Foundations and Trends® in Systems and Control

Synchronous Reinforcement Learning-Based Control for Cognitive Autonomy

Suggested Citation: Kyriakos G. Vamvoudakis and Nick-Marios T. Kokolakis (2020), "Synchronous Reinforcement Learning-Based Control for Cognitive Autonomy", Foundations and Trends® in Systems and Control: Vol. 8, No. 1–2, pp 1–176. DOI: 10.1561/2600000022.

Kyriakos G. Vamvoudakis
Georgia Institute of Technology
USA
kyriakos@gatech.edu

Nick-Marios T. Kokolakis
Georgia Institute of Technology
USA
nmkokolakis@gatech.edu

now
the essence of knowledge
Boston — Delft

Contents

1 Introduction **3**
- 1.1 A Unified Approach . . . 3
- 1.2 RL and Cognitive Autonomy . . . 7
- 1.3 Organization . . . 9
- 1.4 Notation . . . 10

2 Optimal Regulation **11**
- 2.1 Introduction and Motivation . . . 12
- 2.2 Bellman-Based RL . . . 14
- 2.3 RL Based on an Integral Bellman Form . . . 19
- 2.4 Saturating Actuators and Relaxed PE . . . 23

3 Game-Theoretic Learning **31**
- 3.1 Introduction and Motivation . . . 31
- 3.2 Zero-Sum Games . . . 34
- 3.3 Non-Zero-Sum Games . . . 40
- 3.4 Stackelberg Games . . . 46
- 3.5 Graphical Games . . . 57

4 Model-Free RL with Q-Learning **65**
- 4.1 Introduction and Motivation . . . 65
- 4.2 Q-Learning for Optimal Regulation . . . 68

4.3 Q-Learning for Nash Games 73
4.4 Q-Learning for Multi-Agent Systems 78

5 Model-Based and Model-Free Intermittent RL 87
5.1 Introduction and Motivation 87
5.2 Optimal Control of Nonlinear Systems 89
5.3 Optimal Tracking Control of Nonlinear Systems 95
5.4 Intermittent Q-Learning 105

6 Bounded Rationality and Non-Equilibrium RL in Games 112
6.1 Introduction and Motivation 112
6.2 Non-Equilibrium Dynamic Games and RL 114
6.3 Games with Adversaries 117

7 Applications to Autonomous Vehicles 126
7.1 Kinodynamic Motion Planning 126
7.2 Bounded Rationality in Adversarial Target Tracking 138

8 Concluding Remarks 153

Acknowledgements 155

References 156

Synchronous Reinforcement Learning-Based Control for Cognitive Autonomy

Kyriakos G. Vamvoudakis[1] and Nick-Marios T. Kokolakis[2]

[1] *Georgia Institute of Technology, USA; kyriakos@gatech.edu*
[2] *Georgia Institute of Technology, USA; nmkokolakis@gatech.edu*

ABSTRACT

This monograph provides an exposition of recently developed reinforcement learning-based techniques for decision and control in human-engineered cognitive systems. The developed methods learn the solution to optimal control, zero-sum, non zero-sum, and graphical game problems completely online by using measured data along the system trajectories and have proved stability, optimality, and robustness. It is true that games have been shown to be important in robust control for disturbance rejection, and in coordinating activities among multiple agents in networked teams. We also consider cases with intermittent (an analogous to triggered control) instead of continuous learning and apply those techniques for optimal regulation and optimal tracking. We also introduce a bounded rational model to quantify the cognitive skills of a reinforcement learning agent. In order to do that, we leverage ideas from behavioral psychology to formulate differential games where the interacting learning agents have different intelligence skills, and we introduce an iterative method of optimal responses that determine the policy of an agent in adversarial environments. Finally, we present applications of reinforcement learning to motion planning and

Kyriakos G. Vamvoudakis and Nick-Marios T. Kokolakis (2020), "Synchronous Reinforcement Learning-Based Control for Cognitive Autonomy", Foundations and Trends® in Systems and Control: Vol. 8, No. 1–2, pp 1–176. DOI: 10.1561/2600000022.

collaborative target tracking of bounded rational unmanned aerial vehicles.

1

Introduction

1.1 A Unified Approach

This monograph describes the use of principles of reinforcement learning (RL) to design feedback policies for continuous-time dynamical systems that combine features of adaptive control and optimal control. *Adaptive control* (Ioannou and Fidan, 2006) and *optimal control* (Lewis *et al.*, 2012a) represent different philosophies for designing feedback controllers. These methods have been developed by the control systems community.

Optimal controllers minimize user-prescribed performance functions and are normally designed offline, i.e., performing all the calculations before being implemented into a system, by solving Hamilton–Jacobi–Bellman (HJB) equations, for example, the Riccati equation, using complete knowledge of the system dynamics. Determining optimal control policies for nonlinear systems requires the offline solution of nonlinear HJB equations.

Adaptive controllers learn online, i.e., process data and decide in real-time, to control unknown systems using data measured along the system trajectories. In fact, adaptive control is a powerful tool that uses online tuning of parameters to provide effective controllers for nonlinear or linear systems with modeling uncertainties and disturbances.

Closed-loop stability while learning the parameters is guaranteed, often by using Lyapunov design techniques. Parameter convergence, however, often requires that the measured signals carry sufficient information about the unknown parameters known as a persistence of excitation (PE) condition, that is similar to exploration and exploitation in the learning terminology. Nevertheless, adaptive controllers are not usually designed to be optimal in the sense of minimizing user-prescribed performance functions. Indirect adaptive controllers use system identification techniques to first identify the system parameters and then use the obtained model to solve optimal design equations (Ioannou and Fidan, 2006). Adaptive controllers may satisfy certain inverse optimality conditions (Li and Krstic, 1997).

Several machine learning techniques have been employed for enabling adaptive autonomy (Vamvoudakis *et al.*, 2015). Machine learning is grouped, in supervised, unsupervised or *RL*, depending on the amount and quality of feedback about the system or task. In supervised learning, the feedback information provided to learning algorithms is a labeled training data set, and the objective is to build the system model representing the learned relation between the input, output and system parameters. In unsupervised learning, no feedback information is provided to the algorithm and the objective is to classify the sample sets to different groups based on the similarity between the input samples. Finally, RL, that is the subject of this monograph, is a goal-oriented learning tool wherein the agent, decision maker or controller learns a policy to optimize a long-term reward by interacting with the environment. At each step, an RL agent gets evaluative feedback about the performance of its action, allowing it to improve the performance of subsequent actions (Bertsekas and Tsitsiklis, 1996; Cao, 2007; Liu *et al.*, 2017; Sutton and Barto, 2018; Wiering and Van Otterlo, 2012).

In a control engineering context, *RL* bridges the gap between traditional optimal control and adaptive control algorithms (Bertsekas, 2019; Hovakimyan and Cao, 2010; Ioannou and Fidan, 2006; Jiang and Jiang, 2013; Kamalapurkar *et al.*, 2018; Krstić and Kanellakopoulos, 1995; Lewis *et al.*, 2012a,b; Tao, 2003; Zhang *et al.*, 2020). In our framework the goal is to learn the optimal policy and value function for a potentially uncertain physical system. Nevertheless, it is worth pointing

out that the application of RL to the control discipline is not restricted solely in learning the optimal strategy and value function, but rather it is applicable in diverse applications such as system identification, adaptive control and even to the coordination of multi-agent systems (Hunt *et al.*, 1992; Mannor and Shamma, 2007; Poveda *et al.*, 2019; Sontag, 1993; Sontag and Sussmann, 1997; Wang and Hill, 2009). Unlike traditional optimal control, RL finds the solution to the HJB equation online. On the other hand, unlike traditional adaptive controllers, that are not usually designed to be optimal in the sense of minimizing cost functionals, RL algorithms are optimal. This has motivated control system researchers to enable adaptive and cognitive autonomy in an optimal manner by developing RL-based controllers. In continuous-time (CT) linear systems with multiple decision makers and quadratic costs, one has to rely on solving complicated matrix Riccati equations that require complete knowledge of the system matrices and need to be solved offline and then implemented online in the controller. In the era of complex and big data systems, modeling the processes exactly is most of the time infeasible and offline solutions make the systems vulnerable to parameter changes (drift).

Q-learning is a model-free action-dependent RL technique, i.e., does not require information about the environment, developed primarily for discrete-time systems (Watkins, 1989). It learns an action-dependent value function that ultimately gives the expected utility of taking a given action in a given state and following the optimal policy thereafter. When such an action-dependent value function is learned, the optimal policy can be computed easily. The *biggest strength* of Q-learning is that it is model-free. It has been proven in Watkins (1989) that for any finite Markov Decision Process, Q-learning eventually finds an optimal policy. In complex-systems Q-learning needs to store massive amounts of data, which makes the algorithm infeasible. This problem can be solved effectively by using adaptation techniques. Specifically, Q-learning can be improved by using the universal function approximation property that allow us to solve difficult optimization problems online and forward in time. This makes it possible to apply the algorithm to larger problems, even when the state space is continuous, and infinitely large.

Synchronous RL arises from a combination of techniques based on model-free and model-based RL. Specifically, RL techniques are used to design adaptive systems with novel structures that learn the solutions to optimization-based problems by observing data along the system trajectories. We term these as *optimal adaptive controllers*. These adaptive controllers are learned online and the policies converge to the optimal ones by tuning all parameters in all loops *simultaneously*, giving rise to *synchronous RL*. This is accomplished by developing two learning networks that interact with each other as they learn, and so mutually tune their parameters together *simultaneously* without any iterations. This learning mechanism is composed of an actor/critic structure, wherein there are two networks in two control loops – critic-network that evaluates the performance of current control policies and an actor-network that computes those current policies.

Game theory develops mathematical models allowing us to capture the strategic interaction among rational decision-makers/players (Başar and Olsder, 1999; Myerson, 2013). A rational agent can be thought of as an agent that has clear preferences, models uncertainty via expected values, and always chooses to perform the policy with the optimal expected outcome for itself from among all feasible actions. The solutions of several types of non-cooperative games (the cooperation among the agents is not allowed), namely the equilibrium strategies of the game, rely on the assumption of perfect rationality (Myerson, 2013). However, in real-world problems, the assumption of perfect rationality turns out to be quite strong and incapable of interpreting the actual behavior of the players (Crawford and Iriberri, 2007), thereby giving rise in bounded rationality (Simon, 1984) wherein the agents are bounded rational in the sense that the intelligence of the agents is limited by the information they have, the cognitive limitations of their minds, and the finite amount of time they have to make a decision. In the framework of RL, game theory is regraded as a bounded-rational interpretation of how equilibrium may result. Finally, based on the above, it follows that the *synchronous* RL can constitute a means for enabling online gaming by allowing the agents to learn their optimal policies online by measuring data along the players' trajectories, even when the environment is unknown or subject to changes.

1.2 RL and Cognitive Autonomy

Autonomy means having the freedom to act or function independently, i.e., self-government. Concerning the terminology of this term, it originally came from the Greek word "autonomia," which is a combination of the Greek words "auto" (self) and "nomy" (a system of rules). In the discipline of control engineering, this means that the agents can make a decision, namely to select a control policy, without involving a supervisor. Systems featuring these properties are the so-termed "Intelligent Autonomous Systems" (IAS), examples include Unmanned Aerial Vehicles (UAVs), Autonomous Underwater Vehicles (AUVs), office and residential buildings that regulate their energy consumption while adapting to the needs of their inhabitants (smart buildings), safety systems and environmentally friendly energy systems in automobiles (smart cars, smart highways) (Antsaklis *et al.*, 1991; Asama *et al.*, 2013; Vamvoudakis *et al.*, 2015). However, the IAS should be designed so that they are capable of dealing with the endogenous uncertainty imposing by the environment involving the presence of modeling uncertainties, the unavailability of the model, the possibility of cooperative along with non-cooperative goals, and malicious attacks compromising the security of teams of complex systems (Lamnabhi-Lagarrigue *et al.*, 2017). Nevertheless, it is evident that the *Synchronous RL* with the flexibility that it offers in tackling uncertainty, it has facilitated the evolution of cognitive autonomy aiming towards building fully autonomous IAS that are highly cognitive, reflective, multitask-able, and effective in knowledge discovery without external intervention. Ideally, moving towards full autonomy, the control engineering community desires to construct IAS, which should perhaps have the ability to perform even hardware repair if any of their components fails.

In general, there is a need for approaches that respond to situations not programmed or anticipated in the design. Therefore, by leveraging ideas from the recent advances of *Synchronous RL* and game theory, we bring together and combine interdisciplinary ideas from different fields as pictorially illustrated in Figure 1.1, i.e., computational intelligence, game theory, control theory, and information theory to endow IAS with novel cognitive learning algorithms intending to ensuring full autonomy

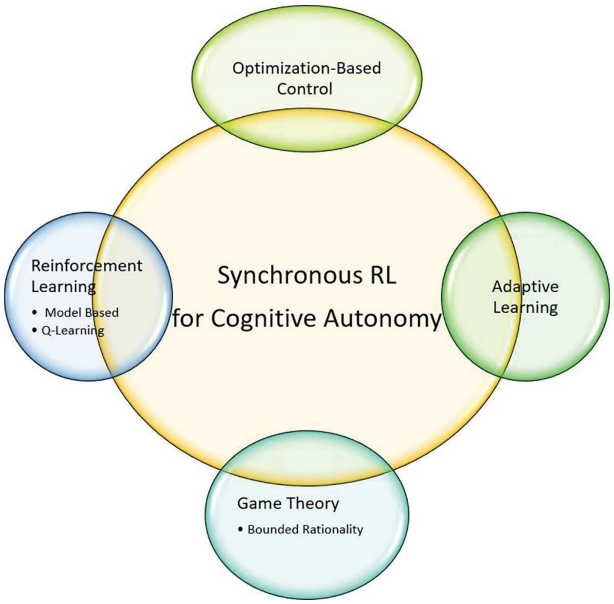

Figure 1.1: The *Synchronous RL*-based framework for enabling cognitive autonomy arises from the intersection of several diverse fields including, optimization-based control, adaptive learning, game theory, and RL.

and secure operation. Exploiting the adaptive nature of *Synchronous RL*, we apply the ideas of synchronous RL to kinodynamic motion planning algorithms that enable IAS to navigate securely and explore an unknown, challenging, environment with obstacles while guaranteeing the avoidance of collision with them. Furthermore, in the aerospace community is of profound importance to develop algorithms that will enable the coordination of autonomous swarms of UAVs to apprehend malicious vehicles that enter a protected zone, a phenomenon that has already been observed. To address that problem, we enforce "geofencing" protocols by constructing *cognitive* hierarchy-based algorithms inspired by the human brain, to coordinate a team of bounded rational UAVs for tracking an intelligent invading moving target. Finally, from the aforementioned, it is obvious that the *Synchronous RL*-based algorithms are featured by strong abilities of learning, and thus, the complex systems will be fully autonomous and tolerant to failures.

In this monograph we present a family of model-free, and model-based online adaptive learning algorithms for single and multi-agent systems using measurements along the system trajectories with continuous and intermittent feedback. The algorithms developed here are based on *Synchronous RL* principles, and rely on actor/critic-network schemes involving simultaneous tuning of the actor/critic neural networks (NNs) while providing online solutions to complex Hamilton–Jacobi (HJ) equations. However, it is worth mentioning that several of these techniques can be implemented without knowing the complete system dynamics, enabling cognitive autonomy.

1.3 Organization

The remainder of this monograph is structured as follows. Section 2 presents an adaptive method based on actor/critic RL for solving online the optimal control problem for deterministic CT input-affine nonlinear systems with known or partially unknown dynamics as well as with saturating and non-saturating actuators. In Section 3, under the assumption of perfect rationality, we develop adaptive controllers that learn optimal solutions for several differential game theory problems, including zero-sum, multi-player non-zero-sum, as well as graphical games. In the sequel, Section 4 proposes online Q-learning algorithms for solving the optimal control problem of a system with completely uncertain/unknown dynamics and shows its applications to differential game theory. Model-free and model-based intermittent control algorithms are displayed in Section 5 using ideas from RL. Next, by relaxing the assumption of perfect rationality, Section 6 introduces the non-equilibrium differential game theory and demonstrates its applications to cyber-physical systems security (CPS). Section 7 applies synchronous RL-based decision-making algorithms to motion planning in robotics as well as to coordinated target tracking using a team of bounded rational UAVs. Finally, Section 8 provides concluding remarks and potential future research perspectives on the area of synchronous RL-based control for cognitive autonomy.

Moreover, it is worth mentioning that throughout the monograph, we omit to include the proofs of the theorems as well as simulation

results to avoid breaking the flow of the document. Nevertheless, we refer the reader to particular references wherein there are complete proofs, and simulation results verifying the efficiency of the presented control algorithms. Last but not least, note that instead of having a "centralized" literature review in this introductory section, and in following with the spirit of this monograph, we adopt a "distributed" literature review approach, where each section itself contains a review of the references that are relevant to the particular section content.

1.4 Notation

The notation used here is standard. $\mathbb{R}_+$ is the set of positive real numbers. $\|\cdot\|$ denotes the Euclidean norm of a vector. The superscript $\star$ is used to denote the optimal solution of an optimization problem, $\underline{\lambda}(A)$ is the minimum eigenvalue of a matrix A, $\overline{\lambda}(A)$ is the maximum eigenvalue of a matrix A, $\mathrm{tr}(A)$ is the trace of a matrix A, and $\mathbf{1}_m$ is the column vector with m ones. The gradient of a scalar-valued function with respect to a vector-valued variable x is defined as a column vector, and is denoted by $\nabla \coloneqq \partial/\partial x$. The $\mathrm{vec}(A)$ and the $\mathrm{vech}(A)$ denote the vectorization and the half-vectorization of a symmetric $n \times n$ matrix A, respectively. The notations $\overline{K}$, $|K|$, and ∂K denote the closure, the cardinality, and the limit points of the set K, respectively. The $U \otimes V$ denotes the Kronecker product of two vectors. The $\oplus$ is the Minkowski sum of two sets.

2

Optimal Regulation

This section is concerned with the presentation of an online approximate solution method, based on policy iteration (PI), for the infinite-horizon optimal control problem for CT non-linear systems with known or partially unknown dynamics. In Subsection 2.2, we present an online adaptive algorithm whose structure is based on the actor/critic structure of PI in RL. However, in PI, the critic and actor-networks are tuned sequentially, that is, the actor network parameters are held constant while the critic-network is tuned. By contrast, the algorithm presented in this section involves simultaneous tuning of both actor and critic NNs (i.e., the parameters in both networks are tuned at the same time). We term this algorithm as synchronous RL. This approach results in a CT controller that operates more along the lines of standard adaptive controllers, still converges to an optimal solution. Next, in Subsection 2.3, we present an online adaptive learning algorithm based on Integral RL (IRL) principles which exploits the advantages of the synchronous RL to solve the infinite-horizon optimal control problem for non-linear systems, where the drift term is considered as unknown. Finally, in Subsection 2.4, we propose an RL algorithm with asymptotically stable equilibrium point to solve the infinite-horizon optimal control problem

for known deterministic nonlinear systems with saturating actuators and non-quadratic cost functional, where the requirement of PE is relaxed by using previously stored data concurrently with current data in the update of the critic NN. Asymptotic stability of the equilibrium point of the closed-loop system is achieved by adding a robustifying term to the controller to eliminate the effect of residual errors.

2.1 Introduction and Motivation

RL describes a family of learning systems that operates based on principles used in animals, social groups and naturally occurring systems. RL was used by Ivan Pavlov in the 1860s to train his dogs (Pavlov and Gantt, 1928; Sutton and Barto, 2018). Methods of RL have been developed by the Computational Intelligence Community in computer science engineering. RL allows the learning of optimal actions without knowing a dynamical model of the system or the environment. RL methods have not been extensively used in the feedback control community until recently.

RL is a sub-area of machine learning concerned with how to methodically modify the actions of an agent (player) based on observed responses from its environment (Barto, 1998; Kaelbling *et al.*, 1996; Sutton *et al.*, 1992). Hence, it brings together optimal and adaptive control in a new setting that we should call adaptive-optimal control (Kamalapurkar *et al.*, 2018; Kiumarsi *et al.*, 2017; Lewis and Liu, 2013; Lewis *et al.*, 2012b; Vrabie *et al.*, 2013). Every decision-making organism interacts with its environment and uses those interactions to improve its own actions in order to maximize the positive effect of its limited available resources; this, in turn, leads to better survival chances. RL is a means of learning optimal behaviors by observing the response from the environment to non-optimal control policies. In engineering terms, RL refers to the learning approach of an actor or agent that modifies its actions, or control policies, based on stimuli received in response to its interaction with its environment. This learning can be extended along two dimensions: (a) nature of the interaction (competitive or collaborative) and (b) the number of decision-makers (single or multi-agent). RL methods have allowed control systems researchers to develop

algorithms to learn online, the solutions to optimal control problems for dynamic systems that are described by difference or ordinary differential equations. These involve a computational intelligence technique known as Policy Iteration (PI) (Sutton *et al.*, 1992). PI provides effective means of learning solutions to HJ equations online. In control theoretic terms, the PI algorithm amounts to learning the solution of the HJ equation, then updating the policy by minimizing a Hamiltonian function. PI techniques have been developed for CT systems in Doya (2000). RL methods have been applied to learn online the solutions for optimal control problems for dynamic systems and differential games in Vrabie *et al.* (2013), Doya (2000), and Vamvoudakis *et al.* (2017a).

PI refers to a class of algorithms built as a two-step iteration: policy evaluation and policy improvement. Instead of trying a direct approach to solving the HJ equations, the PI algorithm starts by evaluating the cost of a given initial admissible (stabilizing and with a finite cost) control policy. The key to solving practically the HJ equations was in the use of approximators (Werbos, 1992, 2007), which can be trained to become approximate solutions of these equations. In fact, the PI algorithm for CT systems can be built on actor/critic structures that involve the use of approximating networks: the critic-networks are trained to become approximations of the Lyapunov equations solution at the policy evaluation step, whereas the actor-networks are trained to approximate improving policies at the policy improving step.

Optimal feedback control design has been responsible for much of the successful performance of engineered systems in aerospace, industrial processes, vehicles, ships, robotics. Traditional optimal feedback control design is performed offline by solving optimal design equations including the algebraic Riccati equation (ARE) for linear systems and HJ equations for nonlinear systems. However, it is usually difficult to solve the HJB equation for nonlinear systems except for special cases. Thus, this gives rise to the development of a new family of optimal adaptive controllers based on RL techniques, i.e., actor/critic mechanisms that converge online, to optimal control solutions by using data measured along the system trajectories. In particular, an RL controller is composed of two NNs, one for value function approximation (VFA) and one to approximate the control law, which could be called the critic

NN and actor NN, respectively. The two NNs are tuned simultaneously, that is, synchronously in time.

2.2 Bellman-Based RL

Consider the nonlinear CT system given by,

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t), \quad x(0) \coloneqq x_0, \ t \geqslant 0, \tag{2.1}$$

where $x(t) \in \mathbb{R}^n$ is the state vector, $u(t) \in U \subseteq \mathbb{R}^m$ is the control input, $f\colon \mathbb{R}^n \to \mathbb{R}^n$ and $g\colon \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are known functions such that $x \mapsto f(x) + g(x)u(x)$ is locally Lipschitz continuous in x for fixed u so that the solution $x(t)$ exists and is unique for any initial condition x_0 and piece-wise continuous feedback control $u(x) \in U$. We assume that the $x(t)$ is available for full state feedback.

It is desired to minimize the following infinite horizon cost functional,

$$V(x(0)) = \int_0^\infty r(x(\tau), u(\tau))\mathrm{d}\tau, \quad \forall x(0), \tag{2.2}$$

with $r(x,u) = Q(x) + R_s(u), \ \forall x, u$, where $Q(x)$ is a positive definite function and $R_s(u) := u^{\mathrm{T}}Ru$ non-negative (with $R \succ 0$), $\forall u \in U$.

The optimal control problem is to find an admissible (stabilizing policy that produces a finite value) control $u^\star(t)$ such that the equilibrium point of the closed-loop system (2.1) is asymptotically stable on $\mathbb{R}^n$ with the value $V(x(0))$ finite.

The Hamiltonian of the problem is defined as,

$$H\left(x, u, \frac{\partial V}{\partial x}\right) = r(x,u) + \frac{\partial V}{\partial x}^{\mathrm{T}} (f(x(t)) + g(x(t))u(t)), \quad \forall x, u,$$

where the dependence of V on x has been omitted for brevity and the optimal cost is given as,

$$V^\star(x(0)) = \min_u \int_0^\infty r\left(x(\tau), u(\tau)\right) \mathrm{d}\tau, \quad \forall x(0). \tag{2.3}$$

The optimal cost (2.3) is also called value function and should satisfy the HJB equation,

$$H\left(x, u^\star, \frac{\partial V^\star}{\partial x}\right) = 0, \quad \forall x, \tag{2.4}$$

with,

$$u^\star = \arg\min_u H\left(x, u, \frac{\partial V^\star}{\partial x}\right) := -\frac{1}{2}R^{-1}g(x)^{\mathrm{T}}\frac{\partial V^\star}{\partial x}, \quad \forall x. \tag{2.5}$$

Substituting the optimal control policy (2.5) we obtain the HJB equation in terms of $\frac{\partial V^\star}{\partial x}$. For the linear system case, considering a quadratic cost functional (i.e., $V^\star(x(0)) = x(0)^{\mathrm{T}}Px(0),\ \forall x(0)$), the equivalent of this HJB equation is the ARE. In order to find the optimal control solution for the problem, one shall need to solve the HJB Equation (2.4) for the value function and then substitute the solution in (2.5) to obtain the optimal control.

2.2.1 Approximate Solution

The structure used for our approximate solution is motivated by the Policy Iteration Algorithm that follows, where ϵ_{ac} is a small number used to terminate the algorithm when two consecutive value functions differ by less than ϵ_{ac}. In the linear case, this algorithm reduces to Kleinman's algorithm (Kleinman, 1968).

Algorithm 1: Policy Iteration for Nonlinear Systems

1: **procedure**
2: Given admissible policies $\mu^{(0)}$.
3: **while** $\|V^{\mu^{(i)}} - V^{\mu^{(i-1)}}\| \geqslant \epsilon_{\mathrm{ac}}$ **do**
4: Solve for the value $V^{(i)}(x)$ using Bellman's equation

$$Q(x) + \nabla V^{\mu^{(i)}\,\mathrm{T}}(f(x) + g(x)\mu^{(i)}) + \mu^{(i)\,\mathrm{T}}R\mu^{(i)} = 0, V^{\mu^{(i)}}(0) = 0.$$

5: Update the control policy $\mu^{(i+1)}$ using

$$\mu^{(i+1)} = -\frac{1}{2}R^{-1}g(x)^{\mathrm{T}}\nabla V^{\mu^{(i)}}.$$

6: $i \coloneqq i + 1$.
7: **end while**
8: **end procedure**

We should use a critic-network to approximate the cost and an actor-network to approximate the control. The critic-network is based on VFA.

In the following, it is desired to determine a rigorously justifiable form for the critic. We are interested in the approximation of the value $V(x)$ as well as its gradient. We assume that there exist weights W_{c} such that the value function $V(x)$ is approximated as,

$$V(x) = W_{\mathrm{c}}^{\mathrm{T}}\phi_c(x) + \epsilon(x), \quad \forall x.$$

Then, $\phi_c(x)$: $\mathbb{R}^n \to \mathbb{R}^N$ is called the basis function vector, N the number of neurons in the hidden layer, and $\epsilon(x)$ the approximation error.

But since we do not know the optimal weights W_{c}, we shall use current critic weights as,

$$\hat{V}(x) = \hat{W}_{\mathrm{c}}^{\mathrm{T}}\phi_c(x), \quad \forall x, \tag{2.6}$$

where $\hat{W}_c$ are the current estimated values of the ideal critic weights W_c. Now, the approximate HJB equation is given as,

$$\hat{H}(\cdot) := \hat{W}_c^{\mathrm{T}}\nabla\phi_c(f + g\hat{u}) + Q(x) + \hat{u}^{\mathrm{T}}R\hat{u} = e_1,$$

where $e_1 \in \mathbb{R}^n$ is the residual error after using current critic weights and $\hat{u}$ is given by,

$$\hat{u}(x) = -\frac{1}{2}R^{-1}g(x)^{\mathrm{T}}\nabla\phi_c(x)^{\mathrm{T}}\hat{W}_u, \quad \forall x, \tag{2.7}$$

that is an approximation of the optimal control policy, estimated by the actor NN whose weights are denoted as $\hat{W}_u$.

The following definition is required for the analysis below.

Definition 2.1 (PE). A vector signal $\Phi(t)$ is PE over the interval $[t, t+T]$, with $T \in \mathbb{R}^+$ if there exists $\beta_1, \beta_2 \in \mathbb{R}^+$ such that $\beta_1 I \leqslant \int_t^{t+T} \Phi(\tau)\Phi^{\mathrm{T}}(\tau)\mathrm{d}\tau \leqslant \beta_2 I,\ t \geqslant 0$, with I an identity matrix of appropriate dimensions. □

As we shall find out in the analysis below, the PE property of the regressor vector is required for establishing exponential convergence of the estimation error to the origin when the adaptive laws that generate the online estimate of the unknown parameters are derived through the gradient descent of a convex function of the estimation error (Ioannou and Fidan, 2006).

Given an approximation of the optimal control policy $\hat{u}(x)$, we define the squared residual error,

$$E_1 = \frac{1}{2} e_1^{\mathrm{T}} e_1.$$

Now it is desired to select $\hat{W}_c$ such that $e_1 \to 0$ in order for the weights $\hat{W}_c \to W_c$. Hence by using a normalized gradient descent, one has,

$$\dot{\hat{W}}_c = -a \frac{\partial E_1}{\partial \hat{W}_c} = -\alpha \frac{\sigma}{(\sigma^{\mathrm{T}}\sigma + 1)^2} (\sigma^{\mathrm{T}} \hat{W}_c + Q(x) + \hat{u}^{\mathrm{T}} R \hat{u}), \tag{2.8}$$

where $\alpha \in \mathbb{R}_+$ is a tuning gain that determines the speed of convergence and $\sigma = \nabla \phi_c (f + g\hat{u})$. To guarantee convergence of $\hat{W}_{\mathrm{c}}$ to W_{c}, the signal $\bar{\sigma} := \sigma/(\sigma^{\mathrm{T}}\sigma + 1)$ has to be PE.

In general, the PE assumption is needed in adaptive control if one desires to perform system identification (Ioannou and Fidan, 2006; Tao, 2003). It is needed here because one effectively desires to perform value function identification, namely, to identify the critic parameters to approximate $V(x)$.

The weights for the actor $\hat{W}_u$ need to be picked appropriately in order to guarantee closed-loop stability. Hence one has,

$$\dot{\hat{W}}_u = -\alpha_u \Bigg((F_1 \hat{W}_u - \mathbf{1}^{\mathrm{T}} \bar{\sigma}^{\mathrm{T}} \hat{W}_c) - \frac{1}{4} (\nabla \phi_c(x) g(x) R^{-1} g(x)^{\mathrm{T}} \nabla \phi_c^{\mathrm{T}}(x)) \hat{W}_u \times \left(\frac{\sigma}{(\sigma^{\mathrm{T}}\sigma + 1)^2} \right)^{\mathrm{T}} \hat{W}_c \Bigg), \tag{2.9}$$

where $\alpha_u \in \mathbb{R}_+$ is a tuning gain that determines the speed of convergence and $F_1 > 0$ is a matrix picked appropriately to guarantee stability and $\mathbf{1}$ is a row vector of ones with appropriate dimensions.

At this point, before proceeding in the presentation of our main results, we need to state the next definition along with some standard assumptions in neuro-inspired control.

Definition 2.2 (Uniform Ultimate Boundedness)**.** A time signal $\zeta(t)$ is said to be uniformly ultimately bounded (UUB) if there exists a compact set $S \subset R^n$ so that for all $\zeta(0) \in S$ there exists a bound B and a time $T(B, \zeta(0))$ such that $\|\zeta(t)\| \leqslant B$ for all $t \geqslant t_0 + T$. □

Assumption 2.1. For a given compact set $\Omega \subseteq \mathbb{R}^n$:

(a) $f(\cdot)$ is Lipschitz so that $\|f(x)\| \leqslant b_f \|x\|$.

(b) $g(\cdot)$ is bounded by a constant $\|g(x)\| < b_{\mathrm{g}}$.

(c) The NN approximation error and its gradient are bounded so that

$$\|\epsilon(x)\| < b_\epsilon, \quad \|\nabla \epsilon(x)\| < b_{\epsilon_x}.$$

(d) The NN activation function and its gradients are bounded so that

$$\|\phi_1(x)\| < b_\phi, \quad \|\nabla \phi_1(x)\| < b_{\phi_x}. \qquad \square$$

Assumption 2.1(d) is satisfied, by sigmoids, tanh, and other standard NN activation functions.

We will now present the main theorem, which provides simultaneous tuning laws for the actor and critic NNs that guarantee convergence online to the optimal policy while ensuring closed-loop stability.

Theorem 2.2 (Vamvoudakis and Lewis, 2010, Thm. 2)**.** Let the critic NN be given by (2.6) and the control input be given by actor NN (2.7). Let tuning for the critic NN be provided by (2.8), and assume that $\bar{\sigma}$ is PE. Let the actor NN be tuned as (2.9). Let Assumption 2.1 hold, and the tuning parameter for the critic being sufficiently larger than the actor. Then there exists an N_0 such that, for the number of hidden-layer units $N > N_0$ the closed-loop signals are UUB. ■

Remark 2.1. The theorem shows that PE is needed for proper identification of the value function by the critic NN, and that a non-standard tuning algorithm is required for the actor NN to guarantee stability. The second term in (2.9) is a cross-product term that involves both the critic weights and the actor weights. It is needed to guarantee good behavior of the Lyapunov function, that is, that the energy decreases to a bounded compact region. □

Remark 2.2. It is important to note that the Theorem 2.2 does not rely at all on convergence of the PI Algorithm. PI, a RL method, is used to obtain the structure of the policy, whereas its performance is carried out using adaptive control Lyapunov techniques. □

Remark 2.3. The PI must be initialized with a stabilizing control policy. The convergence proof requires this, because the Bellman equation has a positive definite solution only if the policy is admissible. A stabilizing policy can be difficult to find for non-linear systems. By contrast, the online synchronous RL does not require an initial stabilizing policy. It converges to the optimal control solution as long as the initial weight estimation errors are not large. □

On the basis of the above, a pseudocode that describes the proposed RL algorithm has the following form,

Algorithm 2: RL Algorithm for Optimal Regulation

1: **procedure**
2: Start with initial state $x(0)$ and random initial weights $\hat{W}_c(0), \hat{W}_u(0)$.
3: Propagate $t, x(t)$.
4: Propagate $\hat{W}_c(t), \hat{W}_u(t)$ ▷ $\dot{\hat{W}}_c$ as in (2.8) and $\dot{\hat{W}}_u$ as in (2.9).
5: Compute (2.6), and (2.7).
6: **end procedure**

2.2.2 Further Reading

The interested reader is directed to Lewis *et al.* (2012b), Vamvoudakis and Lewis (2010), and Vrabie *et al.* (2013) for detailed theorems and proofs of the above statements and algorithms and to Bhasin *et al.* (2013) wherein an RL-based solution is developed for the infinite-horizon optimal control problem for CT uncertain nonlinear systems. Finally, the works of Grondman *et al.* (2012) and Wang *et al.* (2017a) provide a survey of several standard and natural actor-critic algorithms.

2.3 RL Based on an Integral Bellman Form

In this subsection, we present a partially model-free online optimal control algorithm with integral RL. The PI algorithm given previously requires full system dynamics, because both $f(x)$ and $g(x)$ appear in the Bellman Equation (2.4). To find an equivalent formulation of the Bellman equation that does not involve the dynamics, we note that for any time t_0 and time interval $T \in \mathbb{R}_+$ the value function (2.2)

satisfies $\forall x_{t_0}$,

$$V\left(x_{t_0-T}\right) = \int_{t_0-T}^{t_0} r(x(\tau), u(x(\tau)))\mathrm{d}\tau + V\left(x_{t_0}\right). \tag{2.10}$$

In Vrabie *et al.* (2009) it is shown that (2.10) and (2.4) are equivalent, that is, they both have the same solution. Therefore, (2.10) can be viewed as a Bellman equation for CT systems. Note that this form does not involve the system dynamics. We call this the IRL form of the Bellman equation. Therefore, by using a critic NN for VFA, the Bellman error based on (2.10) becomes,

$$\int_{t-T}^{T} \left(Q(x) + u^{\mathrm{T}} \frac{R}{T} u\right) \mathrm{d}\tau + W_1^{\mathrm{T}} \phi(x(t)) - W_1^{\mathrm{T}} \phi(x(t-T)) \equiv \varepsilon_B, \tag{2.11}$$

where the parameter in the control weighting term is selected as the time $T \in \mathbb{R}_+$ of the integral. This has two objectives, first, to determine the control input effort, that is, the smaller the T is, the less control effort is required to steer the state space trajectory to the origin, and second, T is used for stability guarantees in the Lyapunov proof. We define the integral reinforcement as,

$$p(t) = \int_{t-T}^{T} \left(Q(x) + u^{\mathrm{T}} \frac{R}{T} u\right) \mathrm{d}\tau, \quad t \geqslant 0. \tag{2.12}$$

Now (2.11) can be written as,

$$\varepsilon_B - p = W_1^{\mathrm{T}} \Delta\phi(x(t)),$$

where

$$\Delta\phi(x(t)) \equiv \phi(x(t)) - \phi(x(t-T)), \quad t \geqslant 0.$$

Under the Lipschitz assumption on the dynamics, the residual error ε_B is bounded on a compact set.

Remark 2.4. Note that, as $N \to \infty$, the $\varepsilon_B \to 0$ uniformly (Abu-Khalaf and Lewis, 2005). □

2.3.1 Approximate Solution

Standard PI algorithms for CT systems are off-line methods that require complete knowledge on the system dynamics to obtain the solution

(i.e., the functions $f(x)$ and $g(x)$ in Bellman (2.4) need to be known). It is desired to change the off-line character of PI for CT systems and implement it online as in adaptive control mechanisms. Therefore, we present an adaptive learning algorithm that uses simultaneous CT tuning for the actor and critic NN and does not need the drift term $f(x)$ in the dynamics. We term this an *online IRL algorithm*. The online IRL can be computed by introducing an integrator and is equivalent to the Bellman equation (Vrabie *et al.*, 2009). It is added as an extra continuous-time state that functions as the memory or controller dynamics. Therefore, the policy is piecewise constant in time.

The weights of the critic NN, W_1 that solve (2.11) are unknown. The output of the critic NN is,

$$\hat{V}(x) = \hat{W}_1^{\mathrm{T}}\phi(x), \quad \forall x,$$

where $\hat{W}_1$ are the current estimated values of the critic NN weights. Recall that $\phi(x)$: $\mathbb{R}^n \to \mathbb{R}^N$ is the activation functions vector, with N the number of neurons in the hidden layer. The approximate Bellman error $\forall x$ is then,

$$\int_{t-T}^{T} \left(Q(x) + u^{\mathrm{T}} \frac{R}{T} u \right) \mathrm{d}\tau + \hat{W}_1^{\mathrm{T}}\phi(x(t)) - \hat{W}_1^{\mathrm{T}}\phi(x(t-T)) = e_1,$$

which according to (2.12) can be written as,

$$\hat{W}_1^{\mathrm{T}}\Delta\phi(x(t)) = e_1 - p. \tag{2.13}$$

It is desired to select $\hat{W}_1$ to minimize the squared residual error,

$$E_1 = \frac{1}{2} e_1^{\mathrm{T}} e_1.$$

We select the tuning law for the critic weights as the normalized gradient descent algorithm,

$$\dot{\hat{W}}_1 = -a_1 \frac{\Delta\phi_2(t)}{\left(\Delta\phi_2(t)^{\mathrm{T}}\Delta\phi_2(t) + 1\right)^2} \times \left(\Delta\phi_2(t)^{\mathrm{T}}\hat{W}_1 + \int_{t-T}^{T} \left(Q(x) + \frac{1}{4}\hat{W}_2^{\mathrm{T}}\bar{D}_1\hat{W}_2 \right) \mathrm{d}\tau \right), \tag{2.14}$$

where $a_1 \in \mathbb{R}_+$ is a tuning parameter that determines the speed of learning, $\bar{D}_1(x) := T\nabla\phi(x)g(x)R^{-1}g(x)^{\mathrm{T}}\nabla\phi^{\mathrm{T}}(x)$, and $u_2(x)$ is the approximated control policy computed by an action NN in the structured

form by,

$$u_2(x) = -\frac{1}{2}R^{-1}g^{\mathrm{T}}(x)\nabla\phi^{\mathrm{T}}\hat{W}_2, \quad \forall x,$$

where $\hat{W}_2$ denotes the current known values of the actor NN weights.

Note that the data required in this tuning algorithm at each time are $(\Delta\phi(t), p(t))$. The system dynamics $f(x)$ and $g(x)$ are not needed.

Although it is traditional to use critic tuning algorithms of the form (2.14), it is not generally understood when convergence of the critic weights can be guaranteed. In this subsection, we address this issue in a formal manner. To guarantee convergence of $\hat{W}_1$ to W_1, the PE assumption is required. According to (2.13) the signal vector $\bar{\Delta}\phi(t) := \Delta\phi(t)/(\Delta\phi(t)^{\mathrm{T}}\Delta\phi(t) + 1)$, must be PE to solve for W_1 in a least squares sense.

Moreover, the actor NN be tuned as,

$$\dot{\hat{W}}_2 = -a_2\left\{(F_2\hat{W}_2 - \mathbf{1}^{\mathrm{T}}T\bar{\Delta}\phi^{\mathrm{T}}\hat{W}_1) - \frac{1}{4m_s}\bar{D}_1(x)\hat{W}_2\bar{\Delta}\phi^{\mathrm{T}}\hat{W}_1\right\}, \tag{2.15}$$

where $a_2 \in \mathbb{R}_+$ is a tuning parameter that determines the speed of learning, $m_s := \left(\Delta\phi(t)^{\mathrm{T}}\Delta\phi(t) + 1\right)$, and $F_2 > 0$ is chosen appropriately.

We now present the main result, which provides the synchronous tuning laws for the actor and critic NN that guarantee convergence to the optimal policy along with closed-loop stability.

Theorem 2.3 (Vamvoudakis *et al.*, 2014, Thm. 1)**.** Let the tuning for the critic NN be provided by (2.14) and assume that $\bar{\Delta}\phi(t) := \Delta\phi(t)/(\Delta\phi(t)^{\mathrm{T}}\Delta\phi(t) + 1)$ is PE. Let the actor NN be tuned as (2.15). Let Assumption 2.1 hold. Then, there exists a N_0 and a time T_0 such that, for the number of hidden layer units $N > N_0$ and the time interval $T < T_0$, the closed-loop signals are UUB. ■

Remark 2.5. The theorem shows that PE is needed for proper identification of the value function by the critic NN, and that nonstandard tuning algorithm is required for the actor NN to guarantee stability while learning the optimal control solution. The PE condition in Theorem 2.3 is equivalent to the exploration paradigm in RL, which ensures sufficient sampling of the state space and convergence to the optimal policy. Nevertheless, a limitation of this approach is that we

cannot guarantee the PE condition for all present and future time, and also the PE condition can be even unfeasible for certain applications. Thus, in the next subsection, we relax the PE condition by utilizing current and recorded data. □

2.3.2 Further Reading

We refer the reader to Vamvoudakis *et al.* (2014), Vrabie *et al.* (2013), and Lewis *et al.* (2012b), for additional information in terms of the proofs of the theorems presented in this subsection. The works of Modares and Lewis (2014), Modares *et al.* (2013), and Bhasin *et al.* (2013), developed model-based algorithms with identifying structures. An off-policy iteration algorithm for optimal regulation without requiring the dynamics was presented in Jiang and Jiang (2012).

2.4 Saturating Actuators and Relaxed PE

In industrial applications, physical inputs to devices (such as voltages, currents, flows, and torques) are subject to saturations, which must be considered in the optimal control problem. This subsection develops a framework based on adaptive dynamic programming to solve the infinite-horizon optimal control problem for known deterministic nonlinear systems with saturating actuators and non-quadratic cost functionals.

The proposed algorithm is based on an actor/critic framework, where a critic NN is used to learn the optimal cost, and an actor NN is used to learn the optimal control policy. In the previous subsections, we figured out that owing to the adaptive control nature of the PI algorithm is required a PE condition to be a priori validated, but this can be relaxed using previously-stored data concurrently with current data in the update of the critic NN. A robustifying control term is added to the controller to eliminate the effect of residual errors, leading to the asymptotically stability of the closed-loop system.

To force *bounded inputs* in the algorithm derived in Subsection 2.2, (e.g., $|u_i| \leqslant \bar{u}, \forall i \in \{1, \ldots, m\}$) we follow the approach in Lyshevski (1996, 1998) and Abu-Khalaf and Lewis (2005) and use a non-quadratic

penalty function in (2.2) of the form,

$$R_s(u) = 2\sum_{i=1}^{m}\int_0^{u_i}(\theta^{-1}(v_i))^{\mathrm{T}}\varrho_i \mathrm{d}v_i, \quad \forall u,$$

with weighting factor $\varrho_i \in \mathbb{R}_+$, $i = \{1, \ldots, m\}$ and with abuse of notation we can write the component-wise operations in compact form as,

$$R_s(u) = 2\int_0^{u}(\theta^{-1}(v))^{\mathrm{T}} R \mathrm{d}v, \quad \forall u,$$

where R is a diagonal positive definite matrix consisting of the $\varrho_i > 0$, $i = \{1, \ldots, m\}$ terms, $v \in \mathbb{R}^m$, and $\theta(\cdot)$ is a continuous, one-to-one real-analytic integrable function of class C^{μ}, $\mu \geqslant 1$, used to map $\mathbb{R}$ onto the interval $(-\bar{u}, \bar{u})$ satisfying $\theta(0) = 0$. Also note that $R_s(u)$ is positive definite because $\theta^{-1}(v)$ is monotonic odd, e.g., one could select,

$$R_s(u) = 2\int_0^{u}(\theta^{-1}(v))^{\mathrm{T}} R \mathrm{d}v \coloneqq 2\int_0^{u}(\bar{u}\tanh^{-1}(v/\bar{u}))^{\mathrm{T}} R \mathrm{d}v > 0, \quad \forall u. \tag{2.16}$$

The optimal value function is defined as,

$$V^{\star}(x(t)) = \min_{u \in U}\int_t^{\infty} r(x,u)\mathrm{d}\tau, \quad \forall x,\ t \geqslant 0, \tag{2.17}$$

subject to the state dynamics in (2.1). The Hamiltonian of (2.1) associated with the cost function (2.2), can be written as,

$$H(x,u,\nabla V) = \frac{\partial V}{\partial x}^{\mathrm{T}}(f(x) + g(x)u(x)) + Q(x) + R_s(u), \quad \forall x, u. \tag{2.18}$$

The constrained optimal control input for the system (2.1), with cost (2.16), (2.17), can be obtained using the stationarity condition in the Hamiltonian (2.18):

$$u^{\star}(x) = -\theta\left(\frac{1}{2}R^{-1}g(x)^{\mathrm{T}}\frac{\partial V^{\star}}{\partial x}\right), \quad \forall x. \tag{2.19}$$

By substituting (2.19) into (2.18) one gets the HJB equation with bounded (saturated) control inputs as follows,

$$H\left(x, u^{\star}, \frac{\partial V^{\star}}{\partial x}\right) = 0, \quad \forall x. \tag{2.20}$$

2.4.1 Approximate Solution

The structure used for our approximate solution is motivated by the Policy Iteration Algorithm that follows, where ϵ_{ac} is a small number used to terminate the algorithm when two consecutive value functions differ by less than ϵ_{ac}. In the linear case, this algorithm reduces to Kleinman's algorithm (Kleinman, 1968).

Algorithm 3: Policy Iteration for Nonlinear Systems with Saturating Actuators

procedure
  Given admissible policies $\mu^{(0)}$ and $i = 1$.
  while $\|V^{\mu^{(i)}} - V^{\mu^{(i-1)}}\| \geqslant \epsilon_{\mathrm{ac}}$ **do**
    Solve for the value $V^{(i)}(x)$ using Bellman's equation

$$Q(x) + \frac{\partial V^{\star\,\mu^{(i)}\mathrm{T}}}{\partial x}(f(x) + g(x)\mu^{(i)}) + R_s(\mu^{(i)}) = 0, \quad V^{\mu^{(i)}}(0) = 0.$$

    Update the control policy $\mu^{(i+1)}$ using

$$\mu^{(i+1)} = -\theta\left(\frac{1}{2}R^{-1}g(x)^{\mathrm{T}}\frac{\partial V^{\star\,\mu^{(i)}}}{\partial x}\right).$$

    $i \coloneqq i + 1$.
  end while
end procedure

We should use a critic to approximate the cost and an actor to approximate the control. The first step to solve the HJB Equation (2.20) is locally to approximate the value function $V^\star(x)$ in (2.17) with a critic, within a set $\Omega \subseteq \mathbb{R}^n$ that contains the origin, as follows,

$$V^\star(x) = W^{\star\mathrm{T}}\phi(x) + \epsilon(x), \quad \forall x \in \Omega, \tag{2.21}$$

where $W^\star \in \mathbb{R}^N$ is an ideal weight vector satisfying $\|W^\star\| \leqslant W_{\mathrm{m}}$; $\phi(x)\colon \Omega \to \mathbb{R}^N$, $\phi(x) = [\varphi_1(x)\ \varphi_2(x)\ \ldots\ \varphi_N(x)]^{\mathrm{T}}$ are the basis functions such that $\varphi_i(0) = 0$ and $\nabla\varphi_i(0) = 0$, $\forall i = 1, \ldots, N$; N is the number of neurons in the hidden layer; and $\epsilon(x)$ is the approximation error.

One should pick the basis functions $\varphi_i(x)$, $\forall i \in \{1, 2, \dots, N\}$ as quadratic, radial basis or sigmoidal functions so that they define a complete independent basis set for $V^\star$. In this case, $V^\star$ and its derivatives,

$$\frac{\partial V^\star}{\partial x} = \left[\frac{\partial}{\partial x}\phi(x)\right]^{\mathrm{T}} W^\star + \frac{\partial}{\partial x}\epsilon(x) =: \nabla\phi(x)^{\mathrm{T}} W^\star + \nabla\epsilon(x), \quad \forall x \in \Omega,$$

can be uniformly approximated on any given compact set Ω. According to the Weierstrass Higher Order Approximation Theorem (Abu-Khalaf and Lewis, 2005; Hornik *et al.*, 1989), as the number of basis sets N increases, the approximation error on a compact set Ω goes to zero, i.e., $\epsilon(x) \to 0$ as $N \to \infty$. Since the ideal weights $W^\star$ for the (approximate) value function $V^\star(x)$ that appear in (2.21) are unknown, one must consider the *critic weight estimates* $\hat{W} \in \mathbb{R}^N$, associated with the approximate value function:

$$\hat{V}(x) = \hat{W}^{\mathrm{T}}\phi(x), \quad \forall x.$$

Our objective is to find an update law for the weight estimates $\hat{W}$ so that they converge to the ideal values $W^\star$, and thus provide a good estimate,

$$\hat{H}(x, u, \hat{W}^{\mathrm{T}}\nabla\phi) \coloneqq \hat{W}^{\mathrm{T}}\nabla\phi(x)(f + gu) + Q(x) + R_s(u), \quad \forall x, u, \tag{2.22}$$

for the (approximate) Hamiltonian where we have substituted the approximate value function in the Hamiltonian, see Vamvoudakis and Lewis (2010). To achieve convergence of (2.22) to the (approximate) Hamiltonian along the closed-loop trajectories, one would typically need PE for the vector $\omega(t)$ defined by,

$$\omega(t) \coloneqq \nabla\phi(x(t))\,(f(x(t)) + g(x(t))u(t)),$$

along the closed-loop trajectories (Ioannou and Fidan, 2006).

To weaken the need to guarantee a-priori, a PE condition for infinite-time, we follow the approach proposed in Chowdhary and Johnson (2010) that uses *past recorded data, concurrently with current data*. To this effect, we define the Hamiltonian error corresponding to the data collected at the current time t,

$$\begin{aligned} e(t) &:= \hat{H}(x(t), u(t), \hat{W}(t)^{\mathrm{T}}\nabla\phi(x(t))) - H^\star(x, u^\star(x), \nabla V^\star) \\ &= \hat{H}(x(t), u(t), \hat{W}(t)^{\mathrm{T}}\nabla\phi(x(t))), \quad \forall x, u, \end{aligned}$$

where this is due to (2.20), and the error corresponding to the previously collected data at times $t_0, t_1, \ldots, t_k < t$,

$$\begin{aligned} e_{\text{buff}_i}(t_i, t) &:= \hat{H}(x(t_i), u(t_i), \hat{W}(t)^{\mathrm{T}} \nabla\phi(x(t_i))) \\ &:= \hat{W}(t)^{\mathrm{T}} \nabla\phi(x(t_i)(f(x(t_i)) + g(x(t_i))u(t_i)). \\ &\quad + Q(x(t_i)) + R_S(u(t_i)). \end{aligned}$$

We draw attention to the reader that, while the error $e_{\text{buff}_i}(t_i, t)$ uses past state and input data $x(t_i)$ and $u(t_i)$, respectively, it is defined based on the current weight estimates $\hat{W}(t)$.

The current and previous errors defined above can be combined into the following (normalized) global error:

$$E(t) = \frac{1}{2}\left(\frac{e(t)^2}{(\omega(t)^{\mathrm{T}}\omega(t)+1)^2} + \sum_{i=1}^{k} \frac{e^2_{\text{buff}_i}(t_i, t)}{(\omega(t_i)^{\mathrm{T}}\omega(t_i)+1)^2}\right), \quad t \geqslant 0,$$

where $\omega(t_i) := \nabla\phi(x(t_i))(f(x(t_i)) + g(x(t_i))u(t_i))$.

The tuning for the critic is obtained by a gradient-descent-like rule as follows:

$$\begin{aligned} \dot{\hat{W}} &= -\alpha \frac{\partial E}{\partial \hat{W}} \\ &= -\alpha \frac{\omega(t)e(t)}{(\omega(t)^{\mathrm{T}}\omega(t)+1)^2} - \alpha \sum_{i=1}^{k} \frac{\omega(t_i)e_{\text{buff}_i}(t_i, t)}{(\omega(t_i)^{\mathrm{T}}\omega(t_i)+1)^2} \\ &= -\alpha \frac{\omega(t)(\omega(t)^{\mathrm{T}}\hat{W}(t) + R_s(u(t)) + Q(x(t)))}{(\omega(t)^{\mathrm{T}}\omega(t)+1)^2} \\ &\quad - \alpha \sum_{i=1}^{k} \frac{\omega(t_i)(\omega(t_i)^{\mathrm{T}}\hat{W}(t) + Q(x(t_i)) + R_s(u(t_i)))}{(\omega(t_i)^{\mathrm{T}}\omega(t_i)+1)^2}, \end{aligned} \tag{2.23}$$

$\forall t > t_i \geqslant 0$, where $\alpha \in \mathbb{R}_+$ is a constant gain that determines the speed of convergence. We define the weight estimation error of the critic by,

$$\tilde{W} \coloneqq W^\star - \hat{W} \in \mathbb{R}^N.$$

Remark 2.6. Typical RL control algorithms (Vamvoudakis and Lewis, 2010) do not have the extra past-data terms,

$$\Lambda := \sum_{i=1}^{k} \left(\frac{\omega(t_i)\omega(t_i)^{\mathrm{T}}}{(\omega(t_i)^{\mathrm{T}}\omega(t_i)+1)^2}\right),$$

in the tuning law of the critic (2.23) and, thus, need a PE condition on $\omega(t)/(\omega(t)^{\mathrm{T}}\omega(t)+1)$. This is equivalent to requiring that the matrix $\int_t^{t+\mathrm{T}}(\omega(t)\omega(t)^{\mathrm{T}}/(\omega(t)^{\mathrm{T}}\omega(t)+1)^2)\mathrm{d}\tau \in \mathbb{R}^{n\times n}$ is positive definite over any finite interval, which in turn, is equivalent to requiring that the signal $\omega(t)$ contains at least n spectral lines. This condition cannot be verified during learning, especially for nonlinear systems. The relaxed PE condition comes through the requirement that at least N of the vectors $\{\omega(t_1),\ldots,\omega(t_k)\}$ must be linearly independent, which is equivalent to the matrix Λ being positive definite. In practice, as one collects each additional vector $\omega(t_i)$, one adds a new term to the matrix Λ, and one can stop recording points as soon as this matrix becomes full rank (i.e., t_k time has been reached). From that point forward, one does not need to record new data, regardless of whether or not future data provide additional excitation. Notice, that the selection of the times t_i is somewhat arbitrary. □

The optimal control policy (2.19) can be approximated by an actor as follows,

$$u^\star(x) = W_u^{\star\mathrm{T}}\phi_u(x) + \epsilon_u(x), \quad \forall x, \tag{2.24}$$

where $W_u^\star \in \mathbb{R}^{N_2\times m}$ is an ideal weight matrix, $\phi_u(x)$ are the actor basis functions defined similarly to the critic, N_2 is the number of basis, and ϵ_u is the actor approximation error. As before, the basis functions must define a complete independent basis set so that $u^\star(x)$ can be uniformly approximated on Ω.

Since the ideal weighs $W_u^\star$ are not known, we introduce *actor estimate weights* $\hat{W}_u \in \mathbb{R}^{N_2\times m}$ to approximate the optimal control in (2.24) by the following estimate:

$$\hat{u}(x) = \hat{W}_u^{\mathrm{T}}\phi_u(x), \quad \forall x.$$

The tuning for the actor is obtained by a gradient-descent-like rule as follows:

$$\dot{\hat{W}}_u = -\alpha_u\phi_u\left(\hat{W}_u^{\mathrm{T}}\phi_u + \theta\left(\frac{1}{2}R^{-1}g(x)^{\mathrm{T}}\nabla\phi^{\mathrm{T}}\hat{W}\right)\right)^{\mathrm{T}}, \tag{2.25}$$

where $\alpha_u \in \mathbb{R}_+$ is a constant gain that determines the speed of convergence. We define the weight estimation error for the actor by,

$$\tilde{W}_u := W_u^\star - \hat{W}_u \in \mathbb{R}^{N_2\times m}.$$

A pseudocode that describes the proposed RL control algorithm has the following form,

Algorithm 4: RL Control Algorithm with Saturating Actuators and Relaxed PE

procedure
  Start with initial state $x(0)$, random initial weights $\hat{W}_u(0), \hat{W}(0)$ and $i = 1$.
  Propagate $t, x(t)$.
  Propagate $\hat{W}(t), \hat{W}_u(t)$ ▷ $\dot{\hat{W}}$ as in (2.23) and $\dot{\hat{W}}_u$ as in (2.25).
  Compute $\hat{V}(x) = \hat{W}^{\mathrm{T}}\phi(x)$ ▷ output of the Critic, and $\hat{u}(x) = \hat{W}_u^{\mathrm{T}}\phi_u(x)$ ▷ output of the Actor.
  while $i \neq k$ **do** ▷ Until $\{\omega(t_1), \omega(t_2), \ldots, \omega(t_k)\}$ has N linearly independent elements.
    Select arbitrary data points to be included in the history stack.
    $i \coloneqq i + 1$.
  end while
end procedure

However, aiming at removing the effect of the NN approximation errors and obtaining a closed loop system with an asymptotically stable equilibrium point, one needs to add a robustifying term to the control law and use:

$$\hat{u}(x) = \hat{W}_u^{\mathrm{T}}\phi_u(x) + \eta, \quad \forall x,$$

where

$$\eta := -\bar{B}\|x\|^2 \frac{\mathbf{1}_m}{(\bar{A} + x^{\mathrm{T}}x)}, \quad \forall x,$$

with $\bar{A}$, $\bar{B} \in \mathbb{R}^+$ exhibiting some particular properties shown in Vamvoudakis *et al.* (2016).

We now present the main result which guarantees the asymptotic stability of the learning algorithm of the resulting closed-loop dynamics,

$$\dot{x} = f(x) + g(x)((W_u^\star - \tilde{W}_u)^{\mathrm{T}}\phi_u(x) + \eta), \quad t \geqslant 0. \tag{2.26}$$

Assumption 2.4. The actor activation functions in ϕ_u, and the actor residual error ϵ_u are all uniformly bounded on a set $\Omega \subseteq \mathbb{R}^n$ in the sense that there exist finite constant $\phi_{\mathrm{um}},\ \epsilon_{\mathrm{um}} \in \mathbb{R}_+$, such that $|\phi_u(x)| \leqslant \phi_{\mathrm{um}}$ and $|\epsilon_u(x)| \leqslant \epsilon_{\mathrm{um}},\ \forall x \in \Omega$. In order to get ϵ_u small, we also assume that we have a large number of basis sets. □

Theorem 2.5 (Vamvoudakis *et al.*, 2016, Thm. 3)**.** Consider the closed-loop dynamics given by (2.26) together with the tuning laws for the critic and actor NNs given by (2.23) and (2.25), respectively. Suppose that the HJB Equation (2.20) has a positive definite and smooth solution, Assumption 2.1 hold, and that $\{\omega(t_1), \omega(t_2), \ldots, \omega(t_k)\}$ has N linearly independent elements. Then, there exists a triple $(\Omega_x \times \Omega_W \times \Omega_{W_u}) \subset \Omega$ with Ω compact, such that the solution $\tilde{Z} := \left(x(t), \tilde{W}(t), \tilde{W}_u(t)\right) \in (\Omega_x \times \Omega_W \times \Omega_{W_u})$ exists globally and converges asymptotically to zero for all the NN weights $W(0)$ inside Ω_W, $\tilde{W}_u(0)$ inside Ω_{W_u}, and state $x(0)$ inside Ω_x, provided that the following inequalities are satisfied:

$$\alpha > \sqrt{\frac{1}{8\lambda_{\min}\left(\sum_{i=1}^{k} \frac{\omega(t_i)\omega(t_i)^{\mathrm{T}}}{(\omega(t_i)^{\mathrm{T}}\omega(t_i)+1)^2}\right)}}; \quad \phi_{\mathrm{um}} > \frac{1+\sqrt{65}}{8}.$$

When the set Ω that appears in Assumption 2.1 is the whole $\mathbb{R}^n$, the triple $\Omega_W \times \Omega_{W_u} \times \Omega_x$ can also be the whole $\mathbb{R}^n$. ■

2.4.2 Further Reading

The interested reader is directed to Vamvoudakis *et al.* (2016) for detailed theorems and proofs. The authors in Modares *et al.* (2014) develop an IRL algorithm on an actor–critic structure to learn online the solution to the HJB equation for partially-unknown constrained-input systems by using the technique of experience replay, i.e., storing the agent's experiences at each time step in a data set called the replay memory, to update the critic weights to solve an IRL Bellman equation. The work of Benosman (2018) presents an overview of adaptive control by contrasting model-based approaches with data-driven approaches. The author classifies adaptive controllers into two main sub-fields, namely, model-based adaptive control and data-driven adaptive control.

3

Game-Theoretic Learning

In this section, we develop learning policies based on the synchronous RL algorithm that learns optimal solutions online for several differential game theory problems, including zero-sum (Subsection 3.2); multi-player non-zero-sum (Subsection 3.3); Stackelberg (Subsection 3.4) as well as graphical games (Subsection 3.5). The design procedure is to first formulate policy iteration algorithms for these problems, then use the structure of policy iteration to motivate the structure of multi-loop RL structures. The tuning laws for these novel adaptive policies are given, and the result is a family of online adaptive policies that converge to optimal game-theoretic solutions online using data generated along the system trajectories.

3.1 Introduction and Motivation

Complex human-engineered systems involve an interconnection of multiple decision makers (or agents) whose collective behavior depends on a compilation of local decisions that are based on partial information about each other and the state of the environment (Lewis *et al.*, 2014; Marden and Shamma, 2015, 2018a,b; Rantzer, 2008; Semsar-Kazerooni and Khorasani, 2009). Strategic interactions among agents in these

systems can be modeled as a multi-player simultaneous-move game (Arthur, 2018; Başar and Bernhard, 2008; Camerer, 2011; Engwerda, 2005). The agents involved can have conflicting objectives, and it is natural to make decisions based upon optimizing individual payoffs or costs.

Game theory has been mostly pioneered in the field of economics; (Rabin, 1957) considered a finite win-loss game with perfect information between two players, and this classic example of computable economics stands in the long and distinguished tradition of game theory that goes back to Schwalbe and Walker (2001) and Euwe *et al.* (1982). Reference Velupillai (2011) discusses game theory in algorithmic modes but not in what is today referred to as algorithmic game theory after realizing the futility of "algorithmizing" the uncompromisingly subjective von Neumann and Nash approach (Morgenstern and Von Neumann, 1953; Nash, 1951).

Study in the control systems community (Cao, 2020; Marden and Shamma, 2018a; Ungureanu, 2018) has primarily focused on noncooperative (Hespanha, 2017) zero-sum games arising in the form of H_∞ robust control of single-agent systems. Non-zero-sum multi-player game-theoretic control methods have also been employed to provide a basis for the study of coordination, conflict, and control for a single dynamical system with multiple players or control inputs. Graphical games (Kearns, 2007) are also used for the case where players communicate using a graph network topology and can only receive information along the graph edges. Multi-player games arise in real-world scenarios to find ways, for example, to optimally allocate the resources of optical networks (Pavel, 2006), design optimal motion planning for multiple robots with different goals (LaValle and Hutchinson, 1998), study the coalition formation of robots for detecting intrusion (Liang and Xiao, 2009), coordinate the charging of autonomous plug-in electric vehicles (Ma *et al.*, 2011), design cooperative policies across the electricity network to find new transmission routes when a power line is broken (MacKenzie and Wicker, 2001; Saad *et al.*, 2012), to protect a network from adversaries (Alpcan and Başar, 2010; Roy *et al.*, 2010), and provide behavioral decision-making models for planning and operating transportation systems (Fisk, 1984). Additionally, aggregative games

provide a rich abstraction to model strategic multi-agent interactions and can be applied to electricity markets (Paccagnan *et al.*, 2016a,b; 2019).

Strategies for team decision problems, including N-player games (both non-zero-sum and zero-sum), are normally solved offline by solving the coupled Hamilton–Jacobi (HJ) equations for nonlinear systems or coupled Riccati equations for linear systems. For example, max-plus basis methods have been widely used to solve the HJ equations (Fleming and McEneaney, 2000; McEneaney, 2006). This method employs grid-based methods, such as finite-difference or finite-element methods, to approximate the solution to the HJ equations. Although elegant, this approach is offline, requires complete knowledge of the system dynamics, and the computational time grows exponentially with the state-space dimension. Moreover, using these offline approaches, players cannot change their objectives online without calling for a completely new offline solution for the new strategies. However, given the nature of cooperation (or conflict) and that the environment is highly uncertain and dynamic, enabling autonomous agents to gracefully adapt their decision-making strategies to changes in the environment and in the behavior of the other agents is of paramount importance. As a learning technique that does not require a model of the environment and can be used online, RL is well suited for multi-player games, where each agent knows little about other agents and the environment. Using RL, the agents can learn new behaviors online, such that the performance of the individual or all players gradually improves.

In game theory, RL is considered to be a bounded-rational interpretation of how equilibrium may arise. As we witnessed in Section 2, RL methods allow the development of *Synchronous RL*-based algorithms leading to optimal feedback decision-makers (Kiumarsi *et al.*, 2017; Lewis *et al.*, 2012b; Vrabie *et al.*, 2013). In particular, we saw that PI offers an effective means to learn solutions to HJ equations. In general, every decision-maker tries to optimize its own objective defined in terms of the system dynamics. The actions by one decision-maker will influence the actions of the other decision-makers and will eventually lead to some kind of "negotiation."

The purpose of this section is to show how to solve multiplayer games *online* using *Synchronous RL*, namely by adaptive learning using data measured along the trajectories of the players. The full dynamics of the players need to be known for these online solution techniques. These methods implicitly solve the required game design equations without ever explicitly solving them. The algorithms are based on an actor/critic framework where the critic approximators are used to learn the optimal costs and the actor approximators are used to learn the optimal control policies. This section provides a truly dynamic framework for team decision making, since players or teams can change their objectives or optimality criteria on the fly, and the new strategies for all players, appropriate to the new situation, are then recomputed in real time. This online gaming approach also allows for time-varying team dynamics. It will be evident in networked systems that an agent affects the agents who are close enough to her. The team does better in terms of achieving its goals by using a distributed machine-learning approach that enables agents to exchange what they have learned.

3.2 Zero-Sum Games

The $\mathcal{H}_\infty$ control problem is a mini-max optimization problem, and hence a zero-sum (ZS) game where the controller is a minimizing player and the disturbance a maximizing one. Since the work of Zames and Francis (1983) in the early 1980s, $\mathcal{H}_\infty$ techniques have been used in control systems, for sensitivity reduction and disturbance rejection. This subsection is concerned with two-player ZS games that are related to the $\mathcal{H}_\infty$ control problem, as formulated by Başar and Olsder (1999), Başar and Bernhard (2008), and Schaft (1992). In essence, ZS two-player game theory and $\mathcal{H}_\infty$ solutions rely on solving the Hamilton–Jacobi–Isaacs (HJI) equations, a generalized version of the Hamilton–Jacobi–Bellman equations appearing in optimal control problems.

In this subsection, we provide methods for online gaming, that is for solution of two-player ZS infinite-horizon games online, through learning the saddle-point strategies. The dynamics may be nonlinear in CT and are assumed known. A novel neural-network adaptive control technique is given here that is based on RL techniques, whereby the control and

disturbance policies are tuned online using data generated along the system trajectories. Also, it is tuned a "critic" approximator structure whose function is to identify the value or outcome of the current control and disturbance policies. Based on this value estimate, the policies are continuously updated. All three loops are tuned simultaneously, or synchronously. This is a sort of indirect adaptive control algorithm, still, due to the direct form dependence of the policies on the learned value, it is affected online as "direct" optimal adaptive control.

3.2.1 Problem Formulation

Consider the nonlinear time-invariant system given by,

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t) + k(x(t))d(t), \quad t \geqslant 0, \tag{3.1}$$

where $x \in \mathbb{R}^n$ is a measurable state vector, $u(t) \in \mathbb{R}^m$ is the control input, $d(t) \in \mathbb{R}^q$ is the disturbance, $f\colon \mathbb{R}^n \to \mathbb{R}^n$ is the drift dynamics, and $g\colon \mathbb{R}^n \to \mathbb{R}^{n\times m}$ is the input dynamics. It is assumed that $f(0) = 0$ and $x \mapsto f(x) + g(x)u(x) + k(x)d(x)$ is locally Lipschitz and that the system is stabilizable.

Now we shall define the performance index as,

$$J(x(0), u, d) = \int_0^\infty (r(x, u, d))\mathrm{d}t \equiv \int_0^\infty (Q(x) + u^{\mathrm{T}}Ru - \gamma^2\|d\|^2)\mathrm{d}t, \quad \forall x(0), u, d,$$

where $Q(x)$ is a positive definite function, $R = R^{\mathrm{T}} \succ 0$ and $\gamma \geqslant \gamma^\star \geqslant 0$ with $\gamma^\star$ the smallest γ such that the system is stabilized. The value function with feedback control and disturbance policies can be defined as,

$$V(x(t), u, d) = \int_t^\infty (r(x, u, d))\mathrm{d}\tau \equiv \int_t^\infty (Q(x) + u^{\mathrm{T}}Ru - \gamma^2\|d\|^2)\mathrm{d}\tau, \quad \forall x, u, d.$$

Given that, the value is finite, a differential equivalent to this is the Bellman equation,

$$r(x, u, d) + \nabla V^{\mathrm{T}}(f(x) + g(x)u + k(x)d), \quad V(0) = 0,$$

and the Hamiltonian is given by,

$$H(\cdot) = r(x,u,d) + \nabla V^{\mathrm{T}}(f(x) + g(x)u + k(x)d), \quad \forall x,u,d.$$

Define the two-player ZS differential game as,

$$V^{\star}(x(0)) = \min_{u} \max_{d} J(x(0),u,d), \quad \forall x(0),$$

subject to (3.1). It is worth noting that u is the minimizing player, while d is the maximizing one.

This two-player optimal control problem has a unique solution if a game theoretic saddle point exists, that is, if the Nash condition holds,

$$\min_{u} \max_{d} J(x(0),u,d) = \max_{d} \min_{u} J(x(0),u,d), \quad \forall x(0).$$

In order to solve this ZS game we need to solve the following HJ–Isaacs (HJI) equation,

$$r(x,u^{\star},d^{\star}) + \nabla V^{\star\mathrm{T}}(f(x) + g(x)u^{\star} + k(x)d^{\star}) = 0, \quad \forall x,$$

given a solution $V^{\star}\colon \mathbb{R}^n \to \mathbb{R}_+$ to this equation and

$$u^{\star} = \arg\min_{u} H\left(x, \frac{\partial V^{\star}}{\partial x}, u, d^{\star}\right) = -\frac{1}{2}R^{-1}g(x)^{\mathrm{T}}\nabla V^{\star}, \quad \forall x,$$
$$d^{\star} = \arg\max_{d} H\left(x, \frac{\partial V^{\star}}{\partial x}, u^{\star}, d\right) = \frac{1}{2\gamma^2}k(x)^{\mathrm{T}}\nabla V^{\star}, \quad \forall x.$$

3.2.2 Approximate Solution

The structure used for our approximate solution is motivated by the Policy Iteration Algorithm that follows, where it terminates when two consecutive value functions do not differ (regarding a corresponding suitable norm error).

We should use a critic to approximate the cost and two actors to approximate the optimal control and the worst case disturbance. The critic is based on VFA. In what follows, we shall approximate the value $V(x)$ as well as its gradient. It is justified to assume there exist weights W_1 such that the value function $V(x)$ is approximated as,

$$V(x) = W_c^{\mathrm{T}}\phi_c(x) + \epsilon(x), \quad \forall x.$$

Algorithm 5: Policy Iteration for $\mathcal{H}_\infty$ Control (Zero-Sum Games)

1: **procedure**
2: Given admissible policies $u^{(0)}$
3: for $j = 0, 1, \ldots$ given u_j
4: for $i = 0, 1, \ldots$ set $d^0 = 0$ solve for the value $V_j^{(i)}(x)$ using Bellman's equation

$$Q(x)+\nabla V_j^{i\mathrm{T}}(x)(f(x)+g(x)u_j+k(x)d^i)+u_j^{\mathrm{T}}Ru_j-\gamma^2\left\|d^2\right\|^2=0, \quad V_j^i(0)=0,$$

$$d^{i+1}=\frac{1}{2\gamma^2}k(x)^{\mathrm{T}}\nabla V_j^i(x)$$

on convergence, set $V_{j+1}(x) = V_j^i(x)$.
5: Update the control policy u_{j+1} using

$$u_{j+1}=-\frac{1}{2}R^{-1}g(x)^{\mathrm{T}}\nabla V_{j+1}(x)$$

6: go to 3
7: **end procedure**

Then, $\phi_c(x)$: $\mathbb{R}^n \to \mathbb{R}^N$ is called the basis function vector, N the number of neurons in the hidden layer, and $\epsilon(x)$ the approximation error.

But since we do not know the optimal weights W_c, we shall use current critic weights as,

$$\hat{V}(x) = \hat{W}_c^{\mathrm{T}}\phi_c(x), \quad \forall x, \tag{3.2}$$

where $\hat{W}_c$ are the current estimated values of the ideal critic weights W_c. Now, the approximate HJI equation is given as,

$$\hat{H}(\cdot) := \hat{W}_c^{\mathrm{T}}\nabla\phi_c(f + g\hat{u} + k\hat{d}) + Q(x) + \hat{u}^{\mathrm{T}}R\hat{u} - \gamma^2\left\|d\right\|^2 = e_1,$$

where $e_1 \in \mathbb{R}^n$ is the approximation (residual) error after using current critic weights and $\hat{u}, \hat{d}$ are given by,

$$\hat{u}(x) = -\frac{1}{2}R^{-1}g(x)^{\mathrm{T}}\nabla\phi_c(x)^{\mathrm{T}}\hat{W}_u, \tag{3.3}$$

and

$$\hat{d}(x) = \frac{1}{2\gamma^2}k(x)^{\mathrm{T}}\nabla\phi_c(x)^{\mathrm{T}}\hat{W}_d, \tag{3.4}$$

respectively.

Now it is desired to select $\hat{W}_c$ such that the squared residual error $E_1 = \frac{1}{2}e_1^{\mathrm{T}}e_1$ is minimized. Hence by using a normalized gradient descent, one has,

$$\begin{aligned}\dot{\hat{W}}_c &= -a_1\frac{\partial E_1}{\partial \hat{W}_1}\\ &= -\alpha\frac{\nabla\phi_c(f+g\hat{u}+k\hat{d})}{((\nabla\phi_c(f+g\hat{u}+k\hat{d}))^{\mathrm{T}}\nabla\phi_c(f+g\hat{u}+k\hat{d})+1)^2}\\ &\quad\times((\nabla\phi_c(f+g\hat{u}+k\hat{d}))^{\mathrm{T}}\hat{W}_c+Q(x)+\hat{u}^{\mathrm{T}}R\hat{u}-\gamma^2\|\hat{d}\|^2),\end{aligned}\tag{3.5}$$

where $\alpha \in \mathbb{R}_+$ is a tuning gain that determines the speed of convergence.

The weights for the actor $\hat{W}_u$ need to be picked in order to guarantee closed-loop stability. Hence one has,

$$\begin{aligned}\dot{\hat{W}}_u = -\alpha_u\Bigg(&(F_2\hat{W}_u-\mathbf{1}^{\mathrm{T}}(\nabla\phi_c(f+g\hat{u}+k\hat{d}))^{\mathrm{T}}\hat{W}_c)\\ &-\frac{1}{4}(\nabla\phi_c(x)g(x)R^{-1}g(x)^{\mathrm{T}}\nabla\phi_c^{\mathrm{T}}(x))\hat{W}_u\\ &\times\left(\frac{\nabla\phi_c(f+g\hat{u}+k\hat{d})}{((\nabla\phi_c(f+g\hat{u}+k\hat{d}))^{\mathrm{T}}\nabla\phi_c(f+g\hat{u}+k\hat{d})+1)^2}\right)^{\mathrm{T}}\hat{W}_c\Bigg),\end{aligned}\tag{3.6}$$

where $\alpha_u \in \mathbb{R}_+$ is a tuning gain that determines the speed of convergence and $F_2 \succ 0$ is a user defined positive definite matrix picked appropriately for stability. Similarly the weights for the disturbance $\hat{W}_d$ are given by,

$$\begin{aligned}\dot{\hat{W}}_d = -\alpha_d\Bigg(&(F_4\hat{W}_d-\mathbf{1}^{\mathrm{T}}(\nabla\phi_c(f+g\hat{u}+k\hat{d}))^{\mathrm{T}}\hat{W}_c)\\ &+\frac{1}{4\gamma^2}(\nabla\phi_c(x)k(x)k(x)^{\mathrm{T}}\nabla\phi_c^{\mathrm{T}}(x))\hat{W}_d\\ &\times\left(\frac{\nabla\phi_c(f+g\hat{u}+k\hat{d})}{((\nabla\phi_c(f+g\hat{u}+k\hat{d}))^{\mathrm{T}}\nabla\phi_c(f+g\hat{u}+k\hat{d})+1)^2}\right)^{\mathrm{T}}\hat{W}_c\Bigg),\end{aligned}\tag{3.7}$$

where $\alpha_d \in \mathbb{R}_+$ is a tuning gain that determines the speed of convergence and $F_3, F_4 \succ 0$ are user defined positive definite matrices picked

appropriately for establishing stability by using the Lyapunov's direct method.

The main theorem is now given and guarantees convergence to the ZS game Nash equilibrium solution while also guaranteeing closed-loop stability. The practical notion of UUB is used.

Theorem 3.1 (Vamvoudakis and Lewis, 2012, Thm. 2)**.** Let the dynamics be given by (3.1), the critic NN be given by (3.2), the control input be given by actor NN (3.3) and the disturbance input be given by disturbance NN (3.4). Let the tuning for the critic NN be provided by (3.5), let the actor NN be tuned as (3.6), and the disturbance NN be tuned as (3.7). Let $Q(x)$ a positive definite function. Suppose that $\frac{\nabla\phi_c(f+g\hat{u}+k\hat{d})}{((\nabla\phi_c(f+g\hat{u}+k\hat{d}))^{\mathrm{T}}\nabla\phi_c(f+g\hat{u}+k\hat{d})+1)}$ is PE. Let the tuning parameters F_3, F_4 in (3.6), and (3.7) be selected appropriately. Then there exists an N_0 such that, for the number of hidden-layer units $N > N_0$ the closed-loop signals are UUB. ■

Remark 3.1. The theorem shows that PE is needed for proper identification of the value function by the critic NN, and that nonstandard tuning algorithms are required for the actor and the disturbance NN to guarantee stability. □

A pseudocode that describes the proposed adaptive $\mathcal{H}_\infty$ control algorithm has the following form,

Algorithm 6: Adaptive $\mathcal{H}_\infty$ Control Algorithm

1: **procedure**
2: Start with initial state $x(0)$ and, random initial weights
3: $\hat{W}_u(0), \hat{W}_d(0), \hat{W}_c(0)$.
4: Propagate $t, x(t)$.
5: Propagate $\hat{W}_c(t), \hat{W}_u(t), \hat{W}_d(t)$ ▷ $\dot{\hat{W}}_c$ as in (3.5), $\dot{\hat{W}}_u$ as in (3.6) and $\dot{\hat{W}}_d$ as in (3.7).
6: Compute (3.2), (3.3) and (3.4).
7: **end procedure**

3.2.3 Further Reading

The interested reader is directed to Vamvoudakis and Lewis (2012), Vamvoudakis *et al.* (2011), Vrabie *et al.* (2013) and Vamvoudakis *et al.* (2017a) for detailed proofs, theorems and simulation examples. The authors in Luo *et al.* (2014), Modares *et al.* (2015), Zhang *et al.* (2014b), Wu and Luo (2012), Liu *et al.* (2013), Wei *et al.* (2015), and Luo *et al.* (2014) base their works on the results shown in this subsection to develop adaptive learning $\mathcal{H}_\infty$ control schemes.

3.3 Non-Zero-Sum Games

Multi-player cooperative games rely on solving coupled HJ equations, which in the linear quadratic case reduce to the coupled AREs (Başar and Olsder, 1999). Solution methods are generally offline and generate fixed control policies that are then implemented in online controllers. These coupled equations are difficult to solve.

This subsection shows how to solve multi-player non-zero-sum (NZS) games online using novel adaptive control structures based on RL. For the most part, interest in the control systems community has been in the (non-cooperative) ZS games, which provide the solution of the $\mathcal{H}_\infty$ robust control problem (Başar and Olsder, 1999). However, dynamic team games may have some cooperative objectives and some selfish objectives among the players. This cooperative/non-cooperative balance is captured in the NZS games, as detailed herein.

In particular, in this subsection we are interested in feedback policies with full state information. We provide methods for online gaming, that is for solution of infinite-horizon NZS games online, through learning the Nash equilibrium. The dynamics are non-linear in CT and are assumed known. A novel adaptive control structure is given that is based on RL techniques, whereby each player's control policies are tuned online using data generated along the system trajectories. Also tuned by each player are "critic" approximator structures whose function is to identify the values of the current control policies for each player. Based on these value estimates, the players' policies are continuously updated. This is a sort of indirect adaptive control algorithm, still, due to the simple form

dependence of the control policies on the learned value, it is affected online as direct ("optimal") adaptive control.

Furthermore, this subsection proposes an algorithm for non-linear CT systems with known dynamics to solve the N-player NZS game problem where each player wants to optimize his own performance index (Başar and Olsder, 1999; Hespanha, 2017). The number of parametric approximator structures that are used is 2N. Each player maintains a critic approximator NN to learn his optimal value and a control actor NN to learn his optimal control policy. Parameter update laws are given to tune the N-critic and N-actor NNs simultaneously online to converge to the solution to the coupled HJ equations while also guaranteeing closed-loop stability.

3.3.1 Problem Formulation

Consider the N-player nonlinear time-invariant differential game,

$$\dot{x}(t) = f(x(t)) + \sum_{j=1}^{N} g_j(x(t))u_j(t), \quad t \geqslant 0, \tag{3.8}$$

where $x \in \mathbb{R}^n$ is a measurable state vector, $u_j(t) \in \mathbb{R}^{m_j}$ are the control inputs, $f\colon \mathbb{R}^n \to \mathbb{R}^n$ is the drift dynamics, $g_j\colon \mathbb{R}^n \to \mathbb{R}^{n\times m_j}$ is the input dynamics. It is assumed that $f(0) = 0$ and $x \mapsto f(x) + \sum_{j=1}^{N} g_j(x)u_j(x)$ is locally Lipschitz with $u_j(x)$ the feedback control, and that the system is stabilizable.

The cost functionals associated with each player are given by,

$$\begin{aligned} J_i(x(0); u_1, u_2, \ldots, u_N) &= \int_0^\infty (r_i(x, u_1, \ldots, u_N))\mathrm{d}t \\ &\equiv \int_0^\infty \left(Q_i(x) + \sum_{j=1}^{N} u_j^{\mathrm{T}} R_{ij} u_j \right) \mathrm{d}t, \quad \forall i \in \mathcal{N}, \end{aligned}$$

where $Q_i(x)$ is a generally nonlinear positive definite function and $R_{ii} \succ 0, \forall i \in \mathcal{N}$, $R_{ij} \succeq 0$, $\forall j \neq i \in \mathcal{N}$ are symmetric matrices and $\mathcal{N} := \{1, 2, \ldots, N\}$.

The value can be defined as,

$$V_i(x, u_1, u_2, \ldots, u_N) = \int_t^\infty (r_i(x, u_1, \ldots, u_N))\mathrm{d}\tau, \quad \forall i \in \mathcal{N},\ x,\ u_1, u_2, \ldots, u_N.$$

Now we are ready to define the N-player game as,

$$V_i^\star(x, u_1, u_2, \ldots, u_N) = \min_{u_i} \int_t^\infty (r_i(x, u_1, \ldots, u_N))\mathrm{d}\tau, \quad \forall i \in \mathcal{N}.$$

By assuming that all the players have the same hierarchical level, we focus on the so-called Nash equilibrium that is given by the following definition adopted from Başar and Olsder (1999), Hespanha (2017).

Definition 3.1. An N-tuple of strategies $\{u_1^\star, u_2^\star, \ldots, u_N^\star\}$ is said to constitute a Nash equilibrium solution for an N-player finite game in extensive form if the following N inequalities are satisfied for all $u_i^\star,\ i \in \mathcal{N}$,

$$J_i^\star \equiv J_i(u_1^\star, u_2^\star, \ldots, u_i^\star, \ldots, u_N^\star) \leqslant J_i(u_1^\star, u_2^\star, \ldots, u_i, \ldots, u_N^\star), \quad \forall i \in \mathcal{N}.$$

The N-tuple of quantities $\{J_1^\star, J_2^\star, \ldots, J_N^\star\}$ is known as a Nash equilibrium outcome of the N-player game. □

Differential equivalents to each value function are given by the following Bellman equations,

$$r_i(x, u_1, \ldots, u_N) + \nabla V_i^{\mathrm{T}}(x)\left(f(x) + \sum_{j=1}^N g_j(x)u_j\right) = 0, \quad V_i(0) = 0, \forall i \in \mathcal{N}.$$

We shall define the Hamiltonian functions as,

$$H_i(\cdot) = r_i(x, u_1, \ldots, u_N) + \nabla V_i^{\mathrm{T}}(x)\left(f(x) + \sum_{j=1}^N g_j(x)u_j\right), \quad \forall i \in \mathcal{N}.$$

According to the stationarity conditions, associated feedback control policies are given by,

$$u_i = \arg\min_{u_i} H_i(\cdot) = -\frac{1}{2}R_{ii}^{-1} g_i^{\mathrm{T}}(x)\nabla V_i(x), \quad \forall x, \forall i \in \mathcal{N}.$$

After substituting the feedback control policies into the Hamiltonian one has the coupled HJ equations,

$$0 = Q_i(x) + \frac{1}{4}\sum_{j=1}^{N} \nabla V_j^{\mathrm{T}}(x) g_j(x) R_{jj}^{-1^{\mathrm{T}}} R_{ij} R_{jj}^{-1} g_j^{\mathrm{T}}(x) \nabla V_j(x) + \nabla V_i^{\mathrm{T}}(x)\left(f(x) - \frac{1}{2}\sum_{j=1}^{N} g_j(x) R_{jj}^{-1} g_j^{\mathrm{T}}(x) \nabla V_j(x)\right), \quad \forall i \in \mathcal{N}.$$

In linear systems of the form $\dot{x} = Ax + \sum_{j=1}^{N} B_j u_j$ with $Q_i \geq 0$, the N coupled HJ equations become the N coupled generalized AREs,

$$0 = P_i\left(A - \frac{1}{2}\sum_{j=1}^{N} B_i R_{ii}^{-1} B_i^{\mathrm{T}} P_i\right) + \left(A - \frac{1}{2}\sum_{j=1}^{N} B_i R_{ii}^{-1} B_i^{\mathrm{T}} P_i\right)^{\mathrm{T}} P_i + Q_i + \frac{1}{4}\sum_{j=1}^{N} P_j B_j R_{jj}^{-1^{\mathrm{T}}} R_{ij} R_{jj}^{-1} B_j^{\mathrm{T}} P_j, \quad \forall i \in \mathcal{N},$$

where the pair $(A - \frac{1}{2}\sum_{j=1, j\neq i}^{N} B_j R_{jj}^{-1} B_j^{\mathrm{T}} P_j,\ B_i)$ is stabilizable and the pair $(A - \frac{1}{2}\sum_{j=1, j\neq i}^{N} B_j R_{jj}^{-1} B_j^{\mathrm{T}} P_j, Q_i + \frac{1}{4}\sum_{j=1, j\neq i}^{N} P_j B_j R_{jj}^{-1^{\mathrm{T}}} R_{ij} R_{jj}^{-1} \cdot B_j^{\mathrm{T}} P_j)$ is detectable, then the stationary feedback policies form a Nash equilibrium solution for the linear quadratic N-player differential game among feedback policies with full state information. Furthermore, the resulting system dynamics, have an asymptotically stable equilibrium point.

3.3.2 Approximate Solution

The structure used for our approximate solution is motivated by the Policy Iteration Algorithm that follows, where $\epsilon_{\mathrm{iac}} \in \mathbb{R}_+,\ \forall i \in \mathcal{N}$, is a small number used to terminate the algorithm when two consecutive value functions differ by less than $\epsilon_{\mathrm{iac}}, \forall i \in \mathcal{N}$.

We should use N critics to approximate the costs and N actors to approximate the optimal controls. This work uses nonlinear approximator structures for VFA to solve the coupled HJ equations. Therefore, assume there exist constant weights W_i and such that the value functions V_i are approximated on a compact set Ω as,

$$V_i(x) = W_i^{\mathrm{T}} \phi_i(x) + \epsilon_i(x), \quad \forall x,\ \forall i \in \mathcal{N},$$

Algorithm 7: Policy Iteration for Non-Zero-Sum Games

procedure
  Given N-tuple of admissible policies $\mu_i^k(0),\ \forall i \in \mathcal{N}$ and $k = 1$.
  while $\|V_i^{\mu^{(k)}} - V_i^{\mu^{(k-1)}}\| \geqslant \epsilon_{\text{iac}},\ \forall i \in \mathcal{N}$ **do**
    Solve for the N-tuple of costs $V_i^k(x)$ using the coupled Bellman equations

$$Q_i(x) + \nabla V_i^{k\,\mathrm{T}} \left(f(x) + \sum_{i=1}^{N} g_i(x)\mu_i^k \right) + \mu_i^{k\,\mathrm{T}} R_{ii} \mu_i^k + \sum_{j=1}^{N} \mu_j^{k\,\mathrm{T}} R_{ij} \mu_j^k = 0,$$

$$V^{\mu_i^k}(0) = 0.$$

    Update the N-tuple of control policies $\mu_i^{k+1},\ \forall i \in \mathcal{N}$ using

$$\mu_i^{k+1} = -\frac{1}{2} R_{ii}^{-1} g_i^{\mathrm{T}}(x) \nabla V_i^k.$$

    $k \coloneqq k + 1$.
  end while
end procedure

where $\phi_i(x)$: $\mathbb{R}^n \to \mathbb{R}^{K_i},\ \forall i \in \mathcal{N}$ are the activation function basis set vectors, $K_i,\ \forall i \in \mathcal{N}$ is the number of neurons in the hidden layer and $\epsilon_i(x)$ the approximation error. From the approximation literature, the basis functions can be selected as sigmoids, tanh, polynomials, etc.

Assuming current weight estimates $\hat{W}_{ic}, \forall i \in \mathcal{N}$, the outputs of the critics are given by,

$$\hat{V}_i(x) = \hat{W}_{ic}^{\mathrm{T}} \phi_i(x), \quad \forall i \in \mathcal{N}. \tag{3.9}$$

Using the current weight estimates, the approximate Lyapunov-like equations are given by,

$$\hat{H}_i(\cdot) = r_i(x, \hat{u}_1, \ldots, \hat{u}_N) + \nabla V_i^{\mathrm{T}}(x) \left(f(x) + \sum_{j=1}^{N} g_j(x)\hat{u}_j \right) = e_i,$$

$$\forall i \in \mathcal{N},$$

where $e_i \in \mathbb{R}, \forall i \in \mathcal{N}$ are the residual errors due to approximation and

$$\hat{u}_i = -\frac{1}{2} R_{ii}^{-1} g_i^{\mathrm{T}}(x) \nabla \phi_i^{\mathrm{T}}(x) \hat{W}_{iu}, \quad \forall i \in \mathcal{N}, \tag{3.10}$$

where $\hat{W}_{iu}$ denote the current estimated values of the ideal weights W_i.

Now, it is desired to pick the tuning for the critic in order to minimize the square residual errors e_i in order to guarantee that $\hat{W}_{ic} \to W_i$, $\forall i \in \mathcal{N}$. Hence the tuning law for the critic is given as,

$$\begin{aligned}\dot{W}_{ic} = -\alpha_i \frac{\nabla\phi_i(f + \sum_{j=1}^N g_j\hat{u}_j)}{((\nabla\phi_i(f + \sum_{j=1}^N g_j\hat{u}_j))^{\mathrm{T}}\nabla\phi_i(f + \sum_{j=1}^N g_j\hat{u}_j) + 1)^2} \\ \times \left(\left(\nabla\phi_i\left(f + \sum_{j=1}^N g_j\hat{u}_j\right)\right)^{\mathrm{T}}\hat{W}_{ic}\right. \\ \left. + Q_i(x) + \hat{u}_i^{\mathrm{T}}R_{ii}\hat{u}_i + \sum_{j=1}^N \hat{u}_j^{\mathrm{T}}R_{ij}\hat{u}\right), \quad \forall i \in \mathcal{N}, \end{aligned} \tag{3.11}$$

where $\alpha_i \in \mathbb{R}_+$ is a tuning gain that determines the speed of convergence, and for the actor is given as,

$$\begin{aligned}\dot{\hat{W}}_{iu} = -\alpha_{iu}\left(\left(F_i\hat{W}_{iu} - \mathbf{1}^{\mathrm{T}}\left(\nabla\phi_i\left(f + \sum_{j=1}^N g_j\hat{u}_j\right)\right)^{\mathrm{T}}\hat{W}_{ic}\right)\right. \\ - \frac{1}{4}\sum_{j=1}^N (\nabla\phi_i(x)g_i(x)R_{ii}^{-1^{\mathrm{T}}}R_{ij}R_{ii}^{-1}g_j(x)^{\mathrm{T}}\nabla\phi_i^{\mathrm{T}}(x))\hat{W}_{iu} \\ \left. \times \left(\frac{\nabla\phi_i(f + \sum_{j=1}^N g_j\hat{u}_j)}{((\nabla\phi_i(f + \sum_{j=1}^N g_j\hat{u}_j))^{\mathrm{T}}\nabla\phi_i(f + \sum_{j=1}^N g_j\hat{u}_j) + 1)^2}\right)^{\mathrm{T}}\hat{W}_{jc}\right), \\ \forall i \in \mathcal{N}, \end{aligned} \tag{3.12}$$

where $\alpha_{iu} \in \mathbb{R}_+$ is a tuning gain that determines the speed of convergence, and $F_i > 0$ are matrices that guarantee stability of the system.

Theorem 3.2 (Vamvoudakis and Lewis, 2011, Thm. 2)**.** Consider the dynamics be given by (3.8). Let the critic-networks be given by (3.9) and the control inputs be given by action networks (3.10). Let tuning for the critic NNs be provided by (3.11) and for the actor NNs be given by (3.12) where $F_i > 0$ are tuning parameters. Assume positive definite functions $Q_1(x)$ and $Q_2(x)$. Suppose that $\frac{\nabla\phi_i(f+\sum_{j=1}^N g_j\hat{u}_j)}{((\nabla\phi_i(f+\sum_{j=1}^N g_j\hat{u}_j))^{\mathrm{T}}\nabla\phi_i(f+\sum_{j=1}^N g_j\hat{u}_j)+1)}$, $\forall i \in \mathcal{N}$, are persistently exciting. Let the tuning parameter for the critic selected sufficiently larger than the actor. Then there exists a K_0 such that, for the number of hidden-layer units $K > K_0$ the closed-loop signals are UUB. ■

A pseudocode that describes the proposed adaptive NZS game algorithm has the following form,

Algorithm 8: Adaptive Non-Zero-Sum Game Algorithm

1: **procedure**
2: Start with initial state $x(0)$ and random initial weights $\hat{W}_{iu}(0), \hat{W}_{ic}(0), \forall i \in \mathcal{N}$.
3: Propagate $t, x(t)$.
4: Propagate $\hat{W}_{iu}(t), \hat{W}_{ic}(t), \forall i \in \mathcal{N}$ $\triangleright$ $\dot{\hat{W}}_{ic}$ as in (3.11), and $\dot{\hat{W}}_{iu}$ as in (3.12).
5: Compute (3.9) and (3.10).
6: **end procedure**

3.3.3 Further Reading

The interested reader is directed to Vamvoudakis and Lewis (2011), Vrabie *et al.* (2013), and Vamvoudakis *et al.* (2017a) for detailed proofs, theorems and simulation examples. Still, the authors of this article (Yang *et al.*, 2020a) develop a novel actor-critic-barrier structure for the multiplayer safety-critical systems. Finally, we refer the reader to Liu *et al.* (2014) and Zhang *et al.* (2012) wherein online synchronous approximate optimal learning algorithms for multi-player non-zero-sum games are developed. An overview of the state of the art is presented in the magazine article (Vamvoudakis *et al.*, 2017a).

3.4 Stackelberg Games

This subsection considers non-cooperative non-zero-sum games, called Stackelberg games, named after Heinrich von Stackelberg in recognition of his pioneering work (Von Stackelberg, 2010) Stackelberg games provide a framework to analyze and design hierarchical interactions among self-interested players, where the objectives are no longer independent. In Stackelberg games, one needs to differentiate between open-loop, closed-loop, and feedback strategies. Specifically, open loop refers to a decision by each player based on the initial condition, and closed loop refers to the ability of the players to change their decisions based on current information. Feedback strategies correspond to the

ability of the leader to further change her strategy in reaction to the follower's closed-loop strategy. In this subsection, we consider open-loop Stackelberg strategies.

These hierarchical games consist of two groups of players: leaders who have complete information about other players' strategies and followers who lack such information. Each leader selects her action by solving a 2-level optimization problem that seeks to minimize her utility subject to the followers' actions as estimated by that leader. The followers then select their actions, according to their observations from the aggregate impact of other users. Applications of Stackelberg strategies include military intelligence, social behaviors, marketing, network communications, and multilevel optimization for power systems (He *et al.*, 2007). Two applications of Stackelberg games are the ARMOR program at LAX airport (Pita *et al.*, 2008), where police are able to set up checkpoints on roads leading to particular terminals and the IRIS program used by the US Federal Air Marshals (Tsai *et al.*, 2009), where armed Marshals are assigned to commercial flights to defeat terrorist attacks. Generally, the information structure in a Stackelberg game is the set of all available information for the players to make their decisions. When an open-loop information structure is considered, no measurement of the state of the system is available, and the players are committed to follow a predetermined strategy based on their knowledge of the initial state, the system's model, and the cost functional to be minimized. Two possible approaches describe interactions in a Stackelberg game. In the first approach, the follower asks the leader to choose a reaction to specify the leader's best response to the follower's optimal behavior during the game. In the second approach, one may assume that the leader announces the policy to the follower.

Finally, on the basis of the above, it turns out that the Stackelberg games constitute class of NZS games where the strategies are announced sequentially is called. Thus, Stackelberg or hierarchical equilibria refer to non-cooperative solutions, where one or more of the players act as leaders. Because the players have different hierarchical levels, it is natural to focus on Stackelberg equilibrium, which is given by the following definition.

Definition 3.2. The leader knows the cost function mapping of the follower, but the follower may not know the cost function mapping of the leader. The follower knows the control strategy of the leader and the follower takes this into account when computing its strategy within the set of those strategies that minimize $\mathcal{J}_1$ according to definition,

$$\mathcal{J}_1(u_1^\star, u_2) \leqslant \mathcal{J}_1(u_1, u_2), \quad \text{for a fixed policy } u_2.$$

If there exists a pair $u_1^\star$, $u_2^\star$ on the reaction set of Player 1 such that,

$$\mathcal{J}_2(u_1^\star, u_2^\star) \leqslant \mathcal{J}_2(u_1^\star, u_2),$$

for any pair u_1, u_2 on the reaction set of Player 1, the pair $u_1^\star$, $u_2^\star$ is defined as a Stackelberg equilibrium strategy with Player 2 as the leader. □

3.4.1 Open-Loop Stackelberg Learning Solution for Hierarchical Control Problems

In what follows, we shall present a novel framework based on adaptive learning techniques to solve the CT open-loop Stackelberg games. The method yields approximations of the game value and convergence of the policies to the open-loop Stackelberg-equilibrium solution while also guaranteeing asymptotic stability of the equilibrium point of the closed-loop system. It is implemented as a separate actor/critic parametric network approximator structure for every player and involves simultaneous CT adaptation. To introduce and implement the hierarchical structure to the coupled optimization problem, we adjoin to the leader the controller dynamics of the follower. A PE condition guarantees convergence of both critics to the actual game values that eventually solve the hierarchical optimization problem.

3.4.2 Problem Formulation

Consider the two-player differential game,

$$\dot{x}(t) = Ax(t) + B_1u_1(t) + B_2u_2(t), \quad x(0) \coloneqq x_0,\ t \geqslant 0, \tag{3.13}$$

where $x(t) \in \mathbb{R}^n$ is the state available for feedback; $u_1 \in \mathbb{R}^m$, $u_2 \in \mathbb{R}^q$ are the control inputs (i.e., players); and A, B_1, and B_2 are plant and

input matrices of appropriate dimensions. The control inputs or players have different hierarchical levels, i.e., u_1 is the follower, and u_2 is the leader.

Each player has the following cost functionals:

$$\begin{aligned} J_1 &= \frac{1}{2}x_f^{\mathrm{T}}P_{1f}x_f + \frac{1}{2}\int_0^{t_f}(x^{\mathrm{T}}Q_1x + u_1^{\mathrm{T}}R_{11}u_1 + u_2^{\mathrm{T}}R_{12}u_2)\mathrm{d}t \\ &\equiv \frac{1}{2}x_f^{\mathrm{T}}P_{1f}x_f + \frac{1}{2}\int_0^{t_f} r_1(x,u_1,u_2)\mathrm{d}t, \\ J_2 &= \frac{1}{2}x_f^{\mathrm{T}}P_{2f}x_f + \frac{1}{2}\int_0^{t_f}(x^{\mathrm{T}}Q_2x + u_1^{\mathrm{T}}R_{21}u_1 + u_2^{\mathrm{T}}R_{22}u_2)\mathrm{d}t \\ &\equiv \frac{1}{2}x_f^{\mathrm{T}}P_{2f}x_f + \frac{1}{2}\int_0^{t_f} r_2(x,u_1,u_2)\mathrm{d}t, \end{aligned}$$

where $t_f \in \mathbb{R}_+$, a terminal time that can be fixed or variable, P_{1f}, $P_{2f} \in \mathbb{R}^{n\times n} \succ 0$, $Q_i \succeq 0, R_{ii} \succ 0$, and $R_{ij} \succeq 0\ \forall i,\ j = 1,2, i \neq j$ are symmetric matrices. To solve such a problem, we seek optimal controls among the set of control policies with complete state information. However, because the players have a different hierarchical level, we focus on an open-loop Stackelberg equilibrium that is given by the following definition adopted from the works of Simaan and Cruz (1973a,b).

Note that we focus on an open-loop information structure where the players in the Stackelberg game are committed to following a predetermined strategy. We are thus interested in finding the following value:

$$J_1^\star = \min_{u_1}\left(\frac{1}{2}x_f^{\mathrm{T}}P_{1f}x_f + \frac{1}{2}\int_t^{t_f} r_1\left(x,u_1,u_2\right)\mathrm{d}\tau\right), \quad t \geqslant 0, \tag{3.14}$$

for all policies u_2 as functions of the state with the following associated Hamiltonian for the follower (Abou-Kandil and Bertrand, 1985; Fleming and McEneaney, 2000; Johnson *et al.*, 2010):

$$H_1\left(x,\lambda_1,u_1,u_2\right) = \frac{1}{2}r_1\left(x,u_1,u_2\right) + \lambda_1^{\mathrm{T}}\left(Ax + B_1u_1 + B_2u_2\right), \tag{3.15}$$

where the necessary conditions (see the work of Chen and Cruz, 1972) for optimality are (3.13), and

$$\frac{\partial H_1}{\partial u_1} = 0 \Rightarrow u_1^\star = -R_{11}^{-1} B_1^{\mathrm{T}} \lambda_1, \tag{3.16}$$

$$\dot{\lambda}_1 = -\left(\frac{\partial H_1}{\partial x}\right)^{\mathrm{T}} = -A^{\mathrm{T}}\lambda_1 - Q_1 x, \lambda_1(t_f) = P_{1f} x_f. \tag{3.17}$$

For the leader, we are interested in computing the following optimal value:

$$J_2^\star = \min_{u_2}\left(\frac{1}{2} x_f^{\mathrm{T}} P_{1f} x_f + \frac{1}{2}\int_t^{t_f} r_2\left(x, u_1^\star, u_2\right) \mathrm{d}\tau\right),$$

with constraints (3.13) and (3.17). This extra constraint shall quantify how good the follower does after choosing (3.14).

3.4.3 Stackelberg Game and Leader-Follower Riccati Equations

The Hamiltonian associated with the leader with constraints (3.13) and (3.17) is,

$$H_2 = \frac{1}{2} r_2\left(x, u_1^\star, u_2\right) + \lambda_2^{\mathrm{T}}\left(Ax + B_1 u_1^\star + B_2 u_2\right) + y^{\mathrm{T}}\dot{\lambda}_1, \tag{3.18}$$

with $u_1^\star$ given by (3.16) and y, a Lagrangian multiplier to adjoin constraint (3.17). The necessary conditions for optimality of the leader are,

$$\frac{\partial H_2}{\partial u_2} = 0 \Rightarrow u_2^\star = -R_{22}^{-1} B_2^{\mathrm{T}} \lambda_2, \tag{3.19}$$

$$\dot{\lambda}_2 = -\left(\frac{\partial H_2}{\partial x}\right)^{\mathrm{T}} = -A^{\mathrm{T}}\lambda_2 - Q_2 x + Q_1 y,$$

$$\lambda_2(t_f) = P_{2f} x_f - P_{1f} y(t_f),$$

and

$$\dot{y} = -\left(\frac{\partial H_2}{\partial \lambda_1}\right) = Ay - B_1 {R_{11}^{-1}}^{\mathrm{T}} R_{21} R_{11}^{-1} B_1^{\mathrm{T}} \lambda_1 + B_1 {R_{11}^{-1}}^{\mathrm{T}} B_1^{\mathrm{T}} \lambda_2,\ y(0) = 0. \tag{3.20}$$

Remark 3.2. Note that y of Equation (3.20) will adjoin the constraint (3.17) to the optimization problem of the leader. This will actually solve the difference in hierarchies of the two players in the game. □

Since this subsection shall consider the linear quadratic case, the costate variables shall have the form (Abou-Kandil and Bertrand, 1985; Johnson *et al.*, 2015),

$$\lambda_1 = P_1(t)x, \quad \lambda_2 = P_2(t)x, \quad y = P_3(t)x, \quad \forall x,$$

where $P_1(t),\ P_2(t),\ P_3(t) \in \mathbb{R}^{n\times n} \succ 0$ are time varying and block diagonal matrices.

Remark 3.3. Note that $y = P_3x$ describes a linear transformation $T\colon \mathbb{R}^n \to \mathbb{R}^n$, i.e., $y = T(x)$. □

We are ready to state the following lemma adopted from previous studies (Freiling *et al.*, 1996, 2003; Simaan and Cruz, 1973a).

Lemma 3.3 (Vamvoudakis *et al.*, 2019, Lemma 1)**.** Assume that $x_0 \neq 0$, the matrix B_2 is full rank, the pair (Q_1, A) is observable, and at least one of the pairs (A, B_1) and (A, B_2) is controllable. Let $R_{11} \succ 0$, $R_{21} \succeq 0, R_{21} \succeq 0, Q_1 \succeq 0$, and $Q_2 \succeq 0$ and the linear open-loop control inputs issued from an open-loop Stackelberg strategy be given by Equation (3.16) and P_1, P_2, and P_3 satisfy the coupled differential Riccati equations $\forall t \geqslant 0$,

$$\begin{aligned}
\dot{P}_1 &= -P_1A - A^{\mathrm{T}}P_1 + P_1B_1R_{11}^{-1}B_1^{\mathrm{T}}P_1 + P_1B_2R_{22}^{-1}B_2^{\mathrm{T}}P_2 - Q_1,\\
\dot{P}_2 &= -P_2A - A^{\mathrm{T}}P_2 + P_2B_1R_{11}^{-1}B_1^{\mathrm{T}}P_1 + P_2B_2R_{22}^{-1}B_2^{\mathrm{T}}P_2 - Q_2\\
&\quad + Q_1P_3,\\
\dot{P}_3 &= -P_3A + AP_3 + P_3B_1R_{11}^{-1}B_1^{\mathrm{T}}P_1 + P_3B_2R_{22}^{-1}B_2^{\mathrm{T}}P_2\\
&\quad - B_1R_{11}^{-1^{\mathrm{T}}}R_{21}R_{11}^{-1}B_1^{\mathrm{T}}P_1 + B_1R_{11}^{-1^{\mathrm{T}}}B_1^{\mathrm{T}}P_2,
\end{aligned}$$

and the closed-loop dynamical equation is,

$$\dot{x} = (A - B_1R_{11}^{-1}B_1^{\mathrm{T}}P_1 - B_2R_{22}^{-1}B_2^{\mathrm{T}}P_2)x, \quad t \geqslant 0.$$

Finally, given that the coupled Riccati equations have a unique solution

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix},$$

satisfying the boundary conditions $P_1(0) = 0, P_2(t_f) = P_{2f}, P_3(t_f) = P_{3f} - P_{2f}P_1(t_f)$, and $P_{3f} = 0$ with t_f a sufficient large horizon, then Equations (3.16) and (3.19) form a Stackelberg equilibrium. ■

Remark 3.4. The existence of unique Stackelberg equilibria was shown to be tied to the existence of solutions to certain non-symmetric Riccati equations, which are difficult to solve. In the work of Bagchi and Başar (1981) and Freiling *et al.* (2003), a connection between solutions of a standard ARE and a non-symmetric ARE were given. In a similar manner, sufficient conditions for existence of a unique open-loop Stackelberg equilibrium by constructing appropriate potential functions was given in the work of Freiling *et al.* (2001). □

Using the variation of parameters formula, we have,

$$x(t) = \varphi(t, t_0)\, x_0 + \int_{t_0}^{T} \varphi(t,\tau) B_1(\tau) u_1(\tau) \mathrm{d}\tau + \int_{t_0}^{T} \varphi(t,\tau) B_2(\tau) u_2(\tau) \mathrm{d}\tau,$$

where $\varphi(t, t_0) = A\varphi(t, t_0)$ and $\varphi(t, t) = I$.

The following algorithmic framework will be the basis for our approach.

3.4.4 Open-Loop Stackelberg Games

We need to define the following potential functions with gradients that provide the λ_1 and λ_2, respectively,

$$F_1^\star(x) = x^{\mathrm{T}}\lambda_1 = \mathrm{vec}(P_1)^{\mathrm{T}}\,\phi(x), \quad \forall x,$$

and

$$F_2^\star(x) = x^{\mathrm{T}}\lambda_2 = \mathrm{vec}(P_2)^{\mathrm{T}}\,\phi(x), \quad \forall x,$$

where $\mathrm{vec}(\cdot)$ is a vectorization of the matrix $P_i, i = 1, 2$, and $\phi(x)$ denotes a bounded continuously differentiable basis function. Note that one can pick $\phi(x)$ as radial basis or sigmoid functions so that they define a complete independent basis set for $F_1^\star$ and $F_2^\star$.

Algorithm 9: Algorithmic iteration for open-loop Stackelberg games

1: **procedure**
2: Given stabilizing policies $u_1^{(0)}, u_2^{(0)}$.
3: for $k = 0, 1, \ldots$ given $u_1^{(k)}$ and $u_2^{(k)}$ solve for the costate $\lambda_1^{(k)}$ and $\lambda_2^{(k)}$ using,

$$\begin{aligned}\lambda_1^{(k)} &= \int_t^{t_f} \varphi^{\mathrm{T}}(\sigma, t) Q_1 x(\sigma) \mathrm{d}\sigma + \varphi^{\mathrm{T}}(t_f, t) P_{1f} x_f \\ \lambda_2^{(k)} &= \int_t^{t_f} \varphi^{\mathrm{T}}(\sigma, t)[Q_2 x(\sigma) - Q_1 y^{(k)}] \mathrm{d}\sigma + \varphi^{\mathrm{T}}(t_f, t)[P_{2f} - P_{1f} P_{3f}] x_f \\ y^{(k)} &= \int_{t_0}^{t} \varphi(t, \tau) B_1 R_{11}^{-1} [B_1^{\mathrm{T}} \lambda_2^{(k)} - R_{21} R_{11}^{-1} B_1^{\mathrm{T}} \lambda_1^{(k)}] \mathrm{d}\tau\end{aligned}$$

on convergence, set $\lambda_1^{(k+1)} = \lambda_1^{(k)}$ and $\lambda_2^{(k+1)} = \lambda_2^{(k)}$.
4: Update the control policies using

$$\begin{aligned}u_1^{(k+1)} &= -R_{11}^{-1} B_1^{\mathrm{T}} \lambda_1^{(k)}, \\ u_2^{(k+1)} &= -R_{22}^{-1} B_2^{\mathrm{T}} \lambda_2^{(k)}.\end{aligned}$$

5: Go to 3.
6: **end procedure**

Since the functions $F_1^\star$ and $F_2^\star$ are not available, we shall consider the actual outputs of the 2 approximators, namely, the critics, as,

$$\hat{F}_1(x) = \mathrm{vec}(\hat{P}_1)^{\mathrm{T}} \phi(x), \quad \forall x, \tag{3.21}$$

and

$$\hat{F}_2(x) = \mathrm{vec}(\hat{P}_2)^{\mathrm{T}} \phi(x), \quad \forall x, \tag{3.22}$$

where $\hat{P}_1$ and $\hat{P}_2$ are the approximation matrices of the actual matrices P_1 and P_2. Similarly for the 2 players (3.16) and (3.19) 2-actor approximators can be developed as,

$$\hat{u}_1(x) = -R_{11}^{-1} B_1^{\mathrm{T}} \frac{\partial \phi(x)^{\mathrm{T}}}{\partial x} \hat{W}_1, \quad \forall x, \tag{3.23}$$

and

$$\hat{u}_2(x) = -R_{22}^{-1} B_2^{\mathrm{T}} \frac{\partial \phi(x)^{\mathrm{T}}}{\partial x} \hat{W}_2, \quad \forall x, \tag{3.24}$$

with $\hat{W}_1$ and $\hat{W}_2$ denoting the current estimated values of $\mathrm{vec}(\hat{P}_1)$ and $\mathrm{vec}(\hat{P}_2)$, respectively.

Approximate versions of Equations (3.15) and (3.18) can be written with (3.21) and (3.22) respectively for every u_1 and u_2 as,

$$H_1(x, \lambda_1, u_1, u_2) = \frac{1}{2} r_1(x, u_1, u_2) + \lambda_1^{\mathrm{T}} (Ax + B_1 u_1 + B_2 u_2) = e_1,$$
$$H_2(x, \lambda_2, u_1, u_2) = \frac{1}{2} r_2(x, u_1, u_2) + \lambda_2^{\mathrm{T}} (Ax + B_1 u_1 + B_2 u_2) + y^{\mathrm{T}} \left(-A^{\mathrm{T}} \frac{\partial \phi(x)}{\partial x} \mathrm{vec}(\hat{P}_1) - Q_1 x \right) = e_2,$$

here $e_1,\ e_2 \in \mathbb{R}$ are the residual errors. Hence, it is desired to select $\mathrm{vec}(\hat{P}_1)$ and $\mathrm{vec}(\hat{P}_2)$ to minimize the following summation of squared residual errors:

$$E_i = \frac{1}{2} e_i^2, \quad i = 1, 2.$$

Now, we shall select the tuning laws for the critics such that $e_1 \to 0,\ e_2 \to 0$, $\mathrm{vec}(\hat{P}_1) \to \mathrm{vec}(P_1)$, and $\mathrm{vec}(\hat{P}_2) \to \mathrm{vec}(P_2)$. For the follower and leader critics after using the normalized gradient descent, one has,

$$\begin{aligned} \dot{\mathrm{vec}}(\hat{P}_1) &= -\frac{\alpha_1}{(1 + \sigma^{\mathrm{T}}\sigma)^2} \frac{\partial E_1}{\partial \mathrm{vec}(\hat{P}_1)}, \\ &= -\frac{\alpha_1 \sigma}{(1 + \sigma^{\mathrm{T}}\sigma)^2} \left(\sigma^{\mathrm{T}} \mathrm{vec}(\hat{P}_1) + \frac{1}{2} r_1(x, u_1, u_2) \right), \end{aligned} \tag{3.25}$$

and

$$\begin{aligned} \dot{\mathrm{vec}}(\hat{P}_2) &= -\frac{\alpha_2}{(1 + \sigma^{\mathrm{T}}\sigma)^2} \frac{\partial E_2}{\partial \mathrm{vec}(\hat{P}_2)} \\ &= -\frac{\alpha_2 \sigma}{(1 + \sigma^{\mathrm{T}}\sigma)^2} \Bigg(\sigma^{\mathrm{T}} \mathrm{vec}(\hat{P}_2) + \frac{1}{2} r_2(x, u_1, u_2) \\ &\quad + y^{\mathrm{T}} \Bigg(-A^{\mathrm{T}} \frac{\partial \phi(x)^{\mathrm{T}}}{\partial x} \mathrm{vec}(\hat{P}_1) - Q_1 x \Bigg) \Bigg), \end{aligned} \tag{3.26}$$

where $\alpha_1,\ \alpha_2 \in \mathbb{R}_+$ are constants that determine the learning rate and $\sigma := \frac{\partial \phi(x)}{\partial x} (Ax + B_1 u_1 + B_2 u_2)$. Properties of the tuning laws in Equations (3.25) and (3.26) are given in the work of Vrabie *et al.* (2013). Specifically, for exponential convergence, we require a PE condition for the signal $\bar{\sigma} := \frac{\sigma}{(1+\sigma^{\mathrm{T}}\sigma)}$.

Finally, we shall select the tuning laws for $\hat{W}_1$ and $\hat{W}_2$ for the actors in Equations (3.23) and (3.24) as,

$$\dot{\hat{W}}_1 = -\alpha_3\{(\hat{W}_1 - \text{vec}(\hat{P}_1))\} \tag{3.27}$$

and

$$\dot{\hat{W}}_2 = -\alpha_4\{(\hat{W}_2 - \text{vec}(\hat{P}_2))\}, \tag{3.28}$$

where $\alpha_3,\ \alpha_4 \in \mathbb{R}_+$ are constants that determine the learning rate.

A pseudocode that describes the proposed learning algorithm has the following form.

Algorithm 10: Algorithmic iteration for open-loop Stackelberg games

1: **procedure**
2: Start with initial conditions $x(0)$, and random initial weights $\text{vec}(\hat{P}_1)(0)$, $\text{vec}(\hat{P}_1)(0)$, $\hat{W}_1(0)$, $\hat{W}_2(0)$ for the critics, and actors.
3: Propagate $t, x(t)$.
4: Propagate $\text{vec}(\hat{P}_1)(t)$, $\text{vec}(\hat{P}_2)(t), \hat{W}_1(t), \hat{W}_2(t)$ $\triangleright$ $\text{vec}(\dot{\hat{P}}_1)$ as in Equation (3.25), $\text{vec}(\dot{\hat{P}}_2)$ as in Equation (3.26), $\dot{\hat{W}}_1$ as in Equation (3.27) and $\dot{\hat{W}}_2$ as in Equation (3.28).
5: Compute the potential functions $\hat{F}_1$ and $\hat{F}_2$ from Equation (3.21) and Equation (3.22), the optimal controls $\hat{u}_1$ and $\hat{u}_2$ from Equation (3.23) and Equation (3.24), respectively.
6: **end procedure**

The main theorem is now given. This shall provide the tuning laws for the actor and critic approximators for the leader and follower. The resulting tuning laws will be used to prove the convergence of the 2-player game algorithm to the open-loop Stackelberg equilibrium solution while also guaranteeing closed-loop stability.

Fact 3.4 (Vamvoudakis *et al.*, 2019, Fact 1)**.** The input matrices of (3.13) are bounded as,

$$\|B_1\| < \bar{b}_1, \quad \|B_2\| < \bar{b}_2,$$

the desired matrices $P_1,\ P_2$ and P_3 are bounded as,

$$\|P_1\| < \bar{\rho}_1, \quad \|P_2\| < \bar{\rho}_2, \quad \|P_3\| < \bar{\rho}_3,$$

and finally, the basis functions (e.g., sigmoids and radial basis functions) have bounded gradients,

$$\left\| \frac{\partial \phi(x)}{\partial x} \right\| < \mu. \qquad \square$$

Theorem 3.5 (Vamvoudakis *et al.*, 2019, Thm. 1)**.** Suppose that the assumptions and the statements of Lemma 3.3 hold and that the game is played for a long enough horizon. Consider the system given by Equation (3.13) and let the controller dynamics be given by Equation (3.20), the critic approximators be given as Equations (3.21) and (3.22), the follower control input be given by Equation (3.23) and the leader be given by Equation (3.24). Let the tuning for the follower critic be given by Equation (3.25) and for the leader by (3.26) and assume that the signal $\bar{\sigma}$ is PE. Given the follower actor in (3.23) and the leader actor in Equation (3.24), then after picking the tuning gains and the user-defined matrices according to the following inequalities:

$$\begin{aligned} \underline{\lambda}(Q_1 + Q_2) > \frac{1}{2}(&2\bar{\rho}_1 \bar{b}_1^2 \mu \|R_{11}^{-1}\| + \bar{\rho}_2 \bar{b}_2^2 \mu \|R_{22}^{-1}\| + \bar{\rho}_1 \bar{b}_2^2 \mu \|R_{22}^{-1}\| \\ &+ \bar{\rho}_3 \bar{b}_1^2 \mu \|R_{11}^{-1}\|), \\ \alpha_1 > &2\alpha_3 + 2(\bar{\rho}_3 \bar{b}_1^2 \mu \|R_{11}^{-1^{\mathrm{T}}} R_{21} R_{11}^{-1}\|) + 2(\bar{\rho}_2 \bar{b}_2^2 \mu \|R_{22}^{-1}\|), \end{aligned}$$

one has an asymptotically stable equilibrium for the closed-loop system. ■

3.4.5 Further Reading

The interested reader is directed to Vamvoudakis *et al.* (2019) for detailed theorems and proofs of the above statements and algorithms. The authors in Saleheen and Won (2019) and Mu *et al.* (2020) propose adaptive learning algorithms for the Stackelberg games. Finally, the work of Mu *et al.* (2020) deals with a statistical Stackelberg stochastic differential game in which the leader optimizes the variance or the second cumulant of the cost function.

3.5 Graphical Games

The ability to coordinate heterogeneous agents is important in many real-world tasks. Synchronization behavior among agents is found in flocking of birds, schooling of fish, and other natural systems. Work has been done to develop cooperative control methods for consensus and synchronization (Fax and Murray, 2004; Jadbabaie *et al.*, 2003; Olfati-Saber and Murray, 2004; Qu, 2009, Ren and Beard, 2005, 2008; Tsitsiklis, 1984). See Olfati-Saber *et al.* (2007) and Ren *et al.* (2005) for surveys. Leaderless consensus results in all nodes converging to common value that cannot generally be controlled. This is called cooperative regulator problem. On the other hand, the problem of cooperative tracking requires that all nodes synchronize to a leader or control node (Chen *et al.*, 2007; Wang and Chen, 2002). This has been called pinning control or control with a virtual leader. Consensus has been studied for systems on communication graphs with fixed or varying topologies and communication delays.

Game theory provides an ideal environment to study multi-player decision and control problems, and offers a wide range of challenging and engaging problems. Game theory (Tijs, 2003) has been successful in modeling strategic behavior, where the outcome for each player depends on the actions of himself and all the other players. Every player chooses a control to minimize in dependently from the others his own performance objective. Multi-player cooperative games rely on solving coupled HJ equations, which in the linear quadratic case reduce to the coupled ARE (Başar and Olsder, 1999; Freiling *et al.*, 2001; Gajic and Li, 1988). Solution methods are generally offline and generate fixed control policies that are implemented in online controllers. These coupled equations are difficult to solve.

RL methods have been used to solve multi-player games for finite-state systems in Busoniu *et al.* (2010), Busoniu *et al.* (2008), and Littman (2001). RL methods have been applied to learn online the solutions for optimal control problems for dynamic systems and differential games in Dierks and Jagannathan (2010), Johnson *et al.* (2010), Vamvoudakis and Lewis (2010) and Vamvoudakis and Lewis (2011).

This subsection brings together cooperative control, RL, and game theory to solve multi-player differential games on communication graph topologies. The notion of graphical games is developed for dynamical systems, where the dynamics and performance indices for each node depend only on local neighbor information. It is shown that standard definitions for Nash equilibrium are not sufficient for graphical games. This is because, for the case of a disconnected graph, the agents can be in Nash equilibrium, yet have no influence on each other. In such situations, the definition of coalition-proof Nash equilibrium may also hold, that is, no set of agents has an incentive to break away from the Nash equilibrium and seek a new Nash solution among them. To guarantee that all agents in a graph are involved in the same game, the stronger definition of interactive Nash equilibrium is used.

We give a cooperative policy iteration algorithm for graphical games that converges to the best response when the neighbors of each agent do not update their policies, and to the cooperative Nash equilibrium when all agents update their policies simultaneously. This is used to develop methods for online adaptive learning solutions of graphical games.

3.5.1 Background on Networks and Graphs

In this subsection, multi-agent system interactions are modeled by a fixed strongly connected *graph* $\mathcal{G} = (\mathcal{V}_\mathcal{G}, \mathcal{E}_\mathcal{G})$ defined by a finite set $\mathcal{V}_\mathcal{G} = \{n_1, \ldots, n_N\}$ of N nodes that represent agents and a set of edges $\mathcal{E}_\mathcal{G} \subseteq \mathcal{V}_\mathcal{G} \times \mathcal{V}_\mathcal{G}$ that represent inter-agent information exchange links. The set of *neighbors* of a node n_i is defined as the set of nodes with edges incoming to n_i and is denoted by $\mathcal{N}_i = \{n_j \colon (n_j, n_i) \in \mathcal{E}_\mathcal{G}\}$. We assume the graph is simple (i.e., it has not repeated edges) and has no self loops (i.e., $(n_i, n_i) \notin \mathcal{E}_\mathcal{G}, \forall n_i \in \mathcal{V}_\mathcal{G}$). The *connectivity matrix* $\mathcal{A}_\mathcal{G} = [a_{ij}]_{N\times N}$ of the graph $\mathcal{G}$ is an $N \times N$ matrix of the form,

$$a_{ij} \begin{cases} > 0, & \text{if } (n_j, n_i) \in \mathcal{E}_\mathcal{G}, \\ = 0, & \text{otherwise,} \end{cases}$$

where a_{ij} is the *weight* associated with the edge $(n_j, n_i) \in \mathcal{E}_\mathcal{G}$. The *degree* matrix of the graph $\mathcal{G}$ is an $N \times N$ diagonal matrix whose ith entry of the main diagonal is the weighted degree $d_i = \sum_{j\in\mathcal{N}_i} a_{ij}$ of node i

(i.e., ith row sum of $\mathcal{A}_\mathcal{G}$). The number of the neighbors of agent i is denoted by $|\mathcal{N}_i|$ that is equal to d_i.

3.5.2 Problem Formulation

Consider a networked-system $\mathcal{G}$, consisting of N agents each modeled $\forall i \in \mathcal{N} := \{1, \ldots, N\}$ by the following dynamics,

$$\dot{x}_i(t) = Ax_i(t) + B_i u_i(t), \quad x_i(0) = x_{i0},\ t \geqslant 0,$$

where $x_i(t) \in \mathbb{R}^n$ is a measurable state vector, $u_i(t) \in \mathbb{R}^{m_i},\ i \in \mathcal{N} := \{1, \ldots, N\}$ is each control input (or player as we shall see later), and $A \in \mathbb{R}^{n\times n}$, $B_i \in \mathbb{R}^{n\times m_i},\ i \in \mathcal{N}$ are the plant and input matrices, respectively, that will be considered uncertain/unknown. It is assumed that the pairs $(A, B_i),\ \forall i \in \mathcal{N}$ are controllable. We have a total of N players/controllers that select values for $u_i(t),\ t \geqslant 0,\ i \in \mathcal{N}$. The agents in the network seek to cooperatively asymptotically track the state of a leader node/exosystem with dynamics $\dot{x}_0 = Ax_0$, i.e., $x_i(t) \to x_0(t)$, $\forall i \in \mathcal{N}$ while *simultaneously satisfying user-defined distributed performances*. Before we proceed to the design of the energy-related user-defined distributed performances we will define the following *neighborhood tracking error* for every agent,

$$\delta_i := \sum_{j\in\mathcal{N}_i} a_{ij}(x_i - x_j) + g_i(x_i - x_0), \quad \forall i \in \mathcal{N}, \tag{3.29}$$

where $g_i \in \mathbb{R}_+$ is the pinning gain that shows if an agent is pinned to the leader node (i.e., $g_i \neq 0$) and it is nonzero for at least one node.

The dynamics of (3.29) are given by $\forall t \geqslant 0$,

$$\dot{\delta}_i = A\delta_i + (d_i + g_i)B_i u_i - \sum_{j\in\mathcal{N}_i} a_{ij}B_j u_j, \quad \forall i \in \mathcal{N}, \tag{3.30}$$

with $\delta_i \in \mathbb{R}^n$.

The cost functionals associated to each agent $i \in \mathcal{N}$, that depend on the tracking error δ_i, the control u_i and the controls in the neighborhood

of agent i given as, $u_{\mathcal{N}_i} := \{u_j \colon j \in \mathcal{N}_i\}$, have the following form,

$$\begin{aligned}\mathcal{J}_i(\delta_i(0); u_i, u_{\mathcal{N}_i}) &= \frac{1}{2}\int_0^\infty r_i(\delta_i, u_i, u_{\mathcal{N}_i})\mathrm{d}t, \quad \forall i \in \mathcal{N} \\ &\equiv \frac{1}{2}\int_0^\infty \left(\delta_i^{\mathrm{T}} H_i \delta_i + u_i^{\mathrm{T}} R_{ii} u_i + \sum_{j\in\mathcal{N}_i} u_j^{\mathrm{T}} R_{ij} u_j\right)\mathrm{d}t, \\ &\qquad\qquad \forall i \in \mathcal{N},\end{aligned}$$

with user defined matrices $H_i \geq 0$, $R_{ii} > 0$, $R_{ij} \geq 0$, $\forall i, j \in \mathcal{N}$ of appropriate dimensions and $(\sqrt{H}_i, A)$, $\forall i \in \mathcal{N}$ are detectable.

Thus given a strongly connected graph $\mathcal{G}$, we are interested in finding a graphical Nash equilibrium (Vamvoudakis *et al.*, 2012) $u_i^\star$ for all agents $i \in \mathcal{N}$ in the sense that,

$$\mathcal{J}_i(\delta_i(0); u_i^\star, u_{\mathcal{N}_i}^\star) \leqslant \mathcal{J}_i(\delta_i(0); u_i, u_{\mathcal{N}_i}^\star), \quad \forall u_i,\ i \in \mathcal{N}.$$

This can be expressed by the following coupled distributed minimization problems,

$$\mathcal{J}_i(\delta_i(0); u_i^\star, u_{\mathcal{N}_i}^\star) = \min_{u_i} \mathcal{J}_i(\delta_i(0); u_i, u_{\mathcal{N}_i}^\star), \quad \forall i \in \mathcal{N},$$

with the dynamics given in (3.30).

Hence, the ultimate goal is to find the distributed optimal value functions $V_i^\star$, $\forall i \in \mathcal{N}$ defined by,

$$\begin{aligned}V_i^\star(\delta_i(t)) := \min_{u_i} \int_t^\infty \frac{1}{2}(\delta_i^{\mathrm{T}} H_i \delta_i + u_i^{\mathrm{T}} R_{ii} u_i + \sum_{j\in\mathcal{N}_i} u_j^{\mathrm{T}} R_{ij} u_j)\mathrm{d}t, \\ \forall \delta_i,\ i \in \mathcal{N},\ t \geqslant 0. \qquad (3.31)\end{aligned}$$

One can define the Hamiltonians associated with each agent's neighborhood tracking error (3.30) and each $V_i^\star$ given in (3.31) as follows,

$$\begin{aligned}\mathcal{H}_i\left(\delta_i, u_i, u_{\mathcal{N}_i}, \frac{\partial V_i^\star}{\partial \delta_i}\right) = \frac{\partial V_i^{\star\,\mathrm{T}}}{\partial \delta_i}\left(A\delta_i + (d_i + g_i)B_i u_i - \sum_{j\in\mathcal{N}_i} a_{ij} B_j u_j\right) \\ + \frac{1}{2}\delta_i^{\mathrm{T}} H_i \delta_i + \frac{1}{2} u_i^{\mathrm{T}} R_{ii} u_i \\ + \frac{1}{2}\sum_{j\in\mathcal{N}_i} u_j^{\mathrm{T}} R_{ij} u_j, \quad \forall \delta_i, u_i, i \in \mathcal{N}.\end{aligned}$$

After employing the stationarity condition, in the Hamiltonian, i.e., $\frac{\partial \mathcal{H}_i(\cdot)}{\partial u_i} = 0$, the optimal control for each $i \in \mathcal{N}$ can be found to be,

$$\begin{aligned} u_i^\star(\delta_i) &= \arg\min_{u_i} \mathcal{H}_i \left(\delta_i, u_i, u_{\mathcal{N}_i}, \frac{\partial V_i^\star}{\partial \delta_i} \right) \\ &= -(d_i + g_i) R_{ii}^{-1} B_i^{\mathrm{T}} \frac{\partial V_i^\star}{\partial \delta_i}, \quad \forall \delta_i, \end{aligned} \tag{3.32}$$

that should satisfy the appropriate coupled HJ equations,

$$\mathcal{H}_i \left(\delta_i, u_i^\star, u_{\mathcal{N}_i}^\star, \frac{\partial V_i^\star}{\partial \delta_i} \right) = 0, \quad \forall \delta_i, \ i \in \mathcal{N}.$$

One could represent the value functions as quadratic in the neighborhood tracking error, i.e., $V_i^\star(\delta_i)$: $\mathbb{R}^n \to \mathbb{R}$,

$$V_i^\star(\delta_i) = \frac{1}{2} \delta_i^{\mathrm{T}} P_i \delta_i, \quad \forall \delta_i, \ i \in \mathcal{N}, \tag{3.33}$$

where $P_i \in \mathbb{R}^{n \times n}$, $\forall i \in \mathcal{N}$ are the unique symmetric positive definite matrices that solve the following complicated distributed coupled equations (Vamvoudakis *et al.*, 2012),

$$\begin{aligned} &\delta_i^{\mathrm{T}} P_i \left(A\delta_i - (d_i + g_i)^2 B_i R_{ii}^{-1} B_i^{\mathrm{T}} P_i \delta_i + \sum_{j \in \mathcal{N}} a_{ij} (d_j + g_j) B_j R_{jj}^{-1} B_j^{\mathrm{T}} P_j \delta_j \right) \\ &+ \left(A\delta_i - (d_i + g_i)^2 B_i R_{ii}^{-1} B_i^{\mathrm{T}} P_i \delta_i + \sum_{j \in \mathcal{N}} a_{ij} (d_j + g_j) B_j R_{jj}^{-1} B_j^{\mathrm{T}} P_j \delta_j \right)^{\mathrm{T}} P_i \delta_i \\ &+ \sum_{j \in \mathcal{N}_i} (d_j + g_j)^2 \delta_j^{\mathrm{T}} P_j B_j R_{jj}^{-1^{\mathrm{T}}} R_{ij} R_{jj}^{-1} B_j^{\mathrm{T}} P_j \delta_j \\ &+ (d_i + g_i)^2 \delta_i^{\mathrm{T}} P_i B_i R_{ii} B_i^{\mathrm{T}} P_i \delta_i + \delta_i^{\mathrm{T}} H_i \delta_i = 0, \quad \forall \delta_i, \in \mathcal{N}. \end{aligned}$$

By using (3.33), the optimal control (3.32) for every player $i \in \mathcal{N}$ can be written as,

$$u_i^\star(\delta_i) = -(d_i + g_i) R_{ii}^{-1} B_i^{\mathrm{T}} P_i \delta_i, \quad \forall \delta_i.$$

3.5.3 Approximate Solution

The structure used for our approximate solution is motivated by the Policy Iteration Algorithm that follows, where ϵ_{ac} is a small number

Algorithm 11: Policy Iteration for graphical games

procedure
  Given N-tuple of admissible policies $\mu_i^k(0),\ \forall i \in \mathcal{N}$ and $k = 1$.
  while $\|V_i^{\mu_i^{(k)}}(\delta_i) - V_i^{\mu_i^{(k-1)}}(\delta_i)\| \geqslant \epsilon_{\text{iac}},\ \forall i \in \mathcal{N}$ **do**
    Solve for the N-tuple of costs $V_i^k(x)$ using the coupled Bellman equations

$$\delta_i^{\mathrm{T}} Q_{ii} \delta_i + \frac{\partial V_i^k}{\partial \delta_i}^{\mathrm{T}} \left(A\delta_i + (d_i + g_i) B_i \mu_i^k - \sum_{j \in \mathcal{N}_i} \alpha_{ij} B_j \mu_j^k \right) + \mu_i^{k\,\mathrm{T}} R_{ii} \mu_i^k + \sum_{j \in \mathcal{N}_i} \mu_j^{k\,\mathrm{T}} R_{ij} \mu_j^k = 0, \quad V^{\mu_i^k}(0) = 0.$$

    Update the N-tuple of control policies $\mu_i^{k+1},\ \forall i \in \mathcal{N}$ using

$$\mu_i^{k+1} = -(d_i + g_i) R_{ii}^{-1} B_i^{\mathrm{T}} \frac{\partial V_i^k}{\partial \delta_i}.$$

    $k \coloneqq k + 1$.
  end while
end procedure

used to terminate the algorithm when two consecutive value functions differ by less than $\epsilon_{iac}, \forall i \in \mathcal{N}$.

We should use N critic-networks to approximate the costs and N actor-networks to approximate the control inputs by using only distributed information. We assume there exist constant weights W_i such that the value functions V_i are approximated on a compact set Ω as,

$$V_i(\delta_i) = W_i^{\mathrm{T}} \phi_i(\delta_i) + \epsilon_i(\delta_i), \quad \forall \delta_i,\ \forall i \in \mathcal{N},$$

where $\phi_i(\delta_i)$: $\mathbb{R}^n \to \mathbb{R}^{K_i},\ \forall i \in \mathcal{N}$ are the activation function basis set vectors, K_i is the number of neurons in the hidden layer and $\epsilon_i(\delta_i)$ the approximation error. From the approximation literature, the basis functions can be selected as sigmoids, tanh, polynomials, etc.

Assuming current weight estimates $\hat{W}_{ic}, \forall i \in \mathcal{N}$, the outputs of the critics are given by,

$$\hat{V}_i(\delta_i) = \hat{W}_{ic}^{\mathrm{T}} \phi_i(\delta_i), \quad \forall \delta_i,\ \forall i \in \mathcal{N}. \tag{3.34}$$

Using the current weight estimates, the approximate Lyapunov-like equations are given by,

$$\hat{H}_i(\cdot) = r_i(\delta_i, \hat{u}_i, \hat{u}_{\mathcal{N}_i}) + \nabla V_i^{\mathrm{T}}(\delta_i)\Big(A\delta_i + (d_i + g_i)B_i\hat{u}_i - \sum_{j\mathcal{N}_i} \alpha_{ij}B_j\hat{u}_j \Big) = e_i, \quad \forall \delta_i,\ \forall \hat{u}_i,\ \forall i \in \mathcal{N},$$

where $e_i \in \mathbb{R},\ \forall \delta_i, \forall i \in \mathcal{N}$ are the residual errors due to approximation and,

$$\hat{u}_i = -(d_i + g_i)R_{ii}^{-1}B_i^{\mathrm{T}}\nabla\phi_i^{\mathrm{T}}(\delta_i)\hat{W}_{iu}, \quad \forall \delta_i,\ \forall i \in \mathcal{N}, \tag{3.35}$$

where $\hat{W}_{iu}$ denote the current estimated values of the ideal weights W_i.

Now it is desired to pick the tuning for the critic in order to minimize the square residual errors e_i in order to guarantee that $\hat{W}_{ic} \to W_i$, $\forall i \in \mathcal{N}$. Hence the tuning law for the critic is given as,

$$\dot{\hat{W}}_{ic} = -\alpha_i \frac{\upsilon_i}{(\upsilon_i^{\mathrm{T}}\upsilon_i + 1)^2}\Big(\upsilon_i^{\mathrm{T}}\hat{W}_{ic} + \frac{1}{2}\delta_i^{\mathrm{T}}Q_i\delta_i + \hat{u}_i^{\mathrm{T}}R_{ii}\hat{u}_i + \sum_{j\in\mathcal{N}_i} \hat{u}_j^{\mathrm{T}}R_{ij}\hat{u} \Big), \quad \forall i \in \mathcal{N}, \tag{3.36}$$

where $\upsilon_i = \nabla\phi_i(A\delta_i + (d_i + g_i)B_i\hat{u}_i - \sum_{j\in\mathcal{N}_i} \alpha_{ij}B_j\hat{u}_j)$ and $\alpha_i \in \mathbb{R}_+$ is a tuning gain that determines the speed of convergence. The tuning law for the actor is given as,

$$\dot{\hat{W}}_{iu} = -\alpha_{iu}\Big((F_i\hat{W}_{iu} - \mathbf{1}^{\mathrm{T}}\upsilon_i^{\mathrm{T}}\hat{W}_{ic}) - \frac{1}{4}\sum_{j\in\mathcal{N}_i} (\nabla\phi_i B_i R_{ii}^{-1^{\mathrm{T}}} R_{ij} R_{ii}^{-1} B_j^{\mathrm{T}}\nabla\phi_i^{\mathrm{T}}) \times \hat{W}_{iu}\frac{\upsilon_i^{\mathrm{T}}}{(\upsilon_i^{\mathrm{T}}\upsilon_i + 1)^2}\hat{W}_{jc} \Big), \quad \forall i \in \mathcal{N}, \tag{3.37}$$

where $\alpha_{iu} \in \mathbb{R}_+$ is a tuning gain that determines the speed of convergence, and $F_i > 0$ is a matrix that guarantees stability of the system.

Theorem 3.6 (Vamvoudakis *et al.*, 2012, Thm. 6)**.** Let the error dynamics be given by (3.30), and consider the cooperative game formulation in (3.31). Let the critic NN at each node be given by (3.34) and the control

input be given for each node by actor NN (3.35). Let the tuning law for the i-th critic NN be provided by (3.36) and the tuning law for the i-th actor NN be provided by (3.37). Assume $\bar{\upsilon}_i = \upsilon_i/(\upsilon_i^{\mathrm{T}}\upsilon_i)$ is persistently exciting. Then the closed-loop signals are UUB. ■

A pseudocode that describes the proposed adaptive multi-agent game algorithm has the following form,

Algorithm 12: Multi-Agent Game Algorithm with Distributed Information

1: **procedure**
2: Start with initial state $\delta_i(0),\ \forall i \in \mathcal{N}$ and random initial weights $\hat{W}_{iu}(0), \hat{W}_{ic}(0), \forall i \in \mathcal{N}$.
3: Propagate $t, \delta_i(t),\ \forall i \in \mathcal{N}$.
4: Propagate $\hat{W}_{ic}(t), \hat{W}_{iu}(t), \forall i \in \mathcal{N}$ ▷ $\dot{\hat{W}}_{ic}$ as in (3.36), and $\dot{\hat{W}}_{iu}$ as in (3.37).
5: Compute (3.34) and (3.35).
6: **end procedure**

3.5.4 Further Reading

The interested reader is directed to Vamvoudakis *et al.* (2012, 2017a) for detailed proofs, theorems and simulation examples. In the work of Zhang *et al.* (2014c), an online scheme is presented to design the optimal coordination control for the consensus problem of multiagent differential games by fuzzy adaptive dynamic programming. In the paper of Zhang *et al.* (2014a), the inverse optimal approach is employed to design distributed consensus protocols that guarantee consensus and global optimality with respect to some quadratic performance indexes for identical linear systems on a directed graph. Finally, the authors in Abouheaf *et al.* (2014) and Ye and Hu (2017) develop RL-based solutions for graphical games.

4

Model-Free RL with Q-Learning

Unlike the foregoing sections wherein we primarily demonstrated model-based RL algorithmic schemes, this section discusses model-free RL algorithms based on a CT Q-learning framework. The presented schemes solve infinite-horizon optimization problems of linear time-invariant systems with completely unknown dynamics and single or multiple players/controllers. We first formulate the appropriate Q-functions (action-dependent value functions) and the tuning laws based on actor/critic structures for several cases including, optimal regulation (Subsection 4.2), Nash games (Subsection 4.3), and multi-agent systems (Subsection 4.4).

4.1 Introduction and Motivation

Q-learning was the first provably convergent direct optimal adaptive control algorithm and is a model-free reinforcement learning technique developed primarily for discrete-time systems, namely Markov Decision Processes (Watkins and Dayan, 1992). Specifically, Q-learning can be used to find an optimal action-selection policy based on measurements of previous state and action observations controlled using a "non-optimal policy." It learns an action-dependent value function that ultimately

gives the expected utility of taking a given action in a given state and following the optimal policy thereafter. When such an action-dependent value function is learned, the optimal policy can be computed easily. The biggest strength of Q-learning is that it does not require a model of the environment. It has been proven in Watkins and Dayan (1992) that for any finite Markov Decision Process, Q-learning eventually finds an optimal policy. Q-learning at its simplest uses tables to store data. This very quickly loses viability with increasing levels of complexity and dimensionality of the systems. This problem can be solved effectively by using adapted networks as universal approximators. Specifically, Q-learning can be improved by using the universal function approximation property of NNs and especially in the context of approximate dynamic programming (Werbos, 1992) or neuro-dynamic programming (Bertsekas and Tsitsiklis, 1995) that allow us to solve difficult optimization problems online and forward in time. The work of Melo *et al.* (2008) analyzed how reinforcement learning can be combined with function approximation techniques to find the optimal Q-function in Markov Decision Processes with infinite state-spaces.

Recently, many research efforts after the work of Watkins and Dayan (1992) on Q-learning and model-free approaches have focused on discrete time systems and use sequential multi-step algorithms that are not applicable to feedback control systems. The authors in Al-Tamimi *et al.* (2008), have proposed a sequential update Q-learning approach to solve ZS games in systems with discrete dynamics. Similarly with Al-Tamimi *et al.* (2008) the work of Kiumarsi *et al.* (2014) proposes a Q-learning framework to solve the optimal tracking problem of discrete time systems. An output feedback Q-learning approach for partially observable dynamics processes has been presented in Lewis and Vamvoudakis (2010). Following the work of Watkins and Dayan (1992) the authors in Tsitsiklis (1994) proved convergence for undiscounted problems without assuming that all policies must lead to a zero-cost absorbing state by allowing the next action to depend on past experiences by considering asynchronous stochastic approximation. Motivated by the convergence results of Tsitsiklis (1994) the authors in Beck and Srikant (2012) provided a bound on the first moment of the error in the constant step-size

Q-learning for the synchronous and the asynchronous case given discounted reward, infinite-horizon problem, where the system is a discrete, finite state-space Markov chain with a finite number of available control actions in each state. The authors in Xu *et al.* (2012) have used Q-learning to solve the infinite horizon optimal regulation problem of unknown networked control systems with time-varying system matrices.

However, in CT systems, things are harder since the Hamiltonian functions depend on the system dynamics. Till now to apply to CT systems one has to use discretization of the state and the action space to apply such techniques, and as such lose important information. The authors in Mehta and Meyn (2009) have established connections between Q-learning and nonlinear control of CT models with general state and action space by observing that the Q-function developed in Watkins and Dayan (1992) is an extension of the Hamiltonian that appears in the minimum principle. A variant of Q-learning that provides a model-free off-policy approach for CT systems has been proposed in Jiang and Jiang (2012) where the authors solve the ARE iteratively using system state and input information collected online, without knowing the system matrices. The aforementioned algorithm is a two-step algorithm that requires an initial stabilizing policy to start with. Our algorithm will be simultaneous and will not require such a policy for initialization. An ϵ-integral Q function has been used to propose an ϵ-approximate Q-learning framework for solving the linear quadratic regulator problem of CT systems in Lee *et al.* (2012) but the authors can guarantee convergence and UUB only when the initial policy is stabilizing. The work of Lee *et al.* (2012) was based mostly on the work of Vrabie *et al.* (2009) and Abu-Khalaf *et al.* (2006) where the authors propose a sequential algorithm with the necessity of an initial stabilizing policy to solve the optimal control problem without the drift dynamics. Our work overcomes these limitations by providing asymptotic stability of the closed-loop system without any initial stabilizing policy. The work of Xu *et al.* (2014) proposes a Q-learning approach to solve the finite-horizon optimal control problem which eventually reduces to solve the differential Riccati equation without any proofs of convergence. The authors of Palanisamy *et al.* (2014) propose a Q-learning approach to solve the CT infinite-horizon optimal control problem by writing

the Q-function with respect to the state, the control input and the derivatives of the control input. The algorithm proposed in Palanisamy *et al.* (2014) uses iterations as in Lee *et al.* (2012) on the value function to solve the problem.

4.2 Q-Learning for Optimal Regulation

In this subsection, the infinite horizon optimal control problem is formulated as a Q-learning (model-free) problem by using a quadratic in the state and control parametrization. An IRL approach will be used to design and derive tuning formulas for a critic approximator estimating the action-dependent value of the Q-function and for an actor approximator estimating the optimal control. The actor and the critic approximators are tuned simultaneously by measuring information along the state and the control trajectories *without the need of an initial admissible (i.e., stabilizing) policy*, this is in contrast to existing Q-learning approaches that use sequential policy iteration algorithms which delay the convergence, and require admissibility of the initial policy.

4.2.1 Problem Formulation

Consider the following linear time-invariant CT system,

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0, \quad t \geqslant 0, \tag{4.1}$$

where $x(t) \in \mathbb{R}^n$ is a measurable state vector, $u(t) \in \mathbb{R}^m$ is the control input and $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ are the plant and input matrices, respectively, that will be considered uncertain/unknown. It is assumed that the pair (A, B) is controllable. Our goal is to find a controller u that minimizes a cost functional of the form,

$$J(x(0); u) = \frac{1}{2}\int_0^\infty (x^{\mathrm{T}} M x + u^{\mathrm{T}} R u)\mathrm{d}t,$$

with user defined matrices $M \succeq 0$, $R \succ 0$ of appropriate dimensions and $(\sqrt{M}, A)$ detectable.

Remark 4.1. Note that the assumptions that the pair (A, B) is controllable and the pair $(\sqrt{M}, A)$ is detectable, will guarantee that the ARE will have a unique non-negative solution. □

The ultimate goal is to find the optimal value function $V^\star$ defined by,

$$V^\star(x(t)) := \min_u \int_t^\infty \frac{1}{2}(x^{\mathrm{T}} M x + u^{\mathrm{T}} R u)\mathrm{d}\tau, \quad \forall x,\ t \geqslant 0, \tag{4.2}$$

given (4.1) but without any information of the system dynamics.

Before we proceed to the model-free Q-learning approach with the use of an actor/critic adaptive structure, we shall talk briefly about the of the infinite-horizon, CT linear quadratic optimal control problem.

One can define the Hamiltonian associated with (4.1) and (4.2) as follows,

$$H\left(x, u, \frac{\partial V^\star}{\partial x}\right) = \frac{\partial V^{\star\mathrm{T}}}{\partial x}(Ax + Bu) + \frac{1}{2}x^{\mathrm{T}} M x + \frac{1}{2}u^{\mathrm{T}} R u, \quad \forall x, u. \tag{4.3}$$

After employing the stationarity condition, in the Hamiltonian (4.3), i.e., $\frac{\partial H(\cdot)}{\partial u} = 0$, the optimal control can be found to be,

$$u^\star(x) = \arg\min_u H\left(x, u, \frac{\partial V^\star}{\partial x}\right) = -R^{-1}B^{\mathrm{T}}\frac{\partial V^\star}{\partial x}, \quad \forall x. \tag{4.4}$$

Since the system (4.1) is linear we can represent the value function as quadratic in the state, i.e., $V^\star(x)$: $\mathbb{R}^n \to \mathbb{R}$,

$$V^\star(x) = \frac{1}{2}x^{\mathrm{T}} P x, \quad \forall x,$$

where $P \in \mathbb{R}^{n\times n}$ is the unique symmetric positive definite matrix that solves the following Riccati equation (Lewis *et al.*, 2012a),

$$A^{\mathrm{T}}P + PA - PBR^{-1}B^{\mathrm{T}}P + M = 0, \tag{4.5}$$

and the optimal control (4.4) is given by,

$$u^\star(x) = -R^{-1}B^{\mathrm{T}}P x, \quad \forall x. \tag{4.6}$$

It is important to note that Eqs. (4.5) and (4.6) require complete knowledge of the system dynamics, i.e., the system and input matrices A, B. In what follows, we shall develop a Q-learning based approach to

solve the optimal control problem without an information of the system dynamics, by adjusting parameters in a adaptive way. To this end, the value function needs to be parametrized as a function of the state x and the control u to represent the Q-function. The optimal value given by (4.2) after adding the Hamiltonian can be written as the following Q-function or action-dependent value, $Q(x,u)$: $\mathbb{R}^{n+m} \to \mathbb{R}$,

$$\begin{aligned} Q(x,u) := V^\star(x) + \frac{1}{2}x^{\mathrm{T}}P(Ax+Bu) + \frac{1}{2}(Ax+Bu)^{\mathrm{T}}Px \\ + \frac{1}{2}u^{\mathrm{T}}Ru + \frac{1}{2}x^{\mathrm{T}}Mx, \quad \forall x,u, \end{aligned} \tag{4.7}$$

where the optimal cost is $V^\star(x) = \frac{1}{2}x^{\mathrm{T}}Px$ with $P \succ 0$. The Q-function (4.7) can be written in a compact quadratic in the state x and control u form as follows,

$$\begin{aligned} Q(x,u) &= \frac{1}{2}U^{\mathrm{T}}\begin{bmatrix} P+M+PA+A^{\mathrm{T}}P & PB \\ B^{\mathrm{T}}P & R \end{bmatrix}U \\ &:= \frac{1}{2}U^{\mathrm{T}}\begin{bmatrix} Q_{\mathrm{xx}} & Q_{\mathrm{xu}} \\ Q_{\mathrm{ux}} & Q_{\mathrm{uu}} \end{bmatrix}U := \frac{1}{2}U^{\mathrm{T}}\bar{Q}U, \quad \forall x,u, \end{aligned}$$

where $U := \begin{bmatrix} x^{\mathrm{T}} & u^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$, $Q_{\mathrm{xx}} = P+M+PA+A^{\mathrm{T}}P$, $Q_{\mathrm{xu}} = PB$, $Q_{\mathrm{ux}} = Q_{\mathrm{xu}}^{\mathrm{T}} = B^{\mathrm{T}}P$, $Q_{\mathrm{uu}} = R$ and $\bar{Q} = \begin{bmatrix} Q_{\mathrm{xx}} & Q_{\mathrm{xu}} \\ Q_{\mathrm{ux}} & Q_{\mathrm{uu}} \end{bmatrix} \in \mathbb{R}^{(n+m)\times(n+m)}$.

A model-free optimal control formulation, can be found by solving $\frac{\partial Q_{(x,u)}}{\partial u} = 0$.

Since the ideal weights for computing $Q^\star$ and $u^\star$ are unknown, one must consider the following *weight estimates*. The critic approximator with $\hat{W}_c := \mathrm{vech}(\hat{\bar{Q}})$ can be written as,

$$\hat{Q}(x,u) = \hat{W}_c^{\mathrm{T}}(U \otimes U), \quad \forall x,u, \tag{4.8}$$

where $\hat{W}_c \in \mathbb{R}^{\frac{1}{2}(n+m)(n+m+1)}$ are the estimated weights and $U := \begin{bmatrix} x^{\mathrm{T}} & u^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$. Similarly the actor approximator can be expressed as,

$$\hat{u}(x) = \hat{W}_a^{\mathrm{T}}x, \quad \forall x, \tag{4.9}$$

where $\hat{W}_a \in \mathbb{R}^{n\times m}$ are the weight estimates, note also that the state x is serving as an activation function for the action approximator.

Remark 4.2. It is worth noting that the critic and the actor approximators given by (4.8) and (4.9), respectively, do not have any approximation errors since we have a complete basis representation with respect to the state x and the control u in the form $(U \otimes U)$. For that reason, we do not employ approximation in a compact set but in the whole space. □

We have showed in Vrabie *et al.* (2013) that by using IRL we can write the value function (4.2) as the following Bellman equation,

$$V^\star(x(t)) = V^\star(x(t-T)) - \frac{1}{2}\int_{t-T}^{T}(x^{\mathrm{T}}Mx + u^{\mathrm{T}}Ru)\mathrm{d}\tau, \tag{4.10}$$

where $T \in \mathbb{R}^+$ is a small fixed time interval. Because the value of (4.10) is the same with the action-dependent value of $Q^\star(x, u^\star)$, we can write (4.10) $\forall x$ as,

$$\begin{aligned} Q^\star(x(t), u^\star(t)) = {} & Q^\star(x(t-T), u^\star(t-T)) \\ & - \frac{1}{2}\int_{t-T}^{T}(x^{\mathrm{T}}Mx + u^{\mathrm{T}}Ru)\mathrm{d}\tau. \end{aligned} \tag{4.11}$$

We shall define an error $e \in \mathbb{R}$ that we would like to eventually drive to zero by picking appropriately the tuning law for $\hat{W}_{\mathrm{c}}$. In order to do that, we shall take into consideration the Bellman Equation (4.11) and using actual values for the Q-function, (4.11) can be rewritten as,

$$\begin{aligned} e :=& \hat{Q}(x(t), u(t)) - \hat{Q}(x(t-T), u(t-T)) \\ & + \frac{1}{2}\int_{t-T}^{T}(x^{\mathrm{T}}Mx + u^{\mathrm{T}}Ru)\mathrm{d}\tau \\ =& \hat{W}_{\mathrm{c}}^{\mathrm{T}}(U(t) \otimes U(t)) + \frac{1}{2}\int_{t-T}^{T}(x^{\mathrm{T}}Mx + u^{\mathrm{T}}Ru)\mathrm{d}\tau \\ & - \hat{W}_{\mathrm{c}}^{\mathrm{T}}(U(t-T) \otimes U(t-T)). \end{aligned}$$

Now for the actor approximator, we can define the error $e_{\mathrm{a}} \in \mathbb{R}^m$ as follows, $e_{\mathrm{a}} := \hat{W}_{\mathrm{a}}^{\mathrm{T}}x + \hat{Q}_{\mathrm{uu}}^{-1}\hat{Q}_{\mathrm{ux}}x$, where the values of $\hat{Q}_{\mathrm{uu}}^{-1}$ and $\hat{Q}_{\mathrm{ux}}$ are going to be extracted from the vector $\hat{W}_{\mathrm{c}}$. Now we shall find tuning updates for $\hat{W}_{\mathrm{c}}$ and $\hat{W}_{\mathrm{a}}$ such that the errors e and e_{a} go to zero. By following adaptive control techniques Ioannou and Fidan (2006) we can

define the squared-norm of errors e and e_{a} as,

$$K_1 = \frac{1}{2}\|e\|^2, \tag{4.12}$$

$$K_2 = \frac{1}{2}\|e_{\mathrm{a}}\|^2. \tag{4.13}$$

4.2.2 Learning Algorithm

The estimate of $\hat{W}_{\mathrm{c}}$ (in order to guarantee that $e \to 0$ and $\hat{W}_{\mathrm{c}} \to W_{\mathrm{c}}$) for the critic weights can be constructed by applying gradient descent in (4.12), using the chain rule and normalizing (following adaptive control theory Ioannou and Fidan, 2006) as,

$$\dot{\hat{W}}_{\mathrm{c}} = -\alpha_{\mathrm{c}}\frac{\partial K_1}{\partial \hat{W}_{\mathrm{c}}} = -\alpha_{\mathrm{c}}\frac{\sigma}{(1+\sigma^{\mathrm{T}}\sigma)^2}e^{\mathrm{T}}, \tag{4.14}$$

where $\sigma := U(t)\otimes U(t) - U(t-T)\otimes U(t-T)$, and $\alpha_{\mathrm{c}} \in \mathbb{R}^+$ is a constant gain that determines the speed of convergence.

Similarly, the gradient descent estimate of $\hat{W}_{\mathrm{a}}$ for the actor weights can be constructed by applying gradient descent in (4.13),

$$\dot{\hat{W}}_{\mathrm{a}} = -\alpha_{\mathrm{a}}\frac{\partial K_2}{\partial \hat{W}_{\mathrm{a}}} = -\alpha_{\mathrm{a}} x e_{\mathrm{a}}^{\mathrm{T}}, \tag{4.15}$$

where $\alpha_{\mathrm{a}} \in \mathbb{R}^+$ is a constant gain that determines the speed of convergence.

We now present the main result which guarantees the asymptotic stability of the learning algorithm of the resulting closed-loop dynamics

Theorem 4.1 (Vamvoudakis, 2017, Thm. 2)**.** Consider the system dynamics given by (4.1), the critic approximator given by (4.8) and the optimal control given by (4.9). The tuning law for the weights of the critic is given by (4.14) and for the weights of the actor is given by (4.15). Then the equilibrium point (i.e., origin) of the closed-loop system is asymptotically stable given a tuning gain for the critic α_{c} sufficiently larger than the tuning gain for the actor α_{a} and,

$$1 < \alpha_{\mathrm{a}} < \frac{1}{\delta\bar{\lambda}(R^{-1})}(2\underline{\lambda}(M + Q_{\mathrm{xu}}R^{-1}Q_{\mathrm{xu}}^{\mathrm{T}}) - \bar{\lambda}(Q_{\mathrm{xu}}Q_{\mathrm{xu}}^{\mathrm{T}})), \tag{4.16}$$

where δ is a constant of unity order. ■

Remark 4.3. Note that condition (4.16) can be satisfied by picking appropriately the user defined matrices M, $Q_{uu} = R$ without needing the exact values of the submatrices Q_j^i, where $j \in \{\text{xx}, \text{xu}, \text{ux}, \text{uu}\}$. The condition $\alpha_{\text{c}} \gg \alpha_{\text{a}}$ is because in the actor tuning law (4.15), we need to copy the appropriate weights from the critic tuning law (4.14) (see error e_{a}). □

A pseudocode that describes the algorithm has the following form,

Algorithm 13: Q-learning for Optimal Regulation

1: **procedure**
2: Start with initial conditions $x(0)$, and random initial weights $\hat{W}_{\text{c}}(0), \hat{W}_{\text{a}}(0)$, for the critic, and the actor.
3: Propagate $t, x(t)$.
4: Propagate $\hat{W}_{\text{c}}(t), \hat{W}_{\text{a}}(t)$.
5: Compute the Q-function $\hat{Q}$, and the optimal control $\hat{u}$.
6: **end procedure**

4.2.3 Further Reading

More details about model-free optimal regulation are given in Vamvoudakis (2017). In the work of Li *et al.* (2018), a novel off-policy interleaved Q-learning algorithm is presented for solving the optimal control problem of an affine nonlinear discrete-time system, using only the measured data along the system trajectories. Additionally, the authors in Sahoo and Vamvoudakis (2020) extend the results proposed here to construct an "on–off" learning-based scheme, while optimally stabilizing an unknown system. The authors of Devraj *et al.* (2019) present Zap Q-learning that is a recent class of RL algorithms, motivated primarily as a means to accelerate convergence. Finally, the article of Recht (2019) surveys reinforcement learning from the perspective of optimization and control, with a focus on continuous control applications.

4.3 Q-Learning for Nash Games

This subsection will consider N players playing in a Nash game. The N agent/decision makers can be non-cooperative and also can cooperate

in teams. Each of the agents has access to the full state of the system. Since solving Nash game requires complete knowledge of the centralized system dynamics and complicated offline computation, this subsection presents a completely model-free algorithm to solve the coupled Riccati equations arising in multi-player NZS Nash games in deterministic systems. A parametrized Q-function is derived for every of the N-players in the game that depends on the state and the control inputs of all the players. Then, we derive model-free controllers based on the Q-functions parametrization, and then we use $2N$-universal approximators such to approximate the cost and the control of every player, by using namely a critic and an actor-network.

4.3.1 Problem Formulation

Consider the following linear time-invariant CT system,

$$\dot{x}(t) = Ax(t) + \sum_{j=1}^{N} B_j u_j(t), \quad x(0) = x_0,\ t \geqslant 0, \tag{4.17}$$

where $x(t) \in \mathbb{R}^n$ is a measurable state vector, $u_j(t) \in \mathbb{R}^{m_j},\ j \in \mathcal{N} := \{1, \ldots, N\}$ is each control input (or player), and $A \in \mathbb{R}^{n\times n}$, $B_j \in \mathbb{R}^{n\times m_j},\ j \in \mathcal{N}$, are the plant and input matrices, respectively, that will be considered uncertain/unknown. It is assumed that the unknown pairs $(A, B_j),\ \forall j \in \mathcal{N}$ are controllable. The ultimate goal is to find the optimal value functions $V_i^\star,\ \forall i \in \mathcal{N}$ defined by,

$$V_i^\star(x(t)) := \min_{u_i} \int_t^\infty \frac{1}{2}\left(x^{\mathrm{T}} H_i x + \sum_{j=1}^{N} u_j^{\mathrm{T}} R_{ij} u_j \right) \mathrm{d}\tau, \quad t \geqslant 0, \tag{4.18}$$

with user defined matrices $H_i \geq 0$, $R_{ij} \succ 0,\ \forall i, j \in \mathcal{N}$ of appropriate dimensions and $(\sqrt{H_i}, A),\ \forall i \in \mathcal{N}$ are detectable, and without any information of the system matrices A and $B_i, \forall i \in \mathcal{N}$ that appear in (4.17).

The value functions need to be parametrized as a function of the state x and the controls $u_i,\ \forall i \in \mathcal{N}$ to represent the Q-function for each player in the game. The optimal value given by (4.18) after adding

the Hamiltonian can be written as the following Q-function or action-dependent value $Q_i(x, u_i, u_{-i})$: $\mathbb{R}^{n+\sum_{j=1}^{N} m_j} \to \mathbb{R}$,

$$\begin{aligned} Q_i(x, u_i, u_{-i}) := V_i^\star(x) &+ \frac{1}{2} x^{\mathrm{T}} P_i \left(Ax + \sum_{j=1}^{N} B_j u_j \right) \\ &+ \frac{1}{2} \left(Ax + \sum_{j=1}^{N} B_j u_j \right)^{\mathrm{T}} P_i x + \frac{1}{2} \sum_{j=1}^{N} u_j^{\mathrm{T}} R_{ij} u_j \\ &+ \frac{1}{2} x^{\mathrm{T}} H_i x, \quad \forall x, u_i, u_{-i}, \ \forall i \in \mathcal{N}, \end{aligned}$$

where the optimal cost is $V_i^\star(x) = x^{\mathrm{T}} P_i x$ with $P_i > 0$. A model-free optimal control formulation, can be found by solving $\frac{\partial Q_i(\cdot)}{\partial u_i} = 0$.

Since we do not know the optimal values, we should estimate $Q_i^\star$ and $u_i^\star$ with the following actual values for the critic with $\hat{W}_{ic} := \frac{1}{2}\mathrm{vech}(\hat{\bar{Q}}^i)$,

$$\hat{Q}_i(x, u_i, u_{-i}) = \hat{W}_{ic}^{\mathrm{T}} (U_i \otimes U_i), \quad \forall i \in \mathcal{N}, \tag{4.19}$$

where $\hat{W}_{ic} \in \mathbb{R}^{\frac{1}{2}(n+\sum_{j=1}^{N} m_j)(n+\sum_{j=1}^{N} m_j+1)}$ are the estimated critic weights and $U_i := \begin{bmatrix} x^{\mathrm{T}} & u_i^{\mathrm{T}} & u_1^{\mathrm{T}} & \cdots & u_{i-1}^{\mathrm{T}} & u_{i+1}^{\mathrm{T}} & \cdots & u_N^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$, and for the actor

$$\hat{u}_i(x) = \hat{W}_{ia}^{\mathrm{T}} x, \quad \forall i \in \mathcal{N}, \tag{4.20}$$

where $\hat{W}_{ia} \in \mathbb{R}^{n \times m_i}$ are the estimated actor weights, note also that the state x is serving as an activation function for the action network.

Remark 4.4. It is worth noting that the critic and the actor NNs given by (4.19) and (4.20), respectively, for every player, do not have any approximation errors since we have a complete basis representation with respect to the state x and the controls $u_i, \forall i \in N$ in the form $(U_i \otimes U_i)$. For that reason, we do not employ approximation in a compact set but in the whole space. □

The work of Vrabie *et al.* (2013) showed that by using integral RL we can write, the value function (4.18) for each player $i \in \mathcal{N}$ as the following Bellman equation,

$$V_i^\star(x(t)) = V_i^\star(x(t-T)) - \frac{1}{2} \int_{t-T}^{T} \left(x^{\mathrm{T}} H_i x + \sum_{j=1}^{N} u_j^{\mathrm{T}} R_{ij} u_j \right) \mathrm{d}\tau, \tag{4.21}$$

where $T \in \mathbb{R}^+$ a small fixed-time sampling interval. It holds that the value of (4.21) is the same with the action-dependent value of $Q_i^\star\left(x, u_i^\star, u_{-i}^\star\right)$, thus we can write (4.21) as,

$$Q_i^\star(x(t), u_i^\star(t), u_{-i}^\star(t)) = Q^\star(x(t-T), u_i^\star(t-T), u_{-i}^\star(t-T)) - \frac{1}{2}\int_{t-T}^{T}\left(x^{\mathrm{T}} H_i x + \sum_{j=1}^{N} u_j^{\mathrm{T}} R_{ij} u_j\right)\mathrm{d}\tau, \quad \forall i \in \mathcal{N}, \tag{4.22}$$

where we have substituted the value functions $V_i^\star(x(t))$ and $V_i^\star(x(t-T))$ from (4.21) with the action-dependent Q-functions $Q_i^\star(x(t), u_i^\star(t), u_{-i}^\star(t))$ and $Q_i^\star(x(t-T), u_i^\star(t-T), u_{-i}^\star(t-T))$ respectively, see Vamvoudakis (2015). In order to find the update law for the critic of each player $i \in \mathcal{N}$ we shall define the following error $e_i \in \mathbb{R}$ by rewriting (4.22) after using actual values for each player's Q-function as,

$$\begin{aligned} e_i &= \hat{Q}_i\left(x(t), \hat{u}_i(t), \hat{u}_{-i}(t)\right) - \hat{Q}_i\left(x(t-T), \hat{u}_i(t-T), \hat{u}_{-i}(t-T)\right) \\ &\quad + \frac{1}{2}\int_{t-T}^{T}\left(x^{\mathrm{T}} H_i x + \sum_{j=1}^{N} \hat{u}_j^{\mathrm{T}} R_{ij} \hat{u}_j\right)\mathrm{d}\tau \\ &= \hat{W}_{ic}^{\mathrm{T}}\left(U_i(t) \otimes U_i(t)\right) + \frac{1}{2}\int_{t-T}^{T}\left(x^{\mathrm{T}} H_i x + \sum_{j=1}^{N} \hat{u}_j^{\mathrm{T}} R_{ij} \hat{u}_j\right)\mathrm{d}\tau \\ &\quad - \hat{W}_{ic}^{\mathrm{T}}\left(U_i(t-T) \otimes U_i(t-T)\right), \quad \forall i \in \mathcal{N}, \end{aligned}$$

that we would like to eventually drive to zero by picking appropriately $\hat{W}_{ic}, \forall i \in \mathcal{N}$. Now for each player's actor NN we can define the error $e_{ia} \in \mathbb{R}^{m_i}$ as $e_{ia} := \hat{W}_{ia}^{\mathrm{T}} x + (\hat{Q}^i_{u_i u_i})^{-1} \hat{Q}^i_{u_i x} x, \forall i \in \mathcal{N}$, where the values of $(\hat{Q}^i_{u_i u_i})^{-1}$ and $\hat{Q}^i_{u_i x}$ are going to be extracted from the vector $\hat{W}_{ic}$. Now we shall find tuning updates for $\hat{W}_{ic}$ and $\hat{W}_{ia}$ such that e_i and e_{ia} go to zero asymptotically $\forall i \in \mathcal{N}$. By following adaptive control techniques (Ioannou and Fidan, 2006) we can define the squared-norm of errors of each player $i \in \mathcal{N}, e_i$ and e_{ia} as,

$$K_{i1} = \frac{1}{2}\left\|e_i\right\|^2, \quad \forall i \in \mathcal{N}, \tag{4.23}$$

and

$$K_{i2} = \frac{1}{2}\left\|e_{ia}\right\|^2, \quad \forall i \in \mathcal{N}. \tag{4.24}$$

4.3.2 Learning Algorithm

The gradient descent estimate of $\hat{W}_{ic}$ for the critic weights of each player can be constructed by differentiating (4.23) to yield,

$$\dot{\hat{W}}_{ic} = -\alpha_{ic}\frac{\partial K_{i1}}{\partial \hat{W}_{ic}} = -\alpha_{ic}\frac{v_i}{\left(1+v_i^{\mathrm{T}}v_i\right)^2}e_i^{\mathrm{T}}, \quad \forall i \in \mathcal{N}, \tag{4.25}$$

where $v_i := (U_i(t) \otimes U_i(t) - U_i(t-T) \otimes U_i(t-T))$ and $\alpha_{ic} \in \mathbb{R}^+$ is a constant gain that determines the speed of convergence. The gradient descent estimate of $\hat{W}_{\mathrm{a}}$ for the actor weights can be constructed by differentiating (4.24) as,

$$\dot{\hat{W}}_{ia} = -\alpha_{ia}\frac{\partial K_{i2}}{\partial \hat{W}_{ia}} = -\alpha_{ia}xe_{ia}^{\mathrm{T}}, \quad \forall i \in \mathcal{N}, \tag{4.26}$$

where $\alpha_{ia} \in \mathbb{R}^+$ is a constant gain that determines the speed of convergence.

Theorem 4.2 (Vamvoudakis, 2015, Thm. 2)**.** Consider the system dynamics given by (4.17), the critic NN for each player $i \in \mathcal{N}$ given by (4.19) and the optimal control for each player $i \in \mathcal{N}$ given by (4.20). Assume that the signal $\Delta := \frac{v_i}{1+v_i^{\mathrm{T}}v_i}$ is PE. The tuning law for the weights of the critic is given by (4.25) and for the weights of the actor is given by (4.26). Then, the equilibrium point of the closed loop system is asymptotically stable given that the tuning gain for the critic α_{ic} is sufficiently larger than the tuning gain for the actor $\alpha_{ia}, \forall i \in \mathcal{N}$ and,

$$1 < \alpha_{ia} < \frac{1}{\delta\bar{\lambda}\left(R_{ii}^{-1}\right)}\left(2\underline{\lambda}\left(H_i + \sum_{j=1}^{N} Q_{\mathrm{xu_j}}^j R_{jj}^{-1}R_{ij}R_{jj}^{-1}(Q_{\mathrm{xu_j}}^j)^{\mathrm{T}}\right)\right.$$
$$\left. -\bar{\lambda}\left(\sum_{j=1}^{N} Q_{\mathrm{xu_j}}^i(Q_{\mathrm{xu_j}}^i)^{\mathrm{T}}\right)\right), \tag{4.27}$$

where $\delta \in (0,1)$. ■

Remark 4.5. The necessary conditions for asymptotic stability given in (4.27) can be satisfied by picking appropriately the user defined matrices H_i, R_{ii}, R_{ij}, $\forall i, j \in \mathcal{N}$ and the tuning gains $\alpha_{ic} \gg \alpha_{ia}, \forall i \in \mathcal{N}$ without needing the exact values of the sub-matrices Q_j^i, where $j \in \{\mathrm{xx}, \mathrm{xu}, \mathrm{ux}, \mathrm{uu}\}$. □

A pseudocode that describes the algorithm has the following form,

Algorithm 14: Nash Q-learning

procedure
    Start with initial conditions for every agent $x(0)$, and random initial weights $\hat{W}_{ic}(0), \hat{W}_{ia}(0)$ for the critic and actor approximators.
    Propagate $t, x_(t)$.
    Propagate $\hat{W}_{ic}(t), \hat{W}_{ia}(t)$.
    Compute the Q-function $\hat{Q}_i$, and the control $\hat{u}_i$.
end procedure

4.3.3 Further Reading

More details about model-free Nash games are given in Vamvoudakis (2015).

4.4 Q-Learning for Multi-Agent Systems

A cooperative Q-learning approach is now developed to enable the agents in large networks to synchronize to the behavior of an unknown leader by each optimizing a distributed performance criterion that depends only on a subset of the agents in the network. The novel distributed Q-functions are parametrized as functions of the tracking error, control and adversarial inputs in the neighborhood. In the developed approach, the agents coordinate with the neighbors in order to pick their minimizing model-free policies in such a way to guarantee convergence to a graphical Nash equilibrium and also attenuation of maximizing worst-case adversarial inputs. A structure of 2-actors and a single critic approximators are used for each agent in the network. The 2-actors are used to approximate the optimal control input and the worst-case adversarial input, while the critic approximator is used to approximate the optimal cost of each of the coupled optimizations. Effective tuning laws are proposed to solve the model-free cooperative game problem while also guaranteeing closed-loop stability of the equilibrium point.

4.4.1 Problem Formulation

Consider a networked-system $\mathcal{G}$, consisting of N agents each modeled $\forall i \in \mathcal{N} := \{1, \ldots, N\}$ by the following dynamics,

$$\dot{x}_i(t) = Ax_i(t) + B_i u_i(t) + D_i v_i(t),\ x_i(0) = x_{i0}, \quad t \geqslant 0,$$

where $x_i(t) \in \mathbb{R}^n$ is a measurable state vector available for feedback by each agent and known initial conditions $x_i(0)$, $u_i(t) \in \mathbb{R}^{m_i},\ i \in \mathcal{N} := \{1, \ldots, N\}$ is each control input (or minimizing player), $v_i(t) \in \mathbb{R}^{l_i},\ i \in \mathcal{N} := \{1, \ldots, N\}$ is each adversarial input (or maximizing player), and $A \in \mathbb{R}^{n\times n}$, $B_i \in \mathbb{R}^{n\times m_i}$, $D_i \in \mathbb{R}^{n\times l_i},\ i \in \mathcal{N}$ are the plant, control input and adversarial input matrices, respectively, that will be considered uncertain/unknown. It is assumed that the unknown pairs $(A, B_i),\ \forall i \in \mathcal{N}$ are stabilizable. We have a total of $2N$ players/controllers that select values for $u_i(t),\ t \geqslant 0,\ i \in \mathcal{N}$ and $v_i(t),\ t \geqslant 0,\ i \in \mathcal{N}$. The agents in the network seek to cooperatively asymptotically track the state of a leader node/exosystem with dynamics $\dot{x}_L = Ax_L$, i.e., $x_i(t) - x_L(t) \to 0, \forall i \in \mathcal{N}$ while *simultaneously satisfying user-defined distributed performances*.

Now we shall proceed to the design of the user-defined *distributed performances*. For that reason, we shall define the following *neighborhood tracking error* for every agent,

$$e_i := \sum_{j\in\mathcal{N}_i} a_{ij}(x_i - x_j) + g_i(x_i - x_L), \quad \forall i \in \mathcal{N}, \tag{4.28}$$

where $g_i \in \mathbb{R}_+$ is the pinning gain that shows if an agent is pinned to the leader node (i.e., $g_i \neq 0$) and it is nonzero for at least one node.

The dynamics of (4.28) are given by,

$$\dot{e}_i = Ae_i + (d_i + g_i)(B_i u_i + D_i v_i) - \sum_{j\in\mathcal{N}_i} a_{ij}(B_j u_j + D_j v_j), \quad \forall i \in \mathcal{N}, \tag{4.29}$$

with $e_i \in \mathbb{R}^n$.

The cost functionals associated to each agent $i \in \mathcal{N}$, that depend on the tracking error e_i, the control u_i, the controls in the neighborhood of agent i given as, $u_{\mathcal{N}_i} := \{u_j : j \in \mathcal{N}_i\}$, the adversarial input v_i and the adversarial inputs in the neighborhood of agent i given as

$v_{\mathcal{N}_i} := \{v_j : j \in \mathcal{N}_i\}$, have the following form:

$$\mathcal{J}_i(e_i(0); u_i, u_{\mathcal{N}_i}, v_i, v_{\mathcal{N}_i}) = \frac{1}{2}\int_0^\infty (e_i^{\mathrm{T}} H_i e_i + (u_i^{\mathrm{T}} R_{ii} u_i - \gamma_{ii}^2 v_i^{\mathrm{T}} v_i)) + \sum_{j\in N_i} (u_j^{\mathrm{T}} R_{ij} u_j - \gamma_{ij}^2 v_j^{\mathrm{T}} v_j)\mathrm{d}t, \quad \forall i \in \mathcal{N},$$

with matrices $H_i \succeq 0, R_{ii} \succ 0, R_{ij} \succeq 0, \forall i, j \in \mathcal{N}$ of appropriate dimensions, $\gamma_{ii}, \gamma_{ij} \in \mathbb{R}^+, \forall i \in \mathcal{N}$ and $\left(\sqrt{H}_i, A\right), \forall i \in \mathcal{N}$ are detectable.

Hence, given a strongly connected graph $\mathcal{G}$, we are interested in finding a graphical Nash equilibrium (Başar and Olsder, 1999; Vamvoudakis *et al.*, 2012), that is translated to a saddle point $u_i^\star, v_i^\star$ for every agent $i \in \mathcal{N}$ in the sense that,

$$\begin{aligned}\mathcal{J}_i(e_i(0); u_i^\star, u_{\mathcal{N}_i}^\star, v_i, v_{\mathcal{N}_i}^\star) &\leqslant \mathcal{J}_i(e_i(0); u_i^\star, u_{\mathcal{N}_i}^\star, v_i^\star, v_{\mathcal{N}_i}^\star) \\ &\leqslant \mathcal{J}_i(e_i(0); u_i, u_{N_i}^\star, v_i^\star, v_{\mathcal{N}_i}^\star), \quad \forall u_i, v_i, \forall i \in \mathcal{N}.\end{aligned}$$

This can be expressed by the following coupled distributed optimization problems $\forall e_i(0)$:

$$\begin{aligned}\mathcal{J}_i(e_i(0); u_i^\star, u_{\mathcal{N}_i}^\star, v_i^\star, v_{\mathcal{N}_i}^\star) &= \min_{u_i} \mathcal{J}_i(e_i(0); u_i, u_{\mathcal{N}_i}^\star, v_i^\star, v_{\mathcal{N}_i}^\star) \\ &= \max_{v_i} \mathcal{J}_i(e_i(0); u_i^\star, u_{\mathcal{N}_i}^\star, v_i, v_{\mathcal{N}_i}^\star),\end{aligned}$$

given the dynamics in (4.28).

Thus, the ultimate goal is to find the distributed optimal value functions $V_i^\star, \ \forall i \in \mathcal{N}$ defined by,

$$V_i^\star(e_i(t)) := \min_{u_i}\max_{v_i} \int_t^\infty \frac{1}{2}(e_i^{\mathrm{T}} H_i e_i + (u_i^{\mathrm{T}} R_{ii} u_i - \gamma_{ii}^2 v_i^{\mathrm{T}} v_i) + \sum_{j\in\mathcal{N}_i} (u_j^{\mathrm{T}} R_{ij} u_j - \gamma_{ij}^2 v_j^{\mathrm{T}} v_j))\mathrm{d}\tau, \quad t \geqslant 0, \forall e_i, i \in \mathcal{N}, \quad (4.30)$$

given (4.29), but without any information of the system matrices A, B_i, $D_i, \forall i \in \mathcal{N}$ and pinning gains $g_i, \ \forall i \in \mathcal{N}$.

Remark 4.6. Note that the assumptions that the pairs $(A, B_i), \forall i \in \mathcal{N}$ are stabilizable and the pairs $\left(\sqrt{H}_i, A\right), \forall i \in \mathcal{N}$ are detectable will guarantee that the coupled AREs will have a unique non-negative solution (Başar and Bernhard, 2008). □

The value functions need to be parametrized as functions of the neighborhood tracking error e_i, the controls u_i and $u_{\mathcal{N}_i}$ and the adversarial inputs v_i and $v_{\mathcal{N}_i}$ to represent the distributed Q-function, i.e., sparse cooperative learning, for each agent in the game. The optimal value given by (4.30) after adding the Hamiltonian can be written as the following distributed Q-function or *action-dependent value* $Q_i(e_i, u_i, u_{\mathcal{N}_i}, v_i, v_{\mathcal{N}_i})$: $\mathbb{R}^{n+m_i+l_i+\sum_{j\in\mathcal{N}_i}(m_j+l_j)} \to \mathbb{R}$,

$$\begin{aligned} Q_i(e_i, u_i, u_{\mathcal{N}_i}, v_i, v_{\mathcal{N}_i}) := V_i^\star(e_i) + \frac{\partial V_i^\star}{\partial e_i}^{\mathrm{T}} \Bigg(& Ae_i + (d_i + g_i)(B_i u_i + D_i v_i) \\ & - \sum_{j\in\mathcal{N}_i} a_{ij}(B_j u_j + D_j v_j) \Bigg) \\ & + \frac{1}{2}\Bigg(e_i^{\mathrm{T}} H_i e_i + (u_i^{\mathrm{T}} R_{ii} u_i - \gamma_{ii}^2 v_i^{\mathrm{T}} v_i) \\ & + \sum_{j\in\mathcal{N}_i} (u_j^{\mathrm{T}} R_{ij} u_j - \gamma_{ij}^2 v_j^{\mathrm{T}} v_j) \Bigg), \\ & \forall e_i, u_i, u_{\mathcal{N}_i}, v_i, v_{\mathcal{N}_i}, \forall i \in \mathcal{N}, \end{aligned}$$

where the optimal cost is $V_i^\star(e_i) = \frac{1}{2} e_i^{\mathrm{T}} P_i e_i, \forall i \in \mathcal{N}$.

A model-free formulation, can be found by solving $\frac{\partial Q_i(\cdot)}{\partial u_i} = 0$ to write,

$$\begin{aligned} u_i^\star(e_i) &= \arg\min_{u_i} Q_i(e_i, u_i, u_{\mathcal{N}_i}, v_i, v_{\mathcal{N}_i}) \\ &= -(Q^i_{u_i u_i})^{-1} Q^i_{u_i e_i} e_i, \quad \forall i \in \mathcal{N}, \end{aligned} \tag{4.31}$$

and $\frac{\partial Q_i(\cdot)}{\partial v_i} = 0$ to write,

$$\begin{aligned} v_i^\star(e_i) &= \arg\max_{v_i} Q_i(e_i, u_i, u_{\mathcal{N}_i}, v_i, v_{\mathcal{N}_i}) \\ &= -\left(Q^i_{v_i v_i}\right)^{-1} Q^i_{v_i e_i} e_i, \quad \forall i \in \mathcal{N}. \end{aligned} \tag{4.32}$$

The critic approximator is given by,

$$\hat{Q}_i(e_i, u_i, u_{\mathcal{N}_i}, v_i, v_{\mathcal{N}_i}) = \hat{W}_{ic}^{\mathrm{T}}(U_i \otimes U_i), \quad \forall i \in \mathcal{N}, \tag{4.33}$$

where $\hat{W}_{ic}$ are estimated critic weights of the optimal weights and $U_i := \begin{bmatrix} e_i^{\mathrm{T}} & u_i^{\mathrm{T}} & u_{\mathcal{N}_i}^{\mathrm{T}} & v_i^{\mathrm{T}} & v_{\mathcal{N}_i}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$; the control actor approximator is

given as,

$$\hat{u}_i(e_i) = \hat{W}_{ia}^{\mathrm{T}} e_i, \quad \forall i \in \mathcal{N}, \tag{4.34}$$

where $\hat{W}_{ia} \in \mathbb{R}^{n \times m_i}$ are estimated actor weights of the optimal weights; and the adversarial actor approximator is given by,

$$\hat{v}_i(e_i) = \hat{W}_{id}^{\mathrm{T}} e_i, \quad \forall i \in \mathcal{N}, \tag{4.35}$$

where $\hat{W}_{id} \in \mathbb{R}^{n \times l_i}$ are estimated actor weights of the optimal weights. We have showed in Vrabie *et al.* (2013) that the value function (4.30) for each agent satisfies the following Bellman equation:

$$V_i^\star(e_i(t)) = V_i^\star(e_i(t-T)) - \frac{1}{2}\int_{t-T}^{T} \Bigg(e_i^{\mathrm{T}} H_i e_i + (u_i^{\star\mathrm{T}} R_{ii} u_i^\star - \gamma_{ii}^2 v_i^{\star\mathrm{T}} v_i^\star) + \sum_{j \in \mathcal{N}_i} (u_j^{\star\mathrm{T}} R_{ij} u_j^\star - \gamma_{ij}^2 v_j^{\star\mathrm{T}} v_j^\star) \Bigg) \mathrm{d}\tau, \quad \forall i \in \mathcal{N}, \tag{4.36}$$

where $T \in \mathbb{R}^+$ is a small fixed time interval that defines how fast the agent measures her own state and the neighboring states.

Alternatively, we can write (4.36) in terms of the distributed Q_i function of each agent,

$$\begin{aligned} &Q_i^\star(e_i(t), u_i^\star(t), u_{N_i}^\star(t), v_i^\star(t), v_{\mathcal{N}_i}^\star(t)) \\ &= Q^\star(e_i(t-T), u_i^\star(t-T), u_{\mathcal{N}_i}^\star(t-T), v_i^\star(t-T), v_{\mathcal{N}_i}^\star(t-T)) \\ &\quad - \frac{1}{2}\int_{t-T}^{T} \Bigg(e_i^{\mathrm{T}} H_i e_i + (u_i^{\star\mathrm{T}} R_{ii} u_i^\star - \gamma_{ii}^2 v_i^{\star\mathrm{T}} v_i^\star). \\ &\quad + \sum_{j \in \mathcal{N}_i} (u_j^{\star\mathrm{T}} R_{ij} u_j^\star - \gamma_{ij}^2 v_j^{\star\mathrm{T}} v_j^\star) \Bigg) \mathrm{d}\tau \quad \forall i \in \mathcal{N}. \end{aligned} \tag{4.37}$$

Remark 4.7. Note that although the value of the small fixed time interval T does not affect in any way, the convergence property of the algorithm, it is related to the excitation condition necessary in the setup of a numerically posed least squares problem and the least squares solution. For more details, please see Vamvoudakis *et al.* (2014). □

4.4.2 Cooperative Learning Solution

To find the update law for the critic approximator of each agent $i \in \mathcal{N}$, we define the following error $E_i \in \mathbb{R}$ that we would like to eventually

drive to zero by tuning $\hat{W}_{ic}$, $\forall i \in \mathcal{N}$ appropriately. This error is defined by the difference between the left- and the right-hand sides of (4.37) with the actual weights replaced by $\hat{W}_{ic}$,

$$\begin{aligned} E_i := \hat{W}_{ic}^{\mathrm{T}}(U_i(t) \otimes U_i(t)) + \frac{1}{2}\int_{t-T}^{T} \Big(e_i^{\mathrm{T}} H_i e_i + (\hat{u}_i^{\mathrm{T}} R_{ii} \hat{u}_i \\ - \gamma_{ii}^2 \hat{v}_i^{\mathrm{T}} \hat{v}_i) + \sum_{j \in \mathcal{N}_i} (\hat{u}_j^{\mathrm{T}} R_{ij} \hat{u}_j - \gamma_{ij}^2 \hat{v}_j^{\mathrm{T}} \hat{v}_j) \Big) \mathrm{d}\tau \\ - \hat{W}_{ic}^{\mathrm{T}}(U_i(t-T) \otimes U_i(t-T)), \quad \forall i \in \mathcal{N}. \end{aligned}$$

For each agent's control actor approximator, we can define the error $E_{ia} \in \mathbb{R}^{m_i}$ as the difference between the control (4.34) that uses the estimated weights and the optimal control (4.31) with the matrices $Q^i_{u_i u_i}$, and $Q^i_{u_i e_i}$ replaced by the critic estimates extracted from,

$$E_{ia} := \hat{W}_{ia}^{\mathrm{T}} e_i + (\hat{Q}^i_{u_i u_i})^{-1} \hat{Q}^i_{u_i e_i} e_i, \quad \forall i \in \mathcal{N}.$$

Similarly, for each agent's adversarial actor approximator, we can define the error $E_{id} \in \mathbb{R}^{l_i}$ as the difference between the control (4.35) that uses the estimated weights and the optimal control (4.32) with the matrices $Q^i_{v_i v_i}$, and $Q^i_{v_i e_i}$ replaced by the critic estimates extracted from $\hat{W}_{ic}$,

$$E_{id} := \hat{W}_{id}^{\mathrm{T}} e_i + (\hat{Q}^i_{v_i v_i})^{-1} \hat{Q}^i_{v_i e_i} e_i, \quad \forall i \in \mathcal{N}.$$

To find tuning updates for $\hat{W}_{ic}, \hat{W}_{ia} \in \mathbb{R}^{n \times m_i}$, and $\hat{W}_{id} \in \mathbb{R}^{n \times l_i}$ such that the errors $E_i \in \mathbb{R}, E_{ia} \in \mathbb{R}^{m_i}$ and $E_{id} \in \mathbb{R}_i^{l_i}$ go to zero $\forall i \in \mathcal{N}$, we follow adaptive control techniques as in Ioannou and Fidan (2006) and define the squared norm of errors of each agent $i \in \mathcal{N}$, E_i, E_{ia}, and E_{id} as,

$$K_{i1} = \frac{1}{2} \|E_i\|^2, \quad \forall i \in \mathcal{N}, \tag{4.38}$$

$$K_{i2} = \frac{1}{2} \|E_{ia}\|^2, \quad \forall i \in \mathcal{N}, \tag{4.39}$$

and

$$K_{ie} = \frac{1}{2} \|E_{id}\|^2, \quad \forall i \in \mathcal{N}. \tag{4.40}$$

The gradient descent estimate of $\hat{W}_{ic}$ for the critic weights of each agent can be constructed by differentiating (4.38) to yield,

$$\begin{aligned}\dot{\hat{W}}_{ic} &= -\alpha_{ic}\frac{\partial K_{i1}}{\partial \hat{W}_{ic}} \\ &= -\alpha_{ic}\frac{\nu_i E_i}{(1+(U_i(t)\otimes U_i(t-T)\otimes U_i(t-T))^{\mathrm{T}}\nu_i)^2}, \quad \forall i \in \mathcal{N},\end{aligned} \tag{4.41}$$

where $\nu_i = U_i(t)\otimes U_i(t) - U_i(t-T)\otimes U_i(t-T)$ and $\alpha_{ic} \in \mathbb{R}^+$ is a constant gain that determines the speed of the critic approximator convergence.

Similarly, by using (4.39), the gradient descent estimate of $\hat{W}_{ia}$ for the control actor weights can be constructed as,

$$\dot{\hat{W}}_{ia} = -\alpha_{ia}\frac{\partial K_{i2}}{\partial \hat{W}_{ia}} = -\alpha_{ia} e_i E_{ia}^{\mathrm{T}} \quad \forall i \in \mathcal{N}, \tag{4.42}$$

where $\alpha_{ia} \in \mathbb{R}^+$ is a constant gain that determines the speed of the actor approximator convergence, and finally by using (4.40) the gradient descent estimate of $\hat{W}_{id}$ for the adversarial actor weights can be constructed as,

$$\dot{\hat{W}}_{id} = -\alpha_{id}\frac{\partial K_{i2}}{\partial \hat{W}_{id}} = -\alpha_{id} e_i E_{id}^{\mathrm{T}} \quad \forall i \in \mathcal{N},$$

where $\alpha_{id} \in \mathbb{R}^+$ is a constant gain that determines the speed of the actor approximator convergence.

Theorem 4.3 (Vamvoudakis and Hespanha, 2018, Thm. 2)**.** Assume that the agents are strongly connected in the networked system $\mathcal{G}$. Consider the neighborhood tracking error dynamics given by (4.29), the critic approximator for each agent $i \in \mathcal{N}$ given by (4.33), the optimal control for each agent $i \in \mathcal{N}$ given by (4.34), the worst case adversarial input for each agent $i \in \mathcal{N}$ given by (4.35), the tuning law for the weights of the critic given by (4.41), for the weights of the control actor given by (4.42), and for the weights of the adversarial actor given by (4.35). Assume that the tuning laws for the critic are faster than the tuning

laws for the control and adversarial actors, and

$$\alpha_{ia} > 1; \quad \alpha_{id} > 1, \tag{4.43}$$

$$\alpha_{ia}\bar{\lambda}(R_{ii}^{-1}) + \frac{\alpha_{id}}{\gamma_{ii}^2} < \frac{2}{\delta_i}\Bigg(\underline{\lambda}(H_i + Q^i_{e_iu_i}R_{ii}^{-1}(Q^i_{e_iu_i})^{\mathrm{T}}) - \frac{1}{2}\bar{\lambda}\left(Q^i_{e_1u_i}(Q^i_{e_iu_i})^{\mathrm{T}} + \frac{2}{\gamma_{ii}^2}Q^i_{e_iv_i}(Q^i_{e_iv_i})^{\mathrm{T}}\right)\Bigg) \tag{4.44}$$

$$\alpha_{ja}\bar{\lambda}(R_{jj}^{-1}) + \frac{\alpha_{jd}}{\gamma_{jj}^2} < \frac{2}{\delta_j}\underline{\lambda}(Q^j_{e_iu_j}R_{jj}^{-1}R_{ij}R_{jj}^{-1}(Q^j_{e,ju_j})^{\mathrm{T}}) - \frac{1}{2}\bar{\lambda}\left(Q^i_{e_iu_j}(Q^i_{e_iu_j})^{\mathrm{T}} + \frac{\gamma_{ij}^2}{\gamma_{jj}^4}Q^i_{e_iv_j}(Q^i_{e_iv_j})^{\mathrm{T}}\right), \quad \forall j \in \mathcal{N}_i, \tag{4.45}$$

where $\delta_i \in (0,1)$. Then, the equilibrium point of the closed loop system is globally asymptotically stable. ■

Remark 4.8. In order to satisfy the conditions (4.43)–(4.45), we have to pick the tuning gains $\alpha_{ia}, \alpha_{id}, \forall i \in \mathcal{N}$, for each control and adversarial actor such that all three inequalities are met simultaneously. □

Remark 4.9. The necessary conditions given in (4.44) and (4.45) can be satisfied by picking appropriately the user defined matrices H_i, R_{ii}, R_{ij}, $\gamma_{ii}, \gamma_{ij}, \forall i, j \in \mathcal{N}$ and the tuning gains $\alpha_{ic} \gg -\alpha_{ia}$, $\alpha_{ic} \gg \alpha_{id}$, $\forall i \in \mathcal{N}$, without needing to know the exact values of the submatrices $Q^i_{(.)}$. Regarding the tuning gains conditions, as noted in adaptive control (Ioannou and Fidan, 2006), large adaptive gains can cause high-frequency oscillations in the control signal and reduced tolerance to time-delays that will destabilize the system. There are not any systematic approaches to pick a satisfactory adaptation gain, hence, trial and error, intuition or Monte Carlo simulations can serve as guidelines. □

Remark 4.10. In our algorithm, there are no offline computations, neither requirements for storage of any data in the memory, but everything happens as in a plug-n-play framework for every agent by measuring the system trajectories in the neighborhood. The size of the network N increases the complexity since we will have more distributed optimization criteria to optimize, but on the other side the position of leader(s) does not affect the convergence as long as the graph is strongly connected. □

A pseudocode that describes the algorithm has the following form,

Algorithm 15: Multi-Agent Q-learning

procedure
    Start with initial conditions for every agent $x_i(0)$, and random initial weights $\hat{W}_{ic}(0), \hat{W}_{ia}(0), \hat{W}_{id}(0)$ for the critic, actor and worst case adversarial approximator for each agent.
    Propagate $t, x_i(t)$ and $x_j(t)$ in the neighborhood to compute e_i.
    Propagate $\hat{W}_{ic}(t), \hat{W}_{ia}(t), \hat{W}_{id}(t)$.
    Compute the Q-function $\hat{Q}_i$, the control $\hat{u}_i$, and the adversarial input $\hat{v}_i$ for each agent.
end procedure

4.4.3 Further Reading

More details about multi-agent model-free learning can be found in Vamvoudakis and Hespanha (2018).

5

Model-Based and Model-Free Intermittent RL

Event-triggered control is an emerging control strategy suitable for reducing the traffic of information between sensors and controllers. This is attained by closing the control loop only when a user-designed triggering condition is satisfied, and letting the system evolve in an open loop fashion otherwise. Sparse communication and less computation could result in decongestion of the shared network medium and energy save for devices as long as guarantees of stability and performance are satisfied. In this section, Subsection 5.1 provides the stimulus for developing event-triggered optimal controllers. In the sequel, we demonstrate the state of the art intermittent RL algorithms for nonlinear systems addressing the problems of optimal regulation to the equilibrium point of the system as well as of the desired trajectory optimal tracking in Subsections 5.2 and 5.3, respectively. Finally, in Subsection 5.4, we present a model-free event-triggered optimal control algorithm for CT linear systems.

5.1 Introduction and Motivation

Shared congestion and energy saving objectives demand that every information through a network should be rigorously decided when to

transmit. For that reason, one needs to design "bandwidth" efficient controllers that can function in event-driven environments and update their values only when needed. Bandwidth efficient policy design is a newly developed framework that can potentially enhance the performance of applications that have limited resources and bandwidth and offers a new point of view, with respect to conventional time-driven strategies, on how information could be sampled for control purposes. The bandwidth efficient control algorithms (Heemels *et al.*, 2008; Lemmon, 2010; Tabuada, 2007) are composed of a feedback controller updated based on sampled state, and the event-triggering mechanism that determines when the controller has to be transmitted from a stability and performance point of view. This can reduce the computation and communication resources significantly. All the bandwidth efficient control algorithms available in the literature rely on a combination of offline computations, in the sense of computing the Riccati or HJB equations, and online computations in the sense of updating the controller. Computing and updating controller parameters using online solutions may allow for changing dynamics, e.g., to handle the reduced weight of an aircraft as the fuel burns. All this work has been mostly done for linear systems. For nonlinear systems things are more complicated because of the infeasibility of the HJB equation. For that reason, one needs to combine bandwidth efficient controllers with computational intelligent ideas to solve the complicated HJB equation online by updating the controller only when it is needed but still guaranteeing optimal performance of the original system and not a linearized version of it. To overcome all those limitations we shall use machine learning techniques and specifically an actor/critic NN framework (Lewis *et al.*, 2012b) to find new model-free adaptive learning techniques with formal robustness and optimality guarantees. Most of the ground-breaking work on bandwidth efficient control algorithms (see Donkers and Heemels, 2012; Garcia and Antsaklis, 2013; Heemels *et al.*, 2008; Lemmon, 2010; Tabuada, 2007; Wang and Lemmon, 2011 and the references therein) cannot guarantee optimality, require complete knowledge of the models, and consider mostly linear systems without any self-learning capabilities. On the contrary, the work of Molin and Hirche (2013) proposes two optimal event-based control design under

lossy communication but despite of their computational benefits compared to the optimal solution, it turns out that both algorithms are sub-optimal and their algorithm relies heavily on offline optimization schemes with complete information of the model.

5.2 Optimal Control of Nonlinear Systems

This subsection develops an RL-based event-triggered control algorithm for nonlinear CT systems. The goal is to reduce the controller updates, by sampling the state only when an event is triggered to maintain stability and optimality. The online algorithm is implemented based on an actor/critic NN structure. A critic NN is used to approximate the cost and an actor NN is used to approximate the optimal event-triggered controller. Since in the algorithm proposed there are dynamics that exhibit continuous evolution described by ordinary differential equations and instantaneous jumps or impulses, we will use an impulsive system approach.

5.2.1 Problem Formulation

Consider a nonlinear CT system given by (2.1). In order to save resources, the controller will work with a sampled version of the state. For that reason, one needs to introduce a *sampled-data component* that is characterized by a monotone increasing sequence of sampling instants (broadcast release times) $\{r_j\}_{j=1}^{\infty}$, where r_j is the j-th consecutive sampling instant. The output of the *sampled-data component* is a sequence of sampled states $\hat{x}_j$, where $\hat{x}_j = x(r_j)$ for all $t \in [r_j, r_{j+1})$ and $j \in \mathbb{N}$. The controller maps the sampled state onto a control vector $\hat{u}_j$, which after using a zero-order hold (ZOH) becomes a CT input signal. For simplicity we will assume that the sampled-data systems have zero task delays. In order to decide when to trigger an event we will define the gap or difference between the current state $x(t)$ and the sampled state $\hat{x}_j(t)$ as,

$$e_j(t) := \hat{x}_j(t) - x(t), \quad \forall t \in (r_{j-1}, r_j],$$

and the dynamics of the gap are evolving according to,

$$\dot{e}_j(t) = -\dot{x}(t), \quad t \in (r_{j-1}, r_j], \quad e_j(0) = 0.$$

Remark 5.1. Note that when an event is triggered at $t = r_j$, a new state measurement is rendered that resets the gap e_j to zero. □

We want to find a controller u of the form $u = k(\hat{x}_j(t)) \equiv k(x(t) + e_j(t))$ that minimizes a cost functional similar to the one with the time-triggered controller,

$$J(x(0); u) = \int_0^\infty (u^{\mathrm{T}} u + Q(x))\mathrm{d}\tau, \quad \forall x(0), \tag{5.1}$$

with positive definite function $Q(x)$ on $\mathbb{R}^n$ and with limited updates.

The ultimate goal is to find the optimal cost function $V^\star$ defined by,

$$V^\star(x(t)) := \min_u \int_t^\infty (u^{\mathrm{T}} u + Q(x))\mathrm{d}\tau, \quad t \geqslant 0, \tag{5.2}$$

subject to the constraint (2.1) given an aperiodic event-triggered controller as will be defined in the subsequent analysis.

One can define the Hamiltonian associated with (2.1) and (5.2) for the *time-triggered case* as,

$$H\left(x, u(x), \frac{\partial V^\star(x)}{\partial x}\right) = \frac{\partial V^\star(x)}{\partial x}^{\mathrm{T}} (f(x) + g(x)u(x)) + u(x)^{\mathrm{T}} u(x) + Q(x), \quad \forall x, u. \tag{5.3}$$

Now assume that the controller has unlimited bandwidth. Then one needs to find the control input $u(t)$ such that the performance (5.1) is minimized. Hence we will employ, the stationarity condition (Lewis *et al.*, 2012a) into the Hamiltonian Equation (5.3) and we will have,

$$u^\star(x) = -\frac{1}{2} g(x)^{\mathrm{T}} \frac{\partial V^\star(x)}{\partial x}, \tag{5.4}$$

for the time-triggered case.

The optimal cost and the optimal control satisfy the following HJB equation,

$$H\left(x, u^\star(x), \frac{\partial V^\star(x)}{\partial x}\right) \equiv \frac{\partial V^\star(x)}{\partial x}^{\mathrm{T}}\left(f(x) - \frac{1}{2}g(x)g(x)^{\mathrm{T}}\frac{\partial V^\star(x)}{\partial x}\right) + Q(x) + \frac{1}{4}\frac{\partial V^\star(x)}{\partial x}^{\mathrm{T}} g(x)g(x)^{\mathrm{T}}\frac{\partial V^\star(x)}{\partial x} = 0, \quad \forall x. \tag{5.5}$$

From now on we will call the HJB (5.5) as time-triggered HJB equation.

In order to reduce the communication between the controller and the plant, one needs to use an event-triggered version of the above HJB Equation (5.5) by introducing a *sampled-data component* with aperiodic controller updates that ensure a certain condition on the state of the plant to guarantee stability and performance as we will see in the subsequent analysis. For that reason, the control input uses the sampled-state information instead of the true one and hence (5.4) becomes,

$$u^\star(\hat{x}_j) = -\frac{1}{2}g(\hat{x}_j)^{\mathrm{T}}\frac{\partial V^\star(\hat{x}_j)}{\partial x}, \quad \text{for } t \in (r_{j-1}, r_j] \text{ and } j \in \mathbb{N}. \tag{5.6}$$

By using the event-triggered controller given by (5.6), the HJB Equation (5.5) becomes $\forall x, \hat{x}_j \in \mathbb{R}^n$,

$$H\left(x, u^\star(\hat{x}_j), \frac{\partial V^\star(x)}{\partial x}\right) = \frac{\partial V^\star(x)}{\partial x}^{\mathrm{T}}\left(f(x) - \frac{1}{2}g(x)g(\hat{x}_j)^{\mathrm{T}}\frac{\partial V^\star(\hat{x}_j)}{\partial x}\right) + \frac{1}{4}\frac{\partial V^\star(\hat{x}_j)}{\partial x}^{\mathrm{T}} g(\hat{x}_j)g(\hat{x}_j)^{\mathrm{T}}\frac{\partial V^\star(\hat{x}_j)}{\partial x} + Q(x), \tag{5.7}$$

which is eventually the equation we would like to quantify and compare it to (5.5).

Solving the event-triggered HJB Equation (5.7) for the optimal cost for nonlinear systems is in most of the cases infeasible and has to be done in an offline manner that does not allow the system to change its objective while operating. For that reason, the following subsection will provide an actor/critic NN framework to approximate the solution of the discretely sampled state controller HJB and Riccati equations.

5.2.2 Approximate Solution

The first step to solve the event-triggered HJB Equation (5.7) is to approximate the value function $V^\star(x)$ from (5.2). The value function can be represented in a compact set $\Omega \subseteq \mathbb{R}^n$ by a critic NN of the form,

$$V^\star(x) = {W^\star}^{\mathrm{T}} \phi(x) + \epsilon_{\mathrm{c}}(x), \tag{5.8}$$

where the $W^\star \in \mathbb{R}^h$ denote the ideal weights bounded as $\|W^\star\| \leqslant W_{\max}$, and $\phi \coloneqq [\phi_1 \ \phi_2 \ \dots \ \phi_h]$: $\mathbb{R}^n \leftarrow \mathbb{R}^h$, is a bounded continuously differentiable basis function ($\|\phi\| \leqslant \phi_{\max}$ and $\|\frac{\partial \phi}{\partial x}\| \leqslant \phi_{\mathrm{dmax}}$) the activation functions with h neurons, and $\epsilon_{\mathrm{c}}(x)$ is the corresponding residual error such that $\sup_{x\in\Omega} \|\epsilon_c\| \leqslant \epsilon_{\mathrm{cmax}}$ and $\sup_{x\in\Omega} \|\frac{\partial \epsilon_c}{\partial x}\| \leqslant \epsilon_{\mathrm{dcmax}}$. The activation functions ϕ are selected such as $h \to \infty$ one has a complete independent basis for $V^\star$. For causality issues and due to the online nature of our algorithm, we will define the triggering inter execution release time to be in $t \in (r_{j-1}, r_j]$ with $j \in \mathbb{N}$ (in the subsequent analysis, j will be in this set).

Based on this, the optimal event-triggered controller can be rewritten as,

$$u^\star(\hat{x}_j) = -\frac{1}{2} g(\hat{x}_j)^{\mathrm{T}} \left(\frac{\partial \phi(\hat{x}_j)}{\partial x}^{\mathrm{T}} W^\star + \frac{\partial \epsilon_c(\hat{x}_j)}{\partial x} \right), \quad t \in (r_{j-1}, r_j]. \tag{5.9}$$

Remark 5.2. The control input jumps at the triggering instants and remains constant $\forall t \in (r_{j-1}, r_j]$. This is in general achieved with zero-order hold but we shall see that in our algorithm it is not necessary. □

The optimal event-triggered controller (5.9) can be approximated by another NN which we call an *actor*. This has the following form for all $t \in (r_{j-1}, r_j]$,

$$u^\star(\hat{x}_j) = {W_u^\star}^{\mathrm{T}} \phi_u(\hat{x}_j) + \epsilon_u(\hat{x}_j), \quad \forall \hat{x}_j, \ j \in \mathbb{N}, \tag{5.10}$$

where $W_u^\star \in \mathbb{R}^{h_2 \times m}$ are the optimal weights and $\phi_u(\hat{x}_j)$ are the basis functions defined similarly to the critic and h_2 is the number of basis and ϵ_u is the actor approximation error. Note that in order for $u^\star$ to be uniformly approximated the activation functions must define a complete independent basis set. The residual error ϵ_u and the activation

functions are assumed to be upper bounded by positive constants as $\sup_{\hat{x}_j \in \Omega} \|\epsilon_u\| \leqslant \epsilon_{\text{umax}}$ and $\|\phi_u\| \leqslant \phi_{\text{umax}}$, respectively.

The value function (5.8) and the optimal policy (5.10) using current estimates $\hat{W}_c$ and $\hat{W}_u$, respectively, of the ideal weights $W^\star$ and $W_u^\star$ are given by the following critic and actor NNs,

$$\hat{V}(x(t)) = \hat{W}_c^{\mathrm{T}} \phi(x(t)), \quad \forall x, \tag{5.11}$$

$$\hat{u}(\hat{x}_j) = \hat{W}_u^{\mathrm{T}} \phi_u(\hat{x}_j), \quad \forall \hat{x}_j. \tag{5.12}$$

We define the critic error as $\tilde{W}_{\mathrm{c}} := W^\star - \hat{W}_{\mathrm{c}}$ and the actor error as $\tilde{W}_u := W_u^\star - \hat{W}_u$. Our goal now should be to find the tuning laws for the weights $\hat{W}$ and $\hat{W}_u$. In order to do that, we will use adaptive control techniques (Ioannou and Fidan, 2006).

Hence one needs to pick the weights $\hat{W}_c$ as,

$$\dot{\hat{W}}_c = -\alpha \frac{\omega}{(\omega^{\mathrm{T}}\omega + 1)^2} (\omega^{\mathrm{T}} \hat{W}_c + \hat{r}), \tag{5.13}$$

where α determines the speed of convergence.

Now in order to find the tuning for the actor NN we need to define the error $e_u \in \mathbb{R}^m$ in the following form,

$$e_u := \hat{W}_u \phi_u(\hat{x}_j) + \frac{1}{2} g(\hat{x}_j)^{\mathrm{T}} \frac{\partial \phi_u(\hat{x}_j)}{\partial x}^{\mathrm{T}} \hat{W}_c, \quad \forall \hat{x}_j.$$

The objective is to select $\hat{W}_u$ such that the error e_u goes to zero. For that reason, we will select to minimize the following squared error performance,

$$E_u = \frac{1}{2} \mathrm{tr}\{e_u^{\mathrm{T}} e_u\}.$$

The nature of the update law for the actors will have an aperiodic nature and hence, it has to be *updated only at the trigger instants* and held constant otherwise. This has a form of an impulsive system as described in Haddad *et al.* (2006) and Hespanha *et al.* (2008).

We can then define the following laws,

$$\dot{\hat{W}}_u(t) = 0, \quad \text{for } r_{j-1} < t \leqslant r_j, \tag{5.14}$$

and the jump equation to compute $\hat{W}_u(r_j^+)$ given by,

$$\hat{W}_u^+ = \hat{W}_u(t) - \alpha_u \phi_u(x(t)) \left(\hat{W}_u^{\mathrm{T}} \phi_u(x(t)) + \frac{1}{2} g(x(t))^{\mathrm{T}} \frac{\partial \phi(x(t))}{\partial x}^{\mathrm{T}} \hat{W}_c \right)^{\mathrm{T}}, \quad \text{for } t = r_j. \tag{5.15}$$

We now present the main theorem, which establishes asymptotic stability of the impulsive closed-loop system along with convergence to the optimal solution.

Theorem 5.1 (Vamvoudakis, 2014a, Thm. 3)**.** Consider the nonlinear CT system given by (2.1) with the event-triggered control input given by (5.12) and the critic NN given by (5.11). The tuning laws for the CT critic and impulsive actor NNs are given by (5.13), (5.14) and (5.15), respectively. Then there exists a quadruple $\left(\Omega_x \times \Omega_{\hat{x}_j} \times \Omega_{\tilde{W}_c} \times \Omega_{\tilde{W}_u}\right) \subset \Omega$ with Ω compact such that the solution of the impulsive system $\psi \in \left(\Omega_x \times \Omega_{\hat{x}_j} \times \Omega_{\tilde{W}_c} \times \Omega_{\tilde{W}_u}\right)$ exists globally and converges asymptotically to zero for all $x(0)$ inside $\Omega_x, \hat{x}_j(0)$ inside $\Omega_{\hat{x}_j}, W_c(0)$ inside $\Omega_{\tilde{W}_c}$ and $W_u(0)$ inside $\Omega_{\tilde{W}_u}$ given the following triggering condition:

$$\|e_j\|^2 \leqslant \frac{(1-\beta^2)}{L^2} \underline{\lambda}(Q) \|x\|^2 + \frac{1}{L^2} \|\hat{W}_u^{\mathrm{T}} \phi_u(\hat{x}_j)\|^2, \tag{5.16}$$

where $\beta \in (0,1)$ and the following inequalities are satisfied:

$$\alpha > \sqrt{\frac{1}{8\underline{\lambda}(M)}},$$

for the critic NN and,

$$\left(\phi_{u\max}^2 - \frac{3}{2} - \frac{\alpha_u}{2} \phi_{u\max}^4 \right) > 0,$$

for the actor NN.

Remark 5.3. As the bandwidth increases by tweaking β to be close to 1 one approaches asymptotically the performance of the infinite bandwidth or time-triggered controller. □

A pseudocode that describes the proposed RL algorithm has the following form,

Algorithm 16: Event-Triggered RL Algorithm

procedure
  Start with initial state $x(0)$ and random initial weights $\hat{W}_u(0), \hat{W}(0)$.
  Propagate $t, x(t)$.
  Monitor $e_j(t)$ using condition (5.16).
  Propagate $\hat{W}_u(t), \hat{W}_c(t)$ $\triangleright$ $\dot{\hat{W}}_u$ as in (5.14), and $\hat{W}_u^+$ as in (5.15) and $\dot{\hat{W}}_c$ as in (5.13).
  Compute $\hat{V}(x(t)) = \hat{W}_c^{\mathrm{T}} \phi(x(t)), \forall x$, $\hat{u}(\hat{x}_j) = \hat{W}_u^{\mathrm{T}} \phi_u(\hat{x}_j), \forall \hat{x}_j$ $\triangleright$ output of the Critic, and $\triangleright$ output of the Actor.
end procedure

5.2.3 Further Reading

For more technical details and simulations, the interested reader is directed to Vamvoudakis (2014a,b). The work of Yang *et al.* (2020b) presented an intermittent framework for safe RL algorithms by leveraging an actor/critic structure to solve the problem online while guaranteeing optimality, stability, and safety. Finally, the authors in Zhong and He (2016), Wang *et al.* (2017a), Solowjow and Trimpe (2020), Gao *et al.* (2016), Zhang *et al.* (2016b), Zhu *et al.* (2016), Dong *et al.* (2016a), Wang *et al.* (2016), Dong *et al.* (2016b), and Zhang *et al.* (2016a) inspired by the work demonstrated here, develop event-triggered online adaptive learning schemes.

5.3 Optimal Tracking Control of Nonlinear Systems

The basic problems in control theory are the regulation of the system to an equilibrium point and the tracking of the desired trajectory. In the preceding subsection, we coped with the optimal regulation of nonlinear systems. Now, in this subsection, we propose a novel event-triggered optimal tracking control algorithm for nonlinear systems with an infinite horizon discounted cost. The problem is formulated by appropriately augmenting the system and the reference dynamics and then using ideas from RL to provide a solution. Namely, a critic network is used to estimate the optimal cost, while an actor-network is used to approximate the optimal event-triggered controller. Because

the actor-network updates only when an event occurs, we shall use a zero-order hold along with appropriate tuning laws to encounter for this behavior. Because we have dynamics that evolve in continuous and discrete time, we write the closed-loop system as an impulsive model and prove asymptotic stability of the equilibrium point and Zeno behavior exclusion.

5.3.1 Problem Formulation

Consider the input affine nonlinear system defined by (2.1). In order to achieve the desired output trajectory, we shall use the following exosystem,

$$\dot{z}(t) = f_d(z(t)), \quad z(0) = z_0, \tag{5.17}$$

where $z(t) \in \mathbb{R}^n$ denotes the bounded desired trajectory and $f_d(z(t))$ a Lipschitz continuous function with $f_d(0) = 0$.

Because we would like to achieve tracking of a desired trajectory, we shall define the following tracking error,

$$e_{\text{track}}(t) := x - z,$$

with dynamics,

$$\dot{e}_{\text{track}} = f\left(e_{\text{track}}(t) + z(t)\right) + g\left(e_{\text{track}}(t) + z(t)\right)u(t) - f_d(z(t)), \quad t \geqslant 0. \tag{5.18}$$

One can define the augmented state $x_{\text{aug}} := \begin{bmatrix} e_{\text{track}}^{\text{T}} & z^{\text{T}} \end{bmatrix}^{\text{T}} \in \mathbb{R}^{2n}$ as in (Modares and Lewis, 2014) and write the augmented dynamics of (5.17) and (5.18) as,

$$\dot{x}_{\text{aug}} = f_{\text{aug}}\left(x_{\text{aug}}\right) + g_{\text{aug}}\left(x_{\text{aug}}\right)u(t), \tag{5.19}$$

where

$$f_{\text{aug}}\left(x_{\text{aug}}\right) := \begin{bmatrix} f\left(e_{\text{track}}(t) + z(t)\right) - f_d(z(t)) \\ f_d(z(t)) \end{bmatrix} \quad \text{and}$$

$$g_{\text{aug}}\left(x_{\text{aug}}\right) := \begin{bmatrix} g\left(e_{\text{track}}(t) + z(t)\right) \\ 0 \end{bmatrix}.$$

In order to save resources, the controller will work with a sampled version of the state obtained at the triggering instants, that is, the

controller is updated only when, as we shall see later, the triggering condition is satisfied. For that reason, one needs to introduce a sampled-data component that is characterized by a monotone increasing sequence of sampling instants (broadcast release times) $\{r_j\}_{j=1}^{\infty}$, where r_j is the j th consecutive sampling instant (impulse times) that satisfies $0 \leqslant t_0$ $t_1 < \cdots < r_j < \cdots$ and $\lim_{j\to\infty} r_j = \infty$. The output of the sampled-data component is a sequence of sampled states $\hat{x}_{\text{aug}}$ with $\hat{x}_{\text{aug}} = x_{\text{aug}}(r_j)$ for all $t \in (r_j, r_{j+1}]$ and $j \in \mathbb{N}$. The controller maps the sampled state onto a control vector u, which after using a zero-order hold becomes a CT input signal. For simplicity, we will assume that the sampled-data systems have zero task delays.

In order to decide when to trigger an event, we will define the gap or difference between the current state $x_{\text{aug}}(t)$ and the sampled state $\hat{x}_{\text{aug}}(t)$ as,

$$e_{\text{trig}}(t) := \hat{x}_{\text{aug}} - x_{\text{aug}}, \quad \forall t \in (r_j, r_{j+1}], \ j \in \mathbb{N},$$

where

$$\hat{x}_{\text{aug}}(t) = \begin{cases} x_{\text{aug}}(r_j), & \forall t \in (r_j, r_{j+1}], \\ x_{\text{aug}}(t), & t = r_j. \end{cases}$$

Remark 5.4. Note that when an event is triggered at $t = r_j$, a new state measurement is rendered that resets the gap e_{trig} to zero. □

We shall find a controller u of the form $u = k(\hat{x}_{\text{aug}}(t) + e_{\text{trig}}(t))$ that minimizes a cost functional similar to the one with an infinite bandwidth controller,

$$J(x_{\text{aug}}(0); u) = \int_0^{\infty} e^{-\gamma\tau}(x_{\text{aug}}^{\text{T}} Q_{\text{aug}} x_{\text{aug}} + u^{\text{T}} R u)\mathrm{d}\tau, \quad \forall x_{\text{aug}}(0),$$

where $\gamma \in \mathbb{R}^+$ is the discount factor, and with user defined matrices $R > 0$ and

$$Q_{\text{aug}} := \begin{bmatrix} Q & 0_{n\times n} \\ 0_{n\times n} & 0_{n\times n} \end{bmatrix},$$

where $Q \geq 0$ and $0_{n\times n}$ a square matrix of zeros.

We are interested in finding the optimal control $u^\star$ in the sense that, $J(x_{\text{aug}}(0); u^\star) \leqslant J(x_{\text{aug}}(0); u), \forall u$, which can be expressed by the minimization problem $J(x_{\text{aug}}(0); u^\star) = \min_u J(x_{\text{aug}}(0); u)$ given the augmented dynamics in (5.19).

The ultimate goal is to find the optimal value function $V^\star$ defined by,

$$V^\star(x_{\text{aug}}(t)) := \min_u \frac{1}{2}\int_t^\infty e^{-\gamma(\tau-t)}(x_{\text{aug}}^{\text{T}}Q_{\text{aug}}x_{\text{aug}} + u^{\text{T}}Ru)\text{d}\tau, \quad t \geqslant 0,\ x_{\text{aug}}, \tag{5.20}$$

given an aperiodic event-triggered controller as will be defined in the subsequent analysis.

5.3.2 Existence of Solution

This subsection shall provide a connection between the infinite bandwidth optimal control and the event-triggered control. Let us assume that the continuous sampled controller, namely, $u_{\text{c}} = k\left(x_{\text{aug}}\right)$, has infinite bandwidth, that is, continuous sampling, and hence, one can define the Hamiltonian associated with (5.19) and (5.20) as follows:

$$\begin{aligned}\mathcal{H}\left(x_{\text{aug}}, u_{\text{c}}, \frac{\partial V^\star}{\partial x_{\text{aug}}}\right) &= \frac{\partial V^{\star\text{T}}}{\partial x}(f_{\text{aug}}(x_{\text{aug}}) + g_{\text{aug}}(x_{\text{aug}})u_{\text{c}}) \\ &\quad + \frac{1}{2}(x_{\text{aug}}^{\text{T}}Q_{\text{aug}}x_{\text{aug}} + u_{\text{c}}^{\text{T}}Ru_{\text{c}} - 2\gamma V(x_{\text{aug}})), \\ &\quad \forall x_{\text{aug}}, u_{\text{c}}.\end{aligned} \tag{5.21}$$

After employing the stationarity condition, in the Hamiltonian (5.21), that is, $\frac{\partial \mathcal{H}(\cdot)}{\partial u_{\text{c}}} = 0$, the infinite bandwidth optimal control can be found to be,

$$\begin{aligned}u_{\text{c}}^\star := u_{\text{c}}^\star\left(x_{\text{aug}}\right) &= \arg\min_{u_{\text{c}}} \mathcal{H}\left(x_{\text{aug}}, u_{\text{c}}, \frac{\partial V^\star}{\partial x_{\text{aug}}}\right) \\ &= -R^{-1}g_{\text{aug}}\left(x_{\text{aug}}\right)^{\text{T}}\frac{\partial V^\star}{\partial x_{\text{aug}}}, \quad \forall x_{\text{aug}}.\end{aligned} \tag{5.22}$$

By substituting the optimal control (5.22) into (5.21) one has the HJB equation given as,

$$\mathcal{H}\left(x_{\text{aug}}, u_{\text{c}}^\star, \frac{\partial V^\star}{\partial x_{\text{aug}}}\right) = 0, \quad \forall x_{\text{aug}}. \tag{5.23}$$

Now in order to reduce the communication between the plant and the controller, one needs to use an event-triggered version of the previous

HJB Equation (5.23) by introducing a sampled-data component with sparse and aperiodic controller updates that ensure a certain condition on the state of the plant to guarantee stability and a desired level of performance as we shall see in the subsequent analysis. For that reason, the control input uses the sampled-state information instead of the true one and hence (5.22) for $t \in (r_j, r_{j+1}], j \in \mathbb{N}$ becomes,

$$u^\star := u^\star(\hat{x}_{\text{aug}}) = -R^{-1} g_{\text{aug}}(\hat{x}_{\text{aug}})^{\text{T}} \frac{\partial V^\star}{\partial \hat{x}_{\text{aug}}}, \quad \forall \hat{x}_{\text{aug}}. \tag{5.24}$$

By using the event-triggered controller given by (5.24), the HJB Equation (5.23) becomes,

$$\begin{aligned} \mathcal{H}\left(x_{\text{aug}}, u^\star, \frac{\partial V^\star(x_{\text{aug}})}{\partial x_{\text{aug}}}\right) \equiv \frac{\partial V^\star(x_{\text{aug}})^{\text{T}}}{\partial x_{\text{aug}}} \Bigg(& f_{\text{aug}}(x_{\text{aug}}) - g_{\text{aug}}(x_{\text{aug}}) \\ & \times R^{-1} g_{\text{aug}}(\hat{x}_{\text{aug}})^{\text{T}} \frac{\partial V^\star}{\partial \hat{x}_{\text{aug}}} \Bigg) \\ & + \frac{1}{2}(x_{\text{aug}}^{\text{T}} Q_{\text{aug}} x_{\text{aug}} + u^{\text{T}} R u^\star - 2\gamma V(x_{\text{aug}})), \\ & \forall x_{\text{aug}}, \hat{x}_{\text{aug}}. \end{aligned} \tag{5.25}$$

Notice that solving analytically the event-triggered HJB Equation (5.25) for the optimal cost for nonlinear systems is in most of the cases difficult. Hence, the following subsection will provide an actor/critic NN framework to approximate the solution of the discretely sampled state controller HJB equation.

5.3.3 Approximate Solution

The first step to solve the event-triggered HJB Equation (5.25) is to approximate the value function $V^\star(x_{\text{aug}})$ from (5.20). The value function can be represented in a compact set $\Omega \subseteq \mathbb{R}^{2n}$ by a critic approximator of the form,

$$V^\star(x_{\text{aug}}) = W_{\text{c}}^{\star\text{T}} \phi_{\text{c}}(x_{\text{aug}}) + \epsilon_{\text{c}}(x_{\text{aug}}), \quad \forall x_{\text{aug}}, \tag{5.26}$$

where the $W_{\text{c}}^\star \in \mathbb{R}^h$ denote the ideal weights bounded as $\|W_{\text{c}}^\star\| \leqslant W_{\text{cmax}}$, and $\phi_{\text{c}} = [\phi_1\ \phi_2 \dots \phi_h]$: $\mathbb{R}^{2n} \to \mathbb{R}^h$, is a bounded continuously

differentiable basis function ($\|\phi_{\mathrm{c}}\| \leqslant \phi_{\mathrm{cmax}}$ and $\|\frac{\partial \phi_{\mathrm{c}}}{\partial x}\| \leqslant \phi_{\mathrm{demax}}$) the basis set with h number of functions and $\epsilon_{\mathrm{c}}(x_{\mathrm{aug}})$ is the corresponding residual error.

Assume that the residual error ϵ_{c} is upper bounded such that $\sup_{x_{\mathrm{aug}} \in \Omega} \|\epsilon_{\mathrm{c}}\| \leqslant \epsilon_{\mathrm{cmax}}$ and $\sup_{x_{\mathrm{aug}} \in \Omega} \|\frac{\partial \epsilon_{\mathrm{c}}}{\partial x_{\mathrm{th}}}\| \leqslant \epsilon_{\mathrm{demax}}$. The basis set ϕ are selected such as $h \to \infty$ one has a complete independent basis for $V^{\star}$.

Based on this, the optimal event-triggered controller in (5.23) can be re-written as,

$$u^{\star}\left(\hat{x}_{\mathrm{aug}}\right) = -R^{-1} g_{\mathrm{aug}}\left(\hat{x}_{\mathrm{aug}}\right)^{\mathrm{T}} \left(\frac{\partial \phi\left(\hat{x}_{\mathrm{aug}}\right)^{\mathrm{T}}}{\partial \hat{x}_{\mathrm{aug}}} W_{\mathrm{c}}^{\star} + \frac{\partial \epsilon_{\mathrm{c}}\left(\hat{x}_{\mathrm{aug}}\right)}{\partial \hat{x}_{\mathrm{aug}}} \right), \quad t \in (r_j, r_j + 1]. \tag{5.27}$$

The optimal event-triggered controller (5.27) can be further approximated by another approximator, which we call an actor. This has the following form $\forall t \in (r_j, r_{j+1}]$,

$$u^{\star}\left(\hat{x}_{\mathrm{aug}}\right) = W_u^{\star \mathrm{T}} \phi_u\left(\hat{x}_{\mathrm{aug}}\right) + \epsilon_u\left(\hat{x}_{\mathrm{aug}}\right), \quad \forall \hat{x}_{\mathrm{aug}}, \tag{5.28}$$

where $W_u^{\star} \in \mathbb{R}^{h_2 \times m}$ are the optimal weights, $\phi_u\left(\hat{x}_{\mathrm{aug}}\right)$ are the basis functions defined similarly to the critic approximator, h_2 is the number of basis, and finally, ϵ_u is the actor approximation error. Note that in order for $u^{\star}$ to be uniformly approximated, the basis functions must define a complete independent basis set.

Assumption 5.2. The residual error ϵ_u and the basis functions are assumed to be upper bounded by positive constants as $\sup_{\hat{x}_{\max} \in \Omega} \|\epsilon_u\| \leqslant \epsilon_{\mathrm{umax}}$ and $\|\phi_u\| \leqslant \phi_{\mathrm{umax}}$, respectively. □

The value function (5.26) and the optimal policy (5.28) using current estimates $\hat{W}_{\mathrm{c}}$ and $\hat{W}_u$, respectively, of the ideal weights $W_{\mathrm{c}}^{\star}$ and $W_u^{\star}$ are given by the following critic and actor approximators:

$$\hat{V}\left(x_{\mathrm{aug}}(t)\right) = \hat{W}_{\mathrm{c}}^{\mathrm{T}} \phi_{\mathrm{c}}\left(x_{\mathrm{aug}}(t)\right), \quad \forall x_{\mathrm{aug}}, \tag{5.29}$$

$$\hat{u}\left(\hat{x}_{\mathrm{aug}}\right) = \hat{W}_u^{\mathrm{T}} \phi_u\left(\hat{x}_{\mathrm{aug}}\right), \quad \forall \hat{x}_{\mathrm{aug}}. \tag{5.30}$$

Our goal now should be to find the tuning laws for the weights $\hat{W}_{\mathrm{c}}$ and $\hat{W}_u$. We shall now define the error $e_{\mathrm{c}} \in \mathbb{R}$ as,

$$\begin{aligned} e_{\mathrm{c}} &:= H\left(x_{\mathrm{aug}}, \hat{u}\left(\hat{x}_{\max}\right), \frac{\partial \hat{V}\left(x_{\mathrm{aug}}\right)}{\partial x_{\mathrm{aug}}}\right) \\ &\quad - H\left(x_{\mathrm{aug}}, u_{\mathrm{c}}^{\star}(x_{\mathrm{aug}}), \frac{\partial V^{\star}(x_{\mathrm{aug}})}{\partial x_{\mathrm{aug}}}\right) \\ &= \hat{W}_{\mathrm{c}}^{\mathrm{T}} \frac{\partial \phi_{\mathrm{c}}}{\partial x_{\mathrm{aug}}}\left(f_{\mathrm{aug}}\left(x_{\mathrm{aug}}\right) + g_{\mathrm{aug}}(x_{\mathrm{aug}})\hat{u}\left(\hat{x}_{\mathrm{aug}}\right)\right) - \gamma \hat{W}_{\mathrm{c}}^{\mathrm{T}} \phi_{\mathrm{c}} + \hat{r} \\ &= \hat{W}_{\mathrm{c}}^{\mathrm{T}} \omega + \hat{r}, \end{aligned}$$

with $\omega := \frac{\partial \phi_{\mathrm{c}}}{\partial x_{aug}}(f_{\mathrm{aug}}(x_{\mathrm{aug}}) + g_{\mathrm{aug}}(x_{\mathrm{aug}})\hat{u}(\hat{x}_{\mathrm{aug}})) - \gamma \phi_{\mathrm{c}}$, $\hat{r} := \frac{1}{2}\hat{u}(\hat{x}_{\mathrm{aug}})^{\mathrm{T}} \cdot R\hat{u}(\hat{x}_{\mathrm{aug}}) + \frac{1}{2} x_{\mathrm{aug}}^{\mathrm{T}} Q_{\mathrm{aug}} x_{\mathrm{aug}}$, and $\mathcal{H}(x_{\mathrm{aug}}, u_{\mathrm{c}}^{\star}(x_{\mathrm{aug}}), \frac{\partial V^{\star}(x_{\mathrm{usg}})}{\partial x_{\mathrm{aug}}}) = 0$ from (5.23), and for simplicity, we have written ϕ_{c} instead of $\phi_{\mathrm{c}}\left(x_{\mathrm{aug}}\right)$.

In order to drive the error e_{c} to zero, one has to pick appropriately the critic weights. By defining the squared-norm error as $E_{\mathrm{c}} = \frac{1}{2} e_{\mathrm{c}}^2$, we can apply the gradient descent method to obtain the,

$$\dot{\hat{W}}_{\mathrm{c}} = -\alpha \frac{1}{\left(\omega^{\mathrm{T}} \omega + 1\right)^2} \frac{\partial E_{\mathrm{c}}}{\partial \hat{W}_{\mathrm{c}}} = -\alpha \frac{\omega}{\left(\omega^{\mathrm{T}} \omega + 1\right)^2} e_{\mathrm{c}}, \tag{5.31}$$

where $\alpha \in \mathbb{R}^{+}$ determines the speed of convergence.

Now in order to find the tuning law for the actor approximator, we need to define the error $e_u \in \mathbb{R}^m$ in the following form:

$$\begin{aligned} e_u &:= \hat{u} - u_{\hat{W}_{\mathrm{c}}} \\ &= \hat{W}_u^{\mathrm{T}} \phi_u\left(\hat{x}_{\mathrm{aug}}\right) + R^{-1} g_{\mathrm{aug}}\left(\hat{x}_{\mathrm{aug}}\right)^{\mathrm{T}} \frac{\partial \phi_u\left(\hat{x}_{\mathrm{aug}}\right)}{\partial x_{\mathrm{aug}}}^{\mathrm{T}} \hat{W}_{\mathrm{c}}, \quad \forall \hat{x}_{\mathrm{aug}}, \end{aligned}$$

where $u_{\hat{W}_{\mathrm{c}}}$ is the controller with the critic weights $\hat{W}_{\mathrm{c}}$.

The objective is to select $\hat{W}_u$ such that the error e_u goes to zero. For that reason, we shall select to minimize the following squared error performance:

$$E_u = \frac{1}{2} e_u^{\mathrm{T}} e_u.$$

The update law for the actors will have an aperiodic nature because the updates occur only at the triggering instants, and held constant

otherwise. This has a form of an impulsive system as described in Haddad *et al.* (2006, 2008).

We can then define the following laws:

$$\dot{\hat{W}}_u(t) = 0, \quad t \in \mathbb{R}^+ \backslash \bigcup_{j\in\mathbb{N}} r_j, \tag{5.32}$$

and the jump equation to compute $\hat{W}_u(r_j^+)$ is given by,

$$\hat{W}_u^+ = \hat{W}_u - \alpha_u \phi_u(x_{\text{aug}}(t)) \Bigg(\hat{W}_u^{\text{T}} \phi_u(x_{\text{aug}}(t)) + R^{-1} g_{\text{aug}}\left(x_{\text{aug}}(t)\right)^{\text{T}} \frac{\partial \phi\left(x_{\text{aug}}(t)\right)^{\text{T}}}{\partial x_{\text{aug}}} \hat{W}_{\text{c}} \Bigg)^{\text{T}}, \quad t = r_j. \tag{5.33}$$

By defining the actor error dynamics as $\tilde{W}_u := W_u^\star - \hat{W}_u$ and taking the time derivative using the continuous update (5.32) and by using the jump system (5.33) updated at the trigger instants, one has,

$$\dot{\tilde{W}}_u(t) = 0, \quad t \in \mathbb{R}^+ \backslash \bigcup_{j\in\mathbb{N}} r_j, \tag{5.34}$$

and

$$\begin{aligned} \tilde{W}_u^+ = \tilde{W}_u &- \alpha_u \phi_u\left(x_{\text{aug}}(t)\right) \phi_u\left(x_{\text{aug}}(t)\right)^{\text{T}} \tilde{W}_u(t) \\ &- \alpha_u \phi_u\left(x_{\text{aug}}(t)\right) \phi_u\left(x_{\text{aug}}(t)\right)^{\text{T}} \epsilon_u \\ &- \alpha_u \phi_u\left(x_{\text{aug}}(t) y\right) \tilde{W}_{\text{c}}^{\text{T}} \frac{\partial \phi\left(x_{\text{aug}}(t)\right)}{\partial x_{\text{aug}}} g_{\text{aug}}\left(x_{\text{aug}}(t)\right) R^{-1} \\ &- \alpha_u \phi_u\left(x_{\text{aug}}(t)\right) \frac{\partial \epsilon_{\text{c}}}{\partial x_{\text{aug}}} g_{\text{aug}}\left(x_{\text{aug}}(t)\right) R^{-1}, \quad \text{for } t = r_j, \end{aligned} \tag{5.35}$$

respectively. Note that the solution of (5.34)–(5.35) is left continuous; that is, it is continuous everywhere except at the resetting times r_j and,

$$\hat{W}_u\left(r_j\right) = \lim_{\delta \to 0^+} \hat{W}_u\left(r_j - \delta\right), \quad \forall j \in \mathbb{N},$$

and

$$\hat{W}_u^+ = \hat{W}_u - \alpha_u \phi_u(x_{\text{aug}}(t)) \Bigg(\hat{W}_u^{\text{T}} \phi_u(x_{\text{aug}}(t)) + R^{-1} g_{\text{aug}}\left(x_{\text{aug}}(t)\right)^{\text{T}} \times \frac{\partial \phi(x_{\text{aug}}(t))^{\text{T}}}{\partial x_{\text{aug}}} \hat{W}_{\text{c}} \Bigg)^{\text{T}}, \quad t = r_j.$$

In addition, in order to deal with the presence of the approximation errors and known bounds (Polycarpou *et al.*, 2003), and obtain an asymptotically stable equilibrium for the closed-loop system, one needs to add a robustifying term to the closed-loop system of the form,

$$\eta(t) = -c_2 \frac{\|\hat{x}_{\text{aug}}\|^2 \mathbf{1}_m}{c_1 + \|\hat{x}_{\text{aug}}\|^2},$$

where $c_1, c_2 \in \mathbb{R}^+$ satisfy,

$$c_2 > \frac{c_1 + \|\hat{x}_{\text{aug}}\|^2}{(W_{\text{cmax}}\phi_{\text{dcmax}} + \frac{1}{2}\epsilon_{\text{dcmax}})\|\hat{x}_{\text{aug}}\|^2} \times \left(\frac{1}{8\alpha}\epsilon_{\text{Hcmax}}^2 + \gamma(W_{\text{cmax}}\phi_{\text{cmax}} + \epsilon_{\text{cmax}}) + \rho\right),$$

where

$$\begin{aligned}
\rho :=\ & \frac{1}{2}(\phi_{\text{umax}}^2\epsilon_{\text{umax}})^2 + \frac{1}{8}(\|\tilde{W}_{\text{c}}\|\phi_{\text{umax}}\phi_{\text{dcmax}}\underline{\lambda}(R^{-1}))^2 \\
& + \frac{1}{8}(\phi_{\text{umax}}\epsilon_{\text{dcmax}}\underline{\lambda}(R^{-1}))^2 \\
& + \frac{\alpha_u}{2}\phi_{\text{umax}}^4\epsilon_{\text{umax}}^2 + \frac{\alpha_u}{32}\phi_{\text{umax}}^2\|\tilde{W}_{\text{c}}\|^2\phi_{\text{demax}}^2\underline{\lambda}(R^{-1})^2 \\
& + \frac{\alpha_u}{32}\phi_{u\,\max}^2\epsilon_{\text{dcmax}}^2\underline{\lambda}(R^{-1})^2 + \frac{1}{2}(\alpha_u\phi_{u\,\max}^3\epsilon_{u\,\max}^2)^2 \\
& + \frac{1}{4}\alpha_u\phi_{\text{umax}}\phi_{\text{dcmax}}\underline{\lambda}(R^{-1})\|\tilde{W}_{\text{c}}\| + \frac{1}{4}\alpha_u\phi_{\text{umax}}\epsilon_{\text{dcmax}}\underline{\lambda}(R^{-1}) \\
& + \frac{1}{4}\alpha_u\phi_{u\,\max}\epsilon_{u\,\max}\phi_{\text{dcmax}}\underline{\lambda}(R^{-1})\|\tilde{W}_{\text{c}}\| \\
& + \frac{1}{8}\alpha_u\epsilon_{\text{dcmax}}\phi_{\text{dcmax}}\underline{\lambda}(R^{-1})\|\tilde{W}_{\text{c}}\|.
\end{aligned}$$

The closed-loop system dynamics of (5.18) with control law given by (5.30), can now be written as,

$$\dot{x}_{\text{aug}} = f_{\text{aug}}(x_{\text{aug}}) + g_{\text{aug}}(x_{\text{aug}})((W_u^\star - \tilde{W}_u)^{\text{T}}\phi_u(\hat{x}_{\text{aug}}) + \eta), \quad t \geqslant 0. \tag{5.36}$$

The following theorem establishes asymptotic stability of the impulsive closed-loop system described and convergence to the optimal solution.

Theorem 5.3 (Vamvoudakis *et al.*, 2017b, Thm. 3)**.** Consider the nonlinear CT system given by (5.36) with the event-triggered control input given by (5.30) and the critic approximator given by (5.29). The tuning laws for the CT critic and impulsive actor approximators are given by (5.31) and (5.32)–(5.33), respectively. Then there exists a quadruple $(\Omega_{x_{\rm aug}} \times \Omega_{\hat{x}_{\rm aug}} \times \Omega_{\tilde{W}_{\rm c}} \times \Omega_{\tilde{W}_u}) \subset \Omega$ with Ω compact such that the solution of the impulsive system $\psi \in (\Omega_{x_{\rm aug}} \times \Omega_{\hat{x}_{\rm aug}} \times \Omega_{\tilde{W}_u})$ exists globally and converges asymptotically to zero for all $x_{\rm aug}(0)$ inside $\Omega_{x_{\rm aug}}, \hat{x}_{\rm aug}(0)$ inside $\Omega_{\hat{x}_{\rm avg}}, \tilde{W}_{\rm c}(0)$ inside $\Omega_{\tilde{W}_{\rm c}}$ and $\tilde{W}_u(0)$ inside $\Omega_{\tilde{W}_u}$ given the following triggering condition:

$$\|e_{\rm trig}\|^2 \leqslant \frac{(1-\beta^2)\lambda(Q)}{L^2\bar{\lambda}(R)}\|e_{\rm track}\|^2 + \frac{\lambda(R)}{L^2\bar{\lambda}(R)}\|\hat{W}_u^{\rm T}\phi_u(\hat{x}_{\rm aug})\|^2, \quad (5.37)$$

where $\beta \in (0,1)$ and the following inequalities are satisfied,

$$\alpha > \frac{1}{4}\sqrt{\frac{1}{\underline{\lambda}(M)}},$$

for the critic approximator, and

$$0 < \alpha_u < 2\frac{(\phi_{\rm umax}^2 - \frac{3}{2})}{\phi_{\rm umax}^4}; \quad \phi_{\rm umax} > \sqrt{\frac{3}{2}},$$

for the actor approximator. ■

In the following theorem, we shall guarantee the absence of the Zeno behavior. Toward this aim, we will find a lower bound for the inter-event time $T_j := r_{j+1} - r_j, \forall j \in \mathbb{N}$. Before we state the theorem, the following assumption is needed.

Assumption 5.4. There exists a compact set Ω such that:

1. The function $g_{\rm aug}(\cdot)$ is uniformly bounded on Ω, that is, $\sup_{x_{\rm aug}\in\Omega} \|g_{\rm aug}(x_{\rm aug})\| \leqslant \frac{1}{2}$.
2. The function $f_{\rm aug}(\cdot)$ is uniformly bounded on Ω, that is, $\sup_{x_{\rm aug}\in\Omega} \|f_{\rm aug}(x_{\rm aug})\| \leqslant b_{\rm f}\|x_{\rm aug}\|$, where $b_{\rm f}$ is a constant. □

Theorem 5.5. Suppose that Assumption 5.4, the triggering condition (5.37) and the concluding statement of Theorem 5.3 hold. Then the inter-event time $T_j, \forall j \in \mathbb{N}$, is strictly positive and has a positive lower bound. ■

5.3.4 Further Reading

The interested reader is directed to Vamvoudakis *et al.* (2017b) for detailed theorems and proofs of the above statements and algorithms. Also, the authors in Wang *et al.* (2017b,c,d) rely on the results presented here to construct event-triggered adaptive learning schemes.

5.4 Intermittent Q-Learning

This subsection extends the results derived in Subsection 4.2 to propose a model-free event-triggered optimal control algorithm for CT linear systems. The problem is formulated as an infinite-horizon RL problem, and we are able to simultaneously address the issue of designing a control and a triggering mechanism with guaranteed optimal performance by design. In order to provide a model-free solution, we adopt a Q-learning framework with a critic-network to approximate the optimal cost and a zero-order hold actor network to approximate the optimal control. Since we have dynamics that evolve in continuous and discrete-time, the closed-loop system is an impulsive model with an asymptotically stable equilibrium point.

5.4.1 Problem Formulation

Consider the system given by (4.1) but with the plant and input matrices, to be considered uncertain/unknown.

To save resources, the controller will work with a sampled version of the state defined as follows:

$$\hat{x}(t) = \begin{cases} x\left(r_j\right), & \forall t \in (r_j, r_{j+1}] \\ x(t), & t = r_j. \end{cases}$$

The controller maps the sampled state onto a control vector which after using a zero-order hold becomes a CT input signal. In order to decide when to trigger an event, we will define the gap between the current state $x(t)$ and the sampled state $\hat{x}(t)$ as,

$$e(t) := \hat{x}(t) - x(t). \tag{5.38}$$

Remark 5.5. Note that when an event is triggered at $t = r_j, j \in \mathbb{N}$ a new state measurement is rendered that resets the gap (5.38) to zero. □

We are interested in finding a controller $u(t) = u_{\mathrm{d}}$ of the form $u_{\mathrm{d}} := k(\hat{x}(t))$ that minimizes the cost functional,

$$J(x(0); u_{\mathrm{d}}) = \frac{1}{2}\int_0^\infty (x^{\mathrm{T}} H x + u_{\mathrm{d}}^{\mathrm{T}} R u_{\mathrm{d}})\mathrm{d}\tau,$$

with user-defined matrices $H \succeq 0, R \succ 0$, and with reduced updates of the control input given by the triggering rule.

For that reason, we will find a value function with a quantified guaranteed performance that approximates the time-triggered optimal value function $V^\star$ for the system (5.38) with controller $u(t) = \bar{u}_{\mathrm{c}}$ of the form $u_{\mathrm{c}} := k(x(t))$ as defined by,

$$V^\star(x(t)) := \min_{u_{\mathrm{c}}} \int_t^\infty \frac{1}{2}(x^{\mathrm{T}} H x + u_{\mathrm{c}}^{\mathrm{T}} R u_{\mathrm{c}})\mathrm{d}\tau, \quad \forall x, \tag{5.39}$$

but without any information of the system dynamics and given the event-triggered updates of the control input.

One can define the time triggered Hamiltonian associated with (5.38) and (4.1) with controller u_{c} as follows:

$$\begin{aligned}\mathscr{H}\left(x, u_{\mathrm{c}}, \frac{\partial V^\star}{\partial x}\right) &= \frac{\partial V^{\star\mathrm{T}}}{\partial x}(Ax + Bu_{\mathrm{c}}) + \frac{1}{2}x^{\mathrm{T}} H x \\ &\quad + \frac{1}{2}u_{\mathrm{c}}^{\mathrm{T}} R u_{\mathrm{c}}, \quad \forall x, u_{\mathrm{c}}.\end{aligned} \tag{5.40}$$

After employing the stationarity condition, in the Hamiltonian (5.40), i.e., $\frac{\partial \mathscr{H}\left(x, u_{\mathrm{c}}, \frac{\partial V^\star}{\partial x}\right)}{\partial u_{\mathrm{c}}} = 0$, the time-triggered optimal control can be found to be,

$$\begin{aligned}u_{\mathrm{c}}^\star := u_{\mathrm{c}}^\star(x) &= \arg\min_{u_{\mathrm{c}}} \mathcal{H}\left(x, u_{\mathrm{c}}, \frac{\partial V^\star}{\partial x}\right) \\ &= -R^{-1}B^{\mathrm{T}}\frac{\partial V^\star}{\partial x}, \quad \forall x.\end{aligned} \tag{5.41}$$

Assumption 5.6. We assume that the pair (A, B) is stabilizable and the pair $(\sqrt{H}, A)$ is detectable. □

Since the system (4.1) is linear, we can represent the time-triggered value function as quadratic in the state, i.e., $V^\star(x)$: $\mathbb{R}^n \to \mathbb{R}^+$,

$$V^\star(x) = \frac{1}{2}x^{\mathrm{T}} P x, \quad \forall x, \tag{5.42}$$

where under Assumption 5.6 $P \in \mathbb{R}^{n\times n}$ is the unique positive definite matrix that solves the following Riccati equation (i.e., $\mathcal{H}(x, u_{\mathrm{c}}^{\star}, \frac{\partial V^{\star}}{\partial x}) = 0$),

$$A^{\mathrm{T}}P + PA - PBR^{-1}B^{\mathrm{T}}P + H = 0. \tag{5.43}$$

Furthermore, the optimal control (5.41) is given by,

$$u_{\mathrm{c}}^{\star} := u_{\mathrm{c}}^{\star}(x) = -R^{-1}B^{\mathrm{T}}Px, \quad \forall x. \tag{5.44}$$

It is important to note that Eqs. (5.43) and (5.44) require complete knowledge of the system dynamics, i.e., the matrices A and B. Moreover, the controller needs to continuously monitor the state for updates.

In order to reduce the communication between the controller and the plant, we propose to use an event-triggered version of the above Riccati Equation (5.43) by introducing a sampled-data component with aperiodic controller updates that ensures stability and performance. For that reason, we define the event-triggered controller that uses the sampled version of the state as follows:

$$u_{\mathrm{d}}^{\star} := u_{\mathrm{d}}^{\star}(\hat{x}) = -R^{-1}B^{\mathrm{T}}P\hat{x}, \quad \forall \hat{x}. \tag{5.45}$$

Our goal is to propose an event-triggered Q-learning-based approach to solve the optimal control problem without any information of the system dynamics and by adjusting the parameters of the action-dependent value function in an adaptive way. To this end, the value function (5.42) needs to be parameterized as a function of the state x and the control u_{d} to represent the Q-function. We can write the following Q-function or action-dependent value $\mathcal{Q}(x, u_{\mathrm{d}})$: $\mathbb{R}^{n+m} \to \mathbb{R}^{+}$,

$$\begin{aligned}\mathcal{Q}\left(x, u_{\mathrm{d}}\right) &:= V^{\star}(x) + \mathcal{H}\left(x, u_{\mathrm{d}}, \frac{\partial V^{\star}}{\partial x}\right) - \mathcal{H}\left(x, u_{\mathrm{c}}^{\star}, \frac{\partial V^{\star}(x)}{\partial x}\right) \\ &= V^{\star}(x) + \frac{1}{2}x^{\mathrm{T}}P\left(Ax + Bu_{\mathrm{d}}\right) + \frac{1}{2}\left(Ax + Bu_{\mathrm{d}}\right)^{\mathrm{T}}Px \\ &\quad + \frac{1}{2}u_{\mathrm{d}}^{\mathrm{T}}Ru_{\mathrm{d}} + \frac{1}{2}x^{\mathrm{T}}Hx, \quad \forall x, u_{\mathrm{d}},\end{aligned} \tag{5.46}$$

where $\mathcal{H}\left(x, u_{\mathrm{c}}^{\star}, \frac{\partial v^{\star}}{\partial x}\right) = 0$, and the optimal time-triggered cost is $V^{\star}(x) = x^{\mathrm{T}}Px$.

The Q-function (5.46) can be written in a compact quadratic form in the state x and control u_{d} as follows:

$$\begin{aligned}\mathcal{Q}(x,u_{\mathrm{d}}) &= \frac{1}{2}U^{\mathrm{T}}\begin{bmatrix} P+H+PA+A^{\mathrm{T}}P & PB \\ B^{\mathrm{T}}P & R\end{bmatrix}U \\ &:= \frac{1}{2}U^{\mathrm{T}}\begin{bmatrix} Q_{\mathrm{xx}} & Q_{\mathrm{x}u_{\mathrm{d}}} \\ Q_{u_{\mathrm{d}}\mathrm{x}} & Q_{u_{\mathrm{d}}u_{\mathrm{d}}}\end{bmatrix}U := \frac{1}{2}U^{\mathrm{T}}\bar{Q}U, \end{aligned} \tag{5.47}$$

where $U := \begin{bmatrix} x^{\mathrm{T}} & u_{\mathrm{d}}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$, $Q_{\mathrm{xx}} = P+H+PA+A^{\mathrm{T}}P$, $Q_{\mathrm{x}u_{\mathrm{d}}} = PB$, $Q_{u_{\mathrm{d}}\mathrm{x}} = Q_{\mathrm{x}u_{\mathrm{d}}}^{\mathrm{T}} = B^{\mathrm{T}}P, Q_{u_{\mathrm{d}}u_{\mathrm{d}}} = R$, and positive definite

$$\bar{Q} = \begin{bmatrix} Q_{\mathrm{xx}} & Q_{\mathrm{x}u_{\mathrm{d}}} \\ Q_{u_{\mathrm{d}}\mathrm{x}} & Q_{u_{\mathrm{d}}u_{\mathrm{d}}}\end{bmatrix} \in \mathbb{R}^{(n+m)\times(n+m)}.$$

The next lemma shows that when $u_{\mathrm{d}} = u_{\mathrm{d}}^{\star}$ with $u_{\mathrm{d}}^{\star}$ given by (5.45), then we can show that (5.47) and (5.39) have the same value.

Lemma 5.7. Given the Q-function as in (5.47) and the optimal value of the time-triggered control as in (5.39), then one has $\mathcal{Q}(x,u_{\mathrm{d}}^{\star}) := \min_{u_{\mathrm{d}}} \mathcal{Q}(x,u_{\mathrm{d}}) = V^{\star}(x) + \frac{1}{2}(u_{\mathrm{c}}^{\star}-u_{\mathrm{d}}^{\star})^{\mathrm{T}} R (u_{\mathrm{c}}^{\star}-u_{\mathrm{d}}^{\star}), \forall x.$ ■

A formulation of (5.45) that will be useful in our model-free formulation that follows, can be found by solving $\frac{\partial Q(x,u_{\mathrm{d}})}{\partial u_{\mathrm{d}}} = 0$ to write,

$$u_{\mathrm{d}}^{\star} = \arg\min_{u_{\mathrm{d}}} \mathcal{Q}(x,u_{\mathrm{d}}) = -Q_{u_{\mathrm{d}}u_{\mathrm{d}}}^{-1} Q_{u_{\mathrm{d}}\mathrm{x}}\hat{x}, \quad \forall \hat{x}. \tag{5.48}$$

Since we would like to develop a model-free online tuning of the parameters of the action-dependent value function the next subsection will introduce an actor/critic structure to achieve that.

The critic approximator will approximate the Q-function (5.47) and the actor approximator (implemented with a zero-order hold) will approximate the optimal controller (5.48). Specifically, $\mathcal{Q}(x,u_{\mathrm{d}}^{\star})$ can be written as follows:

$$\mathcal{Q}(x,u_{\mathrm{d}}^{\star}) = \frac{1}{2}U^{\mathrm{T}}\begin{bmatrix} Q_{\mathrm{xx}} & Q_{\mathrm{x}u_{\mathrm{d}}} \\ Q_{u_{\mathrm{d}}\mathrm{x}} & Q_{u_{\mathrm{d}}u_{\mathrm{d}}}\end{bmatrix}U := \frac{1}{2}\mathrm{vech}(\bar{Q})^{\mathrm{T}}(U\otimes U).$$

By denoting as $W_{\mathrm{c}} := \frac{1}{2}\mathrm{vech}(\bar{Q})$ we can write it in a compact form as $\mathcal{Q}(x,u_{\mathrm{d}}^{\star}) = W_{\mathrm{c}}^{\mathrm{T}}(U\otimes U)$, with $W_{\mathrm{c}} \in \mathbb{R}^{\frac{1}{2}(n+m)(n+m+1)}$ the ideal weights

given as, $\text{vech}(Q_{\text{xx}}) := W_{\text{c}}[1\colon \frac{n(n+1)}{2}]$,

$$\text{vech}\left(Q_{\text{x}u_{\text{d}}}\right) := W_{\text{c}}\left[\frac{n(n+1)}{2}+1\colon \frac{n(n+1)}{2}+nm\right],$$

and $\text{vech}\left(Q_{u_{\text{d}}u_{\text{d}}}\right) := W_{\text{c}}\left[\frac{n(n+1)}{2}+nm+1\colon \frac{(n+m)(n+m+1)}{2}\right]$.

Since the ideal weights W_{c} for computing $\mathcal{Q}\left(x, u_{\text{d}}^{\star}\right)$ are unknown, one must consider weight estimates $\hat{W}_{\text{c}}$ to write an approximate solution $\hat{Q}\left(x, u_{\text{d}}\right)$ in the following form:

$$\hat{\mathcal{Q}}\left(x, u_{\text{d}}\right) = \hat{W}_{\text{c}}^{\text{T}}(U \otimes U), \quad \forall x,\ u_{\text{d}}, \tag{5.49}$$

where $\hat{W}_{\text{c}} \in \mathbb{R}^{\frac{1}{2}(n+m)(n+m+1)}$.

Similarly for computing an approximate solution $\hat{u}_{\text{d}}$ of $u_{\text{d}}^{\star}$ given by (5.48), one has the following actor approximator:

$$\hat{u}_{\text{d}} = \hat{W}_{\text{a}}^{\text{T}}\hat{x}, \tag{5.50}$$

where $\hat{W}_{\text{a}} \in \mathbb{R}^{n\times m}$ are the weight estimates.

It was shown in Vrabie *et al.* (2013) that by using IRL we can write the value function (5.39) as the following Bellman equation:

$$V^{\star}(x(t)) = V^{\star}(x(t-T)) - \frac{1}{2}\int_{t-T}^{T}(x^{\text{T}}Hx + u_{\text{d}}^{\star\text{T}}Ru_{\text{d}}^{\star})\text{d}\tau, \tag{5.51}$$

where $T \in \mathbb{R}^{+}$ a small fixed time interval. By using the result from Lemma (5.7) where we have brought together the values with the time-triggered and event-triggered controllers, we can write (5.51) as,

$$\mathcal{Q}(x(t), u_{\text{d}}^{\star}(t)) = \mathcal{Q}(x(t-T), u_{\text{d}}^{\star}(t-T)) - \frac{1}{2}\int_{t-T}^{T}(x^{\text{T}}Hx + u_{\text{d}}^{\star\text{T}}Ru_{\text{d}}^{\star})\text{d}\tau.$$

In order to find the update law for the critic, we shall define the following error $e_{\text{c}} \in \mathbb{R}$ that we would like to eventually drive to zero by picking appropriately $\hat{W}_{\text{c}}$,

$$\begin{aligned} e_{\text{c}} :=\ & \hat{Q}(x(t), \hat{u}_{\text{d}}(t)) - \hat{Q}(x(t-T), \hat{u}_{\text{d}}(t-T)) \\ & + \frac{1}{2}\int_{t-T}^{T}(x^{\text{T}}Hx + \hat{u}_{\text{d}}^{\text{T}}R\hat{u}_{\text{d}})\text{d}\tau = \hat{W}_{\text{c}}^{\text{T}}(U(t)\otimes U(t)) \\ & + \frac{1}{2}\int_{t-T}^{T}(x^{\text{T}}Hx + \hat{u}_{\text{d}}^{\text{T}}R\hat{u}_{\text{d}})\text{d}\tau - \hat{W}_{\text{c}}^{\text{T}}(U(t-T)\otimes U(t-T)). \end{aligned}$$

Now, for the actor approximator, we can define the error $e_{\mathrm{a}} \in \mathbb{R}^m$ as follows:

$$e_{\mathrm{a}} := \hat{W}_{\mathrm{a}}^{\mathrm{T}} x\left(r_j\right) + \hat{Q}_{uu_{\mathrm{d}}}^{-1} \hat{Q}_{u_{\mathrm{d}}\mathrm{x}} x\left(r_j\right), \quad t = r_j,$$

where the values of $\hat{Q}_{u_{\mathrm{d}}u_{\mathrm{d}}}^{-1}$ and $\hat{Q}_{u_{\mathrm{d}}\mathrm{x}}$ are going to be extracted from the vector $\hat{W}_{\mathrm{c}}$. Now, we shall find tuning updates for $\hat{W}_{\mathrm{c}}$ and $\hat{W}_{\mathrm{a}}$ such that the errors e_{c} and e_{a} go to zero.

By following adaptive control techniques (Ioannou and Fidan, 2006), we can define the squared-norm of these errors as,

$$K_1 = \frac{1}{2} \left\| e_{\mathrm{c}} \right\|^2, \tag{5.52}$$

$$K_2 = \frac{1}{2} \left\| e_{\mathrm{a}} \right\|^2. \tag{5.53}$$

5.4.2 Learning Algorithm

The estimate of $\hat{W}_{\mathrm{c}}$ for the critic weights can be constructed by applying gradient descent in (5.52) and normalizing as,

$$\dot{\hat{W}}_{\mathrm{c}} = -\alpha_{\mathrm{c}} \frac{1}{\left(1 + \sigma^{\mathrm{T}}\sigma\right)^2} \frac{\partial K_1}{\partial \hat{W}_{\mathrm{c}}} = -\alpha_{\mathrm{c}} \frac{\sigma}{\left(1 + \sigma^{\mathrm{T}}\sigma\right)^2} e_{\mathrm{c}}^{\mathrm{T}}, \tag{5.54}$$

where $\sigma := U(t) \otimes U(t) - U(t-T) \otimes U(t-T)$, and $\alpha_{\mathrm{c}} \in \mathbb{R}^+$ is a constant gain that determines the speed of convergence.

The update law for the actors will have an aperiodic nature since it occurs only at the triggering instants and the previous value is held constant otherwise. This has a form of an impulsive system as described in Haddad *et al.* (2006), Hespanha *et al.* (2008). Hence, the actor weights $\hat{W}_{\mathrm{a}}$ will be updated based on the following impulsive system (after applying a gradient descent in (5.53)),

$$\begin{aligned}
\dot{\hat{W}}_{\mathrm{a}} &= 0, \quad t \in (r_j, r_{j+1}] \\
\hat{W}_{\mathrm{a}}^{+} &= \hat{W}_{\mathrm{a}} - \alpha_{\mathrm{a}} \frac{1}{\left(1 + x(t)^{\mathrm{T}} x(t)\right)} \frac{\partial K_2}{\partial \hat{W}_{\mathrm{a}}} \\
&= \hat{W}_{\mathrm{a}} - \alpha_{\mathrm{a}} \frac{x(t)}{\left(1 + x(t)^{\mathrm{T}} x(t) e_{\mathrm{a}}^{\mathrm{T}}\right)}, \quad t = r_j,
\end{aligned}$$

where $\alpha_{\mathrm{a}} \in \mathbb{R}^+$ is a constant gain that determines the speed of convergence.

We just saw that the dynamics are continuous but the controller jumps to a new value when an event is triggered. Hence, we need to formulate the closed-loop system as an impulsive system. The closed-loop system dynamics of (4.1), (5.39), (5.50) can now be written as,

$$\dot{x} = Ax + B(-Q_{u_{\mathrm{d}}u_{\mathrm{d}}}^{-1} Q_{u_{\mathrm{d}}\mathrm{x}} - \tilde{W}_{\mathrm{a}}^{\mathrm{T}})\hat{x}, \quad t \geqslant 0. \tag{5.55}$$

The main theorem is presented next establishing the stability properties of the closed loop system.

Theorem 5.8 (Vamvoudakis and Ferraz, 2018, Thm. 2)**.** Consider the system dynamics given by (5.55), the Q function critic approximator given by (5.49) and the actor approximator given by (5.50). The tuning laws for the weights of the critic and for the actor are given by (5.54) and (5.54), respectively. Then, the origin of the closed-loop impulsive system is globally asymptotically stable as long as the following condition on the gap is satisfied:

$$\|e\|^2 \leqslant \frac{\left(1-\beta^2\right)\underline{\lambda}(H)}{4\left(L^2+L_1^2\right)\tilde{\lambda}(R)}\|x\|^2 + \frac{\underline{\lambda}(R)}{4\left(L^2+L_1^2\right)\tilde{\lambda}(R)}\|\hat{u}_{\mathrm{d}}\|^2,$$

where L_1 is a positive constant of unity order, and the following inequalities hold:

$$\alpha_{\mathrm{c}} \gg \alpha_{\mathrm{a}}; \quad 0 < \alpha_{\mathrm{a}} < \frac{8\underline{\lambda}(R)-4}{\underline{\lambda}(R)+2}; \quad \frac{\underline{\lambda}(H)}{\bar{\lambda}(R)} > \frac{L_1^2}{\beta^2}. \qquad \blacksquare$$

5.4.3 Further Reading

The interested reader is directed to Vamvoudakis and Ferraz (2018) for detailed theorems and proofs of the above statements and algorithms. Also, the authors in Yang and He (2018) devised a novel event-triggered robust control strategy for continuous-time nonlinear systems with unknown dynamics.

6

Bounded Rationality and Non-Equilibrium RL in Games

In this section, we relax the assumption of infinitely rational agents, whereby the analysis in Section 3 took place, giving rise to non-equilibrium differential games involving agents with bounded intelligence. We start by motivating the application of bounded rationality in the context of a differential game, and next, we introduce a formulation for non-equilibrium learning behavior analysis in systems with adversarial inputs.

6.1 Introduction and Motivation

The assumption of perfect rationality that permeates the Nash equilibrium solution concept has been shown to fail in explaining experimental data from a plethora of studies (Crawford and Iriberri, 2007). In particular, in many games where human players are involved, equilibrium models fail to predict or explain the observed behavior. Consequently, these observations give rise to several structural non-equilibrium models, such as quantal responses (McKelvey and Palfrey, 1995), level-k thinking, and cognitive hierarchy models (Camerer *et al.*, 2004), systematically outperform Nash models in their predictive abilities.

An alternative explanation is that agents are boundedly rational, i.e., agents might be making mistakes during the game. Thus, decision makers may never play the exact same game very often, but they may also extrapolate between games and learn from experiences. Several recent experimental studies (Ho and Su, 2010) suggest that decision makers' initial responses to games often deviate systematically from equilibrium, and that structural non-equilibrium models (e.g., cognitive hierarchy) often out-predict equilibrium. Non-equilibrium models need to allow for players whose adjustment rules are not a best response to the adjustment rules of the others. As a concrete example of this need, we may consider the interactions between humans and unmanned autonomous vehicles. Since the interacting agents are heterogeneous in their computational capabilities, solution concepts that go beyond the infinite intelligence models need to be investigated.

One of the first works on non-equilibrium game theoretic behavior in static environments has been reported in Fudenberg *et al.* (1998), Tambe (2011), and Brams and Kilgour (1988). In such works, the authors state that, for most purposes the right models involve neither full rationality nor the extreme naivete of most stimulus–response models; "better" models have agents consciously but perhaps imperfectly trying to get a good payoff. Besides specifying the players' forecast rules, an analysis of learning must also address the issue of whether players try to influence their opponents' play. The work of Roth and Erev (1995) and Erev and Roth (1998) examined RL approaches for experimental games with unique equilibrium and discussed implications for developing a low-rationality framework, namely behavioral game theory. The authors in He *et al.* (2016) overcame the irrationality issue with a Win-or-Learn Fast in a mini-max-Q learning framework, but with the penalty of losing convergence assurance. Quantal response models (McKelvey and Palfrey, 1995) assume that the players' actions are perturbed equilibrium policies, altered by stochastic mistakes.

Another class of non-equilibrium models depend on the computation of a finite number of best response strategies up to a certain level. Level-k thinking assumes that each player believes all of her opponents operate at the $(k-1)$ th level of intelligence. On the other hand, in cognitive hierarchy models, each player believes that her opponents' levels follow

a Poisson distribution. Both of these models are examined in Chong *et al.* (2016). In the context of CPS, the authors in Li *et al.* (2016) use a cognitive hierarchy approach to train autonomous vehicles in real-world situations. Cognitive hierarchy was also used for cyber–physical security in Abuzainab *et al.* (2016), for the problem of distributed uplink random access for the Internet of Things.

However, in the control discipline, the non-equilibrium game theory constitutes a useful tool for modeling the class of problems arising from the interaction of strategic agents with bounded rationality. In particular, in the work of Kanellopoulos and Vamvoudakis (2019) and Vamvoudakis and Kanellopoulos (2019) is formulated a CPS operating in an adversarial environment by considering a general non-zero-sum, nonlinear differential game and are derived the expressions for the level-k best response. Furthermore, the work of Kokolakis *et al.* (2020) developed a level-k framework for addressing the problem of tracking an actively evading target by employing a team of coordinating UAVs where the rationality policies are computed by using a RL-based architecture.

6.2 Non-Equilibrium Dynamic Games and RL

This subsection concerns non-equilibrium dynamic games for CPS via a cognitive hierarchy approach. Initially, we describe the mathematical representation that captures the behavior of the competing agents in its general setting, as well as in simple – and easier to tackle – scenarios. Next, we construct a framework that introduces level-k thinking strategies in differential games. Finally, we will propose an iterative approach to deriving boundedly rational decision policies by leveraging structural connections between cognitive hierarchy theory and RL techniques.

6.2.1 Nash Games

Initially, we describe the differential game structure along with the Nash equilibrium solutions. Let $\mathcal{N}$ denote the set of all players, and $\mathcal{N}_{-i} = \mathcal{N} \backslash \{i\}$ the set containing all players except player i. In order

to investigate the interactions between non-cooperative players in dynamically changing environments, we define an underlying nonlinear time-invariant dynamical system upon which the agents act as (3.8).

Each player aims to minimize a cost functional of the form,

$$J_i(u_1,\dots,u_N) = \int_0^\infty r_i(x,u_1,\dots,u_N)\mathrm{d}\tau \\ := \int_0^\infty \Big(Q_i(x) + u_i^{\mathrm{T}} R_{ii} u_i + \sum_{j\in\mathcal{N}_{-i}} u_j^{\mathrm{T}} R_{ij} u_j\Big)\mathrm{d}\tau, \quad \forall i \in \mathcal{N},$$

Due to the coupled nature of the cost functionals and the environment dynamics, there is a plethora of different approaches to the "solution" of a differential game. The most common one, which constitutes a rational equilibrium concept, is the Nash equilibrium solution (Başar and Olsder, 1999; Hespanha, 2017).

Definition 6.1. A tuple of decision policies $(u_1^\star,\dots,u_i^\star,\dots,u_N^\star)$ constitute a Nash equilibrium if it holds that,

$$J_i(u_1^\star,\dots,u_i^\star,\dots,u_N^\star) \leqslant J_i(u_1^\star,\dots,u_i,\dots,u_N^\star), \quad \forall i \in \{1,\dots,N\}.$$

The derivation of the Nash policies $u_i^\star,\ i\in\{1,\dots,N\}$, corresponds to solving the coupled minimization problems,

$$J_i(u_i^\star, u_{-i}^\star) = \min_{u_i} J_i(u_i, u_{-i}^\star), \quad i\in\{1,\dots,N\},$$

where for brevity we define $u_{-i} = (u_1,\dots,u_{i-1},u_{i+1},\dots,u_N)$.

Following methods from the theory of optimal control (Lewis *et al.*, 2012a), the goal of the decision makers is to evaluate the optimal value functions $V_i(x),\ i\in\{1,\dots,N\}$ such that,

$$V_i(x(t)) = \min_{u_i} \int_t^\infty r_i(x,u_i,u_{-i}^\star)\mathrm{d}\tau, \quad \forall x,\ t\geqslant 0.$$

This is achieved with the help of the Hamiltonian function, defined $\forall i\in\{1,\dots,N\}$, as,

$$H_i(x,u_i,u_{-i}) = r_i(x,u_1,\dots,u_N) + \nabla V_i^{\mathrm{T}}\Big(f(x) + \sum_{j\in\mathcal{N}} g_j(x)u_j\Big). \quad (6.1)$$

It is known that minimizing inputs on the system minimize the Hamiltonian (Lewis *et al.*, 2012a). Consequently, this allows for the use of stationarity conditions to derive the Nash policies,

$$\frac{\partial H}{\partial u_i} = 0 \Rightarrow u_i^\star(x) = -R_{ii}^{-1} g_i^{\mathrm{T}}(x) \frac{\partial V_i^\star}{\partial x}. \tag{6.2}$$

Substituting (6.2) into (6.1), yields the following coupled HJ system of N equations,

$$\begin{aligned} Q_i(x) + \frac{1}{4} \sum_{j \in \mathcal{N}} \nabla V_j^{\mathrm{T}} g_j(x) (R_{jj}^{-1})^{\mathrm{T}} R_{ij} R_{jj} g_j^{\mathrm{T}}(x) \nabla V_j \\ + (\nabla V_i)^{\mathrm{T}} \left(f(x) - \frac{1}{2} \sum_{j \in \mathcal{N}} g_j(x) R_{jj}^{-1} g_j^{\mathrm{T}}(x) \nabla V_j \right) = 0. \end{aligned} \tag{6.3}$$

A specific case of the general framework presented above that allows us to exhibit clearer results while capturing a large class of applications is the linear quadratic ZS case. In this scenario, consider that $\mathcal{N} = \{1, 2\}$, i.e., there are two players, whose cost functionals are such that $J_1 = -J_2 = J$. This formulation is extremely useful in the field of CPS, in which the competing players correspond to a defending agent (or team of defenders), whose decision vector is $u(t) \in \mathbb{R}^m$ and an attacking agent whose decision is denoted by $d(t) \in \mathbb{R}^l$. Furthermore, the environment evolves according to the simplified dynamics,

$$\dot{x}(t) = Ax(t) + Bu(t) + Kd(t), \quad x(0) = x_0, \ t \geqslant 0, \tag{6.4}$$

and the cost is quadratic in its arguments,

$$J(u, d) = \frac{1}{2} \int_0^\infty (x^{\mathrm{T}} M x + u^{\mathrm{T}} R u - \gamma^2 \|d\|) \mathrm{d}t.$$

Due to the ZS property of the players' costs, we can redefine the problem of deriving the Nash equilibrium, as one of solving for the mini-max policies, i.e.,

$$V^\star(x) = \min_u \max_d \int_0^\infty (x^{\mathrm{T}} M x + u^{\mathrm{T}} R u - \gamma^2 \|d\|) \mathrm{d}\tau, \quad \forall x.$$

The value function for the linear quadratic case is known to also be a quadratic in the states function, i.e., $V^\star(x) = x^{\mathrm{T}} P x$. The optimal

kernel P satisfies a simplified version of (6.3),

$$A^{\mathrm{T}}P + PA - PBR^{-1}B^{\mathrm{T}}P + \frac{1}{\gamma^2}PKK^{\mathrm{T}}P + M = 0. \tag{6.5}$$

Finally, the Nash policies can be found after solving (6.5) for P and substituting in,

$$u^{\star}(x) = -R^{-1}B^{\mathrm{T}}Px, \quad d^{\star}(x) = \frac{1}{\gamma}K^{\mathrm{T}}Px, \quad \forall x.$$

Computational and cognitive limitations have been shown to prevent players from reaching the Nash equilibrium solution even in simple scenarios (Daskalakis *et al.*, 2009). This behavior, which has been observed in most experimental setups, has lead to the formulation of various non-equilibrium solution concepts that aim to better approximate realistic agents. In our work, we introduce bounded rationality concepts via the level-k thinking and cognitive hierarchy frameworks. Towards this, we will define strategies of lower "level-k" thinkers that perform a certain number of steps of strategic thinking and show methods to derive the distribution of beliefs for their abilities.

6.3 Games with Adversaries

In this subsection, we construct a framework that introduces level-k thinking strategies in differential games. In particular, we will propose an iterative approach to deriving boundedly rational decision policies by leveraging structural connections between cognitive hierarchy theory and reinforcement learning techniques.

6.3.1 Level-k Thinking Model

Consider a player $i \in \mathcal{N}$ belonging to a cognitive type $k \in \mathbb{Z}^{+}$. This player is able to perform k-steps of iterative strategic thinking while her beliefs about the behavior of the rest of the players are constructed based on a fixed, full-support distribution, e.g., according to the work of Camerer *et al.* (2004), a Poisson distribution.

Level-0 Policies. The approach of iterative best responses to model bounded rationality – such as the one we employ in this work – requires the existence of a common anchor strategy; a policy that a level-0 player will adopt. It is clear that the assumption that the structure of the anchor policy is common knowledge to all the players is a strict one. However, in various scenarios, those policies can be derived as completely random responses or as focal policies for the game without loss of generality of the framework (Strzalecki, 2014).

Level-k Policies. All high-level players follow a model inspired by Policy Iteration schemes in order to compute their strategies. Specifically, given the cognitive level-k of an agent, their thinking process is initialized by assigning level-0 intelligence to the opponents and computing the optimal response to this environment. Subsequently, they increase the level of the opponents via a similar process until a predefined distribution of lower levels has been embedded on the rest of the players. This process involves the solution of a number of single agent optimization problems.

The single agent optimization that a level-k player performs corresponds to solving the following HJB equation with respect to a non-equilibrium value function $V^k(x)$: $\mathbb{R}^n \to \mathbb{R}$, which is conditioned by the belief probabilities of the agent of level-k, regarding the intelligence level of player $h \in \mathcal{N}\backslash\{i\}$, denoted as $b^k(h)$,

$$Q(x) + u_i^{k\mathrm{T}} R_{ii} u_i^k + \sum_{j\in\mathcal{N}_{-i}} \sum_{h=0}^{k-1} (b^k(h)u_j)^{h\mathrm{T}} R_{ij} b^k(h) u_j^h$$
$$+ (\nabla V_i^k)^{\mathrm{T}} \left(f(x) + g(x)u_i^k + \sum_{j\in\mathcal{N}_{-i}} \sum_{h=0}^{k-1} (g(x)b^k(h)u_j) \right) = 0,$$

which yields the level-k policy given by,

$$u_i^k(x) = -R_{ii}^{-1} g_i^{\mathrm{T}} \nabla V_i^k, \quad \forall x.$$

A simplified version of this bounded rationality principle can be extracted for linear quadratic ZS games. For ease of exposition, we will focus on the iterative scheme that corresponds to the minimizing player of system (6.4). Initially, for CPS applications, a focal strategy for a minimizing (defending) player, is taken as one that assumes attacker-free

environments. This, leads to a level-0 policy derived by the one-sided optimal control problem,

$$V_u^0(x_0) = \min_u \int_0^\infty (x^{\mathrm{T}} M x + u^{\mathrm{T}} R u)\mathrm{d}\tau, \quad \forall x_0. \tag{6.6}$$

The optimal decision for (6.6) given (6.4) with $d_i = 0$ is,

$$u^0(x) = -R^{-1}B^{\mathrm{T}}\frac{\partial V_u^0(x)}{\partial x} = -R^{-1}B^{\mathrm{T}}P_u^0 x, \quad \forall x,$$

where the value function is taken as quadratic in the states $V_u^0(\cdot) = x^{\mathrm{T}}P_u^0 x$ and the kernel P_u^0 solves the Riccati equation, $A^{\mathrm{T}}P_u^0 + P_u^0 A + M - P_u^0 B R^{-1} B^{\mathrm{T}} P_u^0 = 0$.

The iterative process described above, leads to the formulation of high-level optimal control problems described by,

$$V_u^k(x_0) = \min_u \int_0^\infty (x^{\mathrm{T}} M x + u^{\mathrm{T}} R u - \gamma^2 \|d^{k-1}\|^2)\mathrm{d}\tau, \quad \forall x_0,$$

where the $k-1$-th attack d^{k-1} corresponds to an adversary of limited cognitive ability. Solving this problem through classical optimal control tools yields the level-k Riccati equation,

$$\begin{aligned}\left(A + \frac{1}{\gamma^2}KK^{\mathrm{T}}P_d^{k-1}\right)^{\mathrm{T}} P_u^k + P_u^k\left(A + \frac{1}{\gamma^2}KK^{\mathrm{T}}P_d^{k-1}\right)\\ + \left(M - \frac{1}{\gamma^2}P_d^{k-1}KK^{\mathrm{T}}P_d^{k-1}\right) - P_u^k B R^{-1}B^{\mathrm{T}}P_u^k = 0,\end{aligned} \tag{6.7}$$

with its associated level-k defense strategy given by,

$$u^k(x) = -R^{-1}B^{\mathrm{T}}P_u^k x. \tag{6.8}$$

Similarly, the attacking agent can calculate their level-k response through the solution of the Riccati equation,

$$\begin{aligned}(A - BR^{-1}B^{\mathrm{T}}P_u^k)^{\mathrm{T}}P_d^k + P_d^k(A - BR^{-1}B^{\mathrm{T}}P_u^k)\\ + (M + P_u^k B R^{-1}B^{\mathrm{T}}P_u^k) + \frac{1}{\gamma^2}P_d^k KK^{\mathrm{T}}P_d^k = 0,\end{aligned} \tag{6.9}$$

and the optimal attack policy given by,

$$d^k(x) = \frac{1}{\gamma^2}K^{\mathrm{T}}P_d^k x, \quad \forall x. \tag{6.10}$$

With this iterative procedure, a defending agent is able to compute the strategies of the adversaries with finite cognitive abilities, for a given number of levels. For the linear quadratic zero-sum case, proper conditions that guarantee convergence to the Nash equilibrium of the game as the levels of cognition of the players increase have been derived and are summarized in the theorem that follows.

Theorem 6.1 (Kanellopoulos and Vamvoudakis, 2019, Thm. 1)**.** Consider the pairs of strategies at a specific cognitive level-k, given by (6.8) and (6.7) for the defender, and (6.10) and (6.9) for the adversary. The policies converge to a Nash equilibrium for higher levels if the following conditions hold as the levels increase,

$$(P_d^{k-1} + P_d^k - P_u^{k+1})KK^{\mathrm{T}}(P_d^k - P_d^{k-1}) \succ 0,$$
$$(3P_u^{k-1} - P_u^k + P_d^k)BR^{-1}B^{\mathrm{T}}(P_u^k - P_u^{k-1}) \succ 0. \qquad \blacksquare$$

6.3.2 Model-Free Learning for Interaction with Level-k Attacks

By leveraging the Q-learning algorithm developed in Subsection 4.2, we are able to learn the best responses of all level-k agents without explicit knowledge of the physics of the system. For ease of exposition, we shall introduce the notation $V_j^k(\cdot)$, $a_j^k(x)$, and $H_j^k(x, a_j^k, \nabla V_j^k)$ to mean the value function, action policy, and Hamiltonian of a level-k agent, where $j \in \{u, d\}$, i.e., the defender and the adversary.

The action-dependent function can be defined to be,

$$Q_j^k(x, a_j^k) \coloneqq V_j^k(x) + H_j^k(x, a_j^k, \nabla V_j^k), \quad \forall x, a_j^k,\ j \in \{u, d\}. \tag{6.11}$$

We can rewrite the action-dependent function (6.11) in a compact quadratic in the state and action form as,

$$Q_j^k(x, a_j^k) = (U_j^k)^{\mathrm{T}} \begin{bmatrix} Q_{j,\mathrm{xx}}^k & Q_{j,\mathrm{xa}}^k \\ Q_{j,\mathrm{ax}}^k & Q_{j,\mathrm{aa}}^k, \end{bmatrix} U_j^k \coloneqq (U_j^k)^{\mathrm{T}} \tilde{Q}_j^k U_j^k,$$
$$\forall x, a_j^k,\ j \in \{u, d\}, \tag{6.12}$$

where for the level-k defender's problem, $j \coloneqq u$, we have that $U_u^k = \begin{bmatrix} x^{\mathrm{T}} & (u^k)^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$,

$$\begin{aligned} Q_{u,\mathrm{xx}}^k &= \left(A + \frac{1}{\gamma^2} K K^{\mathrm{T}} P_d^{k-1}\right)^{\mathrm{T}} P_u^k + P_u^k \left(A + \frac{1}{\gamma^2} K K^{\mathrm{T}} P_d^{k-1}\right) \\ &\quad + \left(M - \frac{1}{\gamma^2} P_d^{k-1} K K^{\mathrm{T}} P_d^{k-1}\right) - P_u^k B R^{-1} B^{\mathrm{T}} P_u^k + P_u^k, \\ Q_{u,\mathrm{xa}}^k &= B^{\mathrm{T}} P_u^k, \quad Q_{u,\mathrm{ax}}^k = B P_u^k, \quad Q_{u,\mathrm{aa}}^k = R. \end{aligned}$$

Accordingly, for the level-k adversarial problem, $j \coloneqq d$, we have that $U_d^k = \begin{bmatrix} x^{\mathrm{T}} & (d^k)^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$,

$$\begin{aligned} Q_{d,\mathrm{xx}}^k &= (A - B R^{-1} B^{\mathrm{T}} P_u^k)^{\mathrm{T}} P_d^k + P_d^k (A - B R^{-1} B^{\mathrm{T}} P_u^k) \\ &\quad + (M + P_u^k B R^{-1} B^{\mathrm{T}} P_u^k) + \frac{1}{\gamma^2} P_d^k K K^{\mathrm{T}} P_d^k, \\ Q_{d,\mathrm{xa}}^k &= K^{\mathrm{T}} P_d^k, \quad Q_{d,\mathrm{ax}}^k = K P_d^k, \quad Q_{d,\mathrm{aa}}^k = -\gamma^2. \end{aligned}$$

The action for each agent at every level can be found by solving $\frac{\partial Q_j^k(x, a_j^k)}{\partial a_j^k} = 0$ as,

$$a_j^k(x) = -(Q_{j,\mathrm{aa}}^k)^{-1} Q_{j,\mathrm{ax}}^k x, \quad \forall x,\ j \in \{u, d\}. \tag{6.13}$$

We will now use an actor/critic structure to tune the parameters online by utilizing an integral reinforcement learning approach (Vrabie *et al.*, 2013). The level-k critic approximator shall approximate the action-dependent function (6.12), while the level-k actor shall approximate the appropriate defense or attack policy (6.13).

In order to do that, we write the action-dependent function as,

$$Q_j^k(x, a_j^k) = \mathrm{vech}(\tilde{Q})^{\mathrm{T}} (U_j^k \otimes U_j^k), \quad \forall x, a_j^k,\ j \in \{u, d\}, \tag{6.14}$$

where the Kronecker product-based polynomial quadratic polynomial function $(U_j^k \otimes U_j^k)$ is reduced to guarantee linear independence of the elements. The vectorized action-dependent function (6.14) can be described in terms of the ideal weights $W_j^k = \mathrm{vech}(\tilde{Q})$, leading to the compact form $Q_j^k(x, a_j^k) = (W_j^k)^{\mathrm{T}} (U_j^k \otimes U_j^k)$.

Since the ideal weights are unknown, we will consider the following estimated level-k action-dependent function, according to the estimated

critic weights, $\hat{W}_j^k = \text{vech}(\hat{\hat{Q}})$,

$$\hat{Q}_j^k(x, a_j^k) = (\hat{W}_j^k)^{\mathrm{T}}(U_j^k \otimes U_j^k), \quad \forall x, a_j^k,\ j \in \{u, d\},$$

as well as the estimated actor approximator,

$$\hat{a}_j^k(x) = (\hat{W}_{a,j}^k)^{\mathrm{T}} x, \quad \forall x,\ j \in \{u, d\},$$

where the state x is serving as the basis for the actor approximator and $\hat{W}_{a,j}^k$ denotes the weight estimate of the level-k agent's policy.

The action-dependent function with the ideal weights, (6.12), has been shown in 4.2 to satisfy the integral form of the Hamilton–Jacobi–Bellman equation,

$$\begin{aligned} Q_j^k(x(t), a_j^k(t)) = Q_j^k(x(t - T_{\text{IRL}}), a_j^k(t - T_{\text{IRL}})) \\ - \int_{t-T_{\text{IRL}}}^{t} (x^{\mathrm{T}} \bar{M}_j^k x + (a_j^k)^{\mathrm{T}} \bar{R}_j a_j^k) \mathrm{d}\tau, \\ \forall t \geqslant 0, j \in \{u, d\}, \end{aligned}$$

where $T_{\text{IRL}} \in \mathbb{R}^+$ is the sampling period of the algorithm. For the defender, we define $\bar{M}_u^k \coloneqq M - \frac{1}{\gamma^2} P_d^{k-1} K K^{\mathrm{T}} P_d^{k-1}$, and $\bar{R}_u \coloneqq R$. Similarly, for the adversary we define, $\bar{M}_d^k \coloneqq M + P_u^k B R^{-1} B^{\mathrm{T}} P_u^k$, and $\bar{R}_d \coloneqq -\gamma^2$.

We will now define the error based on the current estimate of the action-dependent function that we wish to drive to zero as,

$$\begin{aligned} e_j^k &= \hat{Q}_j^k(x(t), a_j^k(t)) - \hat{Q}_j^k(x(t - T_{\text{IRL}}), a_j^k(t - T_{\text{IRL}})) \\ &\quad + \int_{t-T_{\text{IRL}}}^{t} (x^{\mathrm{T}} \bar{M}_j^k x + (a_j^k)^{\mathrm{T}} \bar{R}_j a_j^k) \mathrm{d}\tau \\ &= (\hat{W}_j^k)^{\mathrm{T}}(U_j^k(t) \otimes U_j^k(t)) - (\hat{W}_k^k)^{\mathrm{T}}(U_j^k(t - T_{\text{IRL}}) \otimes U_j^k(t - T_{\text{IRL}})) \\ &\quad + \int_{t-T_{\text{IRL}}}^{t} (x^{\mathrm{T}} \bar{M}_j^k x + (a_j^k)^{\mathrm{T}} \bar{R}_j a_j^k) \mathrm{d}\tau, \end{aligned}$$

as well as the policy error $e_{j,a}^k = (W_{a,j}^k)^{\mathrm{T}} x + (\hat{Q}_{j,\text{aa}}^k)^{-1} \hat{Q}_{j,\text{ax}} x,\ \forall x,\ j \in \{u, d\}$, where the appropriate elements of the $\hat{Q}$ matrix will be extracted from the critic estimate $\hat{W}_j^k$.

Defining the squared error functions for the critic and the actor weights $K_1 = \frac{1}{2}\|e_j^k\|^2$, and $K_2 = \frac{1}{2}\|e_{j,a}^k\|^2$ respectively, we derive the

tuning rules by applying normalized gradient descent as,

$$\dot{\hat{W}}_j^k = -\alpha \frac{\sigma_j^k}{(1+(\sigma_j^k)^{\mathrm{T}}\sigma_j^k)^2}(e_j^k)^{\mathrm{T}}, \quad \forall t \geqslant 0,\ j \in \{u,d\}$$

$$\dot{\hat{W}}_{j,a}^k = -\alpha_a x (e_{j,a}^k)^{\mathrm{T}}, \quad \forall t \geqslant 0,\ j \in \{u,d\},$$

where $\sigma_j^k = (U_j^k(t) \otimes U_j^k(t)) - (U_j^k(t - T_{\mathrm{IRL}}) \otimes U_j^k(t - T_{\mathrm{IRL}}))$, and $\alpha,\ \alpha_a \in \mathbb{R}^+$ are tuning gains.

Remark 6.1. The persistence of excitation condition can be guaranteed by adding exploration noise in the control input (Ioannou and Fidan, 2006). □

Remark 6.2. Although the described algorithm does not require explicit knowledge of the system matrices, both opponents require knowledge of the policies of the previous levels. This result agrees with similar claims from non-equilibrium game theory about the difficulty of operating in higher cognitive levels since for an agent to learn a level-k best response, first she has to successfully learn all the previous best responses of all the other agents acting on the system. □

6.3.3 Estimation of the Adversarial Levels

In this subsection, we will propose an algorithmic framework where a defender interacts with adversaries of different cognitive levels for a predefined time period $T_{\mathrm{int}} \in \mathbb{R}_+$. To further generalize our results, we will not restrict the adversaries to statically use only a specific level policy throughout the interaction. This will account for opponents who manage to increase their cognitive level after their initial attack, who try to deceive the defender, or even adversaries that do not follow a best response policy. Also, as we noted in Subsection 6.3.2, the defender computes the level-k responses from a model-free learning algorithm utilizing fictitious inputs. Consequently, instead of identifying the exact distribution of the levels of intelligence in the adversarial environment, we fit an arbitrary attack input to the set of beliefs over the different level-k model-free policies.

Initially, we assume that the defender is able to directly measure the effect of the attack input on the system. We define the error between

the actual measured adversarial input, denoted $f(t)$, and the adversarial input of a level-k adversary as,

$$r^k = \int_{t-T_{\text{int}}}^{t} \|f(\tau) + (Q_{d,\text{aa}}^k)^{-1} Q_{d,\text{xa}} x(\tau)\| \mathrm{d}\tau, \quad \forall t \geqslant 0,\, k \in \{1, \ldots, \mathcal{K}\} \tag{6.15}$$

where $\mathcal{K}$ is the maximum level the defender has computed. We note that (6.15) is the norm of the "distance" of the measured attack from each cognitive level during the time of interaction with the particular adversary.

After each interaction period $[t - T_{\text{int}}, t]$, we will stack the elements r^k into a vector of the form, $\mathbf{r} = \begin{bmatrix} r^1 & r^2 & \ldots & r^{\mathcal{K}} \end{bmatrix}$. Motivated by (Sutton and Barto, 2018), we will use the softmax function to map the error vector $\mathbf{r}$ to a reinforcement-like signal. However, we need to reward those elements of the vector that are closer to the appropriate level, i.e., to reinforce the minimum element. As a result, we apply the softmax function to each element of $\mathbf{r}$, as follows

$$\sigma^k = \frac{e^{-r^k}/\tau}{\sum_{i=1}^{\mathcal{K}} e^{-r^i}/\tau},$$

where $\tau \in \mathbb{R}_+$ is the softmax temperature parameter. This will give us, $\boldsymbol{\sigma} = \begin{bmatrix} \sigma^1 & \sigma^2 & \ldots & \sigma^{\mathcal{K}} \end{bmatrix}^{\mathrm{T}}$. We model the distribution of the players over the different levels as a Poisson distribution with the following probability mass function, $p^k = \frac{\lambda^k e^{-\lambda}}{k!}$, $\lambda \in \mathbb{R}_+$. This probability shall also define the belief of the defender about the relative proportion of level-k adversaries as,

$$b^k = \frac{p^k}{\sum_{i=1}^{\mathcal{K}} p^i}.$$

Our goal is to evaluate the parameter λ. To this end, we use the following update rule for the mean of the observations,

$$\lambda^+ = \lambda + \frac{(\tilde{\mathbf{K}}^{\mathrm{T}} \boldsymbol{\sigma}) n}{n+1},$$

where $\tilde{\mathbf{K}} = \begin{bmatrix} 1 & 2 & \ldots & \mathcal{K} \end{bmatrix}$ is a vector containing the indexes of the levels we have trained and n is the number of different agents we have interacted with.

6.3.4 Further Reading

More details about non-equilibrium dynamic games and cyber–physical security are given in Kanellopoulos and Vamvoudakis (2019) and Vamvoudakis and Kanellopoulos (2019).

7

Applications to Autonomous Vehicles

In this section, we present applications of synchronous-RL based decision-making mechanisms to autonomous vehicles. In Subsection 7.1, by leveraging ideas developed in Section 4, we shall endow the intelligent autonomous robots with online model-free algorithms to allow the kinodynamic motion planning in unknown cluttered environments. Subsection 7.2 extends the results displayed in Section 6 to introduce the non-equilibrium pursuit-evasion games. In particular, we devise a novel framework for coordinating a team of bounded rational UAVs to track an adversarial target while learning the target's profile of intelligence by employing a data-driven algorithm utilizing measurements along the target's trajectory.

7.1 Kinodynamic Motion Planning

This subsection presents an online kinodynamic motion planning algorithmic framework using asymptotically optimal rapidly-exploring random tree (RRT*) (Goretkin *et al.*, 2013; Perez *et al.*, 2012; Webb and Berg, 2013) and CT Q-learning, which we term as RRT-Q. We formulate a model-free Q-based advantage function and we utilize integral RL to develop tuning laws for the online approximation of the optimal cost and

the optimal policy of CT linear systems. Furthermore, a terminal state evaluation procedure is introduced to facilitate the online implementation. Finally, we propose a static obstacle augmentation and a local replanning framework, which are based on topological connectedness, to locally recompute the robot's path and ensure collision-free navigation.

7.1.1 Problem Formulation

Consider the linear time-invariant CT system given by (4.1). For driving our system from an initial state x_0 to a final state $x(T) = x_\mathrm{r}$, we define the difference between the state $x(t)$ and the state x_r, as our new state $\bar{x}(t) := x(t) - x_\mathrm{r}$. The final time is denoted by $T \in \mathbb{R}^+$. Similarly, we define our new control as, $\bar{u}(t) := u(t) - u_\mathrm{r}$, with $u_\mathrm{r} = u(T)$. The new system becomes,

$$\begin{aligned}\dot{\bar{x}}(t) &= \dot{x}(t) - \dot{x}_\mathrm{r} \\ &= A\bar{x}(t) + B\bar{u}(t), \quad \bar{x}_0 = x_0 - x_\mathrm{r}, \quad t \geqslant 0. \end{aligned} \tag{7.1}$$

We define an energy cost functional,

$$J(\bar{x}; \bar{u}; t_0, \mathrm{T}) = \phi(T) + \frac{1}{2}\int_{t_0}^{T} (\bar{x}^\mathrm{T} M \bar{x} + \bar{u}^\mathrm{T} R \bar{u})\mathrm{d}\tau, \quad \forall t_0,$$

where $\phi(T) := (1/2)\bar{x}^\mathrm{T}(T)P(T)\bar{x}(T)$ is the terminal cost with $P(T) := P_T \in \mathbb{R}^{n\times n} \succ 0$ the final Riccati matrix, and $M \in \mathbb{R}^{n\times n} \succeq 0$ and $R \in \mathbb{R}^{m\times m} \succ 0$ are the user-defined matrices that penalize the states and the control input, respectively.

Assumption 7.1. We assume that the unknown pairs (A, B) and $(\sqrt{M}, A)$ are controllable and detectable, respectively. □

We are interested in finding an optimal control $\bar{u}^\star$ such that, $J(\bar{x}; \bar{u}^\star; t_0, T) \leqslant J(\bar{x}; \bar{u}; t_0, T)$, $\forall \bar{x}, \bar{u}$, which can be described by the minimization problem $J(\bar{x}; \bar{u}^\star; t_0, T) = \min_{\bar{u}} J(\bar{x}; \bar{u}; t_0, T)$ subject to (7.1). In other words, we want to obtain the optimal value function $V^\star$ that is defined by,

$$V^\star(\bar{x}; t_0, T) := \min_{\bar{u}} \left(\phi(T) + \frac{1}{2}\int_{t_0}^{T} (\bar{x}^\mathrm{T} M \bar{x} + \bar{u}^\mathrm{T} R \bar{u})\mathrm{d}\tau \right), \quad \forall \bar{x}, \tag{7.2}$$

but without any information about the system dynamics.

Consider the closed obstacle space $\mathcal{X}_{\text{obs}} \subset \mathcal{X}$. The free space is defined as the relative complement of the obstacle space in the state space, $\mathcal{X}_{\text{free}} := (\mathcal{X}_{\text{obs}})^{\text{C}} = \mathcal{X} \backslash \mathcal{X}_{\text{obs}}$ which is an open set (Rudin, 1964). The output of an algorithm is designed to efficiently search nonconvex, high-dimensional spaces by randomly building a space-filling tree (e.g., RRT) that will produce the global path $\pi\left(x_{0,i}, x_{\text{r},i}\right) \in \mathbb{R}^{2(k \times n)}$, for $i = 1, \ldots, k$ with $i \in \mathbb{N}$. The global path $\pi\left(x_{0,i}, x_{\text{r},i}\right)$ will include the initial states $\mathcal{X}_0 = x_{0,i}$ for all i, with $\mathcal{X}_0 \in \mathbb{R}^{k \times n} \subset \mathcal{X}_{\text{free}}$, and the final states $\mathcal{X}_{\text{G}} = x_{\text{r},i}$ for all i, with $\mathcal{X}_{\text{G}} \in \mathbb{R}^{k \times n} \subseteq \mathcal{X}_{\text{free}}$. The algorithm will also provide an initial graph $\mathcal{G} = (V, E)$, where V is the initial set of nodes, $V_{\mathcal{G}} = |V|$ its cardinality, and E the initial set of edges. With a slight abuse of notation, we will refer to the nodes $v \in V$ as states $x \in \mathcal{X}$.

A direct relationship of the global path π in the initial graph is given by the tree $\mathcal{T} = (V_{\mathcal{T}}, E_{\mathcal{T}}) \subseteq \mathcal{G}$. Furthermore, the augmented obstacle space $\mathcal{X}_{\text{obs}}^{\text{aug}} = f\left(t; D_{\text{rob}}\right)$ will change through time depending on the kinodynamic distance D_{rob} of the robot motion at every i-TPBVP. Similarly, the diminished open free space is defined as $\mathcal{X}_{\text{free}}^{\text{dim}} := (\mathcal{X}_{\text{obs}}^{\text{aug}})^{\text{C}} = \mathcal{X} \backslash \mathcal{X}_{\text{obs}}^{\text{aug}}$ Our work will formulate an online implementation framework of safe kinodynamic motion planning, given the global path and the initially randomly sampled graph, with completely unknown dynamics as described in (7.1) since we are solving a finite horizon optimal control problem with a free final state, and we are also setting a new state $\bar{x}(t)$, as the difference of the current state $x(t)$ with the desired state x_{r}, we can make the following approximation $\lim_{t \to T} x(T) \approx x_{\text{r}}$. This means that the final state $x(T)$ may not obtain the exact desired state x_{r} value. Still, the system may be fast enough to approximate the desired state, and stay there, until the end of the fixed finite horizon T. To address this problem, we define the initial distance as the n-norm of the initial state x_0 and the desired state x_{r} as,

$$D_0\left(\bar{x}_0\right) := \left\|\bar{x}_0\right\|_n, \quad \forall \bar{x}_0. \tag{7.3}$$

We shall then measure the relative distance at time $t \geqslant 0$ with the n-norm of the current state $x(t)$ and the desired state x_{r} as,

$$D(\bar{x}; t) := \|\bar{x}(t)\|_n, \quad \forall x,\ t. \tag{7.4}$$

Let us also define the distance error of (7.3) and (7.4) as,

$$e_{\mathrm{d}}\left(\bar{x}_0, \bar{x}; t\right) := \left|D_0\left(\bar{x}_0\right) - D(\bar{x}, t)\right|.$$

7.1.2 Finite Horizon Optimal Control

Define the Hamiltonian with respect to (7.1) and (7.2) as,

$$\mathcal{H}(\bar{x}; \bar{u}; \lambda) = \frac{1}{2}(\bar{x}^{\mathrm{T}} M \bar{x} + \bar{u}^{\mathrm{T}} R \bar{u}) + \lambda^{\mathrm{T}}(A\bar{x} + B\bar{u}), \quad \forall \bar{x}, \bar{u}, \lambda.$$

In order to solve the finite horizon optimal control problem (7.2), by using the sweep method (Bryson, 2018) and $\lambda = (\partial V^{\star}/\partial \bar{x})$, we derive the HJB equation,

$$-\frac{\partial V^{\star}}{\partial t} = \frac{1}{2}(\bar{x}^{\mathrm{T}} M \bar{x} + \bar{u}^{\mathrm{T}} R \bar{u}) + \frac{\partial V^{\star}}{\partial \bar{x}}^{\mathrm{T}} (A\bar{x} + B\bar{u}), \quad \forall \bar{x}.$$

Since our system (7.1) is linear, we can write the value function in a quadratic in the state $\bar{x}$ form as,

$$V^{\star}(\bar{x}; t) = \frac{1}{2}\bar{x}^{\mathrm{T}} P(t) \bar{x}, \quad \forall \bar{x},\ t \geqslant t_0, \tag{7.5}$$

where $P(t) \in \mathbb{R}^{n \times n} \succ 0$ solves the Riccati equation,

$$-\dot{P}(t) = M + P(t)A + A^{\mathrm{T}} P(t) - P(t) B R^{-1} B^{\mathrm{T}} P(t). \tag{7.6}$$

Hence, the optimal control is,

$$\bar{u}^{\star}(\bar{x}; t) = -R^{-1} B^{\mathrm{T}} P(t) \bar{x} \quad \forall \bar{x},\ t. \tag{7.7}$$

7.1.3 Model-Free TPBVP Formulation

Let us now define the following advantage function:

$$\begin{aligned}
\mathcal{Q}(\bar{x}; \bar{u}; t) &:= V^{\star}(\bar{x}; t) + \mathcal{H}\left(\bar{x}; \bar{u}; \frac{\partial V^{\star}}{\partial t}, \frac{\partial V^{\star}}{\partial \bar{x}}\right) \\
&= V^{\star}(\bar{x}; t) + \frac{1}{2}\bar{x}^{\mathrm{T}} M \bar{x} + \frac{1}{2}\bar{u}^{\mathrm{T}} R \bar{u} \\
&\quad + \bar{x}^{\mathrm{T}} P(t)(A\bar{x} + B\bar{u}) + \frac{1}{2}\bar{x}^{\mathrm{T}} \dot{P}(t) \bar{x}, \quad \forall \bar{x}, \bar{u}, t \geqslant 0,
\end{aligned} \tag{7.8}$$

where $\mathcal{Q}(\bar{x}; \bar{u}; t) \in \mathbb{R}^{+}$ is an action-dependent value.

Next, we define $U := [\bar{x}^{\mathrm{T}} \bar{u}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{(n+m)}$ to express the Q-function (7.8) in a compact form as,

$$\mathcal{Q}(\bar{x}; \bar{u}; t) = \frac{1}{2} U^{\mathrm{T}} \begin{bmatrix} Q_{\mathrm{xx}}(t) & Q_{\mathrm{xu}}(t) \\ Q_{\mathrm{ux}}(t) & Q_{\mathrm{uu}}(t) \end{bmatrix} U := \frac{1}{2} U^{\mathrm{T}} \overline{\mathcal{Q}}(t) U, \tag{7.9}$$

where $Q_{\mathrm{xx}}(t) = \dot{P}(t) + P(t) + M + P(t)A + A^{\mathrm{T}}P(t) + P(t)B$ $Q_{\mathrm{xu}}(t) = Q_{\mathrm{ux}}^{\mathrm{T}}(t) = P(t)B$, and $Q_{\mathrm{uu}} = R$, with $\mathcal{Q}$: $\mathbb{R}^{n+m} \times \mathbb{R}^{(n+m)\times(n+m)} \to \mathbb{R}^{+}$. Note that, since the Riccati matrix is symmetric, $\bar{x}^{\mathrm{T}}P(t)A\bar{x} = (1/2)\bar{x}^{\mathrm{T}}(P(t)A + A^{\mathrm{T}}P(t))\bar{x}$, and $\bar{x}^{\mathrm{T}}P(t)B\bar{u} = (1/2)\bar{x}^{\mathrm{T}}(P(t)B + B^{\mathrm{T}}P(t))\cdot \bar{u}$. We can find a model-free formulation of $\bar{u}^{\star}$ given in (7.7) by using the stationarity condition $(\partial \mathcal{Q}(\bar{x}; \bar{u}; t)/\partial \bar{u}) = 0$ to obtain,

$$\bar{u}^{\star}(\bar{x}; t) = \arg\min_{\bar{u}} \mathcal{Q}(\bar{x}; \bar{u}; t) = -Q_{\mathrm{uu}}^{-1} Q_{\mathrm{ux}}(t)\bar{x}. \tag{7.10}$$

7.1.4 Learning Algorithm

We design a critic approximator to approximate the Q-function in (7.9) as,

$$Q^{\star}(\bar{x}; \bar{u}^{\star}; t) = \frac{1}{2} U^{\mathrm{T}} \bar{Q}(t) U := \frac{1}{2} \mathrm{vech}(\bar{Q}(t))^{\mathrm{T}} (U \otimes U),$$

where $\mathrm{vech}(\bar{Q}(t)) \in \mathbb{R}^{((n+m)(n+m+1)/2)}$. This structure exploits the symmetric properties of the $\bar{Q}$ matrix to reduce the computational complexity.

Since the ideal weight estimates are unknown, we employ an adaptive estimation technique (Ioannou and Fidan, 2006) by using current weights. Thus, the critic approximator can be written as,

$$\hat{Q}(\bar{x}; \bar{u}; t) = \hat{W}_{\mathrm{c}}^{\mathrm{T}} v(t)(U \otimes U), \tag{7.11}$$

where $\hat{W}_{\mathrm{c}} v(t) := (1/2)\mathrm{vech}(\hat{Q}(t))$.

By using a similar way of thinking for the actor, we will assign $\mu(t)^{\mathrm{T}} W_{\mathrm{a}} = -Q_{\mathrm{uu}}^{-1} Q_{\mathrm{ux}}(t)$ to write,

$$\bar{u}^{\star}(\bar{x}; t) = W_{\mathrm{a}}^{\mathrm{T}} \mu(t) \bar{x},$$

where $W_{\mathrm{a}} \in \mathbb{R}^{n\times m}$ are the actor weight estimates, $\mu(t) \in \mathbb{R}^{n\times n}$ is a radial basis function of appropriate dimensions that depend explicitly on time.

The actor by using current weight estimates is,

$$\hat{\bar{u}}(\bar{x};t) = \hat{W}_{\mathrm{a}}^{\mathrm{T}}\mu(t)\bar{x}. \tag{7.12}$$

Remark 7.1. The critic and the actor approximators described in (7.11) and (7.12), respectively, do not include any approximation errors. Therefore, we use the whole space and not just a compact set. With this structure, the approximations will converge to the optimal policies, and hence, the superscript $\star$ that denotes the ideal values of the adaptive weight estimation render similarly with the optimal solutions. □

We adopt the integral RL approach from Vrabie *et al.* (2013) that will express the Bellman equation as,

$$V^{\star}(\bar{x}(t);t) = V^{\star}(\bar{x}(t-\Delta t);t-\Delta t) - \frac{1}{2}\int_{t-\Delta t}^{T}(\bar{x}^{\mathrm{T}}M\bar{x} + \hat{\bar{u}}^{\mathrm{T}}R\hat{\bar{u}})\mathrm{d}\tau, \tag{7.13}$$

$$V^{\star}(\bar{x}(T);T) = \frac{1}{2}\bar{x}^{\mathrm{T}}(T)P(T)\bar{x}(T), \tag{7.14}$$

where $\Delta t \in \mathbb{R}^{+}$ is a small fixed value.

It holds that $\mathcal{Q}^{\star}(\bar{x};\hat{\bar{u}}^{\star};t) = V^{\star}(\bar{x};t)$, we can write (7.13) and (7.14) as,

$$\begin{aligned}\mathcal{Q}^{\star}(\bar{x}(t);\hat{\bar{u}}^{\star}(t);t) &= \mathcal{Q}^{\star}(\bar{x}(t-\Delta t);\hat{\bar{u}}^{\star}(t-\Delta t);t-\Delta t)\\ &\quad - \frac{1}{2}\int_{t-\Delta t}^{t}(\bar{x}^{\mathrm{T}}M\bar{x} + \hat{\bar{u}}^{\mathrm{T}}R\hat{\bar{u}})\,\mathrm{d}\tau,\\ \mathcal{Q}^{\star}(\bar{x}(T);T) &= \frac{1}{2}\bar{x}^{\mathrm{T}}(T)P(T)\bar{x}(T).\end{aligned}$$

Next, we select the errors $e_{c_1}, e_{c_2} \in \mathbb{R}$, which we would like to drive to zero by appropriately tuning the critic weights of (7.11). Define the first critic error e_{c_1} as,

$$\begin{aligned}e_{\mathrm{c}_1} &:= \mathcal{Q}(\bar{x}(t);\hat{u}(t);t) - \mathcal{Q}(\bar{x}(t-\Delta t);\hat{u}(t-\Delta t);t-\Delta t)\\ &\quad + \frac{1}{2}\int_{t-\Delta t}^{t}(\bar{x}^{\mathrm{T}}M\bar{x} + \hat{\bar{u}}^{\mathrm{T}}R\hat{\bar{u}})\mathrm{d}\tau\\ &= \tilde{W}_{\mathrm{c}}^{\mathrm{T}}v(t)((\hat{U}(t)\otimes\hat{U}(t)) - (\hat{U}(t-\Delta t)\otimes\hat{U}(t-\Delta t))\\ &\quad + \frac{1}{2}\int_{t-\Delta t}^{t}(\bar{x}^{\mathrm{T}}M\bar{x} + \hat{\bar{u}}^{\mathrm{T}}R\hat{\bar{u}})\mathrm{d}\tau,\end{aligned}$$

with $\hat{U} = [\bar{x}^{\mathrm{T}}\hat{u}^{\mathrm{T}}]^{\mathrm{T}}$ the augmented state that is comprised from the measurable full state vector and the estimated control action. Next, we define the second critic error as,

$$e_{\mathrm{c}_2} := \frac{1}{2}\bar{x}^{\mathrm{T}}(T)P(T)\bar{x}(T) - \hat{W}_{\mathrm{c}}^{\mathrm{T}}v(T)(\hat{U}(T)\otimes\hat{U}(T)).$$

The actor approximator error $e_{\mathrm{a}} \in \mathbb{R}^m$ is defined by,

$$e_{\mathrm{a}} := \hat{W}_{\mathrm{a}}^{\mathrm{T}}\mu(t)\bar{x} + \hat{Q}_{\mathrm{uu}}^{-1}\hat{Q}_{\mathrm{ux}}(t)\bar{x},$$

where $\hat{Q}_{\mathrm{uu}}, \hat{Q}_{\mathrm{ux}}$ will be obtained from the critic weight matrix estimation $\hat{W}_{\mathrm{C}}$. By employing adaptive control techniques (Ioannou and Fidan, 2006) we define the squared norm of errors as,

$$K_1(\hat{W}_{\mathrm{c}}, \hat{W}_{\mathrm{c}}(T)) = \frac{1}{2}\|e_{\mathrm{c}1}\|^2 + \frac{1}{2}\|e_{\mathrm{c}2}\|^2, \tag{7.15}$$

$$K_2(\hat{W}_{\mathrm{a}}) = \frac{1}{2}\|e_{\mathrm{a}}\|^2. \tag{7.16}$$

7.1.5 Learning Framework

The weights of the critic estimation matrix are obtained by applying a normalized gradient descent algorithm in (7.15),

$$\begin{aligned}\dot{\hat{W}}_{\mathrm{c}} &= -\alpha_{\mathrm{c}}\frac{\partial K_1}{\partial \hat{W}_{\mathrm{c}}} \\ &= -\alpha_{\mathrm{c}}\left(\frac{1}{\left(1+\sigma^{\mathrm{T}}\sigma\right)^2}\sigma e_{\mathrm{c}_1} + \frac{1}{\left(1+\sigma_{\mathrm{f}}^{\mathrm{T}}\sigma_{\mathrm{f}}\right)^2}\sigma_{\mathrm{f}}e_{\mathrm{c}_2}\right),\end{aligned} \tag{7.17}$$

where $\sigma(t) = v(t)(\hat{U}(t)\otimes\hat{U}(t) - \hat{U}(t-\Delta t)\otimes\hat{U}(t-\Delta t))$, $\sigma_{\mathrm{f}}(T) = v(T)(U(T)\otimes U(T))$, and $\alpha_{\mathrm{c}} \in \mathbb{R}^+$ is a constant gain that specifies the convergence rate. The critic tuning (7.17) guarantees that as $e_{\mathrm{c}_1} \to 0$ and $e_{\mathrm{c}_2} \to 0$ then $\hat{W}_{\mathrm{c}} \to W_{\mathrm{c}}$ and $\hat{W}_{\mathrm{c}}(T) \to W_{\mathrm{c}}(T)$.

Similarly, the weights of the actor estimation matrix $\hat{W}_{\mathrm{a}}$ by applying a gradient descent algorithm in (7.16) yield,

$$\dot{\hat{W}}_{\mathrm{a}} = -\alpha_{\mathrm{a}}\frac{\partial K_2}{\partial \hat{W}_{\mathrm{a}}} = -\alpha_{\mathrm{a}}\bar{x}e_{\mathrm{a}}^{\mathrm{T}}, \tag{7.18}$$

where $\alpha_{\mathrm{a}} \in \mathbb{R}^+$ is a constant gain that specifies the convergence rate. The actor estimation algorithm (7.18) guarantees that as $e_{\mathrm{a}} \to 0$,

then $\hat{W}_{\mathrm{a}} \to W_{\mathrm{a}}$. Next, we provide the main stability Theorem for the proposed Q-learning framework.

Theorem 7.2 (Kontoudis and Vamvoudakis, 2019, Thm. 2)**.** Consider the linear time-invariant CT system (7.1), the critic, and the actor approximators given by (7.11) and (7.12), respectively. The weights of the critic and the actor estimators are tuned by (7.17) and (7.18), respectively. The origin is a globally uniformly asymptotically stable equilibrium point of the closed-loop system, given that the critic gain α_{c} is sufficiently larger than the actor gain α_{a} and the following inequality holds:

$$0 < \alpha_{\mathrm{a}} < \frac{2\underline{\lambda}(M + Q_{\mathrm{xu}} R^{-1} Q_{\mathrm{xu}}^{\mathrm{T}}) - \overline{\lambda}(Q_{\mathrm{xu}} Q_{\mathrm{xu}}^{\mathrm{T}})}{\delta \overline{\lambda}\left(\frac{\mu(t) R^{-1}}{\| 1 + \mu(t)^{\mathrm{T}} \mu(t) \|^2} \right)},$$

with δ a constant of unity order. ■

In this subsection, we discuss the structure of the proposed model-free, online motion planning algorithm with Q-learning and optimal sampling-based path planners. We also present the algorithmic framework of the proposed RRT-Q*.

7.1.6 RRT-Q* Algorithmic Framework

The structure of the proposed motion planning RRT-Q* is presented in Figure 7.1. The proposed motion planning RRT-Q* consists of an offline global RRT computation; an online actor/critic structure; an online terminal state evaluation; an online static obstacle augmentation; and an online local re-planning. First, we compute offline the global path $\pi\left(x_{0,i}, x_{\mathrm{r},i}\right)$ by using the RRT* algorithm. Then, we continue with the online model-free learning of the optimal policy. More specifically, the policy evaluation is assessed by the critic and the policy improvement is performed by the actor. The actor is an inner-loop feedback controller that drives the system with $\hat{u}$ according to (7.12), where W_{a} are the actor parameters that can be found online by (7.18). The critic's objective is to estimate the Q-function, which is the value function that follows from the Bellman equations (7.13). The critic approximates $\hat{\mathcal{Q}}$ by

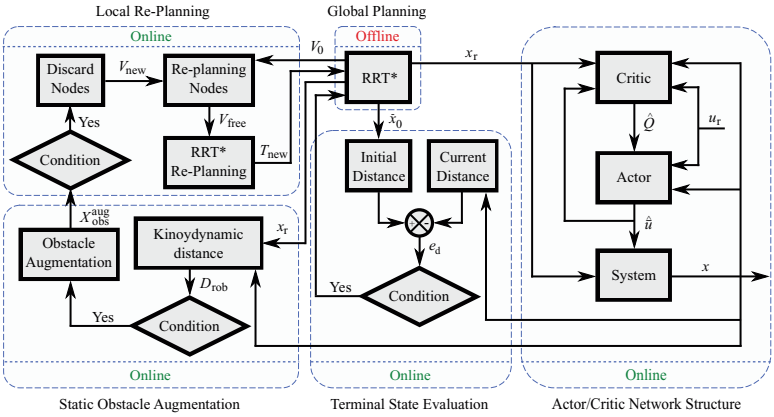

Figure 7.1: The motion planning RRT-Q* structure. The RRT-Q* incorporates five stages, (1) the offline global RRT* computation, (2) the online actor/critic network structure, (3) the online terminal state evaluation, (4) the static obstacle augmentation, and (5) the online local RRT* re-planning.

using (7.11), where W_c are the critic parameters that can be computed online by (7.17). The control actor computes the action $\hat{u}$ according to (7.9), where W_a are the actor parameters that can be estimated online. The disturbance actor computes the action $\hat{d}$ according to (7.10), where $\hat{W}_\mathrm{d}$ are the actor parameters that can be estimated online. The critic's parameters include intrinsic dynamics, which can be obtained by computing the time derivative that yields,

$$\begin{aligned}\dot{p} = {} & \bar{x}^\mathrm{T}(t)M\bar{x}(t) - \bar{x}^\mathrm{T}(t-\Delta t)M\bar{x}(t-\Delta t) + \hat{\bar{u}}^\mathrm{T}(t)R\hat{\bar{u}}(t) \\ & - \hat{\bar{u}}^\mathrm{T}(t-\Delta t)R\hat{\bar{u}}(t-\Delta t). \end{aligned} \tag{7.19}$$

A distance metric will be used to evaluate the final state. The initial distance D_0 is computed by (7.5). Next, the relative distance D is obtained online at every iteration Δt by (7.6). In the case that the distance error (7.7) decreases below an admissible value of the initial distance $e_\mathrm{d} \leqslant \beta D_0, \beta \in \mathcal{B} := \{\beta \in \mathbb{R} | 0 \leqslant \beta \leqslant 1\}$, we continue to the next i-TPBVP, by assigning the current state value as the new initial state $x_{0,i+1} = x(t)$. It is to be noted that the i-TPBVP is specified by the i-set of the initial and the final states x_0, x_r, which final states x_0, x_r, which were initially provided by the global planning with RRT*.

The RRT* algorithm is proven to compute the optimal path, which most of the times passes very close to the obstacles, that is potentially unsafe. Inherently, in kinodynamic motion planning, we cannot track straight lines due to the kinodynamic constraints imposed by the physics of the system. Therefore, when the robot navigates close to the obstacle and deviates from the given RRT* path, then a collision may occur with the obstacle. To address this problem, we propose a static augmentation of the obstacle space and a local re-planning strategy. For the static obstacle augmentation, we compute the maximum deviation of the robot from the straight line at every TPBVP, which we term as kinodynamic distance $D_{\text{rob}}\left(\bar{x}_0, \bar{x}\right)$. The kinodynamic distance is computed by,

$$D_{\text{rob}}\left(\bar{x}_0, \bar{x}\right) = \frac{|\bar{x}_0 \times \bar{x}|}{D_0}. \tag{7.20}$$

Next, if the kinodynamic distance is greater than the previously measured deviations of motion $D_{\text{rob},i} > \max\left\{D_{\text{rob},1}, \ldots, D_{\text{rob},i-1}\right\}$, we compute the augmented obstacle space,

$$\chi_{\text{obs}}^{\text{aug}} = \mathcal{X}_{\text{obs}} \oplus \mathcal{X}_{\text{rob}},$$

where $\mathcal{X}_{\text{rob}} \in \mathbb{R}^2$ is the kinodynamic distance space that is constructed as a rectangle with sides $\delta = 2D_{\text{rob}}$. That is a conservative approach, as we limit the navigation considering the maximum kinodynamic distance. since we tackle the model-free problem, the model of the system is unknown, and hence, we cannot perform offline computations. Therefore, the agent may deviate from the optimal path, still the proposed method ensures collision-free navigation.

We continue on the local replanning stage that will provide a safe path in the open diminished free space $\mathcal{X}_{\text{free}}^{\text{dim}} := \left(\mathcal{X}_{\text{obs}}^{\text{aug}}\right)^{\text{C}} = \mathcal{X} \backslash \mathcal{X}_{\text{obs}}^{\text{aug}}$. We start by evaluating whether the global path $\pi\left(x_{0,i}, x_{\text{r},i}\right)$ collides with the augmented obstacle space $\mathcal{X}_{\text{obs}}^{\text{aug}}$. Then, if a collision occurs, we prune the graph $\mathcal{G}(V, E)$ by discarding the nodes in the augmented obstacle space $V_{\text{aug}} = V \in \mathcal{X}_{\text{obs}}^{\text{aug}}$ from the initial set of nodes, $V_{\text{new}} = V \backslash V_{\text{aug}}$. Since the proposed algorithm operates online, we cannot afford computationally to perform the RRT even in the diminished free state-space $\mathcal{X}_{\text{free}}^{\text{dim}}$. Therefore, a significantly reduced free state-space needs to be specified. The underlying principle to narrow down the local path

planning problem exploits the precomputed nodes V and the initial global path π toward defining a new local free state space $\mathcal{X}_{\text{free}}^{\text{loc}}$. First, we search for the two closest states of the initial global path π outside the area of collision with the augmented obstacle space $\mathcal{X}_{\text{obs}}^{\text{aug}}$. These two states will serve as the local start state $x_{\text{start}}^{\text{loc}}$ and the local goal state $x_{\text{goal}}^{\text{loc}}$, while the rest path will not be affected. If any states of the path π are located in the augmented obstacle space $\mathcal{X}_{\text{obs}}^{\text{aug}}$, we discard them from the updated set of nodes V_{new}. Next, we establish a circle with center point at $O_{\text{loc}} = (x_{\text{start}}^{\text{loc}} + x_{\text{goal}}^{\text{loc}})/2$ and radius $r_{\text{loc}} = \|x_{\text{start}}^{\text{loc}} - x_{\text{goal}}^{\text{loc}}\|$, which forms the local circular space $\mathcal{X}_{\text{circle}}^{\text{loc}} := \{x \in \mathcal{X} | \|x - O_{\text{loc}}\|^2 \leqslant r_{\text{loc}}^2\}$. Then, the local candidate path planning space is defined as the relative complement of the augmented obstacle space in the local circular space, $\mathcal{X}_{\text{cand}}^{\text{loc}} = \mathcal{X}_{\text{circle}}^{\text{loc}} \backslash \mathcal{X}_{\text{obs}}^{\text{aug}}$. To assess the local candidate space $\mathcal{X}_{\text{cand}}^{\text{loc}}$, we introduce the following definitions (Rudin, 1964).

Definition 7.1. If A is a subset of a metric space X, and if ∂A denotes the set of all its limit points, then $\bar{A}$ said to be closure of A if $\bar{A} = A \bigcup \partial A$. □

Definition 7.2. Two subsets A and B of a metric space X are said to be separated if both $A \cap \bar{B} = \varnothing$ and $\bar{A} \cap B = \varnothing$ hold. □

Definition 7.3. A set A is connected if it is not the union of two separated sets. □

We analyze whether the local candidate path planning space $\mathcal{X}_{\text{cand}}^{\text{loc}}$ has separated sub-spaces, as this notion can be handled easier than the connected space. According to the structure of the environment (i.e., obstacle space and free state space), the local candidate path planning space $\mathcal{X}_{\text{cand}}^{\text{loc}}$ may result to be separated, with one space that contain the local goal state $x_{\text{goal}}^{\text{loc}}$ and another space with the local start state $x_{\text{start}}^{\text{loc}}$.

Lemma 7.3. For a given set of states in the diminished free space $\mathcal{X}_{\text{free}}^{\text{dim}}$, the local start state $x_{\text{start}}^{\text{loc}}$, and the local goal state $x_{\text{goal}}^{\text{loc}}$ if there exists a sufficient, connected, and closed local free space $\mathcal{X}_{\text{free}}^{\text{loc}}$ that forms a ring, based on the fixed incremental distance ϵ of the RRT*, then we can obtain a collision-free path with the local replanning framework. ■

Since we obtained a relatively small local free space $\mathcal{X}_{\text{free}}^{\text{loc}}$ that is guaranteed to contain the local start state $x_{\text{start}}^{\text{loc}}$, the local goal state $x_{\text{goal}}^{\text{loc}}$, and sufficient space for the implementation of the path planning with incremental distance ϵ, we can move on the next step, which is the local replanning with RRT. We locate the free replanning nodes from the updated set of nodes V_{new} that lie in the local free space $\mathcal{X}_{\text{free}}^{\text{loc}}$, which we term as $V_{\text{free}} = V_{\text{new}} \in \mathcal{X}_{\text{free}}^{\text{loc}}$. The coordinates of the free replanning nodes V_{free} will further reduce the computational effort, as no random sampling is required in the local free space $\mathcal{X}_{\text{free}}^{\text{loc}}$. The output is a local path π^{loc} that connects the local start state $x_{\text{start}}^{\text{loc}}$ with the local goal state $x_{\text{goal}}^{\text{loc}}$, which along with the previously computed global path π produces the new tree $\mathcal{T}_{\text{new}}$.

The RRT-Q* is presented in Algorithm 17 and its subroutines in Algorithms 18 and 19. However, note that it is beyond the scope of this monograph to present in detail the operation of classic RRT* algorithm. Hence, we omit to present the standard routines of the RRT*. The interested reader is directed to Kontoudis and Vamvoudakis (2019) for the detailed presentation of the RRT-Q* algorithm.

Next, we continue with the online implementation. The function `NoCollision` monitors if there exists a collision in the entire augmented obstacle space $\mathcal{X}_{\text{obs}}^{\text{aug}}$ with the global path π through the whole procedure and returns a binary value. The function `InitialDistance` calculates the distance of the initial and the final state according to (7.3). Then, follows the online approximation of the optimal policy with full state feedback (lines 8–12). The function `Critic` estimates the critic parameters from (7.17). This includes internal dynamics as given in (7.19). The function `EstimateQ` estimates the parameters of the $\hat{\mathcal{Q}}$ from (7.11). The `Actor` calculates the actor parameters from (7.18), that lead the function `Control` to produce the control action $\hat{\hat{u}}$ from (7.12). Next, we perform the terminal state evaluation (lines 13–18). The function `KinodynamicDistance` returns the deviation of the agent from the straight line that connects the initial and final states, by employing (7.20). The distance error is calculated by the function `DistanceError`, which allow the terminal state evaluation to proceed to the next i-TPBVP. The primitive `Augment` inflates the obstacle space by comparing the maximum distance of the previously

obtained deviations $\max\{D_{\text{rob},1}, \ldots, D_{\text{rob},i-1}\}$, with the current kinodynamic distance $D_{\text{rob},i}$.

When a collision of the path π occurs with the augmented obstacle space, then the algorithm continues to the next phase of the online local re-planning. A critical aspect for the feasibility of the online implementation, is to perform the re-planning procedure sufficiently fast. To this end, we narrow down the local free space with `LocalNodes`, that provides the free nodes V_{free} according to the fixed incremental ϵ of the RRT* for feasible local re-planning. Then, the `RRT*` provides the local graph $\mathcal{G}_{\text{new}}$, with a given set of nodes V_{free} that reduce the computational effort even further, as no random sampling is required. Lastly, the primitive `Connect` employs the global graph $\mathcal{G}$ and the locally graph $\mathcal{G}_{\text{new}}$ to find a safe tree $\mathcal{T}_{\text{new}}$ with respect to the kinodynamic constraints.

In Algorithm 18, the function `Circle` establishes a local candidate space $\mathcal{X}_{\text{loc}}^{\text{cand}}$ based on a circle with radius r_{loc} and at center O_{loc}. The function `SemiCircle` attempts to evaluate the connectedness of the $\mathcal{X}_{\text{loc}}^{\text{cand}}$ with respect to the fixed incremental distance ϵ of the RRT*. The `Edge` returns the closest states to the area of collision that we set as the local start state $x_{\text{start}}^{\text{loc}}$ and the local goal state $x_{\text{goal}}^{\text{loc}}$.

7.1.7 Further Reading

More details about the online kinodynamic motion planning algorithm are given in Kontoudis and Vamvoudakis (2019).

7.2 Bounded Rationality in Adversarial Target Tracking

This subsection addresses the problem of tracking an actively evading target by employing a team of coordinating UAVs while also learning the level of intelligence for appropriate countermeasures. Initially, under infinite cognitive resources, we formulate a game between the evader and the pursuing team, with an evader being the maximizing player and the pursuing team being the minimizing one. We derive optimal pursuing and evading policies while taking into account the physical constraints imposed by Dubins vehicles. Subsequently, we relax the

Algorithm 17: RRT_Q$^*(T, \Delta t, M, R, P(T), \alpha_c, \alpha_a, \beta, \mathcal{X}_{obs}, \mathcal{G}(V, E))$

$V_{free} \leftarrow \varnothing$; $\mathcal{X}_{obs}^{aug} \leftarrow \mathcal{X}_{obs}$;
$\mathcal{G}, \pi(x_0, x_r) \leftarrow$ `RRT*`$(\mathcal{G}, N, V_{free})$;
$D_{rob}^{kin} \leftarrow \varnothing$;
while `NoCollision`(π) **do**
  for $i = 1$ to k **do**
    $D_0 \leftarrow$`InitialDistance`(x_0);
    for $t \in T$ **do**
      $\hat{W}_c \leftarrow$ `Critic`$(\bar{x}, \bar{u}, \alpha_c, M, R, \Delta t, P(T))$;
      $\hat{Q} \leftarrow$ `EstimateQ`$(\bar{x}, \bar{u}, \hat{W}_c)$;
      $\hat{W}_a \leftarrow$ `Actor`$(\bar{x}, \bar{u}, \hat{Q}, \alpha_a)$;
      $\hat{\bar{u}} \leftarrow$ `Control`$(\bar{x}, \bar{u}, \hat{W}_a)$;
      Return $\hat{\bar{u}}$;
      $D_{rob} \leftarrow$ `KinodynamicDistance`$(\bar{x}, D_0)$;
      $e_d \leftarrow$ `DistanceError`$(\bar{x}, D_0)$;
      if $e_d \leqslant \beta D_0$ **then**
        $x_{0,i+1} \leftarrow x(t)$;
        break;
      end if
    end for
    if $D_{rob} > D_{rob}^{kin}$ **then**
      $\mathcal{X}_{obs}^{aug} \leftarrow$ `Augment`$(\mathcal{X}_{obs})$; $D_{rob}^{kin} \leftarrow D_{rob}$;
    end if
  end for
end while
$V_{free} \leftarrow$ `LocalNodes`$(\pi, \mathcal{X}_{obs}^{aug}, \epsilon)$;
$\mathcal{G}_{new} \leftarrow$ `RRT*`(V_{free});
$\mathcal{T}_{new} \leftarrow$ `Connect`$(\mathcal{G}, \mathcal{G}_{new})$;
Return $\mathcal{T}_{new}$;

Algorithm 18: LocalNodes $(\pi, \mathcal{X}_{obs}^{aug}, \epsilon)$

$V_{new} \leftarrow V - \{v \in \mathcal{X}_{obs}^{aug}\}$;
$(x_{start}^{loc}, x_{goal}^{loc}) \leftarrow$ `Edge`$(\mathcal{X}_{obs}^{aug}, \pi)$;
$(O_{loc}, r_{loc}, \mathcal{X}_{loc}^{cand}) \leftarrow$ `Circle`$(x_{start}^{loc}, x_{goal}^{loc})$;
for $j = 1$ to l **do**
  $x_{test} \leftarrow$ `SemiCircle`$(O_{loc}, r_{loc}, \epsilon)$
  if $x_{test} \in \mathcal{X}_{obs}^{aug}$ **then**
    $V_{free} \leftarrow \varnothing$;
  else
    $V_{free} \leftarrow V_{new}$;
  end if
end for
Return V_{free};

Algorithm 19: Edge $(\pi, \mathcal{X}_{\text{obs}}^{\text{aug}})$

for $i = 1$ to k **do**
  if $x_0 \in \mathcal{X}_{\text{obs}}^{\text{aug}}$ **then**
    $x_{\text{start}}^{\text{loc}} \leftarrow x_0^{\text{free}}$; break;
  end if
  $x_0 \leftarrow x_0^{\text{free}}$;
end for
for $i = 1$ to k **do**
  if $x_{\text{r}} \in \mathcal{X}_{\text{obs}}^{\text{aug}}$ **then**
    $x_{\text{goal}}^{\text{loc}} \leftarrow x_{\text{r}}^{\text{free}}$;
  end if
  $x_{\text{r}} \leftarrow x_{\text{r}}^{\text{free}}$;
end for
Return $(x_{\text{start}}^{\text{loc}}, x_{\text{goal}}^{\text{loc}})$;

infinite rationality assumption, via the use of level-k thinking. Such rationality policies are computed by using a RL-based architecture, and they converge to the Nash policies as the thinking levels increase.

7.2.1 Problem Formulation

Consider N camera-equipped UAVs tasked with estimating the state of a target vehicle moving evasively in the ground plane. The UAVs fly at a fixed airspeed and constant altitude and are subject to a minimum turning rate. The target vehicle moves in the ground plane and is subject to a maximum turning rate and maximum speed that is less than the UAVs' ground speed, which is the same as its airspeed in the ideal case of no wind. Each UAV takes measurements of the target's position using a gimballed video camera, and we assume that the target can be detected at all times and kept in the center of the camera's field of view by onboard software. We shall first discuss the dynamical models for each of the two types (N UAVs and 1 target) of vehicles and then proceed to describe the relative kinematics of each UAV with respect to the ground moving target.

Vehicle Dynamics

In our approach we consider that the kinematics of each moving vehicle, namely both the UAV and the target can be described via Dubins. The Dubins vehicle is a planar vehicle that moves forward at a fixed speed and has a bounded turning rate.

Consider that each UAV $i \in \mathcal{N} := \{1, 2, \ldots, N\}$ flies at a constant speed s_i, at a fixed altitude, and has a bounded turning rate $u_i \in \mathcal{U}$ in the sense that $\mathcal{U} := \{u_i \in \mathbb{R}\colon |u_i| \leqslant \bar{u}_i\}$ with $\bar{u}_i \in \mathbb{R}_+$.

Denote the state of each vehicle by $\xi^i := \begin{bmatrix}\xi_1^i & \xi_2^i & \xi_3^i\end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^3,\ i \in \mathcal{N}$, which comprises the planar position of each UAV in $\mathrm{p}_i := \begin{bmatrix}\xi_1^i & \xi_2^i\end{bmatrix}^{\mathrm{T}}$ and its heading $\psi_i := \xi_3^i$ all of which are measured in a local East-North-Up coordinate frame. Hence the kinematics of each UAV are given $\forall i \in \mathcal{N}$ by,

$$\dot{\xi}^i = f_v^i(\xi^i, u_i) := \begin{bmatrix}s_i \cos\xi_3^i & s_i \sin\xi_3^i & u_i\end{bmatrix}^{\mathrm{T}},\ t \geqslant 0.$$

On the basis of the above, the target is also modeled as a Dubins vehicle with a bounded turning rate d in the sense that $\mathcal{D} := \{d \in \mathbb{R}\colon |d| \leqslant \bar{d}\}$ where $\bar{d}$ is the maximum turning rate. Denote the state of the target by $\eta := \begin{bmatrix}\eta_1 & \eta_2 & \eta_3\end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^3$, where $\mathrm{p}_T := \begin{bmatrix}\eta_1 & \eta_2\end{bmatrix}^{\mathrm{T}}$ is the planar position of the target in the same local East-North-Up coordinate frame as the UAVs and η_3 is its heading.

To proceed we shall make the following assumption.

Assumption 7.4. The following are needed to make the problem feasible.

- At $t = 0$, the tracking vehicle is observing the target and is not dealing with the problem of initially locating the target.
- Since the UAVs fly at a constant altitude, there is no need to consider the 3-D distance, but only the projection of each UAV's position on the flat-Earth plane where the target is moving.
- The airspeed and the heading rate of the target satisfy, $s_t < \min\{s_1, s_2, \ldots, s_N\}$, and $\bar{d} < \min\{\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_N\}$. □

Relative Kinematics

In a target tracking problem it is necessary to determine the relative motion of the target with respect to the UAV. Thus, we will work in the polar coordinates, i.e., (r_i, θ_i), where r_i is the relative distance of each UAV to the target $r_i = \|\mathrm{p}_i - \mathrm{p}_T\|_2 := \sqrt{(\xi_1^i - \eta_1)^2 + (\xi_2^i - \eta_2)^2}$ and θ_i the azimuth angle defined as, $\theta_i = \arctan \frac{\xi_2^i - \eta_2}{\xi_1^i - \eta_1}$ $\forall i \in \mathcal{N}$. Also, we define the "relative heading" angle (Kokolakis and Koussoulas, 2018), as $\phi_i = \arctan \frac{r_i \dot{\theta}_i}{\dot{r}_i}$ and by taking $\phi_i = \psi_i - \theta_i$ into account, we can derive the tracking dynamics for each UAV as follows,

$$\begin{aligned}
&\dot{r}_i = s_i \cos\phi_i - \dot{\eta}_1 \cos\theta_i - \dot{\eta}_2 \sin\theta_i, \quad \forall i \in \mathcal{N},\\
&\dot{\xi}_3^i = u_i, \quad \forall i \in \mathcal{N},\\
&\dot{\eta}_3 = d.
\end{aligned}$$

We can now write the augmented state $\mathrm{r} := \begin{bmatrix} r_1 & \xi_3^1 & \cdots & r_N & \xi_3^N & \eta_3 \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{2N+1}$ to yield the following dynamics,

$$\begin{aligned}
\dot{\mathrm{r}} &= \begin{bmatrix} \dot{r}_1 \\ 0 \\ \dot{r}_2 \\ 0 \\ \vdots \\ \dot{r}_N \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ 1 & 0 & \cdots & \cdots & \\ 0 & \cdots & \cdots & \cdots & \vdots \\ & 1 & 0 & \cdots & \\ \vdots & & \ddots & & \vdots \\ & \cdots & \cdots & & 0 \\ 0 & & \cdots & 0 & 1 \\ 0 & & \cdots & & 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ \vdots \\ \vdots \\ \vdots \\ 1 \end{bmatrix} d \\
&\equiv F(\mathrm{r}) + Gu + Kd, \quad \mathrm{r}(0) = \mathrm{r}_0, \quad t \geqslant 0,
\end{aligned} \tag{7.21}$$

where, $u := \begin{bmatrix} u_1 & u_2 & \cdots & u_N \end{bmatrix}^{\mathrm{T}}$ is the vector of the turning rates of the UAVs.

Differential Game Formulation

The target tracking problem can be regarded as a two-player ZS game in which the team of UAVs tries to minimize the distance from the target r_i and the target tries to maximize it. Note that the UAVs coordinate

their movements in order to ensure that at least one UAV is close to the target. Additionally, the UAVs should keep their individual distances to the target sufficiently small to maintain the adequate resolution of the target in the camera's image plane for effective visual detection. The above motivates us to choose the following cost functional,

$$J = \int_0^\infty (R_u(u) - R_d(d) + R_r(\mathrm{r}))\mathrm{d}t,$$

where $R_r(\mathrm{r}) := \beta_1 \frac{1}{\sum_{i=1}^N \frac{1}{r_i^2}} + \beta_2 \sum_{i=1}^N r_i^2$, with $\beta_1, \beta_2 \in \mathbb{R}_+$ weighting constants. Specifically, the term being weighted by β_1 enforces distance coordination so that one UAV is always close to the target to improve measurement quality and the term being weighted by β_2 penalizes the individual UAV distances to the target to ensure that the size of the target in each UAV's image plane is sufficiently large for reliable detection by image processing software.

To enforce *bounded UAV inputs* and *bounded target input* we shall use a non-quadratic penalty function of the form,

$$R_u(u) = 2\int_0^u (\theta_1^{-1}(v))^{\mathrm{T}} R\mathrm{d}v, \quad \forall u, \tag{7.22}$$

and,

$$R_d(d) = 2\int_0^d (\theta_2^{-1}(v))^{\mathrm{T}} \Gamma\mathrm{d}v, \quad \forall d, \tag{7.23}$$

where $R \succ 0$, $\Gamma \in \mathbb{R}_+$, and $\theta_i(\cdot), i \in \{1,2\}$ are continuous, one-to-one real-analytic integrable functions of class C^μ, $\mu \geqslant 1$, used to map $\mathbb{R}$ onto the intervals $[-\bar{u}, \bar{u}]$ and $[-\bar{d}, \bar{d}]$, respectively, satisfying $\theta_i(0) = 0,\ i \in \{1,2\}$. Also note that $R_u(u)$ and $R_d(d)$ are positive definite because $\theta_i^{-1}(\cdot),\ i \in \{1,2\}$ are monotonic odd.

First, by assuming infinite rationality (the players in the game are familiar with the decision making mechanism) we are interested in finding the following optimal value function, $\forall \mathrm{r},\ t \geqslant 0$,

$$V^\star(\mathrm{r}(t)) = \min_{u\in U} \max_{d\in\mathcal{D}} \int_t^\infty (R_u(u) - R_d(d) + R_r(\mathrm{r}))\mathrm{d}\tau,$$

subject to (7.21).

7.2.2 Zero-Sum Game

According to the analysis in Subsection 6.2.1, the value function will satisfy the following HJ equation,

$$H\left(\mathrm{r}, \frac{\partial V^\star}{\partial \mathrm{r}}, u^\star, d^\star\right) = 0, \quad \forall r, \tag{7.24}$$

with Nash policies are given by,

$$u^\star(\mathrm{r}) = \arg\min_u H\left(\mathrm{r}, \frac{\partial V^\star}{\partial \mathrm{r}}, u, d^\star\right) = -\theta_1\left(\frac{1}{2}R^{-1}G^{\mathrm{T}}\frac{\partial V^\star}{\partial \mathrm{r}}\right), \quad \forall \mathrm{r}, \tag{7.25}$$

for the UAV, and,

$$d^\star(\mathrm{r}) = \arg\max_d H\left(\mathrm{r}, \frac{\partial V^\star}{\partial \mathrm{r}}, u^\star, d\right) = \theta_2\left(\frac{1}{2}\Gamma^{-1}K^{\mathrm{T}}\frac{\partial V^\star}{\partial \mathrm{r}}\right), \quad \forall \mathrm{r}, \tag{7.26}$$

for the target.

The closed-loop dynamics can be found by substituting (7.25) and (7.26) in (7.21), to write

$$\dot{\mathrm{r}} = F(\mathrm{r}) + Gu^\star + Kd^\star, \quad \mathrm{r}(0) = \mathrm{r}_0, \quad t \geqslant 0. \tag{7.27}$$

For completeness purposes, we characterize the stability of the equilibrium point of the closed-loop system.

Theorem 7.5 (Kokolakis *et al.*, 2020, Thm. 1)**.** Consider the closed-loop system given by (7.27). Assume that the equilibrium point is $\mathrm{r}_\mathrm{e} = 0$. Then, $s_i = s_t, \ \forall i \in \mathcal{N}$. ■

The next theorem provides a sufficient condition for the existence of a saddle-point based on (7.24).

Theorem 7.6 (Kokolakis *et al.*, 2020, Thm. 2)**.** Suppose that, there exists a continuously differentiable radially unbounded positive definite function $V^\star \in C^1$ such that, for the optimal policies given by (7.25) and (7.26), the following is satisfied,

$$R_u(u^\star(\mathrm{r})) - R_d(d^\star(\mathrm{r})) + R_r(\mathrm{r}) \geqslant 0, \quad \forall \mathrm{r},$$

with $V^\star(0) = 0$. Then, the closed-loop system given by (7.27), has a globally asymptotically stable equilibrium point. Moreover the policies (7.25)–(7.26) form a saddle point and the value of the game is, $J^\star(\cdot; u^\star, d^\star) = V^\star(\mathrm{r}(0))$. ■

7.2.3 Cognitive Hierarchy

In this subsection, we apply the proposed bounded rationality framework to the target tracking problem.

Level-0 (Anchor) Policy

We need to introduce an anchor policy for the level-0 player. First, we will define the level-0 UAV strategy as the policy based on the assumption that the target is not maneuvering and moves in a horizontal line which arises by solving an optimal control problem described by,

$$V_u^0(\mathrm{r}_0) = \min_{u \in U} \int_0^\infty (R_u(u) + R_r(\mathrm{r}))\mathrm{d}\tau. \tag{7.28}$$

The optimal control input for the optimization problem (7.28) given (7.21) with $d = 0$ is,

$$u^0(\mathrm{r}) = -\theta_1 \left(\frac{1}{2} R^{-1} G^{\mathrm{T}} \frac{\partial V_u^0}{\partial \mathrm{r}} \right), \quad \forall \mathrm{r},$$

where the value function $V_u^0(\cdot)$ satisfies the HJB equation, namely $H(\mathrm{r}, \frac{\partial V_u^0}{\partial \mathrm{r}}, u^0) = 0$.

Subsequently, the intuitive response of a level-1 adversary target is an optimal policy under the belief that the UAV assumes that the target is not able to perform evasive maneuvers. To this end, we define the optimization problem from the point of view of the target for the anchor input $u = u^0(\mathrm{r})$,

$$V_d^1(\mathrm{r}_0) = \max_{d \in \mathcal{D}} \int_0^\infty (R_u(u^0) - R_d(d) + R_r(\mathrm{r}))\mathrm{d}\tau, \quad \forall \mathrm{r},$$

subject to, $\dot{\mathrm{r}} = F(\mathrm{r}) + Gu^0 + Kd,\ \mathrm{r}(0) = \mathrm{r}_0,\ t \geqslant 0$.

The level-1 target's input is computed as, $d^1(\mathrm{r}) = \theta_2(\frac{1}{2}\Gamma^{-1}K^{\mathrm{T}}\frac{\partial V_d^1}{\partial \mathrm{r}})$, where the value function $V_d^1(\cdot)$ satisfies the HJB equation, i.e., $H(\mathrm{r}, \frac{\partial V_d^1}{\partial \mathrm{r}}, u^0, d^1) = 0$.

Level-k Policies

To derive the policies for the agents of higher levels of rationality, we will follow an iterative procedure, wherein the UAV and the adversary-target optimize their respective strategies under the belief that their opponent is using a lower level of thinking. The UAV performing an arbitrary number of k strategic thinking interactions solves the following minimization problem,

$$V_u^k(\mathrm{r}_0) = \min_{u\in U} \int_0^\infty (R_u(u) - R_d(d^{k-1}) + R_r(\mathrm{r}))\mathrm{d}\tau,$$

subject to the constraint, $\dot{\mathrm{r}} = F(\mathrm{r}) + Gu + Kd^{k-1},\ \mathrm{r}(0) = \mathrm{r}_0,\ t \geqslant 0.$

The corresponding Hamiltonian is,

$$H_u^k\left(\mathrm{r}, \frac{\partial V_u^k}{\partial \mathrm{r}}, u, d^{k-1}\right) = R_u(u) - R_d(d^{k-1}) + R_r(\mathrm{r}) + \left(\frac{\partial V_u^k}{\partial \mathrm{r}}\right)^{\mathrm{T}} (F(\mathrm{r}) + Gu + Kd^{k-1}), \quad \forall \mathrm{r}, u.$$

Substituting the target's input with the policy of the previous level $d^{k-1} = \theta_2(\frac{1}{2}\Gamma^{-1}K^{\mathrm{T}}\frac{\partial V_d^{k-1}}{\partial \mathrm{r}})$, yields,

$$u^k(\mathrm{r}) = -\theta_1\left(\frac{1}{2}R^{-1}G^{\mathrm{T}}\frac{\partial V_u^k}{\partial \mathrm{r}}\right), \quad \forall \mathrm{r}, \tag{7.29}$$

where the level-k UAV value function $V_u^k(\cdot)$ satisfies the HJB equation, namely

$$H_u^k\left(\mathrm{r}, \frac{\partial V_u^k}{\partial \mathrm{r}}, u^k, d^{k-1}\right) = 0, \quad \forall \mathrm{r}. \tag{7.30}$$

Similarly, the target of an arbitrary $k+1$ level of thinking, maximizes her response to the input of a UAV of level-k,

$$V_d^{k+1}(\mathrm{r}_0) = \max_{d\in\mathcal{D}} \int_0^\infty (R_u(u^k) - R_d(d) + R_r(\mathrm{r}))\mathrm{d}\tau,$$

subject to, $\dot{\mathrm{r}} = F(\mathrm{r}) + Gu^k + Kd,\ \mathrm{r}(0) = \mathrm{r}_0,\ t \geqslant 0.$

The corresponding Hamiltonian is,

$$H_d^{k+1}\left(\mathrm{r}, \frac{\partial V_d^{k+1}}{\partial \mathrm{r}}, u^k, d\right) = R_u(u^k) - R_d(d) + R_r(\mathrm{r}) + \frac{\partial V_d^{k+1}{}^{\mathrm{T}}}{\partial \mathrm{r}} (F(\mathrm{r}) + Gu^k + Kd), \quad \forall \mathrm{r}, d. \tag{7.31}$$

Substituting (7.29) in (7.31) yields the following response,

$$d^{k+1}(\mathrm{r}) = \theta_2 \left(\frac{1}{2}\Gamma^{-1} K^{\mathrm{T}} \frac{\partial V_d^{k+1}}{\partial \mathrm{r}}\right), \quad \forall \mathrm{r}, \tag{7.32}$$

where the level-$k+1$ target value function $V_d^{k+1}(\cdot)$ satisfies the HJB equation, namely

$$H_d^{k+1}\left(\mathrm{r}, \frac{\partial V_d^{k+1}}{\partial \mathrm{r}}, u^k, d^{k+1}\right) = 0, \quad \forall \mathrm{r}. \tag{7.33}$$

With this iterative procedure, the UAV is able to compute the strategies of the target with finite cognitive abilities, for a given number of levels.

Theorem 7.7 (Kokolakis *et al.*, 2020, Thm. 3)**.** Consider the pairs of strategies at a specific cognitive level-k, given by (7.29) and (7.30) for the UAV, and (7.32) and (7.33) for the level-$k+1$ adversarial target. The policies converge to a Nash equilibrium for higher levels if the following conditions hold as the levels increase,

$$R_u(u^{k-1}) - R_u(u^{k+1}) > 0, \tag{7.34}$$

$$R_d(d^{k+2}) - R_d(d^k) > 0. \tag{7.35}$$

■

Remark 7.2. It is worth noting that the inequalities (7.34), (7.35) have a meaningful interpretation. The input penalties (7.22), (7.23) are strictly increasing and decreasing functions, respectively, and as the level-k of rationality tends to infinity the players follow a policy such that the corresponding penalty functions become sufficiently small and large, respectively. □

Now, the following theorem provides a sufficient condition which establishes the global asymptotic stability of the equilibrium point $\mathrm{r}_\mathrm{e} = 0$ of the closed-loop system at each level of rationality k.

Theorem 7.8 (Kokolakis *et al.*, 2020, Thm. 4)**.** Consider the system (7.21) under the effect of agents with bounded rationality whose policies are defined by (7.29) for the UAV and (7.32) for the adversarial target. Assuming that the pursuer and evader have the same speed, the game can be terminated at any cognitive level-k as long as the following relationship hold:

$$R_u(u) - R_d(d) + R_r(\mathrm{r}) \geqslant 0, \quad \forall \mathrm{r}, u, d. \qquad \blacksquare$$

7.2.4 Coordination with Non-Equilibrium Game-Theoretic Learning

Due to the inherent difficulties of solving the HJI Equation (7.24), we shall utilize the RL-based machinery developed in Sections 2 and 3. In particular, we shall employ an actor/critic structure. Towards this, initially we will construct a critic approximator to learn the optimal value function that solves (7.24). Specifically, let $\Omega \subseteq \mathbb{R}^{2N+1}$, be a simply connected set, such that $0 \in \Omega$. It is known (Haykin, 2009) that we can rewrite the optimal value function as,

$$V^\star(\mathrm{r}) = W^\mathrm{T}\phi(\mathrm{r}) + \epsilon_\mathrm{c}(\mathrm{r}), \quad \forall \mathrm{r},$$

where $\phi := [\phi_1 \ \phi_2 \ \dots \ \phi_h]^\mathrm{T}$: $\mathbb{R}^{2N+1} \to \mathbb{R}^h$ are activation functions, $W \in \mathbb{R}^h$ are unknown ideal weights, and ϵ_c: $\mathbb{R}^{2N+1} \to \mathbb{R}$ is the approximation error. Specific choices of activation functions can guarantee that $\|\epsilon_\mathrm{c}(\mathrm{r})\| \leqslant \bar{\epsilon}_\mathrm{c}$, $\forall \mathrm{r} \in \Omega$, with $\bar{\epsilon}_\mathrm{c} \in \mathbb{R}^+$ being a positive constant (Hornik *et al.*, 1990; Ioannou and Fidan, 2006; Lewis *et al.*, 1998).

Since the ideal weights W are unknown, we define an approximation of the value function as,

$$\hat{V}(\mathrm{r}) = \hat{W}_\mathrm{c}^\mathrm{T}\phi(\mathrm{r}), \quad \forall \mathrm{r}, \tag{7.36}$$

where $\hat{W}_c \in \mathbb{R}^h$ are the estimated weights. Now, we can write the Hamiltonian utilizing the estimated value function (7.36) as,

$$\hat{H}\left(\mathrm{r}, \hat{W}_c^{\mathrm{T}} \frac{\partial \phi}{\partial \mathrm{r}}, u, d\right) \equiv R_u(u) - R_d(d) + R_r(\mathrm{r}) + \hat{W}_c^{\mathrm{T}} \frac{\partial \phi}{\partial \mathrm{r}}(F(\mathrm{r}) + Gu + Kd), \quad \forall u, d.$$

The approximate Bellman error due to the bounded approximation error and the use of estimated weights is defined as,

$$e_c = R_u(u) - R_d(d) + R_r(\mathrm{r}) + \hat{W}_c^{\mathrm{T}} \frac{\partial \phi}{\partial \mathrm{r}}(F(\mathrm{r}) + Gu + Kd).$$

An update law for $\hat{W}_c$ must be designed, such that the estimated values of the weights converge to the ideal ones. To this end, we define the squared residual error $K_c = \frac{1}{2}\|e_c\|^2$, which we want to minimize. Tuning the critic weights according to a modified Levenberg-Marquardt (Ioannou and Sun, 2012) gradient descent algorithm, yields,

$$\dot{\hat{W}}_c = -\alpha \frac{\omega(t) e_c(t)}{(\omega(t)^{\mathrm{T}} \omega(t) + 1)^2}, \tag{7.37}$$

where $\alpha \in \mathbb{R}^+$ is a constant gain that determines the speed of convergence and $\omega = \nabla \phi(F(\mathrm{r}(t)) + Gu(t) + Kd(t))$.

We use similar ideas to learn the best response policy. For compactness, we denote $a_j(\mathrm{r}),\ j \in \{u, d\}$, that will allow us to develop a common framework for the pursuers and the evaders. Similar to the value function, the feedback policy $a_j(\mathrm{r})$ can be rewritten as,

$$a_j^\star(\mathrm{r}) = {W_{a_j}^\star}^{\mathrm{T}} \phi_{a_j}(\mathrm{r}) + \epsilon_{a_j}, \quad \forall \mathrm{r},\ j \in \{u, d\},$$

where $W_{a_j}^\star \in \mathbb{R}^{h_{a_j} \times N_{a_j}}$ is an ideal weight matrix with $N_{a_u} := N$ and $N_{a_d} := 1$, $\phi_{a_j}(\mathrm{r})$ are the activation functions defined similar to the critic approximator, and ϵ_{a_j} is the actor approximation error. Similar assumptions with the critic approximator are needed to guarantee boundedness of the approximation error ϵ_{a_j}.

Since the ideal weighs $W_{a_j}^\star$ are not known, we introduce $\hat{W}_{a_j} \in \mathbb{R}^{h_{a_j} \times N}$ to approximate the optimal control in (7.25), and (7.26) as,

$$\hat{a}_j(\mathrm{r}) = \hat{W}_{a_j}^{\mathrm{T}} \phi_{a_j}(\mathrm{r}), \quad \forall \mathrm{r},\ j \in \{u, d\}. \tag{7.38}$$

Our goal is then to tune $\hat{W}_{a_j}$ such that the following error is minimized,

$$K_{a_j} = \frac{1}{2} e_{a_j}^{\mathrm{T}} e_{a_j}, \quad j \in \{u, d\},$$

where the reinforcement signal for the actor-network is,

$$e_{a_j} \coloneqq \hat{W}_{a_j}^{\mathrm{T}} \phi_{a_j} - \hat{a}_j^V, \quad j \in \{u, d\},$$

where $\hat{a}_j^V$ is a version of the optimal policy in which $V^\star$ is approximated by the critic's estimate (7.36),

$$\hat{a}_j^V = \begin{cases} -\theta_1 \left(\frac{1}{2} R^{-1} G^{\mathrm{T}} \nabla \phi^{\mathrm{T}} \hat{W}_c \right), & j = u, \\ \theta_2 \left(\frac{1}{2} \Gamma^{-1} K^{\mathrm{T}} \nabla \phi^{\mathrm{T}} \hat{W}_c \right), & j = d. \end{cases}$$

We note that the error considered is the difference between the estimate (7.38) and versions of (7.25) and (7.26). The tuning law for the UAV actor approximator is obtained by a modified Levenberg-Marquardt gradient descent rule,

$$\dot{\hat{W}}_{a_j} = -\alpha_{a_j} \phi_{a_j} e_{a_j}, \quad j \in \{u, d\}, \tag{7.39}$$

where $\alpha_{a_j} \in \mathbb{R}_+$ is a constant gain that determines the speed of convergence. The issue of guaranteeing convergence of the learning algorithms on nonlinear systems has been investigated in the literature. For the proposed approach, rigorous proofs and sufficient conditions of convergence have been presented in Vamvoudakis *et al.* (2016).

We will now propose an algorithmic framework that allows the UAV to estimate the thinking level of an evader that changes her behavior unpredictably by sequentially interacting over time windows of length $T_{\mathrm{int}} \in \mathbb{R}_+$. In essence, we will allow for arbitrary evading policies to be mapped to the level-k policy database.

Let $\mathcal{S} := \{1, 3, 5, \ldots, \mathcal{K}\}$ be the index set including the computed estimated adversarial levels of rationality and $\mathcal{K}$ is the largest number of the set. Assuming that the UAV is able to directly measure the target's heading rate, we define the error between the actual measured turning rate, denoted as $d(t)$ and the estimated one of a level-k adversarial

target,

$$r^k(t) := \int_t^{t+T_{\text{int}}} (d - \hat{a}_d)^2 \, \mathrm{d}\tau, \quad t \geqslant 0, \, k \in \mathcal{S}. \tag{7.40}$$

However, the ith sample shows the estimated target level of intelligence and the sampling period is T_{int}. The classification of the ith sample is found according to the minimum distance classifier, namely

$$x_i = \arg\min_k r^k, \quad \forall k \in \mathcal{S}, \; \forall i \in \{1, \ldots, \mathcal{L}\}, \tag{7.41}$$

where $\mathcal{L}$ is the total number of samples. Note that, the notions of "thinking steps" and "rationality levels" do not coincide as in Camerer *et al.* (2004). Let $k_i = \frac{x_i+1}{2}$ be the random variable counting the target thinking steps per game that follows the Poisson distribution (Camerer *et al.*, 2004) with the following probability mass function, $p(k_i; \lambda) = \frac{\lambda^{k_i} e^{-\lambda}}{k_i!}$, where $\lambda \in \mathbb{R}_+$ is both the mean and variance.

Our goal is to estimate the parameter λ from the observed data by using the sample mean of the observations which forms an unbiased maximum likelihood estimator,

$$\hat{\lambda}(n_{\text{S}}) = \frac{\sum_{i=1}^{n_{\text{S}}} k_i}{n_{\text{S}}}, \quad \forall n_{\text{S}} \in \{1, \ldots, \mathcal{L}\}. \tag{7.42}$$

However, in order to ensure the validity of our estimation we need to make the following assumption.

Assumption 7.9. The target is at most at the $\mathcal{K}$-th level of thinking and does not change policy over the time interval $((i-1)T_{\text{int}}, iT_{\text{int}}), \forall i \in \{1, \ldots, \mathcal{L}\}$. □

On the basis of the above, we write the following learning algorithm that enables the UAV first to learn the level-k policies and then to estimate the intelligence level distribution of a target.

7.2.5 Further Reading

More details and simulation results about coordinated tracking of an evading ground moving target are given in Kokolakis *et al.* (2020).

Algorithm 20: Intelligence Level Learning

procedure
  Given cost weights Γ, R, β_1, β_2 and highest allowable target level defined to be $\mathcal{K}$.
  for $k = 0, \ldots, \mathcal{K} - 1$ **do**
    Set $j := u$ to learn the level-k UAV policy.
    Start with initial state $\mathrm{r}(0)$ and, random initial weights $\hat{W}_u^k(0)$, $\hat{W}_{a_u}^k(0)$.
    Propagate the augmented system with states

$$\chi = \begin{bmatrix} \mathrm{r}^{\mathrm{T}} & (\hat{W}_u^k)^{\mathrm{T}} & (\hat{W}_{a_u}^k)^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}},$$

    according to (7.21), (7.37), and (7.39) until convergence.
    Compute (7.36) and (7.38).
    Set $j := d$ to learn the level-$k+1$ adversarial target policy.
    Start with initial state $\mathrm{r}(0)$ and, random initial weights $\hat{W}_d^{k+1}(0)$, $\hat{W}_{a_d}^{k+1}(0)$.
    Propagate the augmented system with states,

$$\chi = \begin{bmatrix} \mathrm{r}^{\mathrm{T}} & (\hat{W}_d^{k+1})^{\mathrm{T}} & (\hat{W}_{a_d}^{k+1})^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}},$$

    according to (7.21), (7.37), and (7.39) until convergence.
    Compute (7.36) and (7.38). Go to 3.
  end for
  Define the interaction time with each adversary as T_{int}, the number of total samples $\mathcal{L}$.
  for $i = 1, \ldots, \mathcal{L}$ **do**
    for $k = 1, \ldots, \mathcal{K}$ **do**
      Given $t \in [t_i, t_i + T_{\mathrm{int}}]$, measure the value of (7.40).
    end for
    Estimate the level of rationality of target according to (7.41).
    Update λ based on (7.42). Go to 15 to take the next sample.
  end for
end procedure

8

Concluding Remarks

The use of RL to solve control and game-theoretic problems is an area that has attracted a significant amount of research attention in recent years and will continue to grow as cognition and autonomy become a necessity for future engineering systems. The integration of RL with cognitive models can advance human-agent feedback loops while optimizing performance, guaranteeing stability, safety, and advancing data-decision models.

In this monograph, we have reviewed a family of online model-free and model-based adaptive learning techniques for single and multi-agent systems using measurements along the system trajectories with continuous and intermittent feedback for optimal regulation and tracking. Under the assumption of perfect rationality, we developed several *Synchronous* RL-based decision-making mechanisms for online gaming in the context of stabilization, tracking, zero-sum, non-zero-sum, Stackelberg, and graphical games. In the sequel, we relaxed the assumption that agents are rational, and we considered instead that they are bounded-rational, thereby giving rise to non-equilibrium games. Within the bounded rationality framework, by exploiting the principles of *Synchronous* RL, we constructed online adaptive cognitive algorithms, which

in fact are iterative procedures of best responses, allowing the secure operation of a CPS against malicious attacks. Subsequently, moving towards the cognitive autonomy in autonomous vehicles, by leveraging ideas from Q-learning-based control, we endowed the mobile robots with kinodynamic motion planning algorithms allowing them to navigate securely in an unknown, challenging, environment with obstacles while ensuring the avoidance of collision. Still, we introduced the bounded rational pursuit-evasion games enabling the collaborative tracking of an adversarial target by deploying a team of bounded rational cooperative UAVs. Finally, throughout the monograph, it was highlighted that the Synchronous RL-based algorithms are characterized by strong abilities of learning while allowing the complex systems to be fully autonomous and tolerant of failures against the uncertainty imposed by an unknown environment.

Note that, all the aforementioned algorithms use full-state feedback. However, complete knowledge of the agent's state may not be available for measurements in many applications, and it is desired to design output-feedback controllers based on available measured output data. Output feedback has not been fully investigated yet. Another interesting direction to pursue would be to extend the approaches to robot motion planning by combining game theory, formal control synthesis, and learning. Finally, existing RL techniques for large-scale dynamic systems require spending a long time to gather large amounts of data for learning an optimal solution. Moreover, the availability of large amounts of data requires novel decision and control schemes that focus selectively on the data that are relevant for the current situation and ignores unimportant details. The demands for fast response based on available information in a large-scale system impose new requirements for fast and efficient decision.

Future work, will also focus on designing new classes of learning-based controllers that are inspired by behavioral psychology and lead to satisfactory solutions for large-scale systems, which require making fast and skillful decisions in highly constrained, uncertain, and adversarial environments with an overload of information.

Acknowledgements

This work was supported in part by the National Science Foundation (NSF) under Grants SAS-1849264, CPS-1851588, and SATC-1801611, in part by the Office of Naval Research (ONR) Minerva under Grant N00014-18-1-2874, in part by the Army Research Office (ARO) under Grant W911NF-19-1-0270, in part by the Department of Energy (DOE) under Grant DE-EE0008453, and in part by the North Atlantic Treaty Organization (NATO) under Grant SPS G5176.

References

Abouheaf, M. I., F. L. Lewis, K. G. Vamvoudakis, S. Haesaert, and R. Babuska (2014). "Multi-agent discrete-time graphical games and reinforcement learning solutions". *Automatica*. 50(12): 3038–3053.

Abou-Kandil, H. and P. Bertrand (1985). "Analytical solution for an open-loop Stackelberg game". *IEEE Transactions on Automatic Control*. 30(12): 1222–1224.

Abu-Khalaf, M. and F. L. Lewis (2005). "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach". *Automatica*. 41(5): 779–791.

Abu Khalaf, M., F. L. Lewis, A. Al Tamimi, and D. Vrabie (2006). "Model-free adaptive dynamic programming for unknown systems". In: *Proceedings of the First International Conference on Computer Science & Education ICCSE'2006*. 105–114.

Abuzainab, N., W. Saad, and H. V. Poor (2016). "Cognitive hierarchy theory for heterogeneous uplink multiple access in the Internet of things". In: *2016 IEEE International Symposium on Information Theory (ISIT)*. IEEE. 1252–1256.

Alpcan, T. and T. Başar (2010). *Network Security: A Decision and Game-Theoretic Approach*. Cambridge University Press.

Al-Tamimi, A., F. L. Lewis, and M. Abu-Khalaf (2008). "Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof". *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*. 38(4): 943–949.

Antsaklis, P. J., K. M. Passino, and S. J. Wang (1991). "An introduction to autonomous control systems". *IEEE Control Systems Magazine*. 11(4): 5–13.

Arthur, W. B. (2018). *The Economy as an Evolving Complex System II*. CRC Press.

Asama, H., T. Fukuda, T. Arai, and I. Endo (2013). *Distributed Autonomous Robotic Systems 2*. Springer Science & Business Media.

Bagchi, A. and T. Başar (1981). "Stackelberg strategies in linear-quadratic stochastic differential games". *Journal of Optimization Theory and Applications*. 35(3): 443–464.

Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Başar, T. and G. Olsder (1999). *Dynamic Noncooperative Game Theory*. 2nd edn. Philadelphia, PA: SIAM.

Başar, T. and P. Bernhard (2008). *H-Infinity Optimal Control and Related Minimax Design Problems: A Dynamic Game Approach*. Springer.

Beck, C. L. and R. Srikant (2012). "Error bounds for constant step-size Q-learning". *Systems & Control Letters*. 61(12): 1203–1208.

Benosman, M. (2018). "Model-based vs data-driven adaptive control: An overview". *International Journal of Adaptive Control and Signal Processing*. 32(5): 753–776.

Bertsekas, D. P. (2019). *Reinforcement Learning and Optimal Control*. Athena Scientific.

Bertsekas, D. P. and J. N. Tsitsiklis (1995). "Neuro-dynamic programming: An overview". In: *Proceedings of the 34th IEEE Conference on Decision and Control*. Vol. 1. IEEE. 560–564.

Bertsekas, D. P. and J. N. Tsitsiklis (1996). *Neuro-Dynamic Programming*. Athena Scientific, MA.

Bhasin, S., R. Kamalapurkar, M. Johnson, K. G. Vamvoudakis, F. L. Lewis, and W. E. Dixon (2013). "A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems". *Automatica*. 49(1): 89–92.

Brams, S. and D. M. Kilgour (1988). *Game Theory and National Security*. New York: Basil Blackwell.

Bryson, A. E. (2018). *Applied Optimal Control: Optimization, Estimation and Control*. Routledge.

Busoniu, L., R. Babuska, and B. De Schutter (2008). "A comprehensive survey of multiagent reinforcement learning". *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*. 38(2): 156–172.

Busoniu, L., R. Babuska, B. De Schutter, and D. Ernst (2010). *Reinforcement Learning and Dynamic Programming Using Function Approximators*. Vol. 39. CRC Press.

Camerer, C. F. (2011). *Behavioral Game Theory: Experiments in Strategic Interaction*. Princeton University Press.

Camerer, C. F., T.-H. Ho, and J.-K. Chong (2004). "A cognitive hierarchy model of games". *The Quarterly Journal of Economics*. 119(3): 861–898.

Cao, M. (2020). "Merging game theory and control theory in the era of AI and autonomy". *National Science Review*. 7(7): 1122–1124.

Cao, X.-R. (2007). *Stochastic Learning and Optimization*. US: Springer.

Chen, C. and J. Cruz (1972). "Stackelburg solution for two-person games with biased information patterns". *IEEE Transactions on Automatic Control*. 17(6): 791–798.

Chen, T., X. Liu, and W. Lu (2007). "Pinning complex networks by a single controller". *IEEE Transactions on Circuits and Systems I: Regular Papers*. 54(6): 1317–1326.

Chong, J.-K., T.-H. Ho, and C. Camerer (2016). "A generalized cognitive hierarchy model of games". *Games and Economic Behavior*. 99: 257–274.

Chowdhary, G. and E. Johnson (2010). "Concurrent learning for convergence in adaptive control without persistency of excitation". In: *2010 49th IEEE Conference on Decision and Control (CDC)*. IEEE. 3674–3679.

Crawford, V. P. and N. Iriberri (2007). "Level-k auctions: Can a nonequilibrium model of strategic thinking explain the winner's curse and overbidding in private-value auctions?" *Econometrica*. 75(6): 1721–1770.

Daskalakis, C., P. W. Goldberg, and C. H. Papadimitriou (2009). "The complexity of computing a Nash equilibrium". *SIAM Journal on Computing*. 39(1): 195–259.

Devraj, A. M., A. Bušić, and S. Meyn (2019). "Zap Q-learning – A user's guide". In: *2019 Fifth Indian Control Conference (ICC)*. 10–15.

Dierks, T. and S. Jagannathan (2010). "Optimal control of affine nonlinear continuous-time systems using an online Hamilton–Jacobi–Isaacs formulation". In: *49th IEEE Conference on Decision and Control (CDC)*. IEEE. 3048–3053.

Dong, L., Y. Tang, H. He, and C. Sun (2016a). "An event-triggered approach for load frequency control with supplementary ADP". *IEEE Transactions on Power Systems*. 32(1): 581–589.

Dong, L., X. Zhong, C. Sun, and H. He (2016b). "Event-triggered adaptive dynamic programming for continuous-time systems with control constraints". *IEEE Transactions on Neural Networks and Learning Systems*. 28(8): 1941–1952.

Donkers, M. and W. Heemels (2012). "Output-based event-triggered control with guaranteed-gain and improved and decentralized event-triggering". *IEEE Transactions on Automatic Control*. 57(6): 1362–1376.

Doya, K. (2000). "Reinforcement learning in continuous time and space". *Neural Computation*. 12(1): 219–245.

Engwerda, J. (2005). *LQ Dynamic Optimization and Differential Games*. John Wiley & Sons.

Erev, I. and A. E. Roth (1998). "Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria". *American Economic Review*. 88(4): 848–881.

Euwe, M., M. Blaine, and J. Rumble (1982). *The Logical Approach to Chess*. Courier Corporation.

Fax, J. A. and R. M. Murray (2004). "Information flow and cooperative control of vehicle formations". *IEEE Transactions on Automatic Control*. 49(9): 1465–1476.

Fisk, C. (1984). "Game theory and transportation systems modelling". *Transportation Research Part B: Methodological*. 18(4–5): 301–313.

Fleming, W. H. and W. M. McEneaney (2000). "A max-plus-based algorithm for a Hamilton–Jacobi–Bellman equation of nonlinear filtering". *SIAM Journal on Control and Optimization*. 38(3): 683–710.

Freiling, G., G. Jank, and H. Abou-Kandil (1996). "On global existence of solutions to coupled matrix Riccati equations in closed-loop Nash games". *IEEE Transactions on Automatic Control*. 41(2): 264–269.

Freiling, G., G. Jank, and S. Lee (2001). "Existence and uniqueness of open-loop Stackelberg equilibria in linear-quadratic differential games". *Journal of Optimization Theory and Applications*. 110(3): 515–544.

Freiling, G., G. Jank, and D. Kremer (2003). "Solvability condition for a nonsymmetric Riccati equation appearing in stackelberg games". In: *2003 European Control Conference (ECC)*. IEEE. 963–967.

Fudenberg, D., F. Drew, D. K. Levine, and D. K. Levine (1998). *The Theory of Learning in Games*. Vol. 2. MIT Press.

Gajic, Z. and T. Li (1988). "Simulation results for two new algorithms for solving coupled algebraic Riccati equations". In: *Third Int. Symp. on Differential Games. Sophia, Antipolis, France*.

Gao, W., Y. Jiang, Z.-P. Jiang, and T. Chai (2016). "Output-feedback adaptive optimal control of interconnected systems based on robust adaptive dynamic programming". *Automatica*. 72: 37–45.

Garcia, E. and P. J. Antsaklis (2013). "Model-based event-triggered control for systems with quantization and time-varying network delays". *IEEE Transactions on Automatic Control*. 58(2): 422–434.

Goretkin, G., A. Perez, R. Platt, and G. Konidaris (2013). "Optimal sampling-based planning for linear-quadratic kinodynamic systems". In: *IEEE International Conference on Robotics and Automation*. 2429–2436.

Grondman, I., L. Busoniu, G. A. Lopes, and R. Babuska (2012). "A survey of actor-critic reinforcement learning: Standard and natural policy gradients". *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*. 42(6): 1291–1307.

Haddad, W. M., V. Chellaboina, and S. G. Nersesov (2006). *Impulsive and Hybrid Dynamical Systems: Stability, Dissipativity, and Control*. Princeton University Press.

Haykin, S. S. (2009). *Neural Networks and Learning Machines*. Vol. 3. Pearson Upper Saddle River.

He, X., A. Prasad, S. P. Sethi, and G. J. Gutierrez (2007). "A survey of Stackelberg differential game models in supply and marketing channels". *Journal of Systems Science and Systems Engineering*. 16(4): 385–413.

He, X., H. Dai, and P. Ning (2016). "Faster learning and adaptation in security games by exploiting information asymmetry". *IEEE Transactions on Signal Processing*. 64(13): 3429–3443.

Heemels, W., J. Sandee, and P. Van Den Bosch (2008). "Analysis of event-driven controllers for linear systems". *International Journal of Control*. 81(4): 571–590.

Hespanha, J. P. (2017). *Noncooperative Game Theory: An Introduction for Engineers and Computer Scientists*. Princeton University Press.

Hespanha, J. P., D. Liberzon, and A. R. Teel (2008). "Lyapunov conditions for input-to-state stability of impulsive systems". *Automatica*. 44(11): 2735–2744.

Ho, T.-H. and X. Su (2010). "A dynamic level-k model in games". *Tech. rep.* Haas School of Business, University of California, Berkeley.

Hornik, K., M. Stinchcombe, and H. White (1989). "Multilayer feedforward networks are universal approximators". *Neural Networks*. 2(5): 359–366.

Hornik, K., M. Stinchcombe, and H. White (1990). "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks". *Neural Networks*. 3(5): 551–560.

Hovakimyan, N. and C. Cao (2010). *L1 Adaptive Control Theory: Guaranteed Robustness with Fast Adaptation*. Philadelphia, PA: Advances in Design, Control, Society for Industrial, and Applied Mathematics (SIAM).

Hunt, K. J., D. Sbarbaro, R. Żbikowski, and P. J. Gawthrop (1992). "Neural networks for control systems—A survey". *Automatica*. 28(6): 1083–1112.

Ioannou, P. and B. Fidan (2006). "Adaptive control tutorial, society for industrial and applied mathematics". In: *Advances in Design and Control*. PA: SIAM.

Ioannou, P. A. and J. Sun (2012). *Robust Adaptive Control*. Courier Corporation.

Jadbabaie, A., J. Lin, and A. S. Morse (2003). "Coordination of groups of mobile autonomous agents using nearest neighbor rules". *IEEE Transactions on Automatic Control*. 48(6): 988–1001.

Jiang, Y. and Z.-P. Jiang (2012). "Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics". *Automatica*. 48(10): 2699–2704.

Jiang, Z.-P. and Y. Jiang (2013). "Robust adaptive dynamic programming for linear and nonlinear systems: An overview". *European Journal of Control*. 19(5): 417–425.

Johnson, M., T. Hiramatsu, N. Fitz-Coy, and W. E. Dixon (2010). "Asymptotic Stackelberg optimal control design for an uncertain Euler Lagrange system". In: *49th IEEE Conference on Decision and Control (CDC)*. IEEE. 6686–6691.

Johnson, M., R. Kamalapurkar, S. Bhasin, and W. E. Dixon (2015). "Approximate N-player nonzero-sum game solution for an uncertain continuous nonlinear system". *IEEE Transactions on Neural Networks and Learning Systems*. 26(8): 1645–1658.

Kaelbling, L. P., M. L. Littman, and A. W. Moore (1996). "Reinforcement learning: A survey". *Journal of Artificial Intelligence Research*. 4(1): 237–285.

Kamalapurkar, R., P. Walters, J. Rosenfeld, and W. Dixon (2018). *Reinforcement Learning for Optimal Feedback Control*. Springer.

Kanellopoulos, A. and K. G. Vamvoudakis (2019). "Non-equilibrium dynamic games and cyber–physical security: A cognitive hierarchy approach". *Systems & Control Letters*. 125: 59–66.

Kearns, M. (2007). "Graphical games". In: *Algorithmic Game Theory*. Ed. by N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani. Vol. 3. Cambridge, UK: Cambridge University Press. 159–180.

Kiumarsi, B., F. L. Lewis, H. Modares, A. Karimpour, and M.-B. Naghibi-Sistani (2014). "Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics". *Automatica*. 50(4): 1167–1175.

Kiumarsi, B., F. L. Lewis, and Z. P. Jiang (2017). "H_∞ control of linear discrete-time systems: Off-policy reinforcement learning". *Automatica*. 78: 144–152.

Kleinman, D. (1968). "On an iterative technique for Riccati equation computations". *IEEE Transactions on Automatic Control*. 13(1): 114–115.

Kokolakis, N.-M. T. and N. T. Koussoulas (2018). "Coordinated standoff tracking of a ground moving target and the phase separation problem". In: *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 473–482.

Kokolakis, N.-M. T., A. Kanellopoulos, and K. G. Vamvoudakis (2020). "Bounded rational unmanned aerial vehicle coordination for adversarial target tracking". In: *2020 American Control Conference (ACC)*. IEEE. 2508–2513.

Kontoudis, G. P. and K. G. Vamvoudakis (2019). "Kinodynamic motion planning with continuous-time Q-learning: An online, model-free, and safe navigation framework". *IEEE Transactions on Neural Networks and Learning Systems*. 30(12): 3803–3817.

Krstić, M. and I. Kanellakopoulos (1995). *Nonlinear and Adaptive Control Design. Adaptive and Learning Systems for Signal Processing, Communication and Control*. NY: Wiley.

Lamnabhi-Lagarrigue, F., A. Annaswamy, S. Engell, A. Isaksson, P. Khargonekar, R. M. Murray, H. Nijmeijer, T. Samad, D. Tilbury, and P. Van den Hof (2017). "Systems and control for the future of humanity, research agenda: Current and future roles, impact and grand challenges". *Annual Reviews in Control*. 43: 1–64.

LaValle, S. M. and S. A. Hutchinson (1998). "Optimal motion planning for multiple robots having independent goals". *IEEE Transactions on Robotics and Automation*. 14(6): 912–925.

Lee, J. Y., J. B. Park, and Y. H. Choi (2012). "Integral Q-learning and explorized policy iteration for adaptive optimal control of continuous-time linear systems". *Automatica*. 48(11): 2850–2859.

Lemmon, M. (2010). "Event-triggered feedback in control, estimation, and optimization". In: *Networked Control Systems*. Springer. 293–358.

Lewis, F. L. and K. G. Vamvoudakis (2010). "Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data". *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*. 41(1): 14–25.

Lewis, F. L. and D. Liu (2013). *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. Vol. 17. John Wiley & Sons.

Lewis, F., S. Jagannathan, and A. Yesildirak (1998). *Neural Network Control of Robot Manipulators and Non-Linear Systems*. CRC Press.

Lewis, F. L., D. Vrabie, and V. L. Syrmos (2012a). *Optimal Control*. John Wiley & Sons.

Lewis, F. L., D. Vrabie, and K. G. Vamvoudakis (2012b). "Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers". *Control Systems, IEEE*. 32(6): 76–105.

Lewis, F. L., H. Zhang, K. Hengster-Movric, and A. Das (2014). *Cooperative Control of Multi-Agent Systems-Optimal and Adaptive Design Approaches*. New York: Springer-Verlag.

Li, J., T. Chai, F. L. Lewis, Z. Ding, and Y. Jiang (2018). "Off-policy interleaved Q-learning: Optimal control for affine nonlinear discrete-time systems". *IEEE Transactions on Neural Networks and Learning Systems*. 30(5): 1308–1320.

Li, N., D. Oyler, M. Zhang, Y. Yildiz, A. Girard, and I. Kolmanovsky (2016). "Hierarchical reasoning game theory based approach for evaluation and testing of autonomous vehicle control systems". In: *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE. 727–733.

Li, Z.-H. and M. Krstic (1997). "Optimal design of adaptive tracking controllers for nonlinear systems". In: *American Control Conference, 1997. Proceedings of the 1997*. Vol. 2. IEEE. 1191–1197.

Liang, X. and Y. Xiao (2009). "Studying bio-inspired coalition formation of robots for detecting intrusions using game theory". *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*. 40(3): 683–693.

Littman, M. L. (2001). "Value-function reinforcement learning in Markov games". *Cognitive Systems Research*. 2(1): 55–66.

Liu, D., H. Li, and D. Wang (2013). "Neural-network-based zero-sum game for discrete-time nonlinear systems via iterative adaptive dynamic programming algorithm". *Neurocomputing*. 110: 92–100.

Liu, D., H. Li, and D. Wang (2014). "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.

Liu, D., Q. Wei, D. Wang, X. Yang, and H. Li (2017). *Adaptive Dynamic Programming with Application in Optimal Control*. Springer International Publishing.

Luo, B., H.-N. Wu, and T. Huang (2014). "Off-policy reinforcement learning for H_∞ control design". *IEEE Transactions on Cybernetics*. 45(1): 65–76.

Lyshevski, S. (1996). "Constrained optimization and control of nonlinear systems: New results in optimal control". In: *Proceedings of the 35th IEEE Conference on Decision and Control*. Vol. 1. IEEE. 541–546.

Lyshevski, S. E. (1998). "Optimal control of nonlinear continuous-time systems: Design of bounded controllers via generalized nonquadratic functionals". In: *American Control Conference, 1998. Proceedings of the 1998*. Vol. 1. IEEE. 205–209.

Ma, Z., D. S. Callaway, and I. A. Hiskens (2011). "Decentralized charging control of large populations of plug-in electric vehicles". *IEEE Transactions on Control Systems Technology*. 21(1): 67–78.

MacKenzie, A. B. and S. B. Wicker (2001). "Game theory in communications: Motivation, explanation, and application to power control". In: *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No. 01CH37270)*. Vol. 2. IEEE. 821–826.

Mannor, S. and J. S. Shamma (2007). "Multi-agent learning for engineers". *Artificial Intelligence*. 171(7): 417–422.

Marden, J. R. and J. S. Shamma (2015). "Game theory and distributed control". In: *Handbook of Game Theory with Economic Applications*. Vol. 4. Elsevier. 861–899.

Marden, J. R. and J. S. Shamma (2018a). "Game theory and control". *Annual Review of Control, Robotics, and Autonomous Systems*. 1: 105–134.

Marden, J. R. and J. S. Shamma (2018b). "Game-theoretic learning in distributed control". *Handbook of Dynamic Game Theory*: 511–546.

McEneaney, W. M. (2006). *Max-Plus Methods for Nonlinear Control and Estimation*. Springer Science & Business Media.

McKelvey, R. D. and T. R. Palfrey (1995). "Quantal response equilibria for normal form games". *Games and Economic Behavior*. 10(1): 6–38.

Mehta, P. and S. Meyn (2009). "Q-learning and Pontryagin's minimum principle". In: *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) Held Jointly with 2009 28th Chinese Control Conference*. IEEE. 3598–3605.

Melo, F. S., S. P. Meyn, and M. I. Ribeiro (2008). "An analysis of reinforcement learning with function approximation". In: *Proceedings of the 25th International Conference on Machine Learning*. 664–671.

Modares, H. and F. L. Lewis (2014). "Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning". *Automatica*. 50(7): 1780–1792.

Modares, H., F. L. Lewis, and M. B. Naghibi-Sistani (2013). "Adaptive optimal control of unknown constrained-input systems using policy iteration and neural Networks". *IEEE Transactions on Neural Networks and Learning Systems*. 24(10): 1513–1525.

Modares, H., F. L. Lewis, and M.-B. Naghibi-Sistani (2014). "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems". *Automatica*. 50(1): 193–202.

Modares, H., F. L. Lewis, and Z. P. Jiang (2015). "H_∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning". *IEEE Transactions on Neural Networks and Learning Systems*. 26(10): 2550–2562.

Molin, A. and S. Hirche (2013). "On the optimality of certainty equivalence for event-triggered control systems". *IEEE Transactions on Automatic Control*. 58(2): 470–474.

Morgenstern, O. and J. Von Neumann (1953). *Theory of Games and Economic Behavior*. Princeton University Press.

Mu, C., K. Wang, Q. Zhang, and D. Zhao (2020). "Hierarchical optimal control for input-affine nonlinear systems through the formulation of Stackelberg game". *Information Sciences*. 517: 1–17.

Myerson, R. B. (2013). *Game Theory*. Harvard University Press.

Nash, J. (1951). "Non-cooperative games". *Annals of Mathematics*. 54(2): 286–295.

Olfati-Saber, R. and R. M. Murray (2004). "Consensus problems in networks of agents with switching topology and time-delays". *IEEE Transactions on Automatic Control*. 49(9): 1520–1533.

Olfati-Saber, R., J. A. Fax, and R. M. Murray (2007). "Consensus and cooperation in networked multi-agent systems". *Proceedings of the IEEE*. 95(1): 215–233.

Paccagnan, D., B. Gentile, F. Parise, M. Kamgarpour, and J. Lygeros (2016a). "Distributed computation of generalized Nash equilibria in quadratic aggregative games with affine coupling constraints". In: *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE. 6123–6128.

Paccagnan, D., M. Kamgarpour, and J. Lygeros (2016b). "On aggregative and mean field games with applications to electricity markets". In: *2016 European Control Conference (ECC)*. IEEE. 196–201.

Paccagnan, D., B. Gentile, F. Parise, M. Kamgarpour, and J. Lygeros (2019). "Nash and Wardrop equilibria in aggregative games with coupling constraints". *IEEE Transactions on Automatic Control*. 64(4): 1373–1388.

Palanisamy, M., H. Modares, F. L. Lewis, and M. Aurangzeb (2014). "Continuous-time Q-learning for infinite-horizon discounted cost linear quadratic regulator problems". *IEEE Transactions on Cybernetics*. 45(2): 165–176.

Pavel, L. (2006). "A noncooperative game approach to OSNR optimization in optical networks". *IEEE Transactions on Automatic Control*. 51(5): 848–852.

Pavlov, I. P. and W. Gantt (1928). *Lectures on Conditioned Reflexes: Twenty-Five Years of Objective Study of the Higher Nervous Activity (Behaviour) of Animals*. Liverwright Publishing Corporation.

Perez, A., R. Platt, G. Konidaris, L. Kaelbling, and T. Lozano-Perez (2012). "LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics". In: *IEEE International Conference on Robotics and Automation*. 2537–2542.

Pita, J., M. Jain, J. Marecki, F. Ordóñez, C. Portway, M. Tambe, C. Western, P. Paruchuri, and S. Kraus (2008). "Deployed ARMOR protection: The application of a game theoretic model for security at the Los Angeles International Airport". In: *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems: Industrial Track*. 125–132.

Polycarpou, M., J. Farrell, and M. Sharma (2003). "On-line approximation control of uncertain nonlinear systems: Issues with control input saturation". In: *Proceedings of the 2003 American Control Conference, 2003.* Vol. 1. IEEE. 543–548.

Poveda, J. I., M. Benosman, and A. R. Teel (2019). "Hybrid online learning control in networked multiagent systems: A survey". *International Journal of Adaptive Control and Signal Processing*. 33(2): 228–261.

Qu, Z. (2009). *Cooperative Control of Dynamical Systems: Applications to Autonomous Vehicles*. Springer Science & Business Media.

Rabin, M. O. (1957). "Effective computability of winning strategies". *Contributions to the Theory of Games*. 3(39): 147–157.

Rantzer, A. (2008). *Using Game Theory for Distributed Control Engineering*. Department of Automatic Control, Lunds University.

Recht, B. (2019). "A tour of reinforcement learning: The view from continuous control". *Annual Review of Control, Robotics, and Autonomous Systems*. 2: 253–279.

Ren, W. and R. W. Beard (2005). "Consensus seeking in multiagent systems under dynamically changing interaction topologies". *IEEE Transactions on Automatic Control*. 50(5): 655–661.

Ren, W. and R. W. Beard (2008). *Distributed Consensus in Multi-Vehicle Cooperative Control*. Springer.

Ren, W., R. W. Beard, and E. M. Atkins (2005). "A survey of consensus problems in multi-agent coordination". In: *Proceedings of the 2005, American Control Conference, 2005.* IEEE. 1859–1864.

Roth, A. E. and I. Erev (1995). "Learning in extensive-form games: Experimental data and simple dynamic models in the intermediate term". *Games and Economic Behavior*. 8(1): 164–212.

Roy, S., C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu (2010). "A survey of game theory as applied to network security". In: *2010 43rd Hawaii International Conference on System Sciences*. IEEE. 1–10.

Rudin, W. (1964). *Principles of Mathematical Analysis*. Vol. 3. New York: McGraw-Hill.

Saad, W., Z. Han, H. V. Poor, and T. Başar (2012). "Game-theoretic methods for the smart grid: An overview of microgrid systems, demand-side management, and smart grid communications". *IEEE Signal Processing Magazine*. 29(5): 86–105.

Sahoo, P. P. and K. G. Vamvoudakis (2020). "On–off adversarially robust Q-learning". *IEEE Control Systems Letters*. 4(3): 749–754.

Saleheen, F. and C.-H. Won (2019). "Statistical Stackelberg game control: Open-loop minimal cost variance case". *Automatica*. 101: 338–344.

Schaft, A. J. van der (1992). "L_2-gain analysis of nonlinear systems and nonlinear state-feedback H_∞ control". *IEEE Transactions on Automatic Control*. 37(6): 770–784.

Schwalbe, U. and P. Walker (2001). "Zermelo and the early history of game theory". *Games and Economic Behavior*. 34(1): 123–137.

Semsar-Kazerooni, E. and K. Khorasani (2009). "Multi-agent team cooperation: A game theory approach". *Automatica*. 45(10): 2205–2213.

Simaan, M. and J. B. Cruz (1973a). "On the Stackelberg strategy in nonzero-sum games". *Journal of Optimization Theory and Applications*. 11(5): 533–555.

Simaan, M. and J. B. Cruz (1973b). "Additional aspects of the Stackelberg strategy in nonzero-sum games". *Journal of Optimization Theory and Applications*. 11(6): 613–626.

Simon, H. A. (1984). *Models of Bounded Rationality, Volume 1: Economic Analysis and Public Policy*. Vol. 1. MIT Press Books.

Solowjow, F. and S. Trimpe (2020). "Event-triggered learning". *Automatica*. 117: 109009.

Sontag, E. D. (1993). "Neural networks for control". In: *Essays on Control: Perspectives in the Theory and Its Applications*. Ed. by H. L. Trentelman and J. C. Willems. Boston, MA: Birkhäuser Boston. 339–380.

Sontag, E. and H. Sussmann (1997). "Mathematical theory of neural networks". *Tech. rep*. Rutgers, The State University. New Brunswick, NJ.

Strzalecki, T. (2014). "Depth of reasoning and higher order beliefs". *Journal of Economic Behavior & Organization*. 108: 108–122.

Sutton, R. S. and A. G. Barto (2018). *Reinforcement Learning: An Introduction*. 2nd edn. Vol. 1. Cambridge: MIT Press.

Sutton, R. S., A. G. Barto, and R. J. Williams (1992). "Reinforcement learning is direct adaptive optimal control". *Control Systems, IEEE*. 12(2): 19–22.

Tabuada, P. (2007). "Event-triggered real-time scheduling of stabilizing control tasks". *IEEE Transactions on Automatic Control*. 52(9): 1680–1685.

Tambe, M. (2011). *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*. Cambridge University Press.

Tao, G. (2003). "Adaptive control design and analysis". *Adaptive, Cognitive Dynamic Systems: Signal Processing, Learning, Communication and Control*. NY: Wiley.

Tijs, S. (2003). *Introduction to Game Theory*. Springer.

Tsai, J., S. Rathi, C. Kiekintveld, F. Ordonez, and M. Tambe (2009). "IRIS-a tool for strategic security allocation in transportation networks". *AAMAS (Industry Track)*: 37–44.

Tsitsiklis, J. (1984). *Problems in Decentralized Decision Making and Computation.* Ph.D. Thesis, MIT.

Tsitsiklis, J. N. (1994). "Asynchronous stochastic approximation and Q-learning". *Machine Learning*. 16(3): 185–202.

Ungureanu, V. (2018). *Pareto–Nash–Stackelberg Game and Control Theory*. Springer.

Vamvoudakis, K. G. (2014a). "Event-triggered optimal adaptive control algorithm for continuous-time nonlinear systems". *IEEE/CAA Journal of Automatica Sinica*. 1(3): 282–293.

Vamvoudakis, K. (2014b). "An online actor/critic algorithm for event-triggered optimal control of continuous-time nonlinear systems". In: *American Control Conference (ACC), 2014*. 1–6.

Vamvoudakis, K. G. (2015). "Non-zero sum Nash Q-learning for unknown deterministic continuous-time linear systems". *Automatica*. 61: 274–281.

Vamvoudakis, K. G. (2017). "Q-learning for continuous-time linear systems: A model-free infinite horizon optimal control approach". *Systems & Control Letters*. 100: 14–20.

Vamvoudakis, K. G. and F. L. Lewis (2010). "Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem". *Automatica*. 46(5): 878–888.

Vamvoudakis, K. G. and F. L. Lewis (2011). "Multi-player non-zero-sum games: Online adaptive learning solution of coupled Hamilton–Jacobi equations". *Automatica*. 47(8): 1556–1569.

Vamvoudakis, K. G. and F. Lewis (2012). "Online solution of nonlinear two-player zero-sum games using synchronous policy iteration". *International Journal of Robust and Nonlinear Control*. 22(13): 1460–1483.

Vamvoudakis, K. G. and H. Ferraz (2018). "Model-free event-triggered control algorithm for continuous-time linear systems with optimal performance". *Automatica*. 87: 412–420.

Vamvoudakis, K. G. and J. P. Hespanha (2018). "Cooperative Q-learning for rejection of persistent adversarial inputs in networked linear quadratic systems". *IEEE Transactions on Automatic Control*. 63(4): 1018–1031.

Vamvoudakis, K. G. and A. Kanellopoulos (2019). "Non-equilibrium learning and cyber-physical security". In: *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE. 1–6.

Vamvoudakis, K. G., D. Vrabie, and F. Lewis (2011). "Online learning algorithm for zero-sum games with integral reinforcement learning". *Journal of Artificial Intelligence and Soft Computing Research*. 1(4): 315–332.

Vamvoudakis, K. G., F. L. Lewis, and G. R. Hudas (2012). "Multi-agent differential graphical games: Online adaptive learning solution for synchronization with optimality". *Automatica*. 48(8): 1598–1611.

Vamvoudakis, K. G., D. Vrabie, and F. L. Lewis (2014). "Online adaptive algorithm for optimal control with integral reinforcement learning". *International Journal of Robust and Nonlinear Control*. 24(17): 2686–2710.

Vamvoudakis, K. G., P. J. Antsaklis, W. E. Dixon, J. P. Hespanha, F. L. Lewis, H. Modares, and B. Kiumarsi (2015). "Autonomy and machine intelligence in complex systems: A tutorial". In: *2015 American Control Conference (ACC)*. IEEE. 5062–5079.

Vamvoudakis, K. G., M. F. Miranda, and J. P. Hespanha (2016). "Asymptotically stable adaptive–optimal control algorithm with saturating actuators and relaxed persistence of excitation". *IEEE Transactions on Neural Networks and Learning Systems*. 27(11): 2386–2398.

Vamvoudakis, K. G., H. Modares, B. Kiumarsi, and F. L. Lewis (2017a). "Game theory-based control system algorithms with real-time reinforcement learning: How to solve multiplayer games online". *IEEE Control Systems Magazine*. 37(1): 33–52.

Vamvoudakis, K. G., A. Mojoodi, and H. Ferraz (2017b). "Event-triggered optimal tracking control of nonlinear systems". *International Journal of Robust and Nonlinear Control*. 27(4): 598–619.

Vamvoudakis, K. G., F. L. Lewis, and W. E. Dixon (2019). "Open-loop Stackelberg learning solution for hierarchical control problems". *International Journal of Adaptive Control and Signal Processing*. 33(2): 285–299.

Velupillai, K. V. (2011). "Non-linear dynamics, complexity and randomness: Algorithmic foundations". *Journal of Economic Surveys*. 25(3): 547–568.

Von Stackelberg, H. (2010). *Market Structure and Equilibrium*. Springer Science & Business Media.

Vrabie, D., O. Pastravanu, M. Abu-Khalaf, and F. L. Lewis (2009). "Adaptive optimal control for continuous-time linear systems based on policy iteration". *Automatica*. 45(2): 477–484.

Vrabie, D., K. G. Vamvoudakis, and F. L. Lewis (2013). *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*. Vol. 81. IET.

Wang, C. and D. J. Hill (2009). *Deterministic Learning Theory for Identification, Recognition, and Control*. Vol. 32. CRC Press.

Wang, D., C. Mu, H. He, and D. Liu (2016). "Event-driven adaptive robust control of nonlinear systems with uncertainties through NDP strategy". *IEEE Transactions on Systems, Man, and Cybernetics: Systems*. 47(7): 1358–1370.

Wang, D., H. He, and D. Liu (2017a). "Adaptive critic nonlinear robust control: A survey". *IEEE Transactions on Cybernetics*. 47(10): 3429–3451.

Wang, D., H. He, and D. Liu (2017b). "Improving the critic learning for event-based nonlinear H_∞ control design". *IEEE Transactions on Cybernetics*. 47(10): 3417–3428.

Wang, D., H. He, X. Zhong, and D. Liu (2017c). "Event-driven nonlinear discounted optimal regulation involving a power system application". *IEEE Transactions on Industrial Electronics*. 64(10): 8177–8186.

Wang, D., C. Mu, D. Liu, and H. Ma (2017d). "On mixed data and event driven design for adaptive-critic-based nonlinear H_∞ control". *IEEE Transactions on Neural Networks and Learning Systems*. 29(4): 993–1005.

Wang, X. and M. D. Lemmon (2011). "Event-triggering in distributed networked control systems". *IEEE Transactions on Automatic Control*. 56(3): 586–601.

Wang, X. F. and G. Chen (2002). "Pinning control of scale-free dynamical networks". *Physica A: Statistical Mechanics and Its Applications*. 310(3–4): 521–531.

Watkins, C. (1989). *Learning from Delayed Rewards*. Ph.D. Thesis, Cambridge University, Cambridge, UK.

Watkins, C. J. and P. Dayan (1992). "Q-learning". *Machine Learning*. 8(3–4): 279–292.

Webb, D. J. and J. van den Berg (2013). "Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics". In: *IEEE International Conference on Robotics and Automation*. 5054–5061.

Wei, Q., R. Song, and P. Yan (2015). "Data-driven zero-sum neuro-optimal control for a class of continuous-time unknown nonlinear systems with disturbance using ADP". *IEEE Transactions on Neural Networks and Learning Systems*. 27(2): 444–458.

Werbos, P. J. (1992). "Approximate dynamic programming for real-time control and neural modeling". *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*. 15: 493–525.

Werbos, P. J. (2007). "Using ADP to understand and replicate brain intelligence: The next level design?" In: *Neurodynamics of Cognition and Consciousness*. Springer. 109–123.

Wiering, M. and M. Van Otterlo (2012). "Reinforcement learning". *Adaptation, Learning, and Optimization*. 12: 3.

Wu, H.-N. and B. Luo (2012). "Neural network based online simultaneous policy update algorithm for solving the HJI equation in nonlinear H_∞ control". *IEEE Transactions on Neural Networks and Learning Systems*. 23(12): 1884–1895.

Xu, H., S. Jagannathan, and F. L. Lewis (2012). "Stochastic optimal control of unknown linear networked control system in the presence of random delays and packet losses". *Automatica*. 48(6): 1017–1030.

Xu, H., Q. Zhao, and S. Jagannathan (2014). "Optimal regulation of uncertain dynamic systems using adaptive dynamic programming". *Journal of Control and Decision*. 1(3): 226–256.

Yang, X. and H. He (2018). "Adaptive critic designs for event-triggered robust control of nonlinear systems with unknown dynamics". *IEEE Transactions on Cybernetics*. 49(6): 2255–2267.

Yang, Y., K. G. Vamvoudakis, and H. Modares (2020a). "Safe reinforcement learning for dynamical games". *International Journal of Robust and Nonlinear Control*. 30(9): 3706–3726.

Yang, Y., K. G. Vamvoudakis, H. Modares, Y. Yin, and D. C. Wunsch (2020b). "Safe intermittent reinforcement learning with static and dynamic event generators". *IEEE Transactions on Neural Networks and Learning Systems*.

Ye, M. and G. Hu (2017). "Distributed Nash equilibrium seeking by a consensus based approach". *IEEE Transactions on Automatic Control*. 62(9): 4811–4818.

Zames, G. and B. Francis (1983). "Feedback, minimax sensitivity, and optimal robustness". *IEEE Transactions on Automatic Control*. 28(5): 585–601.

Zhang, H., L. Cui, and Y. Luo (2012). "Near-optimal control for nonzero-sum differential games of continuous-time nonlinear systems using single-network ADP". *IEEE Transactions on Cybernetics*. 43(1): 206–216.

Zhang, H., T. Feng, G.-H. Yang, and H. Liang (2014a). "Distributed cooperative optimal control for multiagent systems on directed graphs: An inverse optimal approach". *IEEE Transactions on Cybernetics*. 45(7): 1315–1326.

Zhang, H., C. Qin, B. Jiang, and Y. Luo (2014b). "Online adaptive policy learning algorithm for H_∞ state feedback control of unknown affine nonlinear discrete-time systems". *IEEE Transactions on Cybernetics*. 44(12): 2706–2718.

Zhang, H., J. Zhang, G.-H. Yang, and Y. Luo (2014c). "Leader-based optimal coordination control for the consensus problem of multiagent differential games via fuzzy adaptive dynamic programming". *IEEE Transactions on Fuzzy Systems*. 23(1): 152–163.

Zhang, Q., D. Zhao, and D. Wang (2016a). "Event-based robust control for uncertain nonlinear systems using adaptive dynamic programming". *IEEE Transactions on Neural Networks and Learning Systems*. 29(1): 37–50.

Zhang, Q., D. Zhao, and Y. Zhu (2016b). "Event-triggered H_∞ control for continuous-time nonlinear system via concurrent learning". *IEEE Transactions on Systems, Man, and Cybernetics: Systems*. 47(7): 1071–1081.

Zhang, Y., S. Li, and X. Zhou (2020). *Deep Reinforcement Learning with Guaranteed Performance*. Springer.

Zhong, X. and H. He (2016). "An event-triggered ADP control approach for continuous-time system with unknown internal states". *IEEE Transactions on Cybernetics*. 47(3): 683–694.

Zhu, Y., D. Zhao, H. He, and J. Ji (2016). "Event-triggered optimal control for partially unknown constrained-input systems via adaptive dynamic programming". *IEEE Transactions on Industrial Electronics*. 64(5): 4101–4109.