar{\mathcal{L}}_{i,x}^T\delta x+\sum_{k=1}^{N}\bar{\mathcal{L}}_{iu_k}^T\delta u_k$$
$$+\frac{1}{2}\begin{pmatrix}\delta x\\ \delta u_1\\ \delta u_2\\ \vdots\\ \delta u_N\end{pmatrix}^T\begin{pmatrix}\bar{\mathcal{L}}_{i,xx} & \bar{\mathcal{L}}_{i,xu_1} & \bar{\mathcal{L}}_{i,xu_2} & \cdots & \bar{\mathcal{L}}_{i,xu_N}\\ \bar{\mathcal{L}}_{i,u_1x} & \bar{\mathcal{L}}_{i,u_1u_1} & \bar{\mathcal{L}}_{i,u_1u_2} & \cdots & \bar{\mathcal{L}}_{i,u_1u_N}\\ \bar{\mathcal{L}}_{i,u_2x} & \bar{\mathcal{L}}_{i,u_2u_1} & \bar{\mathcal{L}}_{i,u_2u_2} & \cdots & \bar{\mathcal{L}}_{i,u_2u_N}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \bar{\mathcal{L}}_{i,u_Nx} & \bar{\mathcal{L}}_{i,u_Nu_1} & \bar{\mathcal{L}}_{i,u_Nu_2} & \cdots & \bar{\mathcal{L}}_{i,u_Nu_N}\end{pmatrix}\begin{pmatrix}\delta x\\ \delta u_1\\ \delta u_2\\ \vdots\\ \delta u_N\end{pmatrix}$$
$$+\bar{V}_{i,x}^T\bar{F}+\bar{V}_{i,x}^T\bar{F}_x\delta x+\sum_{k=1}^{N}\bar{V}_{i,x}^T\bar{F}_{u_k}\delta u_k+\delta x^T\bar{V}_{i,xx}\bar{F}+\delta x^T\bar{V}_{i,xx}\bar{F}_x\delta x$$
$$+\delta x^T\bar{V}_{i,xx}\sum_{k=1}^{N}\bar{F}_{u_k}\delta u_k+\frac{1}{2}\bar{W}_i^T\bar{F},\tag{8.23}$$

consider the definition that

$$\frac{1}{2}\bar{W}_i^T\bar{F}=\frac{1}{2}\sum_{j=1}^{n}\left(\delta x^T\bar{V}_{i,xxx}^{(j)}\delta x\bar{F}^{(j)}\right)=\frac{1}{2}\delta x^T\left(\sum_{j=1}^{n}\bar{V}_{i,xxx}^{(j)}\bar{F}^{(j)}\right)\delta x,\tag{8.24}$$

by eliminating extra terms and doing mathematical operations, for $k=1,...,N$, we have

$$
\begin{aligned}
&-\frac{d\bar{V}_i}{dt}-\delta x^T\frac{d\bar{V}_{i,x}}{dt}-\frac{1}{2}\delta x^T\frac{d\bar{V}_{i,xx}}{dt}\delta x=\min_{\delta u_i}\Bigg\{\bar{\mathcal{L}}_i+\bar{\mathcal{L}}_{i,x}^T\delta x+\sum_{k=1}^{N}\bar{\mathcal{L}}_{iu_k}^T\delta u_k\\
&+\frac{1}{2}\begin{pmatrix}\delta x\\ \delta u_1\\ \vdots\\ \delta u_N\end{pmatrix}^T\begin{pmatrix}\bar{\mathcal{L}}_{i,xx} & \bar{\mathcal{L}}_{i,xu_1} & \cdots & \bar{\mathcal{L}}_{i,xu_N}\\ \bar{\mathcal{L}}_{i,u_1x} & \bar{\mathcal{L}}_{i,u_1u_1} & \cdots & \bar{\mathcal{L}}_{i,u_1u_N}\\ \vdots & \vdots & \ddots & \vdots\\ \bar{\mathcal{L}}_{i,u_Nx} & \bar{\mathcal{L}}_{i,u_Nu_1} & \cdots & \bar{\mathcal{L}}_{i,u_Nu_N}\end{pmatrix}\begin{pmatrix}\delta x\\ \delta u_1\\ \vdots\\ \delta u_N\end{pmatrix}\\
&+\bar{V}_{i,x}^T\bar{F}_x\delta x+\sum_{k=1}^{N}\bar{V}_{i,x}^T\bar{F}_{u_k}\delta u_k+\delta x^T\bar{V}_{i,xx}\bar{F}_x\delta x+\delta x^T\bar{V}_{i,xx}\sum_{k=1}^{N}\bar{F}_{u_k}\delta u_k\\
&=\min_{\delta u_i}\{\bar{\mathcal{L}}_i+\delta x^T\bar{\mathcal{Q}}_{i,x}+\sum_{k=1}^{N}\delta u_k^T\bar{\mathcal{Q}}_{i,u_k}\\
&+\frac{1}{2}\begin{pmatrix}\delta x\\ \delta u_1\\ \vdots\\ \delta u_N\end{pmatrix}^T\begin{pmatrix}\bar{Q}_{i,xx} & \bar{Q}_{i,xu_1} & \cdots & \bar{Q}_{i,xu_N}\\ \bar{Q}_{i,u_1x} & \bar{Q}_{i,u_1u_1} & \cdots & \bar{Q}_{i,u_1u_N}\\ \vdots & \vdots & \ddots & \vdots\\ \bar{Q}_{i,u_Nx} & \bar{Q}_{i,u_Nu_1} & \cdots & \bar{Q}_{i,u_Nu_N}\end{pmatrix}\begin{pmatrix}\delta x\\ \delta u_1\\ \vdots\\ \delta u_N\end{pmatrix}\},
\end{aligned}
\tag{8.25}
$$

where

$$\bar{\mathcal{Q}}_{i,x}=\bar{F}_x^T\bar{V}_{i,x}+\bar{\mathcal{L}}_{i,x},\tag{8.26a}$$

$$\bar{\mathcal{Q}}_{i,u_k}=\bar{F}_{u_k}^T\bar{V}_{i,x}+\bar{\mathcal{L}}_{i,u_k},\tag{8.26b}$$

$$\bar{\mathcal{Q}}_{i,xx}=\bar{\mathcal{L}}_{i,xx}+\bar{V}_{i,xx}\bar{F}_x+\bar{F}_x^T\bar{V}_{i,xx},\tag{8.26c}$$

$$\bar{\mathcal{Q}}_{i,u_ku_k}=\bar{\mathcal{L}}_{i,u_ku_k},\tag{8.26d}$$

$$\bar{\mathcal{Q}}_{i,u_kx}=\bar{F}_{u_k}^T\bar{V}_{i,xx}+\bar{\mathcal{L}}_{i,u_kx},\tag{8.26e}$$

$$\bar{\mathcal{Q}}_{i,u_ku_p}=\bar{\mathcal{L}}_{i,u_pu_k}.\tag{8.26f}$$

To find the optimal control law for each player, we open the equation (8.25), set

the "i" equal to the number of player and compute the the gradients of the expression with respect to δu_i and make them equal to zero to obtain

$$\begin{aligned}
\delta u_1 &= -Q_{1,u_1u_1}^{-1}\Big(Q_{1,u_1x}\delta x + \sum_{\substack{p=1\\p\neq 1}}^{N} Q_{1,u_1u_p}\delta u_p + Q_{1,u_1}\Big),\\
\delta u_2 &= -Q_{2,u_2u_2}^{-1}\Big(Q_{2,u_2x}\delta x + \sum_{\substack{p=1\\p\neq 2}}^{N} Q_{2,u_2u_p}\delta u_p + Q_{2,u_2}\Big),\\
&\vdots\\
\delta u_N &= -Q_{i,u_Nu_N}^{-1}\Big(Q_{i,u_Nx}\delta x + \sum_{\substack{p=1\\p\neq N}}^{N} Q_{i,u_Nu_p}\delta u_p + Q_{i,u_N}\Big).
\end{aligned} \tag{8.27}$$

Lets consider

$$\delta u = \begin{pmatrix} \delta u_1\\ \delta u_2\\ \vdots\\ \delta u_N \end{pmatrix}, \quad Q_{ux} = \begin{pmatrix} Q_{1,u_1x}\\ Q_{2,u_2x}\\ \vdots\\ Q_{i,u_Nx} \end{pmatrix}, \quad Q_u = \begin{pmatrix} Q_{1,u_1}\\ Q_{2,u_2}\\ \vdots\\ Q_{i,u_N} \end{pmatrix},$$

$$Q_{uu} = \begin{pmatrix} Q_{1,u_1u_1} & Q_{1,u_1u_2} & \cdots & Q_{1,u_1u_N}\\ Q_{2,u_2u_1} & Q_{2,u_2u_2} & \cdots & Q_{2,u_2u_N}\\ \vdots & \vdots & \ddots & \vdots\\ Q_{i,u_Nu_1} & Q_{i,u_Nu_2} & \cdots & Q_{i,u_Nu_N} \end{pmatrix}, \tag{8.28}$$

$$Q_{uu}\delta u = -Q_{ux}\delta x - Q_u, \tag{8.29}$$

we solve for δu

$$\delta u = -Q_{uu}^{-1}(Q_{ux}\delta x + Q_u). \tag{8.30}$$

From the above equation, the global control update is given by

$$\delta u = -\left(Q_{uu}^{-1}Q_{ux}\right)\delta x - \left(Q_{uu}^{-1}Q_u\right). \tag{8.31}$$

By defining the global feedback gain K_u and feedforward gain I_u as

$$\begin{aligned} K_u &= -Q_{uu}^{-1}Q_{ux}, \\ I_u &= -Q_{uu}^{-1}Q_u, \end{aligned} \tag{8.32}$$

we obtain the desired form for control update laws as

$$\delta u = K_u\delta x + I_u. \tag{8.33}$$

where individual gains K_{u_i} and I_{u_i} for each player are extracted from global gains (8.32). Which this completes the proof. □

The time history of $V_{i,x}$ and $V_{i,xx}$ is needed for evaluating the terms in (8.12). Accordingly, by substituting the control update laws (8.13) into (8.4), we derive the update laws for the value function and its first- and second-order state derivatives. The outcomes are summarized in the following theorem.

Theorem 16. *The backward ordinary differential equations for the value function and*

its first order and second order partial derivatives with respect to x_i are computed as

$$-\frac{d\bar{V}_i}{dt} = \bar{\mathcal{L}}_i + \sum_{k=1}^{N} I_{u_k}^T \bar{Q}_{i,u_k} + \frac{1}{2}\sum_{k=1}^{N}\sum_{p=1}^{N} I_{u_k}^T \bar{Q}_{i,u_k u_p} I_{u_k}, \tag{8.34a}$$

$$-\frac{d\bar{V}_{i,x}}{dt} = \bar{Q}_{i,x} + \sum_{k=1}^{N} K_{u_k}^T \bar{Q}_{i,u_k} + \sum_{k=1}^{N} \bar{Q}_{i,u_k x}^T I_{u_k} + \sum_{k=1}^{N}\sum_{p=1}^{N} K_{u_k}^T \bar{Q}_{i,u_k u_p} I_{u_p}, \tag{8.34b}$$

$$-\frac{d\bar{V}_{i,xx}}{dt} = \sum_{k=1}^{N} K_{u_k}^T \bar{Q}_{i,u_k x} + \sum_{k=1}^{N} \bar{Q}_{i,u_k x}^T K_{u_k} + \bar{Q}_{i,xx} + \sum_{k=1}^{N}\sum_{p=1}^{N} K_{u_k}^T Q_{i,u_k u_p} K_{u_p}, \tag{8.34c}$$

where

$$\bar{\mathcal{Q}}_{i,x} = \bar{F}_x^T \bar{V}_{i,x} + \bar{\mathcal{L}}_{i,x}, \tag{8.35a}$$

$$\bar{\mathcal{Q}}_{i,xx} = \bar{\mathcal{L}}_{i,xx} + \bar{V}_{i,xx}\bar{F}_x + \bar{F}_x^T \bar{V}_{i,xx}, \tag{8.35b}$$

$$\bar{\mathcal{Q}}_{i,u_k u_p} = \bar{\mathcal{L}}_{i,u_k u_p}, \tag{8.35c}$$

$$\tag{8.35d}$$

under terminal conditions

$$\bar{V}_i(t_f) = \phi_i(\bar{x}(t_f), t_f), \tag{8.36a}$$

$$\bar{V}_{i,x}(t_f) = \phi_{i,x}(\bar{x}(t_f), t_f), \tag{8.36b}$$

$$\bar{V}_{i,xx}(t_f) = \phi_{i,xx}(\bar{x}(t_f), t_f). \tag{8.36c}$$

Proof. To obtain the update law, its first- and second-order partial derivatives of the

value function, put the update control laws (8.13) to (8.4). We have

$$
\begin{aligned}
&-\frac{d\bar{V}_i}{dt}-\delta x^T\frac{d\bar{V}_{i,x}}{dt}-\frac{1}{2}\delta x^T\frac{d\bar{V}_{i,xx}}{dt}\delta x=\min_{\delta u_i}\{\bar{\mathcal{L}}_i+\delta x^T\bar{\mathcal{Q}}_{i,x}+\sum_{k=1}^{N}\delta u_k^T\bar{\mathcal{Q}}_{i,u_k}\\
&+\frac{1}{2}\begin{pmatrix}\delta x\\ \delta u_1\\ \vdots\\ \delta u_N\end{pmatrix}^T\begin{pmatrix}\bar{Q}_{i,xx} & \bar{Q}_{i,xu_1} & \cdots & \bar{Q}_{i,xu_N}\\ \bar{Q}_{i,u_1x} & \bar{Q}_{i,u_1u_1} & \cdots & \bar{Q}_{i,u_1u_N}\\ \vdots & \vdots & \ddots & \vdots\\ \bar{Q}_{i,u_Nx} & \bar{Q}_{i,u_Nu_1} & \cdots & \bar{Q}_{i,u_Nu_N}\end{pmatrix}\begin{pmatrix}\delta x\\ \delta u_1\\ \vdots\\ \delta u_N\end{pmatrix}\},
\end{aligned}
\tag{8.37}
$$

First, we open the matrix and set $\delta u_k=(I_{u_k}+K_{u_k}\delta x)$, for $k=1,2,...,N$, in equation (8.37). Next, we collect all terms in the right-hand side of the (8.37) as zeroth-,first-, and second-order expressions of δx. Then, put the coefficients of δx in the left-hand side and right-hand side of the (8.37) and compute the backward propagation equations with respect to the value function and its first- and second-order partial derivatives as

$$
-\frac{d\bar{V}_i}{dt}=\bar{\mathcal{L}}_i+\sum_{k=1}^{N}I_{u_k}^T\bar{Q}_{i,u_k}+\frac{1}{2}\sum_{k=1}^{N}\sum_{p=1}^{N}I_{u_k}^T\bar{Q}_{i,u_ku_p}I_{u_k} \tag{8.38a}
$$

$$
-\frac{d\bar{V}_{i,x}}{dt}=\bar{Q}_{i,x}+\sum_{k=1}^{N}K_{u_k}^T\bar{Q}_{i,u_k}+\sum_{k=1}^{N}\bar{Q}_{i,u_kx}^TI_{u_k}+\sum_{k=1}^{N}\sum_{p=1}^{N}K_{u_k}^T\bar{Q}_{i,u_ku_p}I_{u_p} \tag{8.38b}
$$

$$
\begin{aligned}
&-\frac{d\bar{V}_{i,xx}}{dt}=\sum_{k=1}^{N}K_{u_k}^T\bar{Q}_{i,u_kx}+\sum_{k=1}^{N}\bar{Q}_{i,u_kx}^TK_{u_k}+\bar{Q}_{i,xx}\\
&+\sum_{k=1}^{N}\sum_{p=1}^{N}K_{u_k}^TQ_{i,u_ku_p}K_{u_p}.
\end{aligned}
\tag{8.38c}
$$

At the terminal time, we have the terminal condition $V_i(\bar{x}(t_f),t_f)=\phi_i(\bar{x}(t_f),t_f)$. If

the Taylor series is used to expand around $\bar{x}(t_f)$, we obtain the terms as follow

$$\begin{aligned}
&\phi_i(\bar{x}(t_f), t_f) = \phi_i(\bar{x}(t_f) + \delta x(t_f), t_f) \\
&\approx \phi_i(\bar{x}(t_f), t_f) + \delta x(t_f)^T \phi_{i,x}(\bar{x}(t_f), t_f) \\
&+ \delta x(t_f)^T \phi_{i,xx}(\bar{x}(t_f), t_f) \delta x(t_f).
\end{aligned} \tag{8.39}$$

Henceforth, the terminal conditions at t_f for the backward differential equations are obtain as

$$\bar{V}_i(t_f) = \phi_i(\bar{x}(t_f), t_f), \tag{8.40a}$$

$$\bar{V}_{i,x}(t_f) = \phi_{i,x}(\bar{x}(t_f), t_f), \tag{8.40b}$$

$$\bar{V}_{i,xx}(t_f) = \phi_{i,xx}(\bar{x}(t_f), t_f). \tag{8.40c}$$

Which this completes the proof. □

The nonzero-sum differential dynamic programming is presented in Algorithm 9.

8.4 Numerical Examples

In this section, to validate the effectiveness of the proposed algorithm, the results of three practical examples are demonstrated.

8.4.1 Large-Scale Power System

In the first example, we consider a realistic nonlinear large-scale power system [272] with three interconnected subsystems with mismatched interconnections. In this example, we use the nonzero-sum DDP algorithm for a large-scale system with three subsystems to validate the efficiency of the proposed method to control multi-agent

Algorithm 9 Nonzero-Sum Differential Dynamic Programming

Require: Initial values for x_0, control $u_1, u_2, ..., u_N$, terminal time t_f, multiplier γ, large integer A, and small value ϵ.

Ensure: Optimal minimizing control u_i^*, gains I_{u_i}, K_{u_i}.

procedure UPDATE CONTROL POLICIES:

Compute optimal control u_i through minimizing the cost function J_i for each player i.

for $i = 1, ..., N$ **do** until the convergence is obtained

while $||\sum_i^N \delta u_i|| > \gamma$ **do**

Compute values of V_i, $V_{i,x}$, and $V_{i,xx}$ at t_f from (8.36);

Integrate (8.34) backward in the time;

Compute I_{u_i} and K_{u_i} through (8.14);

Integrate (8.8) by replacing δu_i with $\delta u_i = K_{u_i}\delta x_i + I_{u_i}$ to get δx;

Obtain $\delta u_i = K_{u_i}\delta x + \epsilon I_{u_i}$;

Update $u_i = u_i + \delta u_i$;

Set $u_i^* = u_i$;

end while

end for

return u_i^* for $i = 1, ..., N$;

end procedure

system for the first time. We assume a single player and consider a cost function for each subsystem. The dynamics model of each power station dynamics is given as follows

$$\begin{aligned}
\frac{d(\Delta v_i)}{dt} &= -\frac{1}{T_{g_i}}\Delta v_i + \frac{1}{R_{g_i}T_{g_i}}\Delta f_{G_i} + \frac{1}{T_{g_i}}u_i \\
\frac{d(\Delta P_{m_i})}{dt} &= \frac{K_{t_i}}{T_{t_i}}\Delta v_i - \frac{1}{T_{t_i}}\Delta P_{m_i} \\
\frac{d(\Delta f_{G_i})}{dt} &= \frac{K_{p_i}}{T_{p_i}}\Delta P_{m_i} - \frac{\Delta f_{G_i}}{T_{p_i}} - \frac{K_{p_i}}{T_{p_i}}\Delta P_{G_i}
\end{aligned} \tag{8.41}$$

where $\Delta v_i \in \mathbb{R}, \Delta P_{m_i} \in \mathbb{R}$, and $\Delta f_{G_i} \in \mathbb{R}$, $i = 1, 2, 3$ are incremental change in governor value position, generator output, and frequency deviation, respectively. $u_i \in \mathbb{R}$ is the control input and $\Delta P_{G_i} \in \mathbb{R}$ is the incremental change in the electrical

power and is defined as

$$\Delta P_{G_i} = \zeta_i \sin(\Delta P_{m_i} \Delta f_{G_i}) \tag{8.42}$$

where $\zeta_1 = \sum_{i=1}^{3} e_{1i}\Delta v_i$,$\zeta_2 = \sum_{i=1}^{3} e_{2i}\Delta P_{m_i}$, and $\zeta_3 = \sum_{i=1}^{3} e_{3i}\Delta f_{G_i}$. $e_{is}, i, s = 1, 2, 3$, are chosen as $e_{1i} \in [-0.5, 0.5]$, $e_{2i} \in [-0.5, 0.5]$, and $e_{3i} \in [-0.25, 0.25]$. The goal is to bring the states from $[x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}] = [0.5, -0.5, 0.1, 0.1 - 0.1, 0.1, 0.1 - 0.1, 0.1]$ to $[0, 0, 0, 0, 0, 0, 0, 0, 0]$ in $t_f = 1$. The cost function is defined as

$$J_i = \int_{t_0}^{t_f} \sum_{j=1}^{3} (x_j^T Q_{ij} x_j + u_j^T R_{u_{ij}} u_j) dt, \tag{8.43}$$

and the initial control policies are set as $u_i = 0$, and $\gamma_1 = \gamma_2 = \gamma_3 = 0.1$. The weights are chosen as $Q_{11} = Q_{21} = Q_{31} = [100, 0, 0; 0, 1, 0; 0, 0, 1]$, $Q_{12} = Q_{22} = Q_{32} = [1000, 0, 0; 0, 1, 0; 0, 0, 1]$, $Q_{13} = Q_{23} = Q_{33} = [100, 0, 0; 0, 1, 0; 0, 0, 1]$, $R_{u_{11}} = 10$, $R_{u_{12}} = R_{u_{13}} = 0$, $R_{u_{22}} = 10$, $R_{u_{21}} = R_{u_{23}} = 0$, $R_{u_{31}} = R_{u_{32}} = 0$, and $R_{u_{33}} = 10$.

Fig. 8.1 shows the evolution of the states in the large-scale power system, where three interconnected subsystems interact. The control objective is to stabilize the states by bringing them from their initial values to the desired equilibrium point. The plot demonstrates how the proposed nonzero-sum DDP algorithm effectively handles mismatched interconnections between subsystems and how each player for each subsystem successfully reach its target. As the simulation progresses, the states of all subsystems converge to zero, indicating the success of the control policies in ensuring stability across the entire system.

Fig. 8.2 tracks the cost function of large-scale power system for each player in 200 iterations and reveals how the optimization process evolves. The goal of each player is to minimize their individual cost function despite the effect of other players. This plot

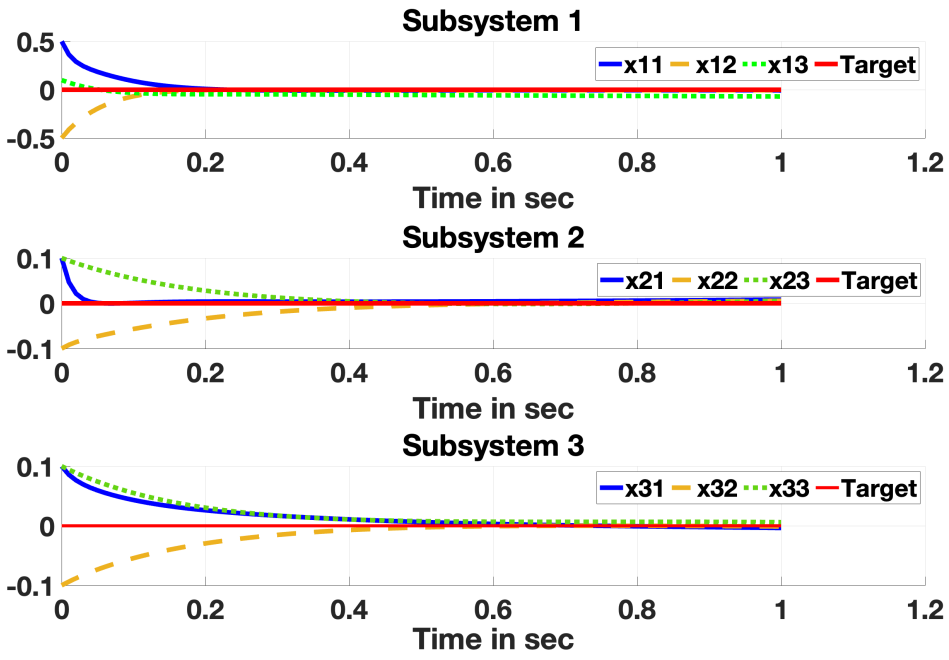


Figure 8.1: States of large-scale power system.

highlights the convergence behavior of the cost values and shows how they decrease iteratively as the players update their control strategies. The results demonstrate that the proposed nonzero-sum DDP algorithm effectively balances the competing objectives of different players and achieves cost minimization for each subsystem.

Fig. 8.3 depicts the control inputs applied by each player over the time. It reflects how the control strategies evolve to meet their objectives while accounting for the influence of other players. The smooth and coordinated control signals indicate that the nonzero-sum DDP method ensures proper cooperation and competition among players. The stabilization of control policies towards the end of the simulation reflects convergence to the optimal solution, where each input becomes steady as the states approach equilibrium.

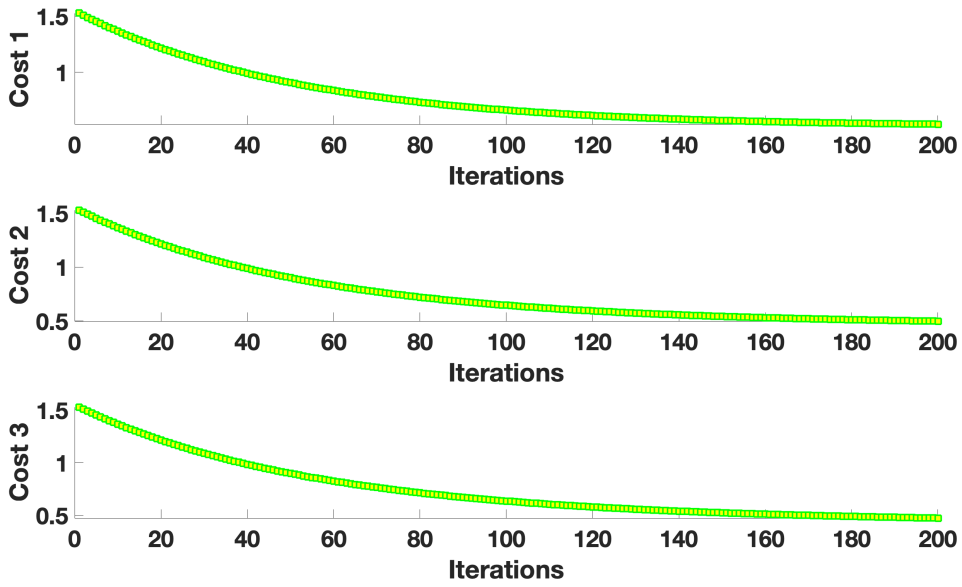


Figure 8.2: Cost per iteration of large-scale power system for each player.

8.4.2 Load-Frequency Control of Power System

In the second example, we consider the following load-frequency control (LFC) of a power system [317] with three players. In this example, we assume the min-max strategy for nonzero-sum DDP framework where the role of other players in each player are intelligent disturbance and while one player is minimizing the cost function, the other players are maximizing it. Consider the following LFC of the power system as

$$\frac{dx(t)}{dt} = Ax + B_1u_1(t) + B_2u_2(t) + B_3u_3(t) \tag{8.44}$$

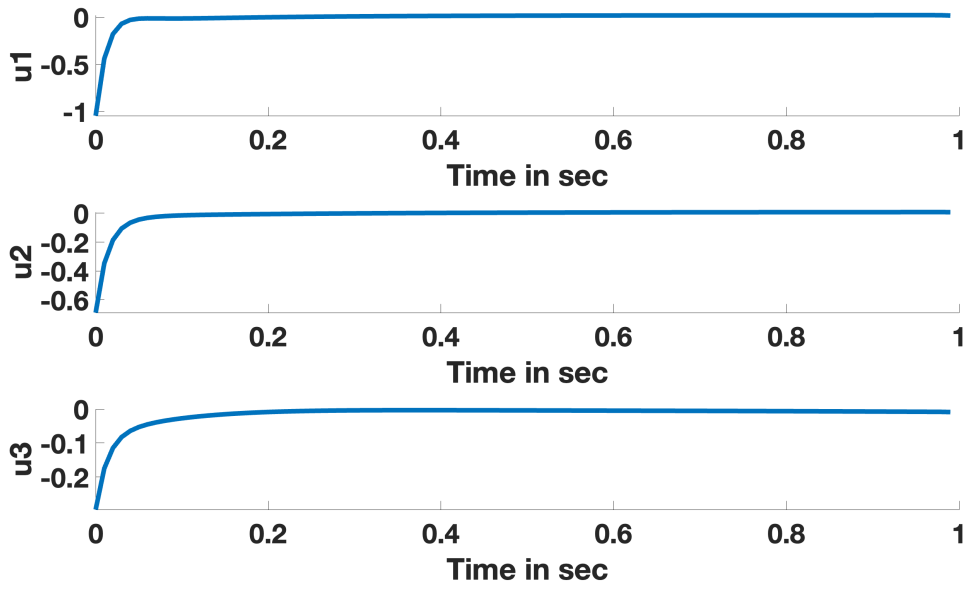


Figure 8.3: Control policies of large-scale power system for each player.

where

$$A = \begin{bmatrix} -0.0665 & 8 & 0 & 0 \\ 0 & -3.663 & 3.633 & 0 \\ -6.86 & 0 & -13.736 & -13.736 \\ 0.6 & 0 & 0 & 0 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0 \\ 0 \\ 13.736 \\ 0 \end{bmatrix}, \quad B_2 = \begin{bmatrix} -8 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

$$B_3 = \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \end{bmatrix}.$$

The goal is to bring the states from $[x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}]$ $= [-0.5, 1, 0.5, -1]$ to $[0, 0, 0, 0]$ in $t_f = 10$. Cost function for each player are defined as

$$
\begin{aligned}
J_1 &= \int_{t_0}^{t_f} (x^T Q_1 x + u_1^T R_{u_{11}} u_1 - u_2^T R_{u_{12}} u_2 - u_3^T R_{u_{13}} u_3) dt, \\
J_2 &= \int_{t_0}^{t_f} (x^T Q_2 x - u_1^T R_{u_{21}} u_1 + u_2^T R_{u_{22}} u_2 - u_3^T R_{u_{23}} u_3) dt, \\
J_1 &= \int_{t_0}^{t_f} (x^T Q_3 x - u_1^T R_{u_{31}} u_1 - u_2^T R_{u_{32}} u_2 + u_3^T R_{u_{33}} u_3) dt.
\end{aligned} \tag{8.45}
$$

and the initial control policies are set as $u_i = 0$, and $\gamma_1 = \gamma_2 = \gamma_3 = 0.01$. The weights are chosen as $Q_1 = [1, 0, 0, 0; 0, 1, 0, 0; 0, 0, 1, 0; 0, 0, 0, 1]$,
$Q_2 = [1, 0, 0, 0; 0, 0.1, 0, 0; 0, 0, 1, 0; 0, 0, 0, 1]$, $Q3 = [1, 0, 0, 0; 0, 1, 0, 0; 0, 0, 1, 0; 0, 0, 0, 10]$, $R_{u_{11}} = 1$, $R_{u_{12}} = 2, R_{u_{13}} = 15$, $R_{u_{21}} = 1$, $R_{u_{22}} = 1, R_{u_{23}} = 10$, $R_{u_{31}} = 10, R_{u_{32}} = 1$, and $R_{u_{33}} = 1$.

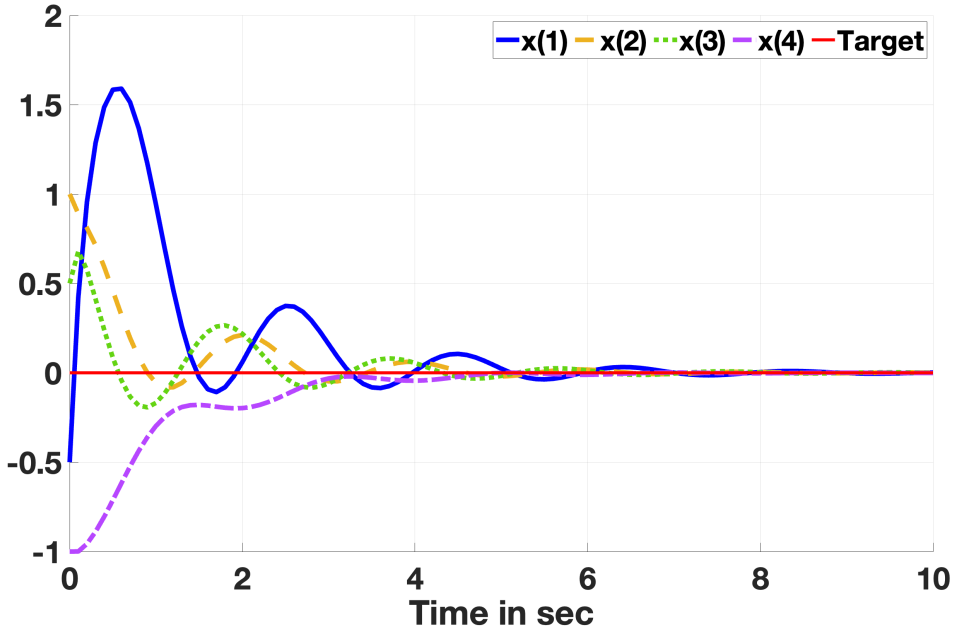


Figure 8.4: States of LFC system.

Fig. 8.4 presents the state trajectories of the LFC system, where three players

manage different components of the power system to maintain frequency stability. The initial states, representing deviations in power and frequency, are driven to zero over the course of the simulation. The results illustrate how the min-max nonzero-sum DDP framework successfully mitigates intelligent disturbances and ensures that the system achieves load-frequency balance. This example showcases the ability of algorithm to handle dynamic interactions and disturbances efficiently.

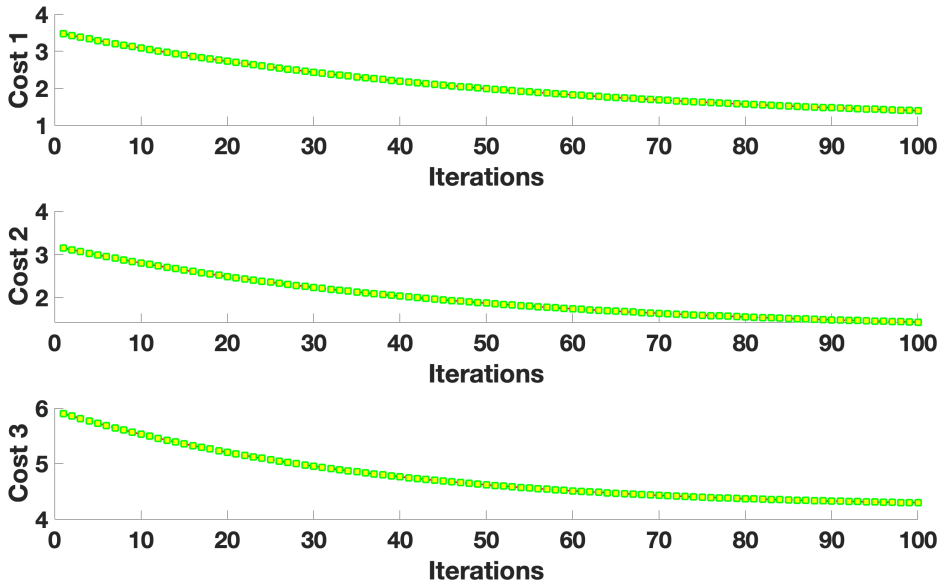


Figure 8.5: Cost per iteration of LFC for each player.

The cost function for each player in the LFC system is shown over the course of 100 iterations in Fig. 8.5. The objective of each player is to minimize its respective cost while considers the effect of maximization of other players, which is affected by both its own actions and those of other players. The plot highlights the min-max nature of the game, where while one player attempts to minimize its cost function, the other players attempts to maximize it. The convergence of cost values demonstrates that the algorithm successfully balances the competing objectives and leads to optimal control policies for maintaining system stability.

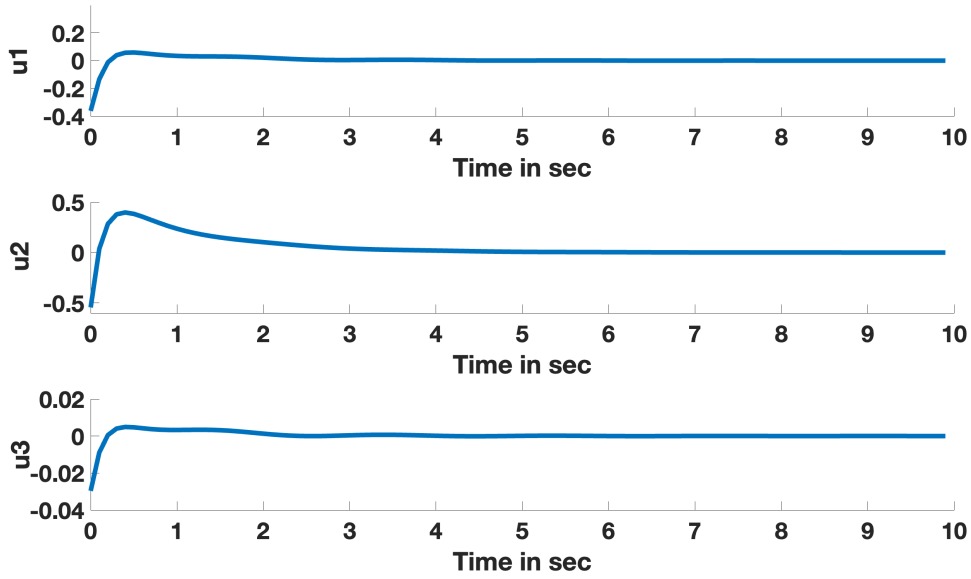


Figure 8.6: Control policies of LFC for each player.

Fig. 8.6 illustrates the control inputs applied by the players during the simulation. Each control policy evolves iteratively and adjusts to the actions of other players while striving to achieve load-frequency control. The plot shows how the control signals stabilize over time as the strategies of players converge to an optimal solution. This example highlights the robustness of the nonzero-sum DDP approach in achieving reliable control in the face of disturbances and conflicting objectives.

8.4.3 Missile Dynamics

In the third example, we apply the proposed method to the missile dynamics with three players [318]. Consider the following missile dynamics as

$$\frac{dx(t)}{dt} = F(x) + B_1 u_1(t) + B_2 u_2(t) + B_3 u_3(t) \tag{8.46}$$

where

$$F(x) = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0 \\ 0 \\ 100 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 \\ 0 \\ 100 \end{bmatrix}, \quad B_3 = \begin{bmatrix} 0 \\ 0 \\ 100 \end{bmatrix}, \tag{8.47}$$

and

$$\begin{aligned} f_1 &= (\frac{180 \cdot \text{grav} \cdot Q \cdot S}{\pi \cdot m \cdot \text{Val}}) \cos\left(\frac{\pi x_1}{180}\right) (1.03 \times 10^{-4} x_1^3 - 9.45 \times 10^{-3} x_1 |x_1| \qquad (8.48a) \\ &\quad - 0.17 x_1) + x_2 - 0.034 \left(\frac{180 \cdot \text{grav} \cdot Q \cdot S}{\pi \cdot m \cdot \text{Val}}\right) \cos\left(\frac{\pi x_1}{180}\right) x_3, \\ f_2 &= \left(\frac{180 \cdot Q \cdot S \cdot D}{\pi \cdot I_{yy}}\right) \left(2.15 \times 10^{-4} x_1^3 - 0.0195 x_1 |x_1| + 0.051 x_1\right) \\ &\quad - 0.206 \left(\frac{180 \cdot Q \cdot S \cdot D}{\pi \cdot I_{yy}}\right) x_3, \qquad (8.48b) \\ f_3 &= -100 x_3. \qquad (8.48c) \end{aligned}$$

Where $Q = 293.638 N/m^2$, $S = 0.04087 m^2$, $m = 4410 kg$, $g = 9.8 m/s^2$, $Val = 947.6 m/s$, and $I_{yy} = 247.44 kg.m^2$. The goal is to bring the states from $[x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}] = [-2, -1, 1]$ to $[0, 1, 0]$ in $t_f = 1$. Cost function for each player are defined as

$$\begin{aligned} J_1 &= \int_{t_0}^{t_f} (x^T Q_1 x + u_1^T R_{u_{11}} u_1 + u_2^T R_{u_{12}} u_2) dt, \\ J_2 &= \int_{t_0}^{t_f} (x^T Q_2 x - u_1^T R_{u_{21}} u_1 + u_2^T R_{u_{22}} u_2) dt, \\ J_3 &= \int_{t_0}^{t_f} (x^T Q_3 x + u_1^T R_{u_{31}} u_1 - u_2^T R_{u_{32}} u_2 + u_3^T R_{u_{33}} u_3) dt. \end{aligned} \tag{8.49}$$

and the initial control policies are set as $u_i = 0$, and $\epsilon_1 = 0.9, \epsilon_2 = 0.8, \epsilon_3 = 0.7$. The weights are chosen as $Q_1 = [1, 0, 0, ; 0, 1, 0; 0, 0, 1]$, $Q_2 = [1, 0, 0, ; 0, 1, 0; 0, 0, 1]$, $Q3 =$

$[1, 0, 0, ; 0, 1, 0; 0, 0, 10]$, $R_{u_{11}} = 10$, $R_{u_{12}} = 1$, $R_{u_{21}} = 10$, $R_{u_{22}} = 1$, $R_{u_{31}} = 1, R_{u_{32}} = 1$, and $R_{u_{33}} = 10$.

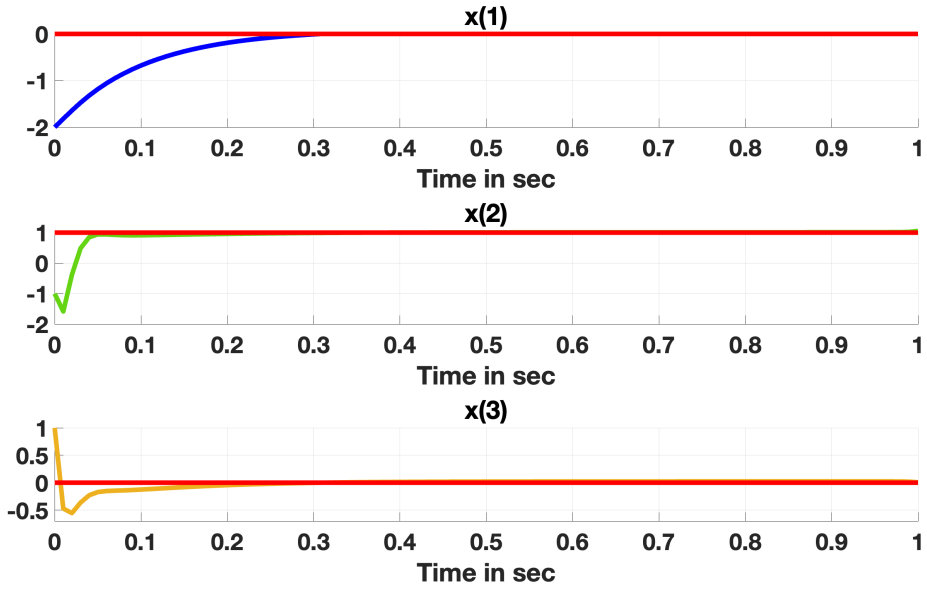


Figure 8.7: States of missile dynamics.

Fig. 8.7 shows the state trajectories of the missile system, where three players interact under the proposed nonzero-sum DDP framework. The goal is to drive the initial state from $[-2, -1, 1]$ to $[0, 1, 0]$ within a short time horizon. The plots demonstrate the system's successful convergence, highlighting the effectiveness of the control strategy in managing nonlinear and coupled missile dynamics. Fig. 8.8 illustrates the control inputs generated by each player, where the distinct influence of each control signal becomes evident as the simulation progresses. These control policies are shaped by each player's cost function and reflect their strategic behavior in achieving individual objectives while responding to the actions of others. Fig. 8.9 presents the cost value for each player over 20 iterations. The consistent reduction in cost values validates the convergence and stability of the optimization process. Overall, this example confirms the robustness and applicability of the proposed nonzero-sum DDP algorithm

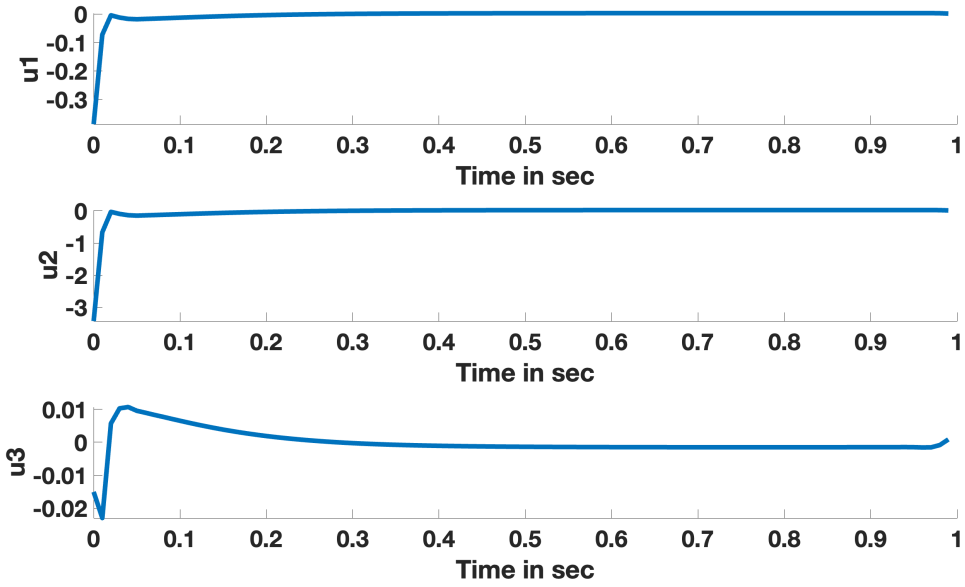


Figure 8.8: Control policies of missile dynamics.

in solving complex, high-speed multi-agent systems such as missile dynamics.

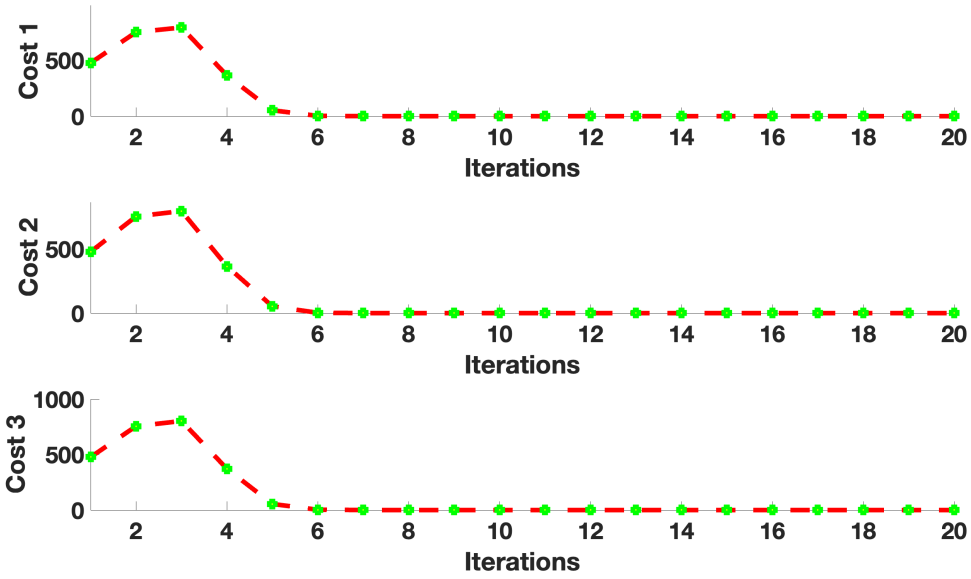


Figure 8.9: Cost per iteration of missile dynamics.

Chapter 9

Conclusion and Future Directions

9.1 Summary of Research Contributions

This dissertation has developed a unified, learning-based framework for **Min-Max Differential Dynamic Programming (DDP)** to address the limitations of existing control algorithms under uncertainty, stochasticity, partial observability, and large-scale interactions. The overall goal was to enhance the adaptability, robustness, and practicality of Min-Max DDP, transforming it into a comprehensive tool for modern control applications. Each chapter of this dissertation contributes a distinct methodological or theoretical advancement, summarized as follows:

Chapter 2 introduced the **Min-Max Adaptive Dynamic Programming (ADP)** algorithm for zero-sum differential games. The framework formulated the Min-Max optimization problem through the **Hamilton–Jacobi–Isaacs (HJI)** equation and solved it iteratively using adaptive critic learning. This contribution demonstrated how adaptive approximation can achieve robust control for nonlinear systems subject to adversarial disturbances without requiring full system linearization.

Chapter 3 presented the **Sliding-Mode-Observer-Based Min-Max DDP** for output-feedback control. A **Higher-Order Sliding Mode (HOSM)** observer was designed to estimate unmeasured or noisy states, enabling the controller to operate under partial observability. This integration provided robustness against modeling

errors and measurement noise while maintaining convergence in the DDP framework.

Chapter 4 developed the **Data-Driven Min-Max DDP** approach for nonlinear systems with unknown dynamics. By employing **Gaussian Process Regression (GPR)** to approximate the drift dynamics directly from measured data, this chapter removed the dependency on explicit mathematical models. The resulting framework achieved high accuracy in policy updates while retaining the Min-Max structure for robust optimization.

Chapter 5 extended the framework to stochastic systems through the **Data-Driven Stochastic Game-Theoretic Min-Max DDP**. In this approach, both the drift and diffusion dynamics were learned from data — using GPR for drift estimation and a nonparametric binning method for diffusion modeling. This enabled the Min-Max DDP algorithm to handle stochastic environments with noise propagation and uncertainty in both process and measurement levels.

Chapter 6 introduced the **Distributed Min-Max DDP** for large-scale interconnected systems with mismatched interconnections. The proposed distributed formulation decomposed the global optimization problem into subsystem-level subproblems that cooperatively solved for local control updates. This advancement achieved scalability and computational efficiency, allowing robust control of networked and multi-agent systems.

Chapter 7 incorporated safety considerations into the control design through the **Chance-Constrained Game-Theoretic Min-Max DDP**. By integrating probabilistic constraints into the optimization process, the method ensured that state and control trajectories satisfied safety limits with a predefined confidence level. This contribution advanced safe trajectory optimization under uncertainty while maintaining computational tractability.

Chapter 8 generalized the framework to **Continuous-Time Nonzero-Sum Dif-**

ferential Games, expanding Min-Max DDP beyond strictly adversarial settings. This formulation allowed multiple agents with distinct objectives to interact cooperatively and competitively, establishing a foundation for multi-player and adaptive extensions in game-theoretic control.

Collectively, these contributions establish a cohesive research trajectory that extends Min-Max DDP from a deterministic, model-dependent method into a **data-driven, stochastic, distributed, and safety-aware control framework**. The methodologies proposed in this dissertation enhance both the theoretical robustness and the practical applicability of differential dynamic programming across a broad range of modern control systems.

9.2 Comparative Advantages and Key Insights

The proposed methodologies throughout this dissertation collectively enhance the capabilities of **Min-Max Differential Dynamic Programming (DDP)** in terms of robustness, adaptability, and scalability. The key advantage of the developed framework lies in its ability to integrate data-driven learning, stochastic modeling, and safety-aware optimization into a unified control strategy. This section highlights the comparative strengths and major insights derived from the research contributions.

1. Robustness Against Uncertainty and Adversarial Disturbances: Traditional DDP formulations are limited to deterministic settings and often degrade in performance when faced with external disturbances or model uncertainties. The proposed **Min-Max DDP** extensions explicitly incorporate adversarial inputs and stochastic effects through a game-theoretic framework, ensuring robust control performance even under worst-case conditions. The adaptive and stochastic variants developed in this work guarantee improved disturbance rejection and resilience against modeling errors.

2. Data-Driven Adaptability Without Explicit Models: One of the most significant advancements achieved in this dissertation is the development of a **data-driven Min-Max DDP** formulation that eliminates the dependency on precise mathematical models. By employing **Gaussian Process Regression (GPR)** and non-parametric diffusion estimation, the algorithm learns the underlying system dynamics directly from observed data. This adaptability allows the controller to generalize across different nonlinear systems and operating conditions, making it suitable for real-world applications where exact models are unavailable.

3. Scalability Through Distributed and Parallel Architectures: The proposed **Distributed Min-Max DDP** addresses the computational challenges associated with large-scale and interconnected systems. By decomposing the optimization problem into local subproblems that coordinate through limited communication, the framework achieves near-optimal performance while maintaining computational tractability. This scalability enables deployment in multi-agent and networked environments such as large-scale power systems, autonomous fleets, and robotic swarms.

4. Safety and Risk-Aware Optimization: Incorporating safety into control design remains a fundamental challenge in uncertain and dynamic environments. The **Chance-Constrained Min-Max DDP** developed in this work ensures probabilistic safety guarantees by maintaining the risk of constraint violation below a specified threshold. This integration of safety within the optimization process enables reliable and risk-sensitive trajectory planning under uncertainty, making the framework applicable to safety-critical systems such as UAVs and autonomous ground vehicles.

5. Generalization to Nonzero-Sum and Multi-Agent Scenarios: Unlike most existing Min-Max formulations limited to two-player zero-sum settings, the proposed **Nonzero-Sum and Multiplayer DDP** extensions enable cooperative–competitive interactions among multiple agents. This generalization allows each agent to optimize

its own objective while accounting for the influence of others, providing a theoretical basis for complex decision-making processes in distributed and adaptive environments.

6. Unified Learning-Based Control Framework: A key insight emerging from this research is that the Min-Max DDP can serve as a unifying structure for robust and learning-based control. By combining adaptive dynamic programming, data-driven modeling, distributed optimization, and probabilistic safety, the framework bridges the gap between classical optimal control and modern reinforcement learning approaches. The resulting architecture offers both theoretical rigor and practical implementability for complex nonlinear systems.

Overall, the proposed methodologies establish a new paradigm for **safe, data-driven, and game-theoretic optimal control**. The insights gained from this research demonstrate that integrating robustness, adaptability, and safety within the Min-Max DDP framework can significantly expand its scope and effectiveness for real-world uncertain systems.

9.3 Limitations and Challenges

While the proposed **Min-Max Differential Dynamic Programming (DDP)** frameworks in this dissertation significantly advance the state of the art in robust, data-driven, and safe control, several limitations and open challenges remain. These limitations highlight potential areas for improvement and future exploration.

1. Computational Complexity in High-Dimensional Systems: Despite algorithmic improvements, the computational cost of Min-Max DDP and its extensions remains a major concern for high-dimensional and large-scale systems. The backward–forward iterative nature of the algorithm, coupled with the evaluation of higher-order derivatives and value function updates, leads to significant computational

overhead. Although distributed and parallelized variants alleviate this issue, further reduction of computational complexity is essential for real-time implementation in high-dimensional systems such as multi-robot coordination or power network control.

2. Convergence and Stability Guarantees in Learning-Based Approaches: While data-driven and adaptive formulations improve flexibility, they also introduce challenges related to convergence and stability. The use of function approximators such as **Gaussian Process Regression (GPR)** or neural networks introduces approximation errors that can affect the convergence of the control policy. Rigorous theoretical guarantees ensuring stability and boundedness across iterative updates are still limited and require further investigation, particularly in stochastic and time-varying environments.

3. Dependence on Hyperparameter Selection and Training Data Quality: The performance of data-driven Min-Max DDP approaches depends heavily on the quality, coverage, and representativeness of the training data. Sparse or biased datasets can lead to poor generalization and suboptimal control actions. Additionally, hyperparameters such as kernel bandwidth in GPR or observer gains in the sliding-mode observer significantly influence algorithm stability and accuracy. Systematic methods for online tuning and adaptive hyperparameter adjustment are still underdeveloped.

4. Limited Scalability for Fully Decentralized Systems: Although the distributed Min-Max DDP formulation achieves scalability for interconnected subsystems, it assumes a partially coordinated structure with shared information among agents. In fully decentralized networks with limited communication or dynamic topologies, achieving convergence to global or equilibrium solutions remains a challenge. Designing decentralized algorithms with reduced communication requirements and guaranteed stability is an important open research direction.

5. Conservatism in Safety and Chance Constraints: The integration of safety

through chance constraints introduces probabilistic guarantees that may be conservative in practice. Approximations of risk bounds and constraint satisfaction can lead to suboptimal solutions, particularly under non-Gaussian uncertainties or nonlinear constraints. Developing tighter, less conservative formulations that maintain computational efficiency is crucial for enhancing safety–performance trade-offs in practical applications.

6. Limited Hardware Validation and Real-Time Implementation: Most results presented in this dissertation are validated through simulation studies. Although the proposed frameworks are designed for real-world applicability, hardware implementation introduces additional challenges such as sensor noise, actuator limits, and time delays. Translating these algorithms into embedded control systems or onboard processors will require further simplifications and computational optimizations.

In summary, while the research presented in this dissertation establishes a strong theoretical and computational foundation for robust and learning-based Min-Max DDP, addressing these limitations is essential to ensure its scalability, stability, and real-time performance in practical autonomous and safety-critical systems.

9.4 Future Research Directions

Building upon the findings and contributions of this dissertation, several promising research directions can further advance the theoretical depth and practical applicability of **Min-Max Differential Dynamic Programming (DDP)**. These directions focus on enhancing computational efficiency, theoretical guarantees, safety, and adaptability of the framework in increasingly complex environments.

1. Real-Time and Event-Triggered Min-Max DDP: Future research should focus on developing **event-triggered** or **real-time adaptive** versions of the Min-

Max DDP algorithm. By activating updates only when significant state deviations or disturbances occur, the computational burden can be reduced while preserving control performance. Such frameworks would be particularly valuable for embedded and hardware-limited systems where computational efficiency is critical.

2. Integration with Deep Reinforcement Learning: A promising direction is the integration of Min-Max DDP with **deep reinforcement learning (DRL)** to approximate value functions and control policies using neural architectures. Deep actor–critic networks or policy gradient methods could provide scalable solutions for high-dimensional nonlinear systems. This combination would bridge the gap between classical model-based control and model-free learning, leading to hybrid algorithms capable of handling real-world complexity.

3. Safe Reinforcement Learning and Risk-Aware Policy Adaptation: The incorporation of **safe learning mechanisms** into the Min-Max DDP structure remains an open challenge. Future studies can focus on combining **control barrier functions (CBFs)** or **Lyapunov-based safety constraints** with DDP to ensure safety during online learning and exploration. This integration will enable adaptive and risk-aware control strategies for uncertain and dynamic environments.

4. Hybrid and Discontinuous Dynamical Systems: Many real-world systems, such as legged robots or switching networks, exhibit hybrid or discontinuous dynamics. Extending Min-Max DDP to **hybrid differential games** with state jumps, mode transitions, or impact dynamics is a valuable next step. This extension would require redefining the backward pass and costate propagation to handle nonsmooth transitions while maintaining convergence.

5. Robustness Under Communication Constraints and Network Uncertainties: Future work can extend the **Distributed Min-Max DDP** to scenarios involving time delays, packet loss, or switching communication topologies. Designing

algorithms that maintain stability and performance under such uncertainties will enable application in large-scale networked systems, such as cooperative UAVs, intelligent transportation networks, and smart grids.

6. Hardware Implementation and Experimental Validation: To bridge the gap between theory and practice, future research should focus on implementing the proposed algorithms on physical platforms such as quadrotors, robotic manipulators, or ground vehicles. Real-time experiments will validate the framework's robustness, computational efficiency, and safety in the presence of unmodeled dynamics, sensor noise, and actuator limitations.

7. Formal Safety Guarantees via Reachability and Verification: Finally, integrating formal verification and reachability analysis into the Min-Max DDP framework could provide **provable safety guarantees** for safety-critical systems. Combining data-driven control with reachability-based safe sets or Hamilton–Jacobi reachability formulations would ensure that trajectories remain within verified safe regions under uncertainty.

In summary, the future extensions of this work lie at the intersection of **robust optimal control**, **machine learning**, and **formal safety assurance**. By combining real-time adaptability, distributed computation, and verifiable safety, these research directions can further enhance the applicability of Min-Max DDP to complex, uncertain, and intelligent autonomous systems. The continued development of these ideas will contribute to establishing a new generation of **learning-based, safe, and scalable optimal control frameworks** for modern engineering applications.

Final Remarks

In conclusion, this dissertation has established a comprehensive and unified framework for **Min-Max Differential Dynamic Programming (DDP)** that bridges

the gap between classical optimal control theory and modern learning-based methods. Through the systematic integration of data-driven modeling, stochastic optimization, distributed coordination, and safety-aware design, the proposed research advances the theoretical understanding and practical implementation of robust and adaptive control in uncertain environments. The results presented across all chapters collectively demonstrate that Min-Max DDP can evolve from a deterministic algorithm into a versatile foundation for **intelligent, resilient, and safe autonomous systems**. The methodologies developed in this work not only contribute to the advancement of nonlinear and stochastic control but also lay the groundwork for future exploration at the intersection of **optimal control, reinforcement learning, and real-world autonomy**.

Reference List

[1] Dimitri Bertsekas. *Dynamic programming and optimal control: Volume I*, volume 4. Athena scientific, 2012.

[2] J Rawlings and D Mayne. Postface to model predictive control: Theory and design. *Nob Hill Pub*, 5:155–158, 2012.

[3] Richard Bellman and Robert Kalaba. Dynamic programming and statistical communication theory. *Proceedings of the National Academy of Sciences*, 43(8):749–751, 1957.

[4] David H Jacobson and David Q Mayne. Differential dynamic programming. *(No Title)*, 1970.

[5] Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1168–1175. IEEE, 2014.

[6] Weiwei Li and Emanuel Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *First International Conference on Informatics in Control, Automation and Robotics*, volume 2, pages 222–229. SciTePress, 2004.

[7] Kemin Zhou and John Comstock Doyle. *Essentials of robust control*, volume 104. Prentice hall Upper Saddle River, NJ, 1998.

[8] Wei Sun, Yunpeng Pan, Jaein Lim, Evangelos A Theodorou, and Panagiotis Tsiotras. Min-max differential dynamic programming: Continuous and discrete time formulations. *Journal of Guidance, Control, and Dynamics*, 41(12):2568–2580, 2018.

[9] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[10] Frank L Lewis and Draguna Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE circuits and systems magazine*, 9(3):32–50, 2009.

[11] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction

method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[12] Kaito Ariu, Cheng Fang, Márcio da Silva Arantes, Claudio Toledo, and Brian Charles Williams. Chance-constrained path planning with continuous time safety guarantees. In *AAAI Workshops*, 2017.

[13] Wei Sun, Evangelos A Theodorou, and Panagiotis Tsiotras. Stochastic game theoretic trajectory optimization in continuous time. In *2016 IEEE 55th conference on decision and control (CDC)*, pages 6167–6172. IEEE, 2016.

[14] Wei Sun and Theodore B Trafalis. H optimal control via game-theoretic differential dynamic programming and gaussian processes. *Journal of Optimization Theory and Applications*, 204(3):40, 2025.

[15] Kun Lin and Steven I Marcus. Dynamic programming with non-convex risk-sensitive measures. In *2013 American Control Conference*, pages 6778–6783. IEEE, 2013.

[16] Yashwanth Kumar Nakka, Anqi Liu, Guanya Shi, Anima Anandkumar, Yisong Yue, and Soon-Jo Chung. Chance-constrained trajectory optimization for safe exploration and learning of nonlinear systems. *IEEE Robotics and Automation Letters*, 6(2):389–396, 2020.

[17] Hassan Almubarak, Evangelos A Theodorou, and Nader Sadegh. Barrier states embedded iterative dynamic game for robust and safe trajectory optimization. In *2022 American Control Conference (ACC)*, pages 5166–5172. IEEE, 2022.

[18] Tamer Başar. Robust designs through risk sensitivity: An overview. *Journal of Systems Science and Complexity*, 34(5):1634–1665, 2021.

[19] David H Jacobson. New second-order and first-order algorithms for determining optimal control: A differential dynamic programming approach. *Journal of Optimization Theory and Applications*, 2:411–440, 1968.

[20] Katsuhisa Ohno. A new approach to differential dynamic programming for discrete time systems. *IEEE Transactions on Automatic Control*, 23(1):37–47, 1978.

[21] Daniel M Murray and Sidney J Yakowitz. Constrained differential dynamic programming and its application to multireservoir control. *Water Resources Research*, 15(5):1017–1027, 1979.

[22] David Q Mayne. Differential dynamic programming–a unified approach to the

optimization of dynamic systems. In *Control and dynamic systems*, volume 10, pages 179–254. Elsevier, 1973.

[23] TC Lin and JS Arora. Differential dynamic programming technique for optimal control. *Optimal Control Applications and Methods*, 15(2):77–100, 1994.

[24] Chao-Chung Yang, Liang-Cheng Chang, Chao-Hsien Yeh, and Chang-Shian Chen. Multiobjective planning of surface water resources by multiobjective genetic algorithm with constrained differential dynamic programming. *Journal of Water Resources Planning and Management*, 133(6):499–508, 2007.

[25] Daniel M Murray and Sidney J Yakowitz. Differential dynamic programming and newton's method for discrete optimal control problems. *Journal of Optimization Theory and Applications*, 43(3):395–414, 1984.

[26] JFA DE O. PANTOJA. Differential dynamic programming and newton's method. *International Journal of Control*, 47(5):1539–1553, 1988.

[27] Etienne Pellegrini and Ryan Russell. Quasi-newton differential dynamic programming for robust low-thrust optimization. In *AIAA/AAS Astrodynamics Specialist Conference*, page 4425, 2012.

[28] LaDon Jones, Robert Willis, and William W-G Yeh. Optimal control of nonlinear groundwater hydraulics using differential dynamic programming. *Water Resources Research*, 23(11):2097–2106, 1987.

[29] Shi-Chung Chang, Chun-Hung Chen, I-Kong Fong, and Peter B Luh. Hydroelectric generation scheduling with an effective differential dynamic programming algorithm. *IEEE transactions on power systems*, 5(3):737–743, 1990.

[30] J Brandão. Stochastic differential dynamic programming for multireservoir system control. *Water Resour Management*, 24:3101–3114, 2010.

[31] Janejira Tospornsampan, Ichiro Kita, Masayuki Ishii, and Yoshinobu Kitamura. Optimization of a multiple reservoir system operation using a combination of genetic algorithm and discrete differential dynamic programming: a case study in mae klong system, thailand. *Paddy and Water Environment*, 3:29–38, 2005.

[32] Yuval Tassa, Tom Erez, and William Smart. Receding horizon differential dynamic programming. *Advances in neural information processing systems*, 20, 2007.

[33] Yuichiro Aoyama, Augustinos D Saravanos, and Evangelos A Theodorou. Reced-

ing horizon differential dynamic programming under parametric uncertainty. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 3761–3767. IEEE, 2021.

[34] OMRY BEN-AVRAHAM and Daniel Tabak. Solution of an emergency air traffic control problem by differential dynamic programming. *International Journal of Systems Science*, 9(10):1167–1178, 1978.

[35] Weiwei Huang, Xiaojun Wu, Qun Zhang, Ning Wu, and Zhiwei Song. Trajectory optimization of autonomous driving by differential dynamic programming. In *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, pages 1758–1763. IEEE, 2014.

[36] Noélie Ramuzat, Florent Forget, Vincent Bonnet, Maxime Gautier, Sébastien Boria, and Olivier Stasse. Actuator model, identification and differential dynamic programming for a talos humanoid robot. In *2020 European Control Conference (ECC)*, pages 724–730. IEEE, 2020.

[37] Rohan Budhiraja, Justin Carpentier, Carlos Mastalli, and Nicolas Mansard. Differential dynamic programming for multi-phase rigid contact dynamics. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pages 1–9. IEEE, 2018.

[38] John N Nganga and Patrick M Wensing. Accelerating second-order differential dynamic programming for rigid-body systems. *IEEE Robotics and Automation Letters*, 6(4):7659–7666, 2021.

[39] He Li and Patrick M Wensing. Hybrid systems differential dynamic programming for whole-body motion planning of legged robots. *IEEE Robotics and Automation Letters*, 5(4):5448–5455, 2020.

[40] Guoxu Zhang, Changxuan Wen, Hongwei Han, and Dong Qiao. Aerocapture trajectory planning using hierarchical differential dynamic programming. *Journal of Spacecraft and Rockets*, 59(5):1647–1659, 2022.

[41] Cheng Zhou, Yanbo Long, Lei Shi, Longfei Zhao, and Yu Zheng. Differential dynamic programming based hybrid manipulation strategy for dynamic grasping. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8040–8046. IEEE, 2023.

[42] Aayushya Agarwal and Larry Pileggi. Large scale multi-period optimal power flow with energy storage systems using differential dynamic programming. *IEEE Transactions on Power Systems*, 37(3):1750–1759, 2021.

[43] Shatil Rahman and Steven L Waslander. Uncertainty-constrained differential dynamic programming in belief space for vision based robots. *IEEE Robotics and Automation Letters*, 6(2):3112–3119, 2021.

[44] Naoya Ozaki, Stefano Campagnola, Ryu Funase, and Chit Hong Yam. Stochastic differential dynamic programming with unscented transform for low-thrust trajectory design. *Journal of Guidance, Control, and Dynamics*, 41(2):377–387, 2018.

[45] Evangelos Theodorou, Yuval Tassa, and Emo Todorov. Stochastic differential dynamic programming. In *Proceedings of the 2010 American Control Conference*, pages 1125–1132. IEEE, 2010.

[46] Yunpeng Pan and Evangelos Theodorou. Probabilistic differential dynamic programming. *Advances in Neural Information Processing Systems*, 27, 2014.

[47] Francesco Romano, Andrea Del Prete, Nicolas Mansard, and Francesco Nori. Prioritized optimal control: A hierarchical differential dynamic programming approach. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3590–3595. IEEE, 2015.

[48] Augustinos D Saravanos, Yuichiro Aoyama, Hongchang Zhu, and Evangelos A Theodorou. Distributed differential dynamic programming architectures for large-scale multiagent control. *IEEE Transactions on Robotics*, 2023.

[49] Akihiko Yamaguchi and Christopher G Atkeson. Neural networks and differential dynamic programming for reinforcement learning problems. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5434–5441. IEEE, 2016.

[50] Zhaoming Xie, C Karen Liu, and Kris Hauser. Differential dynamic programming with nonlinear constraints. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 695–702. IEEE, 2017.

[51] Bolei Di and Andrew Lamperski. Newton's method and differential dynamic programming for unconstrained nonlinear dynamic games. In *2019 IEEE 58th conference on decision and control (CDC)*, pages 4073–4078. IEEE, 2019.

[52] Wei Sun, Evangelos A Theodorou, and Panagiotis Tsiotras. Continuous-time differential dynamic programming with terminal constraints. In *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 1–6. IEEE, 2014.

[53] Yunpeng Pan and Evangelos A Theodorou. Data-driven differential dynamic programming using gaussian processes. In *2015 American Control Conference (ACC)*, pages 4467–4472. IEEE, 2015.

[54] Dennis Gramlich, Carsten W Scherer, and Christian Ebenbauer. Robust differential dynamic programming. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 1714–1721. IEEE, 2022.

[55] Astghik Hakobyan and Insoon Yang. Distributionally robust differential dynamic programming with wasserstein distance. *IEEE Control Systems Letters*, 2023.

[56] Wei Sun, Evangelos A Theodorou, and Panagiotis Tsiotras. Game theoretic continuous time differential dynamic programming. In *2015 American control conference (ACC)*, pages 5593–5598. IEEE, 2015.

[57] Jun Morimoto and Christopher Atkeson. Minimax differential dynamic programming: An application to robust biped walking. *Advances in neural information processing systems*, 15, 2002.

[58] Alexander Poznyak, Sebastian Noriega-Marquez, Alejandra Hernandez-Sanchez, Mariana Ballesteros-Escamilla, and Isaac Chairez. Min–max dynamic programming control for systems with uncertain mathematical models via differential neural network bellman's function approximation. *Mathematics*, 11(5):1211, 2023.

[59] Xiaobo Zheng, Haodong Guan, Defu Lin, Shaoming He, and Hyo-Sang Shin. Game-theoretic flexible-final-time differential dynamic programming using gaussian quadrature. *Journal of Guidance, Control, and Dynamics*, 46(4):742–751, 2023.

[60] Wei Sun, Panagiotis Tsiotras, and Anthony J Yezzi. Multiplayer pursuit-evasion games in three-dimensional flow fields. *Dynamic Games and Applications*, 9(4):1188–1207, 2019.

[61] Wei Sun, Panagiotis Tsiotras, Tapovan Lolla, Deepak N Subramani, and Pierre FJ Lermusiaux. Multiple-pursuer/one-evader pursuit–evasion game in dynamic flowfields. *Journal of guidance, control, and dynamics*, 40(7):1627–1637, 2017.

[62] Wei Sun and Theodore B Trafalis. Learning-based nonlinear h control via game-theoretic differential dynamic programming. In *2023 American Control Conference (ACC)*, pages 1283–1288. IEEE, 2023.

[63] Rushikesh Kamalapurkar, Patrick Walters, Joel Rosenfeld, and Warren Dixon.

Reinforcement learning for optimal feedback control. Springer, 2018.

[64] Dimitri Bertsekas. *Reinforcement learning and optimal control*, volume 1. Athena Scientific, 2019.

[65] Anuradha M Annaswamy. Adaptive control and intersections with reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 6(1):65–93, 2023.

[66] Guoxing Wen, CL Philip Chen, Shuzhi Sam Ge, Hongli Yang, and Xiaoguang Liu. Optimized adaptive nonlinear tracking control using actor–critic reinforcement learning strategy. *IEEE transactions on industrial informatics*, 15(9):4969–4977, 2019.

[67] Tao Bian and Zhong-Ping Jiang. Reinforcement learning and adaptive optimal control for continuous-time nonlinear systems: A value iteration approach. *IEEE transactions on neural networks and learning systems*, 33(7):2781–2790, 2021.

[68] Yu Jiang and Zhong-Ping Jiang. Global adaptive dynamic programming for continuous-time nonlinear systems. *IEEE Transactions on Automatic Control*, 60(11):2917–2929, 2015.

[69] Phu-Cuong Pham and Yong-Lin Kuo. Online reinforcement learning based real-time robust adaptive control design for robot manipulators. *International Journal of Control, Automation and Systems*, 23(10):3048–3059, 2025.

[70] Yongfeng Yin, Zhetao Wang, Lili Zheng, Qingran Su, and Yang Guo. Autonomous uav navigation with adaptive control based on deep reinforcement learning. *Electronics*, 13(13):2432, 2024.

[71] Liqun Zhao, Konstantinos Gatsis, and Antonis Papachristodoulou. Stable and safe reinforcement learning via a barrier-lyapunov actor-critic approach. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 1320–1325. IEEE, 2023.

[72] Puze Liu, Haitham Bou-Ammar, Jan Peters, and Davide Tateo. Safe reinforcement learning on the constraint manifold: Theory and applications. *IEEE Transactions on Robotics*, 2025.

[73] Vincent Andrieu, Lucas Brivadis, Jean-Paul Gauthier, Ludovic Sacchelli, and Ulysse Serres. Exponential stabilizability and observability at the target imply semiglobal exponential stabilizability by templated output feedback. *Systems & Control Letters*, 195:105971, 2025.

[74] Anh-Tu Nguyen, Víctor Campos, Thierry-Marie Guerra, Juntao Pan, and Wenbo Xie. Takagi–sugeno fuzzy observer design for nonlinear descriptor systems with unmeasured premise variables and unknown inputs. *International Journal of Robust and Nonlinear Control*, 31(17):8353–8372, 2021.

[75] Jixing Lv, Xiaozhe Ju, and Changhong Wang. Neural network prescribed-time observer-based output-feedback control for uncertain pure-feedback nonlinear systems. *Expert Systems with Applications*, 264:125813, 2025.

[76] A Raza, FM Malik, N Mazhar, and R Khan. Output feedback control of a class of two-time-scale nonlinear systems using high-gain observers. *International Journal of Control*, 97(11):2612–2624, 2024.

[77] Dario Piga, Simone Formentin, and Alberto Bemporad. Direct data-driven control of constrained systems. *IEEE Transactions on Control Systems Technology*, 26(4):1422–1429, 2017.

[78] Matthias Seeger. Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106, 2004.

[79] Yuhan Liu, Pengyu Wang, and Roland Tóth. Learning for predictive control: A dual gaussian process approach. *arXiv preprint arXiv:2211.03699*, 2022.

[80] Achin Jain, Truong Nghiem, Manfred Morari, and Rahul Mangharam. Learning and control using gaussian processes. In *2018 ACM/IEEE 9th international conference on cyber-physical systems (ICCPS)*, pages 140–149. IEEE, 2018.

[81] Manish Prajapat, Johannes Köhler, Amon Lahr, Andreas Krause, and Melanie N Zeilinger. Finite-sample-based reachability for safe control with gaussian process dynamics. *arXiv preprint arXiv:2505.07594*, 2025.

[82] Xiaobing Dai, Zewen Yang, Sihua Zhang, Di-Hua Zhai, Yuanqing Xia, and Sandra Hirche. Cooperative online learning for multiagent system control via gaussian processes with event-triggered mechanism. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[83] Jehyun Park and Jongeun Choi. Gaussian process online learning with a sparse data stream. *IEEE Robotics and Automation Letters*, 5(4):5977–5984, 2020.

[84] Truong X Nghiem, Trong-Doan Nguyen, and Viet-Anh Le. Fast gaussian process based model predictive control with uncertainty propagation. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1052–1059. IEEE, 2019.

[85] Wendell H Fleming and Raymond W Rishel. *Deterministic and stochastic optimal control*, volume 1. Springer Science & Business Media, 2012.

[86] Stefano Massaroli, Michael Poli, Stefano Peluchetti, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. Learning stochastic optimal policies via gradient descent. *IEEE Control Systems Letters*, 6:1094–1099, 2021.

[87] Argenis Arriojas, Jacob Adamczyk, Stas Tiomkin, and Rahul V Kulkarni. Bayesian inference approach for entropy regularized reinforcement learning with stochastic dynamics. In *Uncertainty in Artificial Intelligence*, pages 99–109. PMLR, 2023.

[88] Ali Mesbah. Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, 36(6):30–44, 2016.

[89] Emanuel Todorov. Stochastic optimal control and estimation methods adapted to the noise characteristics of the sensorimotor system. *Neural computation*, 17(5):1084–1108, 2005.

[90] Jun-Sheng Wang and Guang-Hong Yang. Output-feedback control of unknown linear discrete-time systems with stochastic measurement and process noise via approximate dynamic programming. *IEEE transactions on cybernetics*, 48(7):1977–1988, 2017.

[91] Bin Li, Yuan Tan, Ai-Guo Wu, and Guang-Ren Duan. A distributionally robust optimization based method for stochastic model predictive control. *IEEE Transactions on Automatic Control*, 67(11):5762–5776, 2021.

[92] Kazuhide Okamoto, Maxim Goldshtein, and Panagiotis Tsiotras. Optimal covariance control for stochastic systems under chance constraints. *IEEE Control Systems Letters*, 2(2):266–271, 2018.

[93] Ali Mesbah, Stefan Streif, Rolf Findeisen, and Richard D Braatz. Stochastic nonlinear model predictive control with probabilistic constraints. In *2014 American control conference*, pages 2413–2419. IEEE, 2014.

[94] An Liu, Vincent KN Lau, and Borna Kananian. Stochastic successive convex approximation for non-convex constrained stochastic optimization. *IEEE Transactions on Signal Processing*, 67(16):4189–4203, 2019.

[95] Onur Celik, Hany Abdulsamad, and Jan Peters. Chance-constrained trajectory optimization for non-linear systems with unknown stochastic dynamics. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*,

pages 6828–6833. IEEE, 2019.

[96] Thomas Lew, Riccardo Bonalli, and Marco Pavone. Chance-constrained sequential convex programming for robust trajectory optimization. In *2020 European Control Conference (ECC)*, pages 1871–1878. IEEE, 2020.

[97] Joshua Pilipovsky and Panagiotis Tsiotras. Distributionally robust density control with wasserstein ambiguity sets. In *2024 IEEE 63rd Conference on Decision and Control (CDC)*, pages 1081–1086. IEEE, 2024.

[98] Zhichao Shi, Hao Liang, Shengjun Huang, and Venkata Dinavahi. Distributionally robust chance-constrained energy management for islanded microgrids. *IEEE Transactions on Smart Grid*, 10(2):2234–2244, 2018.

[99] Jiankun Sun, Jun Yang, and Zhigang Zeng. Safety-critical control with control barrier function based on disturbance observer. *IEEE Transactions on Automatic Control*, 69(7):4750–4756, 2024.

[100] Guokai Hao, Yuanzheng Li, Yang Li, Lin Jiang, and Zhigang Zeng. Lyapunov-based safe reinforcement learning for microgrid energy management. *IEEE transactions on neural networks and learning systems*, 2024.

[101] Samuel Pfrommer, Tanmay Gautam, Alec Zhou, and Somayeh Sojoudi. Safe reinforcement learning with chance-constrained model predictive control. In *Learning for Dynamics and Control Conference*, pages 291–303. PMLR, 2022.

[102] Shawon Dey and Hao Xu. Hierarchical game theoretical distributed adaptive control for large scale multi-group multi-agent system. *IET Control Theory & Applications*, 17(17):2332–2352, 2023.

[103] Tingxiang Zhang, Xiangdong Liu, and Xiaoyu Lang. Differential game-based distributed vibration control for gyroelastic beam using control moment gyroscopes. *Acta Astronautica*, 2025.

[104] Huaqing Li, Jun Li, Liang Ran, Lifeng Zheng, and Tingwen Huang. A survey of distributed algorithms for aggregative games. *IEEE/CAA Journal of Automatica Sinica*, 2025.

[105] Tamer Başar and Geert Jan Olsder. *Dynamic noncooperative game theory*. SIAM, 1998.

[106] Chonglin Jing, Chaoli Wang, Hongkai Song, Yibo Shi, and Longyan Hao. Optimal asymptotic tracking control for nonzero-sum differential game systems

with unknown drift dynamics via integral reinforcement learning. *Mathematics*, 12(16):2555, 2024.

[107] Bao-Qiang Zhang, Bing-Chang Wang, and Ying Cao. An online q-learning method for linear-quadratic nonzero-sum stochastic differential games with completely unknown dynamics. *Journal of Systems Science and Complexity*, 37(5):1907–1922, 2024.

[108] Qichao Zhang and Dongbin Zhao. Data-based reinforcement learning for nonzero-sum games with unknown drift dynamics. *IEEE Transactions on Cybernetics*, 49(8):2874–2885, 2018.

[109] Ning Bin, Huainian Zhu, and Qiyou Liu. Non-zero-sum stochastic differential games on investment, consumption and proportional reinsurance. *Journal of Industrial and Management Optimization*, 20(6):2032–2049, 2024.

[110] Jian-liang Zhang, Dong-lian Qi, and Miao Yu. A game theoretic approach for the distributed control of multi-agent systems under directed and time-varying topology. *International Journal of Control, Automation and Systems*, 12(4):749–758, 2014.

[111] Peng Yi, Jinlong Lei, Xiuxian Li, Shu Liang, Min Meng, and Jie Chen. A survey on noncooperative games and distributed nash equilibrium seeking over multi-agent networks. *CAAI Artificial Intelligence Research*, 1(1):8–27, 2022.

[112] Paul Werbos. Approximate dynamic programming for realtime control and neural modelling. *Handbook of intelligent control: neural, fuzzy and adaptive approaches*, pages 493–525, 1992.

[113] Dimitri P Bertsekas and John N Tsitsiklis. Neuro-dynamic programming: an overview. In *Proceedings of 1995 34th IEEE conference on decision and control*, volume 1, pages 560–564. IEEE, 1995.

[114] Jennie Si, Andrew G Barto, Warren B Powell, and Don Wunsch. *Handbook of learning and approximate dynamic programming*, volume 2. John Wiley & Sons, 2004.

[115] Qing Guo. Optimal robust control of electro-hydraulic system based on hamilton–jacobi–bellman solution with backstepping iteration. *IEEE Transactions on Control Systems Technology*, 31(1):459–466, 2022.

[116] Jingwei Lu, Qinglai Wei, and Fei-Yue Wang. Parallel control for optimal tracking via adaptive dynamic programming. *IEEE/CAA Journal of Automatica Sinica*,

7(6):1662–1674, 2020.

[117] Qinglai Wei and Tao Li. Constrained-cost adaptive dynamic programming for optimal control of discrete-time nonlinear systems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[118] Derong Liu and Qinglai Wei. Finite-approximation-error-based optimal control approach for discrete-time nonlinear systems. *IEEE Transactions on Cybernetics*, 43(2):779–789, 2013.

[119] Ding Wang, Derong Liu, Dongbin Zhao, Yuzhu Huang, and Dehua Zhang. A neural-network-based iterative gdhp approach for solving a class of nonlinear optimal control problems with control constraints. *Neural Computing and Applications*, 22(2):219–227, 2013.

[120] Alejandro Gonzalez-Garcia, David Barragan-Alcantar, Ivana Collado-Gonzalez, and Leonardo Garrido. Adaptive dynamic programming and deep reinforcement learning for the control of an unmanned surface vehicle: Experimental results. *Control Engineering Practice*, 111:104807, 2021.

[121] Jichao Liu, Yangzhou Chen, Jingyuan Zhan, and Fei Shang. Heuristic dynamic programming based online energy management strategy for plug-in hybrid electric vehicles. *IEEE Transactions on Vehicular Technology*, 68(5):4479–4493, 2019.

[122] Yaqian Wang and Xiaohong Jiao. Dual heuristic dynamic programming based energy management control for hybrid electric vehicles. *Energies*, 15(9):3235, 2022.

[123] Danil V Prokhorov and Donald C Wunsch. Adaptive critic designs. *IEEE transactions on Neural Networks*, 8(5):997–1007, 1997.

[124] Jennie Si and Yu-Tsung Wang. Online learning control by association and reinforcement. *IEEE Transactions on Neural networks*, 12(2):264–276, 2001.

[125] Nailah Firdausiyah, E Taniguchi, and AG Qureshi. Modeling city logistics using adaptive dynamic programming based multi-agent simulation. *Transportation Research Part E: Logistics and Transportation Review*, 125:74–96, 2019.

[126] He Jiang, Huaguang Zhang, Yanhong Luo, and Ji Han. Neural-network-based robust control schemes for nonlinear multiplayer systems with uncertainties via adaptive dynamic programming. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(3):579–588, 2018.

[127] Shan Xue, Biao Luo, Derong Liu, and Yueheng Li. Adaptive dynamic programming based event-triggered control for unknown continuous-time nonlinear systems with input constraints. *Neurocomputing*, 396:191–200, 2020.

[128] Dimitri Bertsekas. Multiagent reinforcement learning: Rollout and policy iteration. *IEEE/CAA Journal of Automatica Sinica*, 8(2):249–272, 2021.

[129] Shuping He, Haiyang Fang, Maoguang Zhang, Fei Liu, Xiaoli Luan, and Zhengdao Ding. Online policy iterative-based h-infinity optimization algorithm for a class of nonlinear systems. *Information Sciences*, 495:1–13, 2019.

[130] Huaguang Zhang, Lili Cui, Xin Zhang, and Yanhong Luo. Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method. *IEEE Transactions on Neural Networks*, 22(12):2226–2236, 2011.

[131] Rufus Isaacs. *Differential Games: a Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*. Courier Dover Publications, Mineola, NY, 1999.

[132] Mircea-Bogdan Radac and Timotei Lala. Robust control of unknown observable nonlinear systems solved as a zero-sum game. *IEEE Access*, 8:214153–214165, 2020.

[133] Mohammad Sarbaz, Iman Zamani, Mohammad Manthouri, and Asier Ibeas. Decentralized robust interval type-2 fuzzy model predictive control for takagi–sugeno large-scale systems. *Automatika*, 63(1):49–63, 2022.

[134] Qinglai Wei, Derong Liu, Qiao Lin, and Ruizhuo Song. Adaptive dynamic programming for discrete-time zero-sum games. *IEEE transactions on neural networks and learning systems*, 29(4):957–969, 2017.

[135] Derong Liu, Hongliang Li, and Ding Wang. Neural-network-based zero-sum game for discrete-time nonlinear systems via iterative adaptive dynamic programming algorithm. *Neurocomputing*, 110:92–100, 2013.

[136] Syed Ali Asad Rizvi and Zongli Lin. Output feedback adaptive dynamic programming for linear differential zero-sum games. *Automatica*, 122:109272, 2020.

[137] He Jiang, Huaguang Zhang, Ji Han, and Kun Zhang. Iterative adaptive dynamic programming methods with neural network implementation for multi-player zero-sum games. *Neurocomputing*, 307:54–60, 2018.

[138] Draguna Vrabie and Frank Lewis. Adaptive dynamic programming for online solution of a zero-sum differential game. *Journal of Control Theory and Applications*, 9(3):353–360, 2011.

[139] Huaguang Zhang, Qinglai Wei, and Derong Liu. An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games. *Automatica*, 47(1):207–214, 2011.

[140] Asma Al-Tamimi, Murad Abu-Khalaf, and Frank L Lewis. Adaptive critic designs for discrete-time zero-sum games with application to H_∞ control. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(1):240–247, 2007.

[141] Chunbin Qin, Ziyang Shang, Zhongwei Zhang, Dehua Zhang, and Jishi Zhang. Robust tracking control for non-zero-sum games of continuous-time uncertain nonlinear systems. *Mathematics*, 10(11):1904, 2022.

[142] Yongliang Yang, Hamidreza Modares, Kyriakos G Vamvoudakis, Wei He, Cheng-Zhong Xu, and Donald C Wunsch. Hamiltonian-driven adaptive dynamic programming with approximation errors. *IEEE Transactions on Cybernetics*, 52(12):13762–13773, 2021.

[143] Derong Liu and Qinglai Wei. Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems. *IEEE Transactions on Neural Networks and Learning Systems*, 25(3):621–634, 2013.

[144] Huaguang Zhang, He Jiang, Chaomin Luo, and Geyang Xiao. Discrete-time nonzero-sum games for multiplayer using policy-iteration-based adaptive dynamic programming algorithms. *IEEE transactions on cybernetics*, 47(10):3331–3340, 2016.

[145] Bo Dong, Tianjiao An, Fan Zhou, Shenquan Wang, Yulian Jiang, Keping Liu, Fu Liu, Huiqiu Lu, and Yuanchun Li. Decentralized robust optimal control for modular robot manipulators based on zero-sum game with adp. In *Advances in Neural Networks–ISNN 2019: 16th International Symposium on Neural Networks, ISNN 2019, Moscow, Russia, July 10–12, 2019, Proceedings, Part II 16*, pages 3–14. Springer, 2019.

[146] Bo Dong, Tianjiao An, Xinye Zhu, Yuanchun Li, and Keping Liu. Zero-sum game-based neuro-optimal control of modular robot manipulators with uncertain disturbance using critic only policy iteration. *Neurocomputing*, 450:183–196, 2021.

[147] Derong Liu, Hongliang Li, and Ding Wang. Online synchronous approximate optimal learning algorithm for multi-player non-zero-sum games with unknown dynamics. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 44(8):1015–1027, 2014.

[148] Kyriakos G Vamvoudakis and Frank L Lewis. Multi-player non-zero-sum games: Online adaptive learning solution of coupled hamilton–jacobi equations. *Automatica*, 47(8):1556–1569, 2011.

[149] Hojjat Salehinejad, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaee. Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*, 2017.

[150] Mohammad Sarbaz, MohammadReza Soltanian, Mohammad Manthouri, and Iman Zamani. Adaptive optimal control of chaotic system using backstepping neural network concept. In *2022 8th International Conference on Control, Instrumentation and Automation (ICCIA)*, pages 1–5. IEEE, 2022.

[151] Jianping Cai, Gang Chen, Xiushan Wu, Qiuzhen Yan, and Jianning Li. Adaptive output feedback control for uncertain nonlinear systems with unknown modeling errors. *Advanced Theory and Simulations*, 7(5):2301136, 2024.

[152] Zhangbao Xu, Chuanbin Sun, and Qingyun Liu. Output-feedback prescribed performance control for the full-state constrained nonlinear systems and its application to dc motor system. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 53(7):3898–3907, 2023.

[153] Emmanuel Cruz-Zavala, Emmanuel Nuño, and Jaime A Moreno. Trajectory-tracking in finite-time for robot manipulators with bounded torques via output-feedback. *International Journal of Control*, 96(4):907–921, 2023.

[154] Guichao Yang, Tao Zhu, Fengbo Yang, Longfei Cui, and Hua Wang. Output feedback adaptive rise control for uncertain nonlinear systems. *Asian Journal of Control*, 25(1):433–442, 2023.

[155] Liang Cao, Hongru Ren, Hongyi Li, and Renquan Lu. Event-triggered output-feedback control for large-scale systems with unknown hysteresis. *IEEE Transactions on Cybernetics*, 51(11):5236–5247, 2020.

[156] Anqing Wang, Lu Liu, Jianbin Qiu, and Gang Feng. Event-triggered adaptive fuzzy output-feedback control for nonstrict-feedback nonlinear systems with asymmetric output constraint. *IEEE Transactions on Cybernetics*, 52(1):712–722, 2020.

[157] Na Zhang, Jinling Liang, and Dehao Li. Pid output-feedback control and filtering for positive roesser system. *Circuits, Systems, and Signal Processing*, 43(1):152–171, 2024.

[158] Jun Zhao and Yongfeng Lv. Output-feedback robust tracking control of uncertain systems via adaptive learning. *International Journal of Control, Automation and Systems*, 21(4):1108–1118, 2023.

[159] Xiaoyang Gao, Yue Long, Tieshan Li, Xin Hu, CL Philip Chen, and Fuchun Sun. Optimal fuzzy output feedback control for dynamic positioning of vessels with finite-time disturbance rejection under thruster saturations. *IEEE Transactions on Fuzzy Systems*, 31(10):3447–3458, 2023.

[160] Jianchen Hu and Baocang Ding. Output feedback mpc for nonlinear system in large operation range. *IEEE Transactions on Automatic Control*, 68(12):7903–7910, 2023.

[161] Li-Ying Hao, Yu-Qing Zhang, Chao Shen, and Fengqiu Xu. Fault-tolerant control for unmanned marine vehicles via quantized integral sliding mode output feedback technique. *IEEE Transactions on Intelligent Transportation Systems*, 24(5):5014–5023, 2023.

[162] Puranjit Singh, Nariman Niknejad, Sushan Ru, and Yin Bao. A deep learning-based smartphone app for field-based blueberry yield prediction. In *ASA, CSSA, SSSA International Annual Meeting*, pages 21–31. ASA-CSSA-SSSA, 2023.

[163] Avner Friedman. Differential games. *Handbook of game theory with economic applications*, 2:781–799, 1994.

[164] P-L Lions and Panagiotis E Souganidis. Differential games, optimal control and directional derivatives of viscosity solutions of bellman's and isaacs' equations. *SIAM journal on control and optimization*, 23(4):566–583, 1985.

[165] Syed Ali Asad Rizvi and Zongli Lin. Output feedback q-learning for discrete-time linear zero-sum games with application to the h-infinity control. *Automatica*, 95:213–221, 2018.

[166] Xiaoyu Lang and Anton de Ruiter. Non-cooperative differential game based output feedback control for spacecraft attitude regulation. *Acta Astronautica*, 193:370–380, 2022.

[167] Jingliang Sun and Chunsheng Liu. Distributed zero-sum differential game for multi-agent systems in strict-feedback form with input saturation and output

constraint. *Neural Networks*, 106:8–19, 2018.

[168] MDS Aliyu. A new hamilton–jacobi differential game framework for nonlinear estimation and output feedback control. *Circuits, Systems, and Signal Processing*, 39(4):1831–1852, 2020.

[169] Jun Zhao, Yongfeng Lv, and Ziliang Zhao. Adaptive learning based output-feedback optimal control of ct two-player zero-sum games. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 69(3):1437–1441, 2021.

[170] Yongfeng Lv, Jun Zhao, Rong Li, and Xuemei Ren. Robust optimal control of the multi-input systems with unknown disturbance based on adaptive integral reinforcement learning q-function. *International Journal of Robust and Nonlinear Control*, 34(6):4234–4251, 2024.

[171] Hongye Lv and Huanhuan Yuan. Hierarchical differential game output feedback control for approaching non-cooperative target. In *2023 42nd Chinese Control Conference (CCC)*, pages 8846–8852. IEEE, 2023.

[172] Junhao Guo, Zhijian Ji, and Yungang Liu. Controllability of game-based multi-agent system. *Science China Information Sciences*, 66(12):222206, 2023.

[173] Ren-Ren Zhang and Lei Guo. Controllability of nash equilibrium in game-based control systems. *IEEE Transactions on Automatic Control*, 64(10):4180–4187, 2019.

[174] Muneomi Sagara and Hiroaki Mukaidani. Sign-indefinite linear quadratic differential game for stochastic systems via static output feedback strategy. *Electronics and Communications in Japan*, 106(3):e12401, 2023.

[175] Nariman Niknejad and Hamidreza Modares. Physics-informed data-driven safe and optimal control design. *IEEE Control Systems Letters*, 2023.

[176] Nadav Berman and Uri Shaked. H-inf like control for nonlinear stochastic systems. *Systems & Control Letters*, 55(3):247–257, 2006.

[177] Weihai Zhang and Bor-Sen Chen. State feedback h-inf control for a class of nonlinear stochastic systems. *SIAM journal on control and optimization*, 44(6):1973–1991, 2006.

[178] Qingguo Li and Shahram Payandeh. Planning for dynamic multiagent planar manipulation with uncertainty: A game theoretic approach. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 33(5):620–626,

2003.

[179] Li Ma, Ning Xu, Xudong Zhao, Guangdeng Zong, and Xin Huo. Small-gain technique-based adaptive neural output-feedback fault-tolerant control of switched nonlinear systems with unmodeled dynamics. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(11):7051–7062, 2020.

[180] Yuri Shtessel, Christopher Edwards, Leonid Fridman, Arie Levant, et al. *Sliding mode control and observation*, volume 10. Springer, 2014.

[181] Roberto Franco, Héctor Ríos, Denis Efimov, and Wilfrid Perruquetti. Adaptive estimation for uncertain nonlinear systems with measurement noise: A sliding-mode observer approach. *International Journal of Robust and Nonlinear Control*, 31(9):3809–3826, 2021.

[182] Mateusz Czyżniewski and Rafał Łangowski. A robust sliding mode observer for non-linear uncertain biochemical systems. *ISA transactions*, 123:25–45, 2022.

[183] Alberto Isidori, J van Schuppen, E Sontag, M Thoma, and M Krstic. Communications and control engineering. *Nonlinear control systems*, 1995.

[184] Jean-Paul Gauthier and Ivan AK Kupka. Observability and observers for nonlinear systems. *SIAM journal on control and optimization*, 32(4):975–994, 1994.

[185] Siri Schlanbusch and Jing Zhou. Adaptive backstepping control of a 2-dof helicopter. In *2019 7th International Conference on Control, Mechatronics and Automation (ICCMA)*, pages 210–215. IEEE, 2019.

[186] Hui Bi, Weitian He, Xiaowei Wang, Tao Zou, and Zhijia Zhao. Adaptive fault-tolerant control of a nonlinear 2-dof helicopter system with prescribed performance. *IET Control Theory & Applications*, 2024.

[187] Vijay Kumar Singh and Shyam Kamal. Prescribed-time adaptive backstepping control of an uncertain nonlinear 2-dof helicopter. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 70(11):4138–4142, 2023.

[188] Thanh-Do Huynh, Ngoc-Thanh Bach, Quang-Dao Le, Duong-Dong Le, Viet-Hoang Bui, Hoai-An Vo, Phi-Hung Pham, Huu-Dat Nguyen, et al. Implementation of fuzzy-pid controller for 2-dof helicopter. *Journal of Fuzzy Systems and Control*, 2(2):74–80, 2024.

[189] Jun Morimoto and Kenji Doya. Robust reinforcement learning. *Neural computation*, 17(2):335–359, 2005.

[190] Oliver Mihatsch and Ralph Neuneier. Risk-sensitive reinforcement learning. *Machine learning*, 49:267–290, 2002.

[191] Stefano P Coraluppi and Steven I Marcus. Risk-sensitive and minimax control of discrete-time, finite-state markov decision processes. *Automatica*, 35(2):301–309, 1999.

[192] J Morimioto, Garth Zeglin, and CG Atkeson. Minimax differential dynamic programming: application to a biped walking robot. In *SICE 2003 Annual Conference (IEEE Cat. No. 03TH8734)*, volume 3, pages 2310–2315. IEEE, 2003.

[193] Chi Kong Ng, Li-Zhi Liao, and Duan Li. A globally convergent and efficient method for unconstrained discrete-time optimal control. *Journal of Global Optimization*, 23:401–421, 2002.

[194] Mohammad Sarbaz, Iman Zamani, Mohammad Manthouri, and Asier Ibeas. Hierarchical optimization-based model predictive control for a class of discrete fuzzy large-scale systems considering time-varying delays and disturbances. *International Journal of Fuzzy Systems*, 24(4):2107–2130, 2022.

[195] Christopher Atkeson and Jun Morimoto. Nonparametric representation of policies and value functions: A trajectory-based approach. *Advances in neural information processing systems*, 15, 2002.

[196] Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

[197] Emanuel Todorov and Weiwei Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 300–306. IEEE, 2005.

[198] Chris Atkeson and Benjamin Stephens. Random sampling of states in dynamic programming. *Advances in neural information processing systems*, 20, 2007.

[199] Yuichiro Aoyama, George Boutselis, Akash Patel, and Evangelos A Theodorou. Constrained differential dynamic programming revisited. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9738–9744. IEEE, 2021.

[200] Nikolaus Kriegeskorte and Tal Golan. Neural network models and deep learning. *Current Biology*, 29(7):R231–R236, 2019.

[201] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*, chapter 7, pages 216–261. MIT Press, 2016.

[202] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[203] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[204] Jiayuan Huang and Robert L Nowack. Machine learning using u-net convolutional neural networks for the imaging of sparse seismic data. *Pure and Applied Geophysics*, 177(6):2685–2700, 2020.

[205] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of Mathematical Psychology*, 85:1–16, 2018.

[206] Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459, 2015.

[207] Carl Edward Rasmussen and Hannes Nickisch. Gaussian processes for machine learning (gpml) toolbox. *The Journal of Machine Learning Research*, 11:3011–3015, 2010.

[208] Kevin P Murphy. *Machine learning: a probabilistic perspective*, chapter 14,15, pages 1–22. MIT press, New York, 2012.

[209] Amir Hossein Zaeri, Samsul Bahari Mohd-Noor, Maryam Mohd Isa, Farah Saleena Taip, and Aida Esmaeilian Marnani. Disturbance rejection for a 2-dof nonlinear helicopter model by using mimo fuzzy sliding mode control with boundary layer. In *2012 Third International Conference on Intelligent Systems Modelling and Simulation*, pages 411–416. IEEE, 2012.

[210] H Sira-Ramirez, M Zribi, and Shaheen Ahmad. Dynamical sliding mode control approach for vertical flight regulation in helicopters. *IEE Proceedings-Control Theory and Applications*, 141(1):19–24, 1994.

[211] Jitka Dupacova, Jan Hurt, and Josef Stepan. *Stochastic modeling in economics and finance*, volume 75. Springer Science & Business Media, 2002.

[212] Anthony F Bartholomay. Stochastic models for chemical reactions: I. theory of the unimolecular reaction process. *The bulletin of mathematical biophysics*,

20:175–190, 1958.

[213] Matthew Tsao, Ramon Iglesias, and Marco Pavone. Stochastic model predictive control for autonomous mobility on demand. In *2018 21st International conference on intelligent transportation systems (ITSC)*, pages 3941–3948. IEEE, 2018.

[214] Giorgio Fabbri, Fausto Gozzi, and Andrzej Swiech. Stochastic optimal control in infinite dimension. *Probability and Stochastic Modelling. Springer*, 2017.

[215] Xin Sun, Runqi Chai, Senchun Chai, Baihai Zhang, and Antonios Tsourdos. Flexible final-time stochastic differential dynamic programming for autonomous vehicle trajectory optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 59(5):6658–6669, 2023.

[216] Wenhuai Li, Jiejin Cai, Chengjie Duan, Shu Chen, Peng Ding, Jiming Lin, and Dawei Cui. Learning and ensemble based mpc with differential dynamic programming for nuclear power autonomous control. *Expert Systems with Applications*, 215:119416, 2023.

[217] Jorge Estrela da Silva and Joao Borges de Sousa. A dynamic programming approach for the motion control of autonomous vehicles. In *49th IEEE conference on decision and control (CDC)*, pages 6660–6665. IEEE, 2010.

[218] Jie Su, Rui Gao, Zhenghao Xu, and Weidong Zhang. Vs-ddp: Variable splitting based differential dynamic programming algorithm for autonomous trajectory planning. In *2023 42nd Chinese Control Conference (CCC)*, pages 4382–4387. IEEE, 2023.

[219] Jorge Estrela da Silva and Joao Borges de Sousa. A dynamic programming based path-following controller for autonomous vehicles. *Control and Intelligent Systems*, 39(4):245, 2011.

[220] Hao-Yang Zhu, Yuan-Xin Li, and Shaocheng Tong. Dynamic event-triggered reinforcement learning control of stochastic nonlinear systems. *IEEE Transactions on Fuzzy Systems*, 31(9):2917–2928, 2023.

[221] ZC Su, Andy HF Chow, CL Fang, EM Liang, and RX Zhong. Hierarchical control for stochastic network traffic with reinforcement learning. *Transportation Research Part B: Methodological*, 167:196–216, 2023.

[222] Ali Forootani, Majid Ghaniee Zarch, Massimo Tipaldi, and Raffaele Iervolino. A stochastic dynamic programming approach for the machine replacement problem.

Engineering Applications of Artificial Intelligence, 118:105638, 2023.

[223] Elena Arcari, Andrea Iannelli, Andrea Carron, and Melanie N Zeilinger. Stochastic mpc with robustness to bounded parametric uncertainty. *IEEE Transactions on Automatic Control*, 2023.

[224] Yushi Hamaguchi. On the maximum principle for optimal control problems of stochastic volterra integral equations with delay. *Applied Mathematics & Optimization*, 87(3):42, 2023.

[225] Bin Zhang, Yingmin Jia, and Yuqi Zhang. Differential dynamic programming for finite-horizon zero-sum differential games of nonlinear systems. *International Journal of Robust and Nonlinear Control*, 33(18):11062–11084, 2023.

[226] Wei Sun and Justin Kleiber. Control constrained game theoretic differential dynamic programming. In *2023 American Control Conference (ACC)*, pages 1408–1413. IEEE, 2023.

[227] Krupa Prag, Matthew Woolway, and Turgay Celik. Toward data-driven optimal control: A systematic review of the landscape. *IEEE Access*, 10:32190–32212, 2022.

[228] Rui Cao, YuPing Lu, and Zhen He. System identification method based on interpretable machine learning for unknown aircraft dynamics. *Aerospace Science and Technology*, 126:107593, 2022.

[229] Lu Cheng, Siddharth Ramchandran, Tommi Vatanen, Niina Lietzén, Riitta Lahesmaa, Aki Vehtari, and Harri Lähdesmäki. An additive gaussian process regression model for interpretable non-parametric analysis of longitudinal data. *Nature communications*, 10(1):1798, 2019.

[230] Ivan V Maly. Diffusion approximation of the stochastic process of microtubule assembly. *Bulletin of Mathematical Biology*, 64(2):213–238, 2002.

[231] Silvan Türkcan, Antigoni Alexandrou, and Jean-Baptiste Masson. A bayesian inference scheme to extract diffusivity and potential fields from confined single-molecule trajectories. *Biophysical journal*, 102(10):2288–2298, 2012.

[232] Jin Chen, Per Jönsson, Masayuki Tamura, Zhihui Gu, Bunkei Matsushita, and Lars Eklundh. A simple method for reconstructing a high-quality ndvi time-series data set based on the savitzky–golay filter. *Remote sensing of Environment*, 91(3-4):332–344, 2004.

[233] Edward G Coffman, Gabor Galambos, Silvano Martello, and Daniele Vigo. Bin packing approximation algorithms: Combinatorial analysis. *Handbook of Combinatorial Optimization: Supplement Volume A*, pages 151–207, 1999.

[234] Richard Bellman. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6):503–515, 1954.

[235] Wendell H Fleming and Panagiotis E Souganidis. On the existence of value functions of two-player, zero-sum stochastic differential games. *Indiana University Mathematics Journal*, 38(2):293–314, 1989.

[236] Mohammad Sarbaz and Wei Sun. Min–max adaptive dynamic programming for zero-sum differential games. *International Journal of Control*, pages 1–10, 2024.

[237] Lujie Yang, Hongkai Dai, Alexandre Amice, and Russ Tedrake. Approximate optimal controller synthesis for cart-poles and quadrotors via sums-of-squares. *IEEE Robotics and Automation Letters*, 2023.

[238] Hefu Ye, Changyun Wen, and Yongduan Song. Decentralized and distributed control of large-scale interconnected multi-agent systems in prescribed time. *IEEE Transactions on Automatic Control*, 2024.

[239] Mohamed H Hassan, Salah Kamel, Francisco Jurado, and Umberto Desideri. Global optimization of economic load dispatch in large scale power systems using an enhanced social network search algorithm. *International Journal of Electrical Power & Energy Systems*, 156:109719, 2024.

[240] Zhenlei Ma, Xiaojian Li, and Jie Sun. A data-driven fault detection approach for unknown large-scale systems based on ga-svm. *Information Sciences*, 658:120023, 2024.

[241] Zheng Xu, Yulu Gong, Yanlin Zhou, Qiaozhi Bao, and Wenpin Qian. Enhancing kubernetes automated scheduling with deep learning and reinforcement techniques for large-scale cloud computing optimization. In *Ninth International Symposium on Advances in Electrical, Electronics, and Computer Engineering (ISAEECE 2024)*, volume 13291, pages 1595–1600. SPIE, 2024.

[242] Mojtaba Asadi Jokar, Iman Zamani, Mohammad Manthouri, and Mohammad Sarbaz. A novel robust extended dissipativity state feedback control system design for interval type-2 fuzzy takagi-sugeno large-scale systems. *Automatika*, 64(3):642–657, 2023.

[243] Peter ED Love, Lavagnon A Ika, and Jeffrey K Pinto. Smart heuristics for

decision-making in the 'wild': Navigating cost uncertainty in the construction of large-scale transport projects. *Production Planning & Control*, 35(16):2286–2303, 2024.

[244] Chengzhang Liang. Intelligent monitoring methodology for large-scale logistics transport vehicles based on parallel internet of vehicles. *EURASIP Journal on Wireless Communications and Networking*, 2023(1):75, 2023.

[245] Ahmed M Elberry, Jagruti Thakur, Annukka Santasalo-Aarnio, and Martti Larmi. Large-scale compressed hydrogen storage as part of renewable electricity storage systems. *International journal of hydrogen energy*, 46(29):15671–15690, 2021.

[246] Faisal Alkhaldi and Ali Alouani. Development of a generic model for large-scale healthcare organizations. In *2019 IEEE 19th International Symposium on High Assurance Systems Engineering (HASE)*, pages 200–207. IEEE, 2019.

[247] Mahdi Saadatmand, Babak Mozafari, Gevork B Gharehpetian, and Soodabeh Soleymani. Optimal pid controller of large-scale pv farms for power systems lfo damping. *International Transactions on Electrical Energy Systems*, 30(6):e12372, 2020.

[248] Eleftherios Vlahakis, Leonidas Dritsas, and George Halikias. Distributed lqr design for a class of large-scale multi-area power systems. *Energies*, 12(14):2664, 2019.

[249] Huanqing Wang, Peter Xiaoping Liu, Jialei Bao, Xue-Jun Xie, and Shuai Li. Adaptive neural output-feedback decentralized control for large-scale nonlinear systems with stochastic disturbances. *IEEE transactions on neural networks and learning systems*, 31(3):972–983, 2019.

[250] Changchao Li, Zhongjian Kang, Hongguo Yu, Hao Wang, Zheng Chang, and Kun Xue. Research on synchronization control of complex systems with unknown disturbances. In *2024 IEEE China International Youth Conference on Electrical Engineering (CIYCEE)*, pages 1–5. IEEE, 2024.

[251] Saad Bella, Azeddine Houari, Ali Djerioui, A Chouder, Mohamed Machmoum, M-F Benkhoris, and K Ghedamsi. Robust model predictive control (mpc) for large-scale pv plant based on paralleled three-phase inverters. *Solar Energy*, 202:409–419, 2020.

[252] Mohammad Sarbaz, Mohammad Manthouri, and Iman Zamani. Lmi-based robust fuzzy model predictive control of discrete-time fuzzy takagi-sugeno large-

scale systems based on hierarchical optimization and h performance. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 30(04):649–679, 2022.

[253] Yinliang Xu, Wei Zhang, and Wenxin Liu. Distributed dynamic programming-based approach for economic dispatch in smart grids. *IEEE Transactions on Industrial Informatics*, 11(1):166–175, 2014.

[254] Reza Aliakbarpour Shalmani, Mehdi Rahmani, and Nooshin Bigdeli. Nash-based robust distributed model predictive control for large-scale systems. *Journal of Process Control*, 88:43–53, 2020.

[255] Aaron Kandel, Saehong Park, Hector E Perez, Geumbee Kim, Yohwan Choi, Hyoung Jun Ahn, Won Tae Joe, and Scott J Moura. Distributionally robust surrogate optimal control for large-scale dynamical systems. In *2020 American Control Conference (ACC)*, pages 2225–2231. IEEE, 2020.

[256] Mehdi Hosseinzadeh, Luca Schenato, and Emanuele Garone. A distributed optimal power management system for microgrids with plug&play capabilities. *Advanced Control for Applications: Engineering and Industrial Systems*, 3(1):e65, 2021.

[257] Nasser Sadati, Mehdi Rahmani, and Mehrdad Saif. Two-level robust optimal control of large-scale nonlinear systems. *IEEE Systems Journal*, 9(1):242–251, 2013.

[258] Donghwan Lee, Han-Dong Lim, et al. Continuous-time distributed dynamic programming for networked multi-agent markov decision processes. In *2024 IEEE 18th International Conference on Control & Automation (ICCA)*, pages 960–967. IEEE, 2024.

[259] Graziana Cavone, Ton van den Boom, Lex Blenkers, Mariagrazia Dotoli, Carla Seatzu, and Bart De Schutter. An mpc-based rescheduling algorithm for disruptions and disturbances in large-scale railway networks. *IEEE Transactions on Automation Science and Engineering*, 19(1):99–112, 2020.

[260] Mohammad Sarbaz. Model predictive control for interval type-2 fuzzy systems with unknown time-varying delay in states and input vector. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 32(03):385–401, 2024.

[261] Yiming Teng, Haisheng Li, and Feng Wu. Design of distributed fractional order pid type dynamic matrix controller for large-scale process systems. *IEEE Access*,

8:179754–179771, 2020.

[262] Iman Zamani, Mohsen Shafieirad, Mohammad Manthouri, Mohammad Sarbaz, and Asier Ibeas. Nonlinear pseudo state-feedback controller design for affine fuzzy large-scale systems with h performance. *International Journal of Fuzzy Systems*, 25(1):80–95, 2023.

[263] Mohammad Sarbaz and Wei Sun. Sliding mode observer-based min-max differential dynamic programming for output feedback differential games. *International Journal of Control*, pages 1–16, 2025.

[264] David D Fan and Evangelos A Theodorou. Differential dynamic programming for time-delayed systems. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 573–579. IEEE, 2016.

[265] Hassan K Khalil. *High-gain observers in nonlinear feedback control*. SIAM, 2017.

[266] Xiong Yang and Haibo He. Adaptive dynamic programming for decentralized stabilization of uncertain nonlinear large-scale systems with mismatched interconnections. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(8):2870–2882, 2018.

[267] Petros A Ioannou and Jing Sun. Robust adaptive control. *Journal of Adaptive Control*, 1, 1996.

[268] Dinh Cong Huong. Design of event-triggered finite-time dissipative control for fractional-order time-delay interconnected systems. *Circuits, Systems, and Signal Processing*, pages 1–20, 2025.

[269] Dinh Cong Huong. Design of an event-triggered state feedback control for fractional-order interconnected systems. *Journal of Control, Automation and Electrical Systems*, 35(2):266–275, 2024.

[270] Muhammad Shamrooz Aslam, Hazrat Bilal, and Mohammad Hayajneh. Lqr-based pid controller with variable load tuned with data-driven methods for double inverted pendulum. *Soft Computing*, 28(1):325–338, 2024.

[271] Wenqian Fan, Ao Liu, and Ding Wang. Decentralized tracking control design based on intelligent critic for an interconnected spring-mass-damper system. *Complex Engineering Systems*, 3(1):N–A, 2023.

[272] DP Iracleous and AT Alexandridis. A multi-task automatic generation control for power regulation. *Electric Power Systems Research*, 73(3):275–285, 2005.

[273] Li Dai, Yanye Hao, Huahui Xie, Zhongqi Sun, and Yuanqing Xia. Distributed robust mpc for nonholonomic robots with obstacle and collision avoidance. *Control Theory and Technology*, 20(1):32–45, 2022.

[274] Giulio Ferro, Michela Robba, and Roberto Sacile. A fully distributed robust mpc approach for frequency and voltage regulation in smart grids with active and reactive power constraints. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2024.

[275] Robert F Stengel. Optimal control and estimation. *Book, Princton University*, 24(4):1–10, 1994.

[276] Richard Bellman. Dynamic programming. *science*, 153(3731):34–37, 1966.

[277] Yaolei Shen, Antonio Franchi, and Chiara Gabellieri. Aerial robots carrying flexible cables: Dynamic shape optimal control via spectral method model. *IEEE Transactions on Robotics*, 2025.

[278] R Fazel, AM Shafei, and SR Nekoo. Dynamic modeling and closed-loop control design for humanoid robotic systems: Gibbs–appell formulation and sdre approach. *Multibody System Dynamics*, 62(1):57–86, 2024.

[279] Tianjiao An, Bo Dong, Haoyu Yan, Lei Liu, and Bing Ma. Dynamic event-triggered strategy-based optimal control of modular robot manipulator: a multiplayer nonzero-sum game perspective. *IEEE Transactions on Cybernetics*, 2024.

[280] Xinji Zhu, Yujia Wang, and Zhe Wu. Reinforcement learning for optimal control of stochastic nonlinear systems. *AIChE Journal*, page e18840, 2025.

[281] Antonio Candelieri, Andrea Ponti, Elisabetta Fersini, Enza Messina, and Francesco Archetti. Safe optimal control of dynamic systems: Learning from experts and safely exploring new policies. *Mathematics*, 11(20):4347, 2023.

[282] Lukas Brunke, Melissa Greeff, Adam W Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5(1):411–444, 2022.

[283] Yixuan Wang, Simon Sinong Zhan, Ruochen Jiao, Zhilu Wang, Wanxin Jin, Zhuoran Yang, Zhaoran Wang, Chao Huang, and Qi Zhu. Enforcing hard constraints with soft barriers: Safe reinforcement learning in unknown stochastic environments. In *International Conference on Machine Learning*, pages 36593–36604. PMLR, 2023.

[284] Lukas Hewing, Kim P Wabersich, Marcel Menner, and Melanie N Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3(1):269–296, 2020.

[285] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European control conference (ECC)*, pages 3420–3431. Ieee, 2019.

[286] Sylvia L Herbert, Somil Bansal, Shromona Ghosh, and Claire J Tomlin. Reachability-based safety guarantees using efficient initializations. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 4810–4816. IEEE, 2019.

[287] Gokhan Alcan and Ville Kyrki. Differential dynamic programming with nonlinear safety constraints under system uncertainties. *IEEE Robotics and Automation Letters*, 7(2):1760–1767, 2022.

[288] Hassan Almubarak, Kyle Stachowicz, Nader Sadegh, and Evangelos A Theodorou. Safety embedded differential dynamic programming using discrete barrier states. *IEEE Robotics and Automation Letters*, 7(2):2755–2762, 2022.

[289] Chaoxu Mu, Ke Wang, Xin Xu, and Changyin Sun. Safe adaptive dynamic programming for multiplayer systems with static and moving no-entry regions. *IEEE Transactions on Artificial Intelligence*, 5(5):2079–2092, 2023.

[290] Stephen J Wright. Numerical optimization, 2006.

[291] Matthias Lorenzen, Fabrizio Dabbene, Roberto Tempo, and Frank Allgöwer. Constraint-tightening and stability in stochastic model predictive control. *IEEE Transactions on Automatic Control*, 62(7):3165–3177, 2016.

[292] Nathan Michael, Daniel Mellinger, Quentin Lindsey, and Vijay Kumar. The grasp multiple micro-uav testbed. *IEEE Robotics & Automation Magazine*, 17(3):56–65, 2010.

[293] Gerasimos Rigatos, Krishna Busawon, Jorge Pomares, and Masoud Abbaszadeh. Nonlinear optimal control for the wheeled inverted pendulum system. *Robotica*, 38(1):29–47, 2020.

[294] Runqi Chai, Antonios Tsourdos, Al Savvaris, Senchun Chai, Yuanqing Xia, and CL Philip Chen. Review of advanced guidance and control algorithms for space/aerospace vehicles. *Progress in Aerospace Sciences*, 122:100696, 2021.

[295] Harald Waschl, Ilya Kolmanovsky, Maarten Steinbuch, and Luigi Del Re. *Optimization and optimal control in automotive systems*, volume 455. Springer, 2014.

[296] Javier Arroyo, Fred Spiessens, and Lieve Helsen. Comparison of optimal control techniques for building energy management. *Frontiers in Built Environment*, 8:849754, 2022.

[297] Krishna Bhavithavya Kidambi, Cornelia Fermüller, Yiannis Aloimonos, and Huan Xu. Robust nonlinear control-based trajectory tracking for quadrotors under uncertainty. *IEEE Control Systems Letters*, 5(6):2042–2047, 2020.

[298] Derong Liu, Shan Xue, Bo Zhao, Biao Luo, and Qinglai Wei. Adaptive dynamic programming for control: A survey and recent advances. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(1):142–160, 2020.

[299] Ling Wang, Zixiao Pan, and Jingjing Wang. A review of reinforcement learning based intelligent optimization for manufacturing scheduling. *Complex System Modeling and Simulation*, 1(4):257–270, 2021.

[300] Wiebe Van der Hoek and Michael Wooldridge. Multi-agent systems. *Foundations of Artificial Intelligence*, 3:887–928, 2008.

[301] Shihui Liu, Huanqing Wang, Yunfeng Liu, Ning Xu, and Xudong Zhao. Sliding-mode surface-based adaptive optimal nonzero-sum games for saturated nonlinear multi-player systems with identifier-critic networks. *Neurocomputing*, 584:127575, 2024.

[302] Heng Zhao, Guangdeng Zong, Xudong Zhao, Huanqing Wang, Ning Xu, and Ning Zhao. Hierarchical sliding-mode surface-based adaptive critic tracking control for nonlinear multiplayer zero-sum games via generalized fuzzy hyperbolic models. *IEEE Transactions on Fuzzy Systems*, 31(11):4010–4023, 2023.

[303] Hanguang Su, Huaguang Zhang, He Jiang, and Yinlei Wen. Decentralized event-triggered adaptive control of discrete-time nonzero-sum games over wireless sensor-actuator networks with input constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 31(10):4254–4266, 2020.

[304] Bing Ma, Yuanchun Li, Tianjiao An, and Bo Dong. Compensator-critic structure-based neuro-optimal control of modular robot manipulators with uncertain environmental contacts using non-zero-sum games. *Knowledge-Based Systems*, 224:107100, 2021.

[305] Qi Zhu, Kun Zhang, and Xiangpeng Xie. Multi-event-triggered adaptive dynamic programming for non-zero-sum game of unknown nonlinear system. *International Journal of Robust and Nonlinear Control*, 34(8):5168–5189, 2024.

[306] Lei Guo and Han Zhao. Model-free adaptive optimal control of continuous-time nonlinear non-zero-sum games based on reinforcement learning. *IET Control Theory & Applications*, 17(2):223–239, 2023.

[307] Sholeh Yasini, Mohammad Bagher Naghibi Sitani, and Ali Kirampor. Reinforcement learning and neural networks for multi-agent nonzero-sum games of nonlinear constrained-input systems. *International Journal of Machine Learning and Cybernetics*, 7:967–980, 2016.

[308] Steven Okamoto, Noam Hazon, and Katia P Sycara. Solving non-zero sum multiagent network flow security games with attack costs. In *AAMAS*, pages 879–888, 2012.

[309] Siyu Guo, Yingnan Pan, Hongyi Li, and Liang Cao. Dynamic event-driven adp for n-player nonzero-sum games of constrained nonlinear systems. *IEEE Transactions on Automation Science and Engineering*, 2024.

[310] Yu Huo, Ding Wang, Junfei Qiao, and Menghua Li. Adaptive critic design for nonlinear multi-player zero-sum games with unknown dynamics and control constraints. *Nonlinear Dynamics*, 111(12):11671–11683, 2023.

[311] Haoming Zou, Guoshan Zhang, Wanquan Liu, and Zhiguo Yan. Dynamic event-triggered robust optimal tracking control for multi-player nonzero-sum games with mismatched uncertainties and asymmetric constrained inputs. *Information Sciences*, 662:120177, 2024.

[312] Tianyu Qiu, Han Zhang, and Jingchuan Wang. Nash pursuit strategy for nonzero-sum mpc game via inverse optimal control. In *2022 13th Asian Control Conference (ASCC)*, pages 2354–2359. IEEE, 2022.

[313] Chenyu Zhao, Qing Wang, Xiaofeng Liu, Chun Li, and Lidong Shi. Reinforcement learning based a non-zero-sum game for secure transmission against smart jamming. *Digital Signal Processing*, 112:103002, 2021.

[314] Yongfeng Lv and Xuemei Ren. Approximate nash solutions for multiplayer mixed-zero-sum game with reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(12):2739–2750, 2018.

[315] Kyriakos Vamvoudakis, Draguna Vrabie, and Frank Lewis. Adaptive optimal

control algorithm for zero-sum nash games with integral reinforcement learning. In *AIAA guidance, navigation, and control conference*, page 4773, 2012.

[316] Yinlei Wen, Huaguang Zhang, Hanguang Su, and He Ren. Optimal tracking control for non-zero-sum games of linear discrete-time systems via off-policy reinforcement learning. *Optimal Control Applications and Methods*, 41(4):1233–1250, 2020.

[317] Bosen Lian, Vrushabh S Donge, Wenqian Xue, Frank L Lewis, and Ali Davoudi. Distributed minmax strategy for multiplayer games: Stability, robustness, and algorithms. *IEEE Transactions on Neural Networks and Learning Systems*, 35(3):3265–3277, 2022.

[318] Yuxin Gao, Chunsheng Liu, Sen Jiang, and Shaojie Zhang. Zero-sum differential games-based fast adaptive robust optimal sliding mode control design for uncertain missile autopilot with constrained input. *International Journal of Control*, 95(7):1789–1801, 2022.

VITA

Mohammad Sarbaz received his B.S. degree in Electrical and Electronic Engineering with a focus on Control Systems from Azad University, Tehran, Iran, in 2016, and his M.S. degree in Electrical and Electronic Engineering with a focus on Control Systems from Shahed University, Tehran, Iran, in 2020. He is currently a Ph.D. candidate in Mechanical Engineering at the School of Aerospace and Mechanical Engineering, University of Oklahoma, Norman, Oklahoma, USA. His research interests include optimal control, differential games, reinforcement learning, and trajectory optimization, with applications in motion planning, aircraft systems, and pursuit–evasion games.