Model-based Reinforcement Learning of Nonlinear Dynamical Systems

by

Milad Farsi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Applied Mathematics

Waterloo, Ontario, Canada, 2022

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Guillaume Crevecoeur Associate Professor, Dept. of Electromechanical, Systems and Metal Engineering, University of Ghent
Supervisor(s):	Jun Liu Associate Professor, Dept. of Applied Math., University of Waterloo
Internal Member:	Xinzhi Liu Professor, Dept. of Applied Math., University of Waterloo
Internal Member:	Hans De Sterck Professor, Dept. of Applied Math., University of Waterloo
Internal-External:	Amir Khajepour Professor, Dept. of Mechanical and Mechatronics Engineering, University of Waterloo

Abstract

Model-based Reinforcement Learning (MBRL) techniques accelerate the learning task by employing a transition model to make predictions. In this dissertation, we present novel techniques for online learning of unknown dynamics by iteratively computing a feedback controller based on the most recent update of the model. Assuming a structured continuous-time model of the system in terms of a set of bases, we formulate an infinite horizon optimal control problem addressing a given control objective. The structure of the system along with a value function parameterized in the quadratic form provides flexibility in analytically calculating an update rule for the parameters. Hence, a matrix differential equation of the parameters is obtained, where the solution is used to characterize the optimal feedback control in terms of the bases, at any time step. Moreover, the quadratic form of the value function suggests a compact way of updating the parameters that considerably decreases the computational complexity. In the convergence analysis, we demonstrate asymptotic stability and optimality of the obtained learning algorithm around the equilibrium by revealing its connections with the analogous Linear Quadratic Regulator (LQR). Moreover, the results are extended to the trajectory tracking problem. Assuming a structured unknown nonlinear system augmented with the dynamics of a commander system, we obtain a control rule minimizing a given quadratic tracking objective function. Furthermore, in an alternative technique for learning, a piecewise nonlinear affine framework is developed for controlling nonlinear systems with unknown dynamics. Therefore, we extend the results to obtain a general piecewise nonlinear framework where each piece is responsible for locally learning and controlling over some partition of the domain. Then, we consider the Piecewise Affine (PWA) system with a bounded uncertainty as a special case, for which we suggest an optimization-based verification technique. Accordingly, given a discretization of the learned PWA system, we iteratively search for a common piecewise Lyapunov function in a set of positive definite functions, where a non-monotonic convergence is allowed. Then, this Lyapunov candidate is verified for the uncertain system. To demonstrate the applicability of the approaches presented in this dissertation, simulation results on benchmark nonlinear systems are included, such as quadrotor, vehicle, etc. Moreover, as another detailed application, we investigate Maximum Power Point Tracking (MPPT) problem of solar Photovoltaic (PV) systems. Therefore, we develop an analytical nonlinear optimal control approach that assumes a known model. Then, we apply the obtained nonlinear optimal controller together with the piecewise MBRL technique presented previously.

Acknowledgments

First, I would like to express my sincere gratitude to my supervisor, Dr. Jun Liu, whose valuable guidance in identifying the research objectives and developing methodology in this dissertation was essential. He has been a constant source of inspiration, and his insightful advice helped me in refining my ideas and raising the quality of this research. I would like to thank all my committee members, Dr. Guillaume Crevecoeur, Dr. Amir Khajepour, Dr. Xinzhi Liu, and Dr. Hans De Sterck, who without their precious feedback, this dissertation could not have been accomplished. I am appreciative for everything I learned from my professors; my special thanks goes to Dr. Xinzhi Liu and Dr. Jeff Orchard who helped me to deepen my knowledge through courses. Finally, I want to convey my heartfelt thanks to my family, and friends for their unwavering support and encouragement during my study and research.

Dedication

This dissertation is dedicated to my family ♡.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ACCPM Analytic Center Cutting-Plane Method 8, 83, 84

ARE Algebraic Riccati Equation 6

DNN Deep Neural Network 8

DP Dynamic Programming 2, 4, 6, 7, 15, 20

FPRE Forward-Propagating Riccati Equation 6, 10, 28

GD Gradient Descent 18, 19, 154

HJB Hamilton Jacobi Bellman 2, 7, 11, 16, 25, 56, 60

LQR Linear Quadratic Regulator iv, xiii, 1, 6, 10, 27–29, 37, 59, 61, 88, 91

LQT Linear Quadratic Tracking 9

MBRL Model-based Reinforcement Learning iv, 4, 5, 9, 12, 21–23, 96, 99, 107, 154

MIQP Mixed-Integer Quadratic Program 8, 12, 81, 84, 85, 87

MPC Model-Predictive Control 6, 8, 96

MPP Maximum Power Point xiv, 22, 112–115, 119, 122, 123, 127, 129–133, 137–143

MPPT Maximum Power Point Tracking iv, 112, 113, 120, 122, 126, 129, 130, 133, 142, 154

NN Neural Network xiii, 88, 91

PE Persistence of Excitation 7, 19

PI Policy Iteration 2, 7, 20, 21

PV Photovoltaic iv, xiv, xv, 13, 14, 22, 112–120, 122, 125, 127–131, 133, 136, 137, 140–143, 148, 150–152, 154

PWA Piecewise Affine iv, xii, xiii, 8, 21, 81, 83, 86, 88, 91, 93, 141, 153, 154

PWM Pulse-Width Modulation xiii, 98, 99, 110, 119, 133, 136, 140

RL Reinforcement Learning 2–7, 9, 11, 14, 19, 20, 53, 96

RLS Recursive Least Squares xiii, 13, 40, 97, 99, 104, 107–109, 154

ROA Region of Attraction xiii, 22, 85, 88, 91, 93, 154

SDRE State-dependent Riccati Equations 6

SINDy Sparse Identification of Nonlinear Dynamics 10, 11, 38–40, 42, 62, 63, 154

SMC Sliding Mode Control xiv, 113, 114, 133, 138, 143–147

SOL Structured Online Learning 10, 11, 13, 21, 23, 38, 40, 43, 49, 57, 62, 63, 88, 97, 104, 105, 108, 154

SOS Sum of Squares xiii, 88, 91

TD Temporal Difference 7

VI Value Iteration 2, 3, 7, 20, 21

Chapter 1

Introduction

1.1 Motivation

Lack of a General Nonlinear Optimal Control Technique: Optimal control theory plays an important role in designing effective control systems. Optimal control problem is already solved successfully for linear systems that guarantees stability of the system in the case the system is controllable. Linear Quadratic Regulator (LQR) problem is represented by minimizing a quadratic cost in terms of the control input, where solving that allows us regulate the state and the control input of the system. In the applications of control systems, this provides an opportunity to specifically regulate the behavior of the system by adjusting coefficients used in the cost functional. However, when it turns to nonlinear dynamical systems, there is no systematic method for obtaining an optimal feedback control for the general nonlinear systems. Thus, many of the techniques available in the literature on linear systems do not apply in general.

Despite the complexity of nonlinear dynamical systems, they have attracted much attention from researchers in recent years. This is mostly because of their practical benefit in establishing a wide variety of applications in engineering, including power electronics, flight control, and etc. Considering hybrid systems as the most general category of dynamical system, optimal control involves finding a sequence that denotes the order of switching among subsystems, corresponding control input, and switching times such that a cost functional is minimized.

Importance of an Optimal Feedback Control: In general, there exist two well-know approaches to solving such optimization problems: the minimum principles (Pontryagin)

[128] and the Dynamic Programming (DP) method (Bellman) [137]. To solve an optimization problem that involves dynamics, minimum principles require us to solve a two point boundary value problem, where the solution is not in feedback mode.

There exist plenty of numerical techniques presented in the literature to solve the optimal control problem. Such approaches generally rely on our knowledge of the exact model of the system. In the case where such model exists, the optimal control input is obtained in the open-loop form that is given by a sequence of real values in time. Consequently, their implementation in the real-world problems often involves many complications that are well-known by control community. This is because of the model mismatch, noises, and disturbances that greatly affect the online solution, causing it to diverge from the preplanned offline solution. Therefore, obtaining a closed-loop solution for the optimal control problem is often preferred in the applications.

DP approach analytically results in a feedback control for linear systems with quadratic cost. Moreover, employing Hamilton Jacobi Bellman (HJB) equation with a value function, one might manage to derive an optimal feedback control rule for some real-world application. This motivates us to consider conditions leading to an optimal feedback control rule that can be applied to real-world problems.

Limits of Optimal Feedback Control Techniques: Consider an optimal control problem over an infinite horizon involving a non-quadratic performance measure. Using the idea of inverse optimal control, the cost functional can be then evaluated in closed form as long as the running cost depends somehow on an underlying Lyapunov function by which the asymptotic stability of the nonlinear closed-loop system is guaranteed. Then it can be obtained that the Lyapunov function is indeed the solution of the steady-state HJB equation. Although such formulation allows analytically obtaining an optimal feedback rule, choosing the proper performance measure may not be trivial. Moreover, from practical point of view, because of the nonlinearity in the performance measure, it might cause unpredictable behavior.

A well-studied method for solving an optimal control problem online is employing a value function assuming a given policy. Then, for any state the value function gives a measure of how good the state is by collecting the cost starting from that state while the policy is applied. If such a value function can be obtained, and the system model is known, the optimal policy is actually the one that takes the system in the direction by which the value decreases the most in the space of the states. Such Reinforcement Learning (RL) techniques which are known as value-based methods, including the Value Iteration (VI) and the Policy Iteration (PI) algorithms, are shown to be effective in a finite state and control spaces. However, the computations cannot efficiently scale with the size of the state and control spaces.

Complexity of Approximate DP Algorithms: One way of facilitating the computations regarding the value updates is employing an approximate scheme. This is done by parameterizing the value function, and accordingly adjusting the parameters in the training process. Then, the optimal policy given by the value function is also parameterized and approximated accordingly. The complexity of any value update depends directly on the number of parameters employed, where one may try limiting the number of the parameters by sacrificing the optimality. Therefore, we are motivated to obtain a more efficient update rule for the value parameters, rather than limiting the number of the parameters. We achieve this by reformulating the problem with a quadratically parameterized value function.

Moreover, the classical VI algorithm does not explicitly use the system model for evaluating the policy. This benefits the applications in the way that the full knowledge of the system dynamics is no longer required. However, now, the online training may take much longer time since the model only participates implicitly through the future state. Therefore, the learning can be potentially accelerated by introducing the system model. Furthermore, this creates an opportunity for running a separate identifier unit, where the model obtained can be simulated offline to complete the training or can be used for learning optimal policies for different objectives.

It can be shown that, in the discrete time, the VI algorithm for linear systems results in a Lyapunov recursion in the policy evaluation step based on the system matrices. However, in the continuous time, especially for the general nonlinear case, methods for obtaining an equivalent for that are not clear. Hence, in this dissertation, we investigate the chances of acquiring such an update rule.

Importance of Learning-based Tracking Approaches: One of the most common problems in the control of dynamical systems is to track a desired reference trajectory, which is found in a variety of real-world applications. However, designing an efficient tracking controller using conventional methods often necessitates a thorough understanding of the model, as well as computations and considerations for each application. RL approaches, on the other hand, propose a more flexible framework that requires less information about the system dynamics. While this may create additional problems, such as safety or computing limits, there are already effective outcomes from the use of such approaches in real-world situations. Similar to regulation problems, the applications of tracking control can benefit from an Model-based Reinforcement Learning (MBRL) that can handle the parameter updates more efficiently.

Opportunities for Obtaining a Realtime Control: In the approximate optimal control technique, employing a limited number of parameters can only yield a local approximation of the model and the value function. However, if an approximation within a larger domain is intended, a considerably higher number of parameters may be needed. Then, the identification and the controller's complexity might be rather too high to be performed online in real-world applications. This convinces us to circumvent this constraint by considering a set of local simple learners instead, in a piecewise approach.

As mentioned, there exist already interesting real-world applications of MBRL. Motivated by this, we aim on introducing automated ways of solving optimal control problems that can replace the conventional controllers. Hence, detailed applications of the proposed approaches are included that are done through numerical simulations.

Summary:

- Optimal control is highly favored, while there is no general analytical technique applicable to any nonlinear systems.
- Feedback control techniques are known to be more robust and computationally efficient compared to the numerical techniques, especially in the continuous space.
- The chance of obtaining a feedback control in a closed-loop form is low and the known techniques are limited to some classes of systems.
- Approximate DP provides a systematic way of obtaining an optimal feedback control, while the complexity grows greatly with the number of parameters.
- An efficient parameterization of the optimal value may provide an opportunity for more complex realtime applications in control regulation and tracking problems.

1.2 Literature Review

Reinforcement Learning: RL is a well-known class of machine learning methods that are concerned with learning a particular task through interactions with the environment.

The task is often defined by some reward/cost assigned. The intelligent agent has to take actions in different situations. Then, the reward/cost accumulated is used as a measure to improve the agent's actions in future, where the objective is to accumulate as much as rewards possible or minimize a cost over some time. Therefore, it is expected that the agent's actions approach the optimal behavior in a long term. RL has gained lots of successes in the simulation environment. However, the lack of explainability [46] and data efficiency [44] make them less favorable as an online learning technique that can be directly employed in the real-world problems, unless there exists a way to safely transfer the simulation-based learned experience to the real world. The main challenges in the implementations of the RL techniques are discussed in [46]. Numerous studies are done on this subject. See e.g. [155, 166, 77, 7] for a list of related works. RL has found a variety of interesting applications in robotics [89], multi-agent systems [177, 40, 73], power systems [178, 171], autonomous driving [86] and intelligent transportation [72], healthcare [173], etc.

Model-based Reinforcement Learning: MBRL techniques, as apposed to model-free methods in learning, are known to be more data-efficient. Direct methods usually require enormous data and hours of training even for simple applications [44], while model-based techniques can show optimal behavior in a limited number of trials. This property, in addition to the flexibilities in changing learning objectives and performing farther safety analysis make them more suitable for real-world implementations, such as robotics [127]. In model-based approaches, having a deterministic or probabilistic description of the transition system saves much of the effort spent by direct methods in treating any point in the state-control space individually. Hence, the role of model-based techniques becomes even more significant when it comes to problems with continuous control rather than discrete actions ([154, 9, 130]).

In [115], the authors provide a survey of some recent MBRL methods which are formulated based on Markov decision process. In general, there exist two approaches for approximating a system: parametric and non-parametric. Parametric models are usually preferred over non-parametric since the number of the parameters are independent of the number of samples. Therefore, they can be implemented more efficiently on complex systems, where many samples are needed. On the other hand, in non-parametric approaches, the prediction for a given sample is obtained by comparing it with a set of samples already stored, which represents the model. Therefore, the complexity increases with the size of the dataset. In this dissertation, because of this advantage of parametric models, we focus on the parametric techniques.

Optimal Control: Let us specifically consider implementations of RL on control systems. Regardless of the fact that RL techniques do not require the dynamical model to solve the problem, they are in fact intended to find a solution for the optimal control problem. This problem is extensively investigated by the control community. The LQR problem has been solved satisfactorily for controllable linear using an Algebraic Riccati Equation (ARE) [79] ensuring system stability. However, in the case of nonlinear systems, it seems obtaining such a solution is not trivial.

Model-Predictive Control (MPC) [28, 60, 135, 66, 60, 109, 117]has been frequently used as an optimal control technique, where it is inherently model-based. Furthermore, it deals with the control problem only across a restricted prediction horizon. For this reason, and for the fact that the problem is not considered in the closed-loop form, the stability analysis is hard to establish. For the same reasons, the computational complexity is considerably high compared to a feedback control rule implemented.

Forward-Propagating Riccati Equation (FPRE) [165, 131] is one of the techniques presented for solving the LQR problem. Normally, the differential Riccati equation is solved in backward with a final condition. In an analogues technique, it can be solved in forward-time with some initial condition instead. A comparison between these two schemes is given in [131]. Employing forward-integration method makes it suitable for solving the problem for time-varying [165, 32] system or in the RL setting [98] since the future dynamics are not needed. However, the backward case needs the knowledge of the dynamics at the final condition. FPRE has been shown to be an efficient technique in finding a sub-optimal solution for the linear systems, while for the nonlinear systems, the assumption is that the system is linearized along the system's trajectories.

State-dependent Riccati Equations (SDRE) [37, 48, 38] is another technique can be found in the literature for solving the optimal control problem. This technique relies on the fact that any nonlinear system can be written in the form of a linear system with state-dependent matrices. However, this conversion is not unique. Hence, a sub-optimal solution is expected. Similar to MPC, it does not yield a feedback control rule since the control at each state is computed by solving a different Riccati equation.

Dynamic Programming: Other model-based approaches can be found in the literature that are mainly categorized under RL in two group: value function and policy search methods. In value function-based methods, known also as approximate/adaptive DP techniques [163, 96, 11], a value function is used to construct the policy. However, policy search methods directly improve the policy to achieve optimality. Adaptive DP has found different applications [133, 57, 134, 119, 174, 71, 94, 102, 57] in automotive control, flight control, power control, etc. A review of recent techniques can be found in [80, 27, 140, 127, 83]. The Q-learning approach learns an action-dependent function using Temporal Difference (TD) to obtain the optimal policy. This is inherently a discrete approach. There are continuous extensions of this technique, such as [111, 62, 144, 164]. However, for an efficient implementation, the state and action space ought to be finite that is highly restrictive in the continuous space.

Adaptive controllers [8], as a well-known class of control techniques, may seem similar to RL in methodology, while there are substantial differences in the problem formulation and objectives. Adaptive techniques, as well as RL, learn to regulate unknown systems utilizing data collected in real-time. In fact, an RL technique can be seen as an adaptive technique that converges to the optimal control [96]. However, as apposed to RL and optimal controllers, adaptive controllers are not normally intended to be optimal, with respect to a user-specified cost function. Hence, such methods will not be further discussed in this dissertation.

Value methods in reinforcement learning normally require solving the well-known HJB. However, common techniques for solving such equations suffer from curse of dimensionality. Hence, in approximate DP techniques, a parametric or non-parametric model is used to approximate the solution. In [97], some related approaches are reviewed that fundamentally follow the actor-critic structure [14], such as VI and PI algorithms.

In such approaches, the Bellman error, which is obtained from the exploration of the state space, is used to improve the parameters estimated in a gradient-descent or least-squares loop that require the Persistence of Excitation (PE) condition. Since the Bellman error obtained is only valid along the trajectories of the system, sufficient exploration in the state-space is required to efficiently estimate the parameters. In [83], the authors have reviewed different strategies employed to increase the data-efficiency in exploration. In [160, 114], a probing signal is added to the control to enhance the exploring properties of the policy. In another approach [114], the recorded data of explorations is used as a replay of experience to increase the data efficiency. Accordingly, the model obtained from identification is used to acquire more experience by doing simulation in an offline routine that decreases the need for visiting any point in the state space.

As an alternative method, considering a nonlinear control affine system with a known input coupling function, the work [82] used a parametric model to approximate the value function. Then, they employed a least-squares minimization technique to adjust the parameters according to the Bellman error which can be calculated at any arbitrary point of the state space by having identified internal dynamics of the system and approximated state derivatives under a PE-like rank condition. In [81], the authors proposed an improved technique, where it approximates the value function only in a small neighborhood of the current state that travels within a compact set. It has been shown that the local approximation can be done more efficiently since considerably less number of bases can be used.

Piecewise Learning: There exist different techniques to efficiently fit a piecewise model to data, see e.g. [158, 24, 57, 4, 139, 43]. In [57], a technique for the identification of discrete-time hybrid systems by the piecewise affine model is presented. The algorithm combines clustering, linear identification, and pattern recognition approaches to identify the affine subsystems together with the partitions for which they apply. In fact, the problem of globally fitting a piecewise affine model is considered to be computationally expensive to solve. In [92], it is discussed that global optimality can be reached with a polynomial complexity in the number of data, while it is exponential with respect to the data dimension. In this regard, the work [24] presents an efficient two step technique: first, recursively clustering of the regressor vectors and estimation of the model parameters, and second, computation of a polyhedral partition. A review of some of the techniques can be found in [59, 61].

The flexibility of Piecewise Affine (PWA) systems makes them suitable for different approaches in control. Hence, the control problem of piecewise systems is extensively studied in the literature (see e.g. [108, 180, 142, 13, 150, 35, 143]). Moreover, various applications can be found for PWA systems that includes robotics [5, 107], automotive control [21, 152], and power electronics [63, 162]. In [180], the robust MPC strategy is extended to PWA systems with polytopic uncertainty, where multiple PWA quadratic Lyapunov functions are employed for different vertices of the uncertainty polytope in different partitions. In an other work [108], hybrid MPC is formulated as a mixed-integer program to solve the optimal control problem for PWA systems. However, these techniques are only available in an open-loop form, which decreases their applicability for realtime control.

On the other hand, Deep Neural Network (DNN) offers an efficient technique for control in closed loop. However, one drawback of DNN-based control is the difficulty in stability analysis. This becomes even more challenging when PWA are considered. The work [33] suggested a sample-efficient technique for synthesizing a Lyapunov function for the PWA system controlled through a DNN in closed loop. In this approach, Analytic Center Cutting-Plane Method (ACCPM) [64, 120, 22] is first used for searching for a Lyapunov function. Then, this Lyapunov candidate is verified on the closed-loop system using a Mixed-Integer Quadratic Program (MIQP). This approach relies on our knowledge of the exact model of the system, hence, cannot be directly implemented on an identified PWA with uncertainty.

Tracking Control: For the learning-based tracking problem, several techniques can be found in the literature, in addition to some extensions presented for the techniques reviewed [113, 112, 179, 172, 106]. The authors of [113] have developed an integral RL technique for linear systems based on policy iteration algorithm, starting with an admissible initial controller. It has been shown that the optimal tracking controller converges to Linear Quadratic Tracking (LQT) controller, with a partially-unknown system. In [112], an off-policy method is employed with three neural networks in an actor-critic-disturbance configuration to learn an H_∞-tracking controller for unknown nonlinear systems. The authors of [179] constructed an augmented system using the tracking error and the reference. Neural networks were employed, in an actor-critic structure, to approximate the value function and learn an optimal policy. In another neural network-based approach [172], a single network was used to approximate the value function, where a class of uncertain dynamics were assumed. In addition to the above approaches, there exist other similar ones in the literature. However, applications of RL in tracking control are not only limited to model-based techniques. For instance, [106] suggests a critic-only Q-learning approach for tracking problems, which does not require solving the HJB equation.

Applications: As mentioned, there exist different applications of MBRL, as well as the optimal control, on the real-world problems [133, 57, 134, 119, 174, 71, 94, 102, 57]. Accordingly, we will later provide the detailed literature review for each of the applications including the quadrotor and the solar photo-voltaic systems in Chapters 5 and 6 respectively.

1.3 Contributions

In this section, we discuss the contributions and merits of the proposed techniques while highlighting related results.

Structured Online Learning: In Chapter 2, by proposing a novel technique for parameterizing the system and the value function, we aim on solving a closed-loop optimal control problem online, rather than using a minimization technique to update the parameters based on the Bellman error. Although, Bellman theory is clearly behind all these techniques, the formulation here is contrasted with previous approaches: assuming a particular structure for the identified system allows us to analytically obtain a matrix differential equation in terms of parameters. This relaxes the need of a gradient-descent or least-squares technique that is used in similar approaches. Unlike [176, 17, 82], we assume a quadratic form for the value function that provides a compact way for parameterizing the value in terms of quadratic terms, following the structure considered for the dynamics. Accordingly, we will refer the presented framework as a Structured Online Learning (SOL) algorithm.

Moreover, we present the stability analysis of the approach and its connections with FPRE [165, 131]. Unlike FPRE technique, that only considers a linear system, the proposed technique allow employing various nonlinear bases that may increase the adaptability of the control.

In [25], an algorithm for sparse identification of Sparse Identification of Nonlinear Dynamics (SINDy) is presented to obtain the explicit dynamics of the system. In [78], a control approach is employed based on SINDy that includes two independent stages: the identification with a generated random signal and the model predictive control. This technique is shown to be a data-efficient identification scheme that, in addition, can handle the noise in data. A variant of sparse identification using the alternating direction method of multipliers is applied on a set of real-world experimental data in [65]. Although the identification methods can be used in SOL algorithm are not limited to any particular approach, as another contribution of this dissertation, we employ SINDy to achieve faster convergence of the parameters in an online scheme. This is contrasted with [78] in a way that we iteratively perform both identification and control in a single loop.

Summary:

- A new formulation of the optimal control problem is proposed based on a particular structure of the system in terms of nonlinear bases.
- A novel technique for parameterizing the value function in a quadratic form and obtaining a feedback control is proposed.
- The quadratically parameterized value function in accordance with the structure in the system results in a new technique for updating the value.
- It is shown that the proposed update rule can be run more efficiently compared to previous techniques.
- Compared to FPRE, the proposed method improves adaptability in learning an optimal controller by allowing nonlinear bases.
- The local stability and optimality analysis are provided by making connections with the LQR control.

- In a novel scheme, SINDy is employed for online RL of nolinear systems.

Learning-based Tracking Control: In Chapter 3, we employ SOL scheme in formulating the tracking problem and approximating the solution of the HJB equation. By sampling the input, state, and the reference trajectories, we exploit a system identification method to update the system model and solve a closed-loop approximate optimal control problem in an iterative fashion. In contrast with other learning approaches in the literature, such as [112, 176, 17, 82], which normally use standard gradient descent or least squares techniques, we extend the idea of SOL to the tracking problem. Hence, we analytically develop and solve a matrix differential equation whose forward solution updates the value and yields a tracking optimal control rule that includes the feedback and the feedforward control terms of nonlinear basis functions.

Moreover, assuming a quadratic form for parameterizing the value function and updating the parameters in the matrix form introduces considerably less complexity to value update, compared to [112, 179, 172]. Furthermore, compared to [113], the control can be obtained in terms of different nonlinear bases that can promote the adaptability of the approach.

Summary:

- A novel formulation of optimal tracking control is presented.
- The computational complexity is improved since the quadratic parameterization is employed for approximating the value function.
- The efficiency in computations allows employing more various bases that increases the adaptability.

Piecewise Learning: In Chapter 4, we develop a piecewise learning technique according to the SOL algorithm. In an alternative approach, instead of adding many bases to perform learning in a large domain of interest, we divide the domain into pieces where each can be handled independently with a limited number of bases. Employing a piecewise model will considerably improve the learning by keeping the online computations needed for updating the model and the control in a tractable size for every time step.

Despite the improvement in the computations, data efficiency of learning may be diminished if a large number of the pieces is chosen. Considering that the total number of the model parameters is relative to the number of pieces, a piecewise model may involve more parameters compared to learning in terms of bases. Therefore, they may need much longer time to converge. In fact, there exists a trade-off between the data efficiency and computational efficiency that can be controlled by the number of pieces and parameters employed.

Summary:

- A novel procedure for learning a piecewise model with bounded uncertainty is presented.
- In the reinforcement learning setting, we developed a feedback controller based on the forward integration of a differential equation for different modes of the system.
- The uncertain piecewise model together with the feedback controller is encoded in an MIQP.
- We develop an optimization-based technique for guaranteeing asymptotic stability of the uncertain piecewise system overall.

Learning-based Low-level Control of Quadrotor: In Chapter 5, similar to [91], we use the flight data obtained along random open-loop trajectories to establish an initial model, with no need for any expert demonstrations. However, in a different approach than [91], once an initial model together with the corresponding controller is obtained, we switch to learning in a closed-loop form to refine the model and the performance achieved. To verify the method, we acquire data from the nonlinear model of the quadrotor, treated as a black box.

For a practical framework, the MBRL approach has to be data-efficient while being fast enough to allow real-time implementation. Unlike [91], our approach does not demand a lot of computational efforts considering that we learn the system in terms of a limited number of bases and accordingly obtain a feedback control rule. Hence, it can be used as a lightweight alternative in implementations. Moreover, in [91], the objective is to reach the hovering position, whereas in addition to the attitude control, we also control the position. This means the quadrotor simultaneously learns to reach and stay at a given point in the 3D space. This will also minimize the instances where the quadrotor slides out of the training environment.

For learning purpose, we implemented the SOL approach proposed in Chapter 2 with a Recursive Least Squares (RLS) algorithm that is well known for its high efficiency in online applications. Successful applications of RLS can be found in [103, 167, 170, 147]. In this application, as an alternative to the neural network approach, we used a system structured in terms of a library of bases. Accordingly, by sampling the input and state, we employ RLS to update the system model. Then, by exploiting the structure assumed in the model and a quadratic parametrization of the value function in terms of the same set of bases, we obtain a matrix differential equation to update the controller that can be efficiently integrated online.

Summary:

- A more complex objective is considered that, in addition to stability also includes reaching a position.
- The learned model supports more modes of the real quadrotor since a more complex objective is defined.
- The obtained optimal feedback controller is computationally more efficient compared to the previous technique.

Maximum Power Point Tracking of Solar PV Systems: While there exist various approaches presented in the literature for improving the performance of the control in tracking the maximum power point, there is still no clear connection made between the configuration of the controller implemented and the performance obtained. Hence, performance analysis and defining the problem of maximizing the output power of Photovoltaic (PV) systems in optimal control framework still remain as a challenging problem. In this dissertation, we addressed this problem by proposing three different techniques.

In Chapter 6, we formulate and solve an optimal feedback control problem of solar PV systems that potentially brings many benefits in the applications. To obtain the optimal feedback control law, we consider a nonlinear affine model with a performance measure including a cross-weighting term. Hence, in contrast with the previous approaches, the performance analysis is additionally done by satisfying optimality conditions, which are adapted for a set of equilibrium points based on the incremental conductance approach. The obtained feedback controller, due to its suitable responses around the maximum power point, significantly decreases the undesired oscillations. Considering that the performance of the solar PV system is affected by the changing weather condition, we demonstrate the merits of the proposed controller under changing ambient temperature and solar radiation power.

Summary:

- For the first time, a nonlinear optimal control problem is formulated and analytically solved for tracking maximum power point of the solar PV systems.
- A model-free version of this framework were developed that only requires measurements of the states.
- The proposed technique outperformed the previous control techniques implemented, based on a realistic Matlab simulation.
- In an alternative scheme, the piecewise RL approach proposed in Chapter 4 was implemented on the solar PV system that successfully reached the maximum-power point with no knowledge of the system dynamics.

1.4 Preliminaries

1.4.1 Notation

We will denote the p-norm by $\|\cdot\|_p$. For defining a set of bases, any operator on a vector x, is performed component-wise e.g. $x^2 = [x_1^2, \dots, x_n^2]$. Moreover, a diagonal square matrix A with elements $A_1, \dots, A_n$ on the diagonal is shortened as $A = \text{diag}([A_1, \dots, A_n])$. The set of real and non-negative real numbers are given by $\mathbb{R}$ and $\mathbb{R}^+$, respectively. The set of all interior points of X is denoted by $\text{int}(X)$.

1.4.2 Optimal Control

Control methods in applications generally involve some trial and error process through which design parameters are chosen to satisfy desirable performance of the system. Desirable performance is usually determined in terms of systems responses such as peak overshoot, settling time, and rise time. Furthermore, to reach the desirable response of the system, the control effort usually need to be observed and restricted in some domain. Such design considerations cannot be generally accomplished for complicated systems by classical control methods. Hence, optimal control framework is well-known as a direct approach to the synthesis of such systems. In this regard, the goal of optimal control theory is to obtain the control input that is required to satisfy a performance measure and physical constraints.

First, to define an optimal control problem we need a model of the control system. Hence, consider the general nonlinear system is given

$$\dot{x} = f(t, x, u), \quad x(t_0) = x_0, \tag{1.1}$$

where $x \in \mathbb{R}^n$ is the state, u is the control input taking values in $U \in \mathbb{R}^m$, x_0 is the initial state at the initial time t_0.

Then, a cost functional is required to assign a cost to any trajectory of the system. Behaviors of the control system is determined by the control signal. Hence, the cost functional associates a cost to any given control signal. A general form of cost functional can be written as

$$J(t_0, x_0, t_f, u) = \int_{t_0}^{t_f} L(t, x(t), u(t))\mathrm{dt} + \Phi(t_f, x_f), \tag{1.2}$$

where L and Φ are the running cost and the terminal cost respectively. Given the cost functional, we can then define the optimal control problem as finding a control u that takes the system along a trajectory that minimizes the cost functional. This problem is known as the Bolza problem in optimal control that results in the Lagrange problem as a special case if there is no terminal cost [100]. Moreover, this cost functional can represent different problems in control by adjusting the target set as the final condition. For instance, a free-time fixed-endpoint optimal problem is given by enforcing the target set $[t_0, \infty] \times \{x_f\}$.

Dynamic Programming

Consider the cost functional (1.2) with the fixed final time t_1, where the endpoint is free. Then, instead of $J(t_0, x_0, u)$, DP suggests minimizing

$$J(t, x, u) = \int_{t}^{t_1} L(s, x(s), u(s))\mathrm{ds} + \Phi(x_f),$$

where $t \in [t_0, t_1)$ and $x \in \mathbb{R}^n$.

This brings us to assume a corresponding value function

$$V(t,x) = \inf_{u_{[t,t_1]}} J(t,x,u)$$

that gives the optimal cost-to-go from (t,x), if a solution exists.

Necessary Conditions

Having the value function defined, we can introduce the principle of optimality. The value function satisfies

$$V(t,x) = \inf_{u_{[t,t+\Delta t]}} \{\int_t^{t+\Delta t} L(s,x(s),u(s))\mathrm{ds}\} + V(t+\Delta t, x(t+\Delta t)), \tag{1.3}$$

for any $(t,x) \in [t_0,t_1) \times \mathbb{R}^n$ and $\Delta t \in (0, t_1 - t]$.

The importance of this condition is that we can now search for the optimal solution over a small time interval. Then, the optimal control has to minimize the cost over this interval plus the remaining cost-to-go [100].

HJB Equation

Looking at relation (1.3), the value function appears in both sides with two different times and states. Therefore, a dynamic relation can be derived in the form of a partial differential equation as the following

$$-V_t(t,x) = \inf_{u\in U}\{L(t,x,u) + V_x^T(t,x)f(t,x,u)\}$$

for $t \in [t_0,t_1)$ and $x \in \mathbb{R}^n$. This equation is well known as the HJB equation.

Sufficient Conditions

Suppose that a $\mathbb{C}^1$ function $\hat{V} : [t_0,t_1] \times \mathbb{R}^n \to \mathbb{R}$ satisfies the HJB equation

$$-\hat{V}_t(t,x) = \inf_{u\in U}\{L(t,x,u) + \hat{V}_x^T(t,x)f(t,x,u)\}$$

for any $(t,x) \in [t_0,t_1) \times \mathbb{R}^n$ and the boundary condition

$$\hat{V}(t_1,x) = \Phi(x).$$

Suppose that a control $\hat{u} : [t_0, t_1] \to U$ and the corresponding trajectory $\hat{x} : [t_0, t_1] \to \mathbb{R}^n$, with the given initial condition $\hat{x}(t_0) = x_0$ satisfy everywhere the equation

$$\begin{aligned} L(t, \hat{x}(t), \hat{u}(t)) + & \hat{V}_x^T(t, \hat{x}(t)) f(t, \hat{x}(t), \hat{u}(t)) \\ & = \min_{u \in U} \{ L(t, \hat{x}(t), u) + \hat{V}_x^T(t, \hat{x}(t)) f(t, \hat{x}(t), u) \}, \end{aligned}$$

which is equivalent to the Hamiltonian maximization condition

$$H(t, \hat{x}(t), \hat{u}(t), -\hat{V}_x^T(t, \hat{x}(t))) = \max_{u \in U} H(t, \hat{x}(t), u, -\hat{V}_x^T(t, \hat{x}(t))).$$

Then $\hat{V}(t_0, x_0)$ is the optimal cost and $\hat{u}$ is an optimal control [100].

1.4.3 System Identification

System identification can be defined as a set of techniques used for obtaining a model of a dynamical system based on the observations and our prior knowledge of the system. In practice, the model acquired almost always represents only an approximation of the real plant.

The identification techniques can be categorized in three different groups. In white-box methods, physical laws are used to directly derive the set of equations defining the system. On the other hand, in the grey-box identification, only the structure of the model is known while there exist some unknown or uncertain parameters that need to be estimated through observations. Accordingly, the structure assumed may be a result of a white-box identification with uncertain parameters. Finally, the black-box technique utilizes a generic parameterization that is chosen independently from the physical laws and relationships underlying the system [85]. In what follows, we review some well-known techniques for updating a parameterized model through observations.

Recursive Least Squares

Consider the linear regression

$$y_k = \phi_k^T w + e_k,$$

where y_k and ϕ_k are the observation and the regressor vector. Moreover, e_k denotes the prediction error. The objective is to minimize the sum of squares

$$J(w) = \sum_{k=1}^{N} (y_k - \phi_k^T w)^2,$$

with respect to the parameters w.

Then using the least squares technique, an estimation of the parameters can be obtained as

$$\hat{w}_N = (\sum_{k=1}^{N} \phi_k \phi_k^T)^{-1} \sum_{k=1}^{N} \phi_k y_k,$$

assuming that the matrix inverse exists. However, this technique cannot be implemented online since the complexity grows with the number of observations made. Therefore, the recursive algorithm is used instead that is

$$\begin{aligned} \hat{w}_k &= \hat{w}_{k-1} + L_k(y_k - \phi_k^T \hat{w}_{k-1}), \\ L_k &= P_{k-1}\phi_k[1 + \phi_k^T P_{k-1}\phi_k]^{-1}, \\ P_k &= P_{k-1} - P_{k-1}\phi_k\phi_k^T P_{k-1}[1 + \phi_k^T P_{k-1}\phi_k]^{-1}, \end{aligned}$$

$k = 1, \ldots N$, and for some initial P_0 and w_0 [104].

Gradient Descent

Gradient Descent (GD) is an iterative optimization algorithm that can be used to obtain an estimation of the model parameters. Assume a differentiable objective $J(w)$ is given. Then, by moving in the direction of steepest decent as

$$w_k = w_{k-1} - \gamma \partial J(w_k)/\partial w_k$$

for a small enough $\gamma \in \mathbb{R}^+$, we have $J(w_{k-1}) \leq J(w_k)$. Therefore, we hope the resulting monotonic sequence leads to a desired minimum.

Sparse Identification

In the sparse identification techniques, in addition to minimizing the prediction error, we also minimize the magnitude of the parameters as the following

$$\hat{w} = \arg\min_{w} \quad \sum_{k=1}^{N} (y_k - \phi_k^T w)^2 + \lambda \|w\|_1, \tag{1.4}$$

where $\lambda > 0$ is a weighting factor. This choice of the cost encourages the sparsity in the model. The technique for solving such an optimization problem will be discussed in detail in the next chapter.

Persistence of Excitation

Consider the cost

$$J(w) = \frac{1}{2} e^T e.$$

Then a continuous GD update rule for the parameters can be obtained as

$$\dot{w} = -\gamma \partial J(w)/\partial w = -\gamma \phi(x) e^T.$$

Given $\tilde{w} = w - w^*$, as the estimation error, we have

$$\dot{\tilde{w}} = \dot{w} = -\gamma \phi(x) \phi^T(x) \tilde{w},$$

where we substituted $e = \tilde{w}\phi(x)$. However, this equation does not guarantee exponential convergence since $\phi(x)$ may become zero while convergence is not finished. Therefore, an extra condition is required to assure $\phi(x)$ remains non-zero for some span of time so that the estimation error converges zero. This is known as the PE condition that is given by

$$\int_t^{t+T} \phi(x)\phi^T(x) \mathrm{d}\tau \geq \alpha_0 I,$$

for all $t \geq t_0$ and some $\alpha_0 \geq 0$ [121].

1.4.4 Reinforcement Learning

In [96], a complete review of RL methods and their relation with adaptive and optimal control are provided. In this section, for the sake of self-containedness, we summarize some concepts and techniques in RL.

Consider the following nonlinear affine system in discrete time

$$x_{k+1} = f(x_k) + g(x_k) u_k,$$

where $x_k \in \mathbb{R}^n$ and control input $u_k \in \mathbb{R}^m$. Moreover, the optimal behavior is defined by

$$V_h(x_k) = \sum_{i=k}^{\infty} \gamma^{i-k} r(x_i, u_i),$$

with the discount factor $0 < \gamma \leq 1$, and a feedback policy $u_k = h(x_k)$. The Bellman optimality condition suggests that

$$V_h^*(x_k) = \min_{h(.)} \quad [r(x_k, h(x_k)) + \gamma V_h^*(x_{x+1})],$$

and the optimal control policy is given by the argument of this optimization problem. Accordingly, DP is an approach for solving the optimal control backwards in time. This requires us to have the future optimal policy for obtaining the optimal policy at current time, hence, it is inherently an offline method. Therefore, by employing this technique, we assume the full knowledge of the system dynamics. However, in RL, we seek a solution for this optimal control problem online.

Assume an admissible policy $u_k = h(x_k)$ is given. Moreover, the corresponding value is $V_h(x_k)$. Then,

$$h'(x_k) = \arg\min_{h(.)} \quad [r(x_k, h(x_k)) + \gamma V_h(x_{x+1})]$$

suggests an update rule for the policy where the resulting value satisfies $V_h'(x_k) \leq V_h(x_k)$. Accordingly, an online iterative scheme can be derived instead of solving the optimal control problem offline. In what follows, we review some well-known iterative techniques in RL.

Policy Iteration

Let us assume that a stabilizing initial policy is given by $h_0(x)$. Then the policy can be evaluated by using the Bellman equation.

$$V_{j+1}(x_k) = r(x_k, h_j(x_k)) + \gamma V_{j+1}(x_{k+1}).$$

Accordingly, the value is used to improve the policy by

$$h_{j+1}(x_k) = \arg\min_{h(.)} \quad (r(x_k, h(x_k)), \gamma V_{j+1}(x_{k+1})).$$

Value Iteration

In PI the initial policy needs to be stabilizing. However, this is not a requirement in VI. Assuming an initial policy $h_0(x_k)$, we update the value using

$$V_{j+1}(x_k) = r(x_k, h_j(x_k)) + \gamma V_j(x_{k+1}).$$

Unlike PI which requires solving a nonlinear equation for evaluating the policy, this can be easily done by a recursion in VI. Having the value updated, it is used to improve the policy as below

$$h_{j+1}(x_k) = \underset{h(.)}{\arg\min} \quad (r(x_k, h(x_k)), \gamma V_{j+1}(x_{k+1})).$$

1.5 Outlines

Chapter 1, as the introduction, includes a review of the literature. Moreover, we motivate this dissertation by discussing the limitations of related works, and challenges known in MBRL. We provide preliminary discussions together with a summary of the contributions.

In Chapter 2, we consider an infinite horizon optimal control problem addressing a given control objective. A structured continuous-time description of the system in terms of a collection of bases, together with a value function that is parameterized in quadratic form, is utilized to achieve an update algorithm for the parameters. As a result, a matrix differential equation of the parameters is constructed, the solution of which is used to describe the optimal feedback control in terms of the bases at every time step. In the numerical results, the presented algorithm is implemented on four nonlinear benchmark examples. The regulation problem is solved while an identified model of the system is obtained with a bounded prediction error.

In Chapter 3, we obtain an approximate optimal control framework according to the results achieved in Chapter 2 to generate a tracking controller for a nonlinear system that can be implemented as an online model-based learning technique. We establish a control strategy that minimizes a specified quadratic tracking objective function by assuming a structured unknown nonlinear system enhanced with the dynamics of a commander system. The suggested optimum tracking framework is utilized as an online model-based reinforcement learning strategy in which we iteratively update the system model and construct a corresponding control using a system identification method. In the simulation results, we implement the presented approach to learn a tracking control on two nonlinear benchmark problems.

In chapter 4, a piecewise affine framework is suggested for controlling nonlinear systems with unknown dynamics. We develop the idea of SOL to acquire a piecewise nonlinear framework where each piece is responsible for locally learning and controlling over some partition of the domain. Then, we emphasize on learning in the form of the well-known PWA as a special case of the proposed framework, for which we suggest an optimization-based verification technique with considering the uncertainty bounds estimated. In the numerical results, we implemented the approach on the pendulum system as a benchmark example to obtain an Region of Attraction (ROA). Moreover, the comparison results with related well-know control techniques are provided to better highlight the merits of the approach.

In Chapter 5, we implement the learning technique discussed in chapter 2 on quadrotor systems. We employ a structured model parameterized by a set of bases to identify the governing dynamics of quadrotors where the control objective is to fly to a set goal position and preserve the hovering state at that point. In the simulation results, a nonlinear model of the quadrotor is exploited as a black box that replaces the real quadrotor. Accordingly, the flight data together with the runtime results are reported within the learning process to verify the presented approach.

In Chapter 6, we investigate applications of a nonlinear optimal control approach and the piecewise MBRL technique presented in chapter 4 on solar PV systems. Considering the nonlinearity appearing in the model of the solar PV system, we employ a nonlinear optimal feedback control scheme to reach the Maximum Power Point (MPP), and deal with the oscillations induced by the chattering phenomenon in the control. In an alternative technique, we consider a similar problem with an unknown solar PV system. Then, we implement the piecewise learning technique proposed. To demonstrate the merits of the proposed framework, the obtained optimal feedback control, together with the partial shading condition and model-free approach, is simulated under various weather conditions.

Chapter 2

Structured Online Learning-based Control of Continuous-time Nonlinear Systems

The results presented in this chapter are partially published in [52], and the journal paper is submitted to [54].

2.1 Introduction

In this chapter, we propose an MBRL technique for solving a control regulation problem for unknown nonlinear continuous-time systems. In Section 2.2, we propose an optimal control approach based on a particular structure of dynamics, and characterize the optimal feedback control based on a matrix of parameters obtained by a differential equation. Section 2.4 outlines the SOL algorithm designed based on the obtained results. In Section 2.5, we present the numerical results of this algorithm implemented on a few benchmark examples.

2.2 A Structured Approximate Optimal Control Framework

Consider the nonlinear affine system

$$\dot{x} = F(x, u) = f(x) + g(x)u, \tag{2.1}$$

where $x \in D \subset \mathbb{R}^n$, $u \in \Omega \subset \mathbb{R}^m$, $f : D \to \mathbb{R}^n$, and $g : D \to \mathbb{R}^{n \times m}$.

The cost function to be minimized along the trajectory, started from the initial condition $x_0 = x(0)$, is considered in the following linear quadratic form

$$J(x_0, u) = \lim_{T \to \infty} \int_0^T \mathrm{e}^{-\gamma t} \left(x^T Q x + u^T R u \right) \mathrm{dt}, \tag{2.2}$$

where $Q \in \mathbb{R}^{n \times n}$ is positive semi-definite, $\gamma \geq 0$ is the discount factor, and $R \in \mathbb{R}^{m \times m}$ is a diagonal matrix with only positive values, given by design criteria. With $\gamma > 0$, this defines a discounted optimal control problem. Discounted optimal control problem has been discussed in [129, 58]. Moroever, it is widely used in reinforcement learning to determine the time horizon considered for minimizing the objective [95].

For the closed-loop system, by assuming a feedback control law $u = \omega(x(t))$ for $t \in [0, \infty)$, the optimal control is given by

$$\omega^* = \quad \arg \min_{u(\cdot) \in \Gamma(x_0)} J(x_0, u(\cdot)), \tag{2.3}$$

where Γ is the set of admissible controls.

Assumption 1. *f and g can be identified or effectively approximated within the domain of interest by the linear combination of some basis functions $\phi_i \in C^1 : D \to \mathbb{R}$ for $i = 1, 2, \ldots, p$.*

Accordingly, (2.1) is rewritten as

$$\dot{x} = W\Phi(x) + \sum_{j=1}^{m} W_j \Phi(x) u_j, \tag{2.4}$$

where W and $W_j \in \mathbb{R}^{n \times p}$ are the matrices of the coefficients obtained for $j = 1, 2, \ldots, m$, and $\Phi(x) = [\phi_1(x) \quad \ldots \quad \phi_p(x)]^T$.

In what follows, without loss of generality, the cost defined in (2.2) is transformed to the space of bases $\Phi(x)$, that is

$$J(x_0, u) = \lim_{T\to\infty} \int_0^T \mathrm{e}^{-\gamma t} \left(\Phi(x)^T \bar{Q} \Phi(x) + u^T R u\right) \mathrm{d}t, \tag{2.5}$$

where $\bar{Q} = \mathrm{diag}\left([Q], [\mathbf{0}_{(p-n)\times(p-n)}]\right)$ is a block diagonal matrix that contains all zeros except the first block Q which correspond to the linear bases x.

Then the corresponding HJB equation can be written as

$$-\frac{\partial}{\partial t}(\mathrm{e}^{-\gamma t} V) = \min_{u(\cdot)\in\Gamma(x_0)} H, \tag{2.6}$$

by the Hamiltonian defined as

$$H = \mathrm{e}^{-\gamma t}\left(\Phi(x)^T \bar{Q} \Phi(x) + u^T R u\right) + \mathrm{e}^{-\gamma t} \frac{\partial V}{\partial x}^T (W\Phi(x) + \sum_{j=1}^{m} W_j \Phi(x) u_j). \tag{2.7}$$

In general, there exists no analytical approach that can solve such partial differential equation and obtain the optimal value function. However, it has been shown in the literature that approximate solutions can be computed by numerical techniques.

Assume a parameterization of the optimal value function in the following form

$$V = \Phi(x)^T P \Phi(x), \tag{2.8}$$

where P is symmetric.

Remark 1. *Unlike other approximate optimal approaches in the literature, such as, [176, 17, 81], that use a linear combination of bases to parameterize the value function, we assume a quadratic form. As a result, the value function now is defined in the product space $\Lambda := \Phi \times \Phi$. Hence, it is expected that the resulting quadratic terms better contribute to basing a positive value function around $x = 0$. Furthermore, due to the function-approximating properties of bases Φ itself, one may bring them to Λ in addition by including a constant basis c in Φ. Therefore, compared to other approaches, the structure used in (2.8) suggests a more compact way of formulating the problem where by only a limited number of bases in Φ, we can attain a richer set Λ to parameterize the value function.*

Then the Hamiltonian is given by

$$H = \mathrm{e}^{-\gamma t}(\Phi(x)^T \bar{Q} \Phi(x) + u^T R u)$$

$$+ \mathrm{e}^{-\gamma t}\Phi(x)^T P \frac{\partial \Phi(x)}{\partial x}\left(W\Phi(x) + \sum_{j=1}^{m} W_j \Phi(x) u_j\right)$$
$$+\mathrm{e}^{-\gamma t}\left(\Phi(x)^T W^T + \sum_{j=1}^{m} u_j^T \Phi(x)^T W_j^T\right)\frac{\partial \Phi(x)}{\partial x}^T P\Phi(x).$$

Moreover, based on the structure of R, the quadratic term of u is rewritten in terms of its components.

$$\begin{aligned}H = \mathrm{e}^{-\gamma t}\Bigg(&\Phi(x)^T \bar{Q}\Phi(x) + \sum_{j=1}^{m} r_j u_j^2 + \Phi(x)^T P\frac{\partial \Phi(x)}{\partial x} W\Phi(x) + \\ &\Phi(x)^T P\frac{\partial \Phi(x)}{\partial x}\left(\sum_{j=1}^{m} W_j\Phi(x)u_j\right) + \Phi(x)^T W^T \frac{\partial \Phi(x)}{\partial x}^T P\Phi(x) \\ &+ \left(\sum_{j=1}^{m} u_j \Phi(x)^T W_j^T\right)\frac{\partial \Phi(x)}{\partial x}^T P\Phi(x)\Bigg),\end{aligned} \tag{2.9}$$

where $r_j \neq 0$ is the jth component on the diagonal of matrix R. To minimize the resulting Hamiltonian we need

$$\begin{aligned}\frac{\partial H}{\partial u_j} &= 2r_j u_j + 2\Phi(x)^T P \frac{\partial \Phi(x)}{\partial x} W_j \Phi(x) \\ &= 0, \qquad j = 1, 2, \ldots, m.\end{aligned} \tag{2.10}$$

Hence, the jth optimal control input is obtained as

$$u_j^* = -\Phi(x)^T r_j^{-1} P \frac{\partial \Phi(x)}{\partial x} W_j \Phi(x). \tag{2.11}$$

By plugging in the optimal control and the value function in (2.6) we get

$$\begin{aligned}&-\mathrm{e}^{-\gamma t}\Phi(x)^T \dot{P}\Phi(x) + \gamma \mathrm{e}^{-\gamma t}\Phi(x)^T P\Phi(x) \\ &\quad = \mathrm{e}^{-\gamma t}\Bigg(\Phi(x)^T\bar{Q}\Phi(x) + \\ &\quad + \Phi(x)^T P\frac{\partial \Phi(x)}{\partial x}\left(\sum_{j=1}^{m} W_j\Phi(x) r_j^{-1}\Phi(x)^T W_j^T\right)\frac{\partial \Phi(x)}{\partial x}^T P\Phi(x)\end{aligned}$$

$$- 2\Phi(x)^T P\frac{\partial\Phi(x)}{\partial x}\left(\sum_{j=1}^{m} W_j\Phi(x)r_j^{-1}\Phi(x)^T W_j^T\right)\frac{\partial\Phi(x)}{\partial x}^T P\Phi(x)$$

$$+ \Phi(x)^T P\frac{\partial\Phi(x)}{\partial x}W\Phi(x) + \Phi(x)^T W^T\frac{\partial\Phi(x)}{\partial x}^T P\Phi(x)\Bigg).$$

This is rewritten as

$$\begin{aligned}\Phi(x)^T\dot{P}\Phi(x) + \gamma\Phi(x)^T P\Phi(x) &= \Phi(x)^T\bar{Q}\Phi(x)+\\ &- \Phi(x)^T P\frac{\partial\Phi(x)}{\partial x}\left(\sum_{j=1}^{m} W_j\Phi(x)r_j^{-1}\Phi(x)^T W_j^T\right)\frac{\partial\Phi(x)}{\partial x}^T P\Phi(x)\\ &+ \Phi(x)^T P\frac{\partial\Phi(x)}{\partial x}W\Phi(x) + \Phi(x)^T W^T\frac{\partial\Phi(x)}{\partial x}^T P\Phi(x),\end{aligned} \tag{2.12}$$

where a sufficient condition to hold this equation is

$$\begin{aligned}-\dot{P} =& \bar{Q} + P\frac{\partial\Phi(x)}{\partial x}W + W^T\frac{\partial\Phi(x)}{\partial x}^T P - \gamma P\\ &- P\frac{\partial\Phi(x)}{\partial x}\left(\sum_{j=1}^{m} W_j\Phi(x)r_j^{-1}\Phi(x)^T W_j^T\right)\frac{\partial\Phi(x)}{\partial x}^T P.\end{aligned} \tag{2.13}$$

This equation has to be solved backward to get a value of P that characterizes the optimal value function (2.8) and control (2.11). However, it has been shown that the forward integration of such equation converges to similar results as long as we are not very close to the initial time [131].

Remark 2. *While the similarity between the derived optimal control and the LQR problem cannot be denied, there are substantial differences. It should be noted that the matrix differential equation (2.13) is derived in terms of Φ which is of dimension p in contrast with the LQR formulation that includes only the linear terms of the state with dimension n.*

Remark 3. *Because of the general case considered in obtaining (2.13), where Φ includes arbitrary basis functions of the state, there exists no way to escape from the state-dependency in this equation, except in the linear case as mentioned. Hence, we require (2.13) be solved along the trajectories of the system.*

2.3 Local Stability and Optimality Analysis

In this section, we will present the stability analysis of the approach and its connections with the FPRE for linear systems [165, 131]. To do this, we first formulate the LQR problem for the linearized model of (2.4). Then we will show that once we get close enough to the origin, the integration of (2.13) will be governed by the forward propagated solution of the linearized system, as the dominant part.

Before we start, we need some regulations on the system that can be assured with no loss of generality. Consider the structured system (2.4). We will assume an equilibrium point at the origin. Moreover, we need to redefine the bases by using the following lemma.

Lemma 1. *Assuming that the constant basis $\phi_1(x) = 1$ is included in Φ, we can always redefine Φ such that $\phi_i(0) = 0$ for $i = 2, \dots, p$. To hold this properties, we redefine $\phi_i(x) := \phi_i(x) - \phi_i(0)$ for $i = 2, \dots, p$. Accordingly, we also set the W to zeros in the entries corresponding to basis 1. For instance, the system $\dot{x} = 1 - \cos x = [1 \quad -1] \begin{bmatrix} 1 \\ \cos x \end{bmatrix}$ can be equivalently rewritten as $\dot{x} = [0 \quad -1] \begin{bmatrix} 1 \\ \cos x - 1 \end{bmatrix}$.*

Accordingly, we construct the vector of basis as $\Phi = [1 \quad x^T \quad \Gamma(x)^T]^T$, where Γ includes all the nonlinear basis. Then, according to Lemma 1, the system (2.4) is represented by some block structured matrices.

$$\dot{x} = \begin{bmatrix} 0 & W_2 & W_3 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma(x) \end{bmatrix} + \begin{bmatrix} W_{j_1} & W_{j_2} & W_{j_3} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma(x) \end{bmatrix} u. \tag{2.14}$$

2.3.1 Linear Quadratic Regulator

By defining $\Gamma_1 = \frac{\partial \Gamma(x)}{\partial x}|_{x=0}$, the linearization of (2.14) at the equilibrium point yields

$$\dot{x} = Ax + \sum_{j=1}^{m} B_j u_j, \tag{2.15}$$

where $A = W_2 + W_3\Gamma_1$, and $B_j = W_{j_1}$.

Now consider the LQR problem with quadratic cost (2.2) and $\gamma = 0$ for the linearized system (2.15). Then, the optimal control is given by $u = -r_j^{-1} B^T \bar{S} x$, where $\bar{S}$ is the solution of the well-known algebraic Riccati equation

$$Q + \bar{S}A + A^T\bar{S} - \bar{S}(\sum_{j=1}^{m} B_j r_j^{-1} B_j^T)\bar{S} = 0. \tag{2.16}$$

Alternatively, we consider the forward solution of the following DRE where we update the feedback controller with the solution $S(t)$ at any $t \in [0, \infty)$,

$$\begin{aligned} u_j &= -r_j^{-1} B^T S x, \\ \dot{S} &= Q + SA + A^T S - S(\sum_{j=1}^{m} B_j r_j^{-1} B_j^T)^T S. \end{aligned} \tag{2.17}$$

By substitution of A and B, we get

$$u_j = -r_j^{-1} W_{j1}^T S x, \tag{2.18}$$

$$\begin{aligned} \dot{S} = Q &+ S(W_2 + W_3\Gamma_1) + (W_2^T + \Gamma_1^T W_3^T)S \\ &- S(\sum_{j=1}^{m} W_{j1} r_j^{-1} W_{j1}^T)S. \end{aligned} \tag{2.19}$$

The following theorem helps to illuminate the relation of FPRE with the LQR control.

Lemma 2. *([131]) Assume that (A, B) is controllable, (A, Q) is observable, and Q is positive definite. Consider plant (2.15) with the control law (2.17), where, for all $t \in [0, \infty)$, $S(t)$ is the positive-semi-definite solution of the forward propagate Riccati equation (2.17). Then, there exist some $T_1 > 0$ such that the feedback control law renders the closed-loop system asymptotically stable for all $t > T_1$. Moreover, as $t \to \infty$, $S(t)$ will converge to $\bar{S}$.*

Proof. See Theorems 1 and 4 in [131].

2.3.2 SOL Control

Based on the optimal control framework presented, for a known structured nonlinear system in the form of (2.14), the optimal control is given by (2.11). Moreover, the value is updated by the evolutions of the parameters in (2.13). Accordingly, the following theorem provides guarantees for the stability of the closed-loop system.

Theorem 1. *Let $\bar{D} = \{x \in D, \|x\| < r\}$, then, there exist some $r > 0$,$\gamma > 0$, and $T_2 > 0$ such that the solution $P(t)$ of (2.13) starting from $P(0) = 0$ establishes an stabilizing controller for an initial condition $x_0 \in \bar{D}$ and for all $t > T_2$. Hence, the nonlinear closed loop system is asymptotically stable over $\bar{D}$.*

Proof. Consider a ball of radius r around the origin. We will show that the dominating part of the solutions of (2.13) is equivalent to (2.19) for some small r. For this purpose, we first consider the Taylor expansion of bases in (2.14) and its partial derivatives as below,

$$\Phi(x) = \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}, \text{and} \qquad \frac{\partial \Phi(x)}{\partial x} = \begin{bmatrix} 0 \\ \mathbf{I} \\ \Gamma_1 + O(x) \end{bmatrix}, \tag{2.20}$$

where $O(x^\alpha)$ denotes higher order terms $x_1^{\alpha_1} x_2^{\alpha_2} \dots x_n^{\alpha_n}$ with $\alpha = \sum_{i=1}^n \alpha_i$, and non-negative integers α_i. It should be noted that the expansion of the regulated nonlinear bases does not include any constant term according to Remark 1. Moreover, $\bar{Q}$ and P are structured matrices including rectangular blocks of appropriate dimensions as below,

$$\bar{Q} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & Q & 0 \\ 0 & 0 & 0 \end{bmatrix}, \text{and} \quad P = \begin{bmatrix} P_1 & P_2 & P_3 \\ P_2^T & P_4 & P_5 \\ P_3^T & P_5^T & P_6 \end{bmatrix}.$$

By substituting $\bar{Q}$,P in (2.13), it is easy to investigate that starting from the initial condition $P_0 = 0$, any term in the equations of $\dot{P}_1$, $\dot{P}_2$, and $\dot{P}_3$ will depend on P_1, P_2, or P_3 as below

$$\begin{aligned}
\dot{P}_1 &= -P_2 K_1 K_1^T P_2^T - P_3 \Gamma_1 K_1 K_1^T P_2^T - P_2 K_1 K_1^T P_3^T \\
&\quad - P_3 \Gamma_1 K_1 K_1^T P_3^T, \\
\dot{P}_2 &= P_2 W_2 + P_3 \Gamma_1 W_2 - P_2 K_1 K_1^T P_4 - P_3 \Gamma_1 K_1 K_1^T P_4 \\
&\quad - P_2 K_1 K_1^T P_5^T - P_3 \Gamma_1 K_1 K_1^T P_5^T, \\
\dot{P}_3 &= P_2 W_3 + P_3 \Gamma_1 W_3 - P_2 K_1 K_1^T P_5 - P_3 \Gamma_1 K_1 K_1^T P_5 \\
&\quad - P_2 K_1 K_1^T P_6 - P_3 \Gamma_1 K_1 K_1^T P_6,
\end{aligned}$$

where $K_1 = W_{j_1} + W_{j_2} x + W_{j_3} \Gamma_1 x$. Therefore, solutions P_1, P_2, and P_3 will always stay at zeros, and the matrix P will only grow on the block $\begin{bmatrix} P_4 & P_5 \\ P_5^T & P_6 \end{bmatrix}$. Therefore, for brevity, we will follow the computations only for this block as long as this simplification does not cause ambiguity. Now, let us take one step back and start with

$$\begin{aligned}
&\Phi(x)^T \dot{P} \Phi(x) = \Phi(x)^T \bar{Q} \Phi(x) \\
&\quad + \Phi(x)^T P \frac{\partial \Phi(x)}{\partial x} W \Phi(x) + \Phi(x)^T W^T \frac{\partial \Phi(x)}{\partial x}^T P \Phi(x) \\
&\quad - \Phi(x)^T P \Phi_x W_j \Phi(x) r_j^{-1} \Phi(x)^T W_j^T \frac{\partial \Phi(x)}{\partial x}^T P \Phi(x) \\
&\quad - \gamma \Phi(x)^T P \Phi(x),
\end{aligned} \tag{2.21}$$

as in (2.12). For the non-zero block of P_4, P_5, and P_6 with the corresponding bases, the left hand side can be rewritten as

$$\begin{aligned}
&\begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & \dot{P}_4 & \dot{P}_5 \\ 0 & \dot{P}_5^T & \dot{P}_6 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} = \\
&\begin{bmatrix} 1 \\ x \\ O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & \dot{P}_4 + \Gamma_1^T \dot{P}_5^T + \dot{P}_5 \Gamma_1 + \Gamma_1^T \dot{P}_6 \Gamma_1 & \dot{P}_5 + \Gamma_1^T \dot{P}_6 \\ 0 & \dot{P}_5^T + \dot{P}_6 \Gamma_1 & \dot{P}_6 \end{bmatrix} \\
&\qquad\qquad\qquad\qquad \begin{bmatrix} 1 \\ x \\ O(x^2) \end{bmatrix},
\end{aligned} \tag{2.22}$$

where we shifted the linear term in the third entry of the bases to the second. In the next step, we will consider the following change of variables throughout the matrix differential equation.

$$\begin{aligned}
Z_1 &= P_4 + \Gamma_1^T P_5^T + P_5 \Gamma_1 + \Gamma_1^T P_6 \Gamma_1 \\
Z_2 &= P_5 + \Gamma_1^T P_6 \\
Z_3 &= P_6
\end{aligned} \tag{2.23}$$

For this reason we apply the same modification of bases to all the terms in the right hand side of (2.21). The modification will not affect the first term since $\bar{Q}$ is zero everywhere except in the block corresponding to the second basis, that remained unchanged. Then, let us consider the second term in the right hand side that becomes

$$\Phi(x)^T P \frac{\partial \Phi(x)}{\partial x} W \Phi(x)$$

$$
\begin{aligned}
&= \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & P_4 & P_5 \\ 0 & P_5^T & P_6 \end{bmatrix} \begin{bmatrix} 0 \\ \mathbf{I} \\ \Gamma_1 + O(x) \end{bmatrix} \\
&\qquad \begin{bmatrix} 0 & W_2 & W_3 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & \Upsilon_1 & \Upsilon_2 \\ 0 & \Upsilon_3 & \Upsilon_4 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ x \\ O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & \bar{\Upsilon}_1 & \bar{\Upsilon}_2 \\ 0 & \bar{\Upsilon}_3 & \bar{\Upsilon}_4 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ O(x^2) \end{bmatrix},
\end{aligned}
\tag{2.24}
$$

where

$$
\begin{aligned}
\Upsilon_1 &= P_4 W_2 + P_5 \Gamma_1 W_2 + P_5 O(x) W_2, \\
\Upsilon_2 &= P_4 W_3 + P_5 \Gamma_1 W_3 + P_5 O(x) W_3, \\
\Upsilon_3 &= P_5^T W_2 + P_6 \Gamma_1 W_2 + P_6 O(x) W_2, \\
\Upsilon_4 &= P_5^T W_3 + P_6 \Gamma_1 W_3 + P_6 O(x) W_3, \\
\bar{\Upsilon}_1 &= Z_1 (W_2 + W_3 \Gamma_1) + Z_2 O(x)(W_2 + W_3 \Gamma_1), \\
\bar{\Upsilon}_2 &= Z_1 W_3 + Z_2 O(x) W_3, \\
\bar{\Upsilon}_3 &= Z_2^T (W_2 + W_3 \Gamma_1) + Z_3 O(x)(W_2 + W_3 \Gamma_1), \\
\bar{\Upsilon}_4 &= Z_2^T W_3 + Z_3 O(x) W_3,
\end{aligned}
$$

and we used (2.23) to get

$$
\begin{aligned}
& P_4 W_2 + P_5 \Gamma_1 W_2 + P_4 W_3 \Gamma_1 + P_5 \Gamma_1 W_3 \Gamma_1 \\
&\quad + \Gamma_1^T P_5^T W_2 + \Gamma_1^T P_6 \Gamma_1 W_2 + \Gamma_1^T P_5^T W_3 \Gamma_1 + \Gamma_1^T P_6 \Gamma_1 W_3 \Gamma_1 \\
&\quad = (P_4 + \Gamma_1^T P_5^T + P_5 \Gamma_1 + \Gamma_1^T P_6 \Gamma_1)(W_2 + W_3 \Gamma_1) \\
&\quad = Z_1 (W_2 + W_3 \Gamma_1), \\
& P_5 O(x) W_2 + P_6 O(x) W_3 \Gamma_1 + \Gamma_1^T P_6 O(x) W_2 + \Gamma_1^T P_6 O(x) W_3 \Gamma_1 \\
&\quad = Z_2 O(x)(W_2 + W_3 \Gamma_1), \\
& P_4 W_3 + P_5 \Gamma_1 W_3 + P_5 O(x) W_3 + \Gamma_1^T P_5^T W_3 \\
&\qquad + \Gamma_1^T P_6 \Gamma_1 W_3 + \Gamma_1^T P_6 O(x) W_3
\end{aligned}
$$

$$
\begin{aligned}
&= (P_4 + P_5\Gamma_1 + \Gamma_1^T P_5^T)W_3 + (P_5 + \Gamma_1^T P_6)O(x)W_3 \\
&\quad + \Gamma_1^T P_6\Gamma_1 W_3 \\
&= (Z_1 - \Gamma_1^T Z_3\Gamma 1)W_3 + Z_2 O(x)W_3 + \Gamma_1^T Z_3\Gamma_1 W_3 \\
&= Z_1 W_3 + Z_2 O(x)W_3, \\
P_5^T W_2 &+ P_6\Gamma_1 W_2 + P_6 O(x)W_2 + P_5^T W_3\Gamma_1 + P_6\Gamma_1 W_3\Gamma_1 \\
&\quad + P_6 O(x)W_3\Gamma_1 \\
&= (P_5^T + P_6\Gamma_1)(W_2 + W_3\Gamma_1) + P_6 O(x)(W_2 + W_3\Gamma_1) \\
&= Z_2^2(W_2 + W_3\Gamma_1) + Z_3 O(x)(W_2 + W_3\Gamma_1), \\
P_5^T W_3 &+ P_6\Gamma_1 W_3 + P_6 O(x)W_3 \\
&= Z_2^T W_3 + Z_3 O(x)W_3.
\end{aligned}
$$

Furthermore, for the last term in the right hand side of (2.21), the followings hold.

$$
\begin{aligned}
&\Phi(x)^T P\frac{\partial\Phi(x)}{\partial x}\sum_{j=1}^{m}(W_j\Phi(x)r_j^{-1}\Phi(x)^T W_j^T)\frac{\partial\Phi(x)}{\partial x}^T P\Phi(x) \\
&= \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & P_4 & P_5 \\ 0 & P_5^T & P_6 \end{bmatrix} \begin{bmatrix} 0 \\ \mathbf{I} \\ \Gamma_1 + O(x) \end{bmatrix} \sum_{j=1}^{m} (\\
&\begin{bmatrix} W_{j_1} & W_{j_2} & W_{j_3} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} r_j^{-1} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T \\
&\begin{bmatrix} W_{j_1} \\ W_{j_2} \\ W_{j_3} \end{bmatrix}) \begin{bmatrix} 0 \\ \mathbf{I} \\ \Gamma_1 + O(x) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & P_4 & P_5 \\ 0 & P_5^T & P_6 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & P_4 & P_5 \\ 0 & P_5^T & P_6 \end{bmatrix} \begin{bmatrix} 0 \\ \mathbf{I} \\ \Gamma_1 + O(x) \end{bmatrix} \Omega_2 \\
&\quad \begin{bmatrix} 0 \\ \mathbf{I} \\ \Gamma_1 + O(x) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & P_4 & P_5 \\ 0 & P_5^T & P_6 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T (\begin{bmatrix} 0 \\ P_4 + P_5\Gamma_1 \\ P_5^T + P_6\Gamma_1 \end{bmatrix} + \begin{bmatrix} 0 \\ P_5 O(x) \\ P_6 O(x) \end{bmatrix})\Omega_2
\end{aligned}
$$

$$
\begin{aligned}
&(\begin{bmatrix} 0 \\ P_4 + P_5\Gamma_1 \\ P_5^T + P_6\Gamma_1 \end{bmatrix} + \begin{bmatrix} 0 \\ P_5 O(x) \\ P_6 O(x) \end{bmatrix})^T \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & \Omega_3 & \Omega_4 \\ 0 & \Omega_4^T & \Omega_5 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} \\
&+ \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & \Omega_6 & \Omega_7 \\ 0 & \Omega_7^T & \Omega_8 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ x \\ O(x^2) \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & \bar{\Omega}_3 & \bar{\Omega}_4 \\ 0 & \bar{\Omega}_4^T & \bar{\Omega}_5 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ O(x^2) \end{bmatrix} + \begin{bmatrix} 1 \\ x \\ O(x^2) \end{bmatrix}^T \\
&\qquad \begin{bmatrix} 0 & 0 & 0 \\ 0 & \bar{\Omega}_6 & \bar{\Omega}_7 \\ 0 & \bar{\Omega}_7^T & \bar{\Omega}_8 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ O(x^2) \end{bmatrix},
\end{aligned}
\tag{2.25}
$$

where Ω_2 to Ω_8 are defined as

$$
\begin{aligned}
\Omega_2 &= \sum_{j=1}^{m} r_j^{-1} \begin{bmatrix} W_{j_1} & W_{j_2} & W_{j_3} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x \end{bmatrix}^T \begin{bmatrix} W_{j_1} \\ W_{j_2} \\ W_{j_3} \end{bmatrix}^T \\
&= \sum_{j=1}^{m} r_j^{-1} \begin{bmatrix} W_{j_1} & W_{j_2} & W_{j_3} \end{bmatrix} \begin{bmatrix} 1 & x^T & x^T\Gamma_1^T \\ x & xx^T & xx^T\Gamma_1^T \\ \Gamma_1 x & \Gamma_1 xx^T & \Gamma_1 xx^T\Gamma_1^T \end{bmatrix} \begin{bmatrix} W_{j_1} \\ W_{j_2} \\ W_{j_3} \end{bmatrix}^T \\
&= \sum_{j=1}^{m} \big(W_{j_1}W_{j_1}^T + W_{j_2}xW_{j_1}^T + W_{j_3}\Gamma_1 x W_{j_1}^T + W_{j_1}x^T W_{j_2}^T + \\
&\qquad W_{j_2}xx^T W_{j_2}^T + W_{j_3}\Gamma_1 xx^T W_{j_2}^T + W_{j_1}x^T\Gamma_1^T W_{j_3}^T + \\
&\qquad W_{j_2}xx^T\Gamma_1^T W_{j_2}^T + W_{j_3}\Gamma_1 xx^T\Gamma_1^T W_{j_3}^T \big) r_j^{-1}, \\
\Omega_3 &= (P_4 + P_5\Gamma_1)\Omega_2(P_4 + \Gamma_1^T P_5^T), \\
\Omega_4 &= (P_4 + P_5\Gamma_1)\Omega_2(P_5 + \Gamma_1^T P_6), \\
\Omega_5 &= (P_5^T + P_6\Gamma_1)\Omega_2(P_5 + \Gamma_1^T P_6), \\
\Omega_6 &= (P_4 + P_5\Gamma_1)\Omega_2 O(x) P_5^T + P_5 O(x)\Omega_2(P_4 + \Gamma_1^T P_5^T) \\
&\quad + P_5 O(x^2) P_5^T,
\end{aligned}
$$

$$\begin{aligned}\Omega_7 &= (P_4 + P_5\Gamma_1)\Omega_2 O(x)P_6 + P_5 O(x)\Omega_2(P_5 + \Gamma_1^T P_6)\\ &\quad + P_5 O(x^2)P_6,\\ \Omega_8 &= (P_5^T + P_6\Gamma_1)\Omega_2 O(x)P_6 + P_6 O(x)\Omega_2(P_5 + \Gamma_1^T P_6)\\ &\quad + P_6 O(x^2)P_6.\end{aligned}$$

Moreover, $\bar{\Omega}_3$ to $\bar{\Omega}_3$, are their analogous blocks after modifying the bases, where they can also be rewritten in terms of the new variables defined in (2.23) as below

$$\begin{aligned}\bar{\Omega}_3 &= (P_4 + \Gamma_1^T P_5^T + P_5\Gamma_1 + \Gamma_1^T P_6\Gamma_1)\Omega_2\\ &\quad\ (P_4 + \Gamma_1^T P_5^T + P_5\Gamma_1 + \Gamma_1^T P_6\Gamma_1)^T\\ &= Z_1 W_{j_1} W_{j_1}^T Z_1^T,\\ \bar{\Omega}_4 &= (P_4^T + P_5\Gamma_1 + \Gamma_1^T P_5^T + \Gamma_1^T P_6\Gamma_1)\Omega_2(P_5 + \Gamma_1^T P_6)\\ &= Z_1 W_{j_1} W_{j_1}^T Z_2,\\ \bar{\Omega}_5 &= \Omega_5 = Z_2^T W_{j_1} W_{j_1}^T Z_2,\\ \bar{\Omega}_6 &= \Omega_6 + \Gamma_1^T\Omega_7^T + \Omega_7\Gamma_1 + \Gamma_1^T\Omega_8\Gamma_1\\ &= (P_4 + P_5\Gamma_1 + \Gamma_1^T P_5^T + \Gamma_1^T P_6\Gamma_1)\Omega_2 O(x)P_5^T\\ &\quad + (P_5 + \Gamma_1^T P_6)O(x)\Omega_2(P_4 + \Gamma_1^T P_5^T)\\ &\quad + (P_4 + P_5\Gamma_1 + \Gamma_1^T P_5^T + \Gamma_1^T P_6\Gamma_1)\Omega_2 O(x)P_6\Gamma_1\\ &\quad + (P_5 + \Gamma_1^T P_6)O(x)\Omega_2(P_5 + \Gamma_1^T P_6)\Gamma_1\\ &\quad + (P_5 + \Gamma_1^T P_6)O(x^2)P_5^T + (P_5 + \Gamma_1^T P_6)O(x^2)P_6\Gamma_1\\ &= (P_4 + P_5\Gamma_1 + \Gamma_1^T P_5^T + \Gamma_1^T P_6\Gamma_1)\Omega_2 O(x)(P_5^T + P_6\Gamma_1)\\ &\quad + (P_5 + \Gamma_1^T P_6)O(x)\Omega_2(P_4 + \Gamma_1^T P_5^T + P_5\Gamma_1 + \Gamma_1^T P_6\Gamma_1)\\ &\quad + (P_5 + \Gamma_1^T P_6)O(x^2)(P_5^T + P_6\Gamma_1)\\ &= Z_1\Omega_2 O(x)Z_2^T + Z_2 O(x)\Omega_2 Z_1^T + Z_2 O(x^2)Z_2^T,\\ \bar{\Omega}_7 &= \Omega_7 + \Gamma_1^T\Omega_8 = Z_1\Omega_2 O(x)Z_3 + Z_2 O(x)\Omega_2 Z_2\\ &\quad\ + Z_2 O(x^2)Z_3,\\ \bar{\Omega}_8 &= \Omega_8 = Z_2^T\Omega_2 O(x)Z_3 + Z_3 O(x)\Omega_2 Z_2 + Z_3 O(x^2)Z_3,\end{aligned}$$

where we also used the fact that the constant term will dominate in Ω_2 as $x \to 0$. Hence, we get $\Omega_2 \to \sum_{j=1}^{m} r_j^{-1} W_{j1} W_{j1}^T$.

By substituting (2.22), (2.24), and (2.25) in (2.21), one can obtain

$$\dot{Z}_1 = Q + Z_1(W_2 + W_3\Gamma_1) + (W_2^T + \Gamma_1^T W_3^T)Z_1^T$$

$$
\begin{aligned}
&- Z_1(\sum_{j=1}^{m} W_{j_1} r_j^{-1} W_{j_1}^T) Z_1^T + Z_2 O(x)(W_2 + W_3\Gamma_1) \\
&- \gamma Z_1 + (W_2 + W_3\Gamma_1)^T O(x) Z_2^T - Z_1 \Omega_2 O(x) Z_2^T \\
&- Z_2 O(x) \Omega_2 Z_1^T - Z_2 O(x^2) Z_2^T \\
=& Q + Z_1(W_2 + W_3\Gamma_1) + (W_2^T + \Gamma_1^T W_3^T) Z_1^T \\
&- Z_1(\sum_{j=1}^{m} W_{j_1} r_j^{-1} W_{j_1}^T) Z_1^T - \gamma Z_1 + O(x), \qquad (2.26)
\end{aligned}
$$

$$
\begin{aligned}
\dot{Z}_2 =& Z_1 W_3 + (W_2 + W_3\Gamma_1)^T Z_2 - Z_1(\sum_{j=1}^{m} W_{j_1} r_j^{-1} W_{j_1}^T) Z_2 \\
&- \gamma Z_2 + Z_2 O(x) W_3 + (W_2 + W_3\Gamma_1)^T O(x) Z_3^T \\
&- Z_1 \Omega_2 O(x) Z_3 - Z_2 O(x) \Omega_2 Z_2 \\
&- Z_2 O(x^2) Z_3, \qquad (2.27)
\end{aligned}
$$

$$
\begin{aligned}
\dot{Z}_3 =& Z_2^T W_3 + W_3^T Z_2 - Z_2(\sum_{j=1}^{m} W_{j_1} r_j^{-1} W_{j_1}^T) Z_2 \\
&- \gamma Z_3 + Z_3 O(x) W_3 + W_3^T O(x) Z_3^T - Z_2^T \Omega_2 O(x) Z_3 \\
&- Z_3 O(x) \Omega_2 Z_2 - Z_3 O(x^2) Z_3 \qquad (2.28)
\end{aligned}
$$

Moreover, for the optimal control (2.11) takes the following form,

$$
\begin{aligned}
u_j^* =& - \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T r_j^{-1} \begin{bmatrix} 0 & 0 & 0 \\ 0 & P_4 & P_5 \\ 0 & P_5^T & P_6 \end{bmatrix} \begin{bmatrix} 0 \\ \mathbf{I} \\ \Gamma_1 + O(x) \end{bmatrix} \\
& \begin{bmatrix} W_{j_1} & W_{j_2} & W_{j_3} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} \\
=& - \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix}^T r_j^{-1} \begin{bmatrix} 0 \\ P_4 + P_5\Gamma_1 \\ P_5^T + P_6\Gamma_1 \end{bmatrix} \begin{bmatrix} W_{j_1} & W_{j_2} & W_{j_3} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ \Gamma_1 x + O(x^2) \end{bmatrix} \\
=& -r_j^{-1} x^T (P_4 + P_5\Gamma_1 + \Gamma_1^T P_5^T + \Gamma_1^T P_6 \Gamma_1)(W_{j_1} + W_{j_2} x + W_{j_1}\Gamma_1 x) \\
=& -r_j^{-1} x^T (P_4 + P_5\Gamma_1 + \Gamma_1^T P_5^T + \Gamma_1^T P_6 \Gamma_1) W_{j_1} \\
&- r_j^{-1} x^T (P_4 + P_5\Gamma_1 + \Gamma_1^T P_5^T + \Gamma_1^T P_6 \Gamma_1)(W_{j_2} x + W_{j_1}\Gamma_1 x)
\end{aligned}
$$

$$= -r_j^{-1} x^T Z_1 W_{j_1} + O(x^2)$$

where the linear term will dominate as $x \in \bar{D}$. Hence, the control rule will take the form of (2.18).

Accordingly, among the solutions of (2.26) to (2.27), only $Z_1(t)$ takes part in the control. Hence, to guarantee the stability of the closed-loop system, $Z_1(t)$ should be stabilizing as $t \to \infty$. In fact, by looking at (2.26), it is not difficult to verify it as a differential Riccati equation for the linearized system (2.15). Therefore we can use Lemma 2 to conclude that there is some $T_2 > 0$ such that the integration of (2.26) will lead to an stabilizing controller for all $t > T_2$, as long as $Q - \gamma Z_1 + O(x) > 0$, where this can be assured by appropriate choices of r and γ.

Furthermore, although $Z_2(t)$ and $Z_3(t)$ do not appear in the control, we require them to remain bounded. (2.27) and (2.28) can be rewritten as

$$\dot{Z}_2 = (A_{cl} - \gamma I)Z_2 + G_1(Z_1) + O(x),$$
$$\dot{Z}_3 = (-\gamma I)Z_3 + G_2(Z_2) + O(x),$$

where $A_{cl} = W_2 + W_3\Gamma_1 - r_j^{-1} Z_1 W_{j_1} W_{j_1}^T$, and functions $G_1(Z_1)$ and $G_2(Z_2)$ cam be seen as inputs to these differential equations. In fact, A_{cl} is the closed-loop system matrix of (2.15), hence, is Hurwitz for $t > T_2$. Accordingly, in addition, considering $\gamma > 0$, both autonomous dynamics are asymptotically stable with Hurwitz system matrices, $A_{cl} - \gamma I$, and $-\gamma I$. This guarantees that $Z_2(t)$ will stay bounded for the bounded solution $Z_1(t)$. In a similar way, bounded $Z_3(t)$ can be concluded by the bounded $Z_2(t)$.

□

Once we assured the asymptotic stability in some region around the equilibrium point, we can use the following results to investigate the optimality.

Theorem 2. *Assume that the conditions of Theorem 1 holds and a local stabilizing controller is obtained, with a sufficiently small choice of the discounting factor, $\gamma \to 0$. Then, the feedback control rule converges to the LQR control of the linearized system given by (2.16). Hence, the local optimality of the obtained controller is guaranteed.*

Proof. Assuming that the closed loop system is asymptotically stable, we have $x \to 0$. Furthermore, if γ can be made small enough, then the steady state solution of (2.26) will converge to the steady state solution of (2.19), and hence to the solution of the algebraic Riccati equation (2.16). □

In the next section, we will establish an online learning algorithm based on the proposed optimal control framework.

2.4 A Structured Online Learning (SOL) Algorithm

By considering a general description of the nonlinear input affine system in terms of some bases as in (2.4), we obtained a structured optimal control framework that suggests using the state-dependent matrix differential equation (2.13) to achieve the parameters of the nonlinear feedback control. Next, we exploit this framework to propose the SOL algorithm. Hence the focus of this section will be on the algorithm and practical properties of SOL.

The learning procedure is done in the following order. First, initialize P with a zero matrix. Then, in the control loop:

- We acquire the samples of the states at any time step t_k and evaluate the set of bases accordingly.
- We update the structured system model by a system identification technique.
- Using the measurements and the updated model coefficients, we integrate (2.13) to update P.
- We calculate the control value using (2.11) for the next step t_{k+1} using P.
- $k++$.

In what follows, we discuss the steps involved in more details with focusing on SINDy algorithm.

2.4.1 ODE Solver and Control Update

In this approach, we run the system from some $x_0 \in D$, then solve the matrix differential equation (2.13) along the trajectories of the system. Different solvers are already developed that can efficiently integrate differential equations. In the simulation, we use a Runge–Kutta solver to integrate the dynamics of the system that replaces the real system in a real-world application. Although the solver may take smaller steps, we only allow the measurements and control update at time steps $t_k = kh$, where h is the sampling time and $k = 0, 1, 2, \dots$. For solving (2.13) in continuous time, we use the Runge–Kutta solver with a similar setting, where the weights and the states in this equation are updated by a system identification algorithm and the measurements x_k at each iteration of the control loop, respectively. A recommended choice for P_0 is a matrix with components of zero or very small values.

The differential equation (2.13) also requires evaluations of $\partial\Phi/\partial x_k$ at any time step. Since the bases Φ are chosen beforehand, the partial derivatives can be analytically calculated and stored as functions. Hence, they can be evaluated for any x_k in a similar way as Φ itself. By solving (2.13), we can calculate the control update at any time step t_k according to (2.11). Although, at the very first steps of learning, control is not expected to take effective steps toward the control objective, it can help in exploration of the state space and gradually improve by learning more about the dynamics.

Remark 4. *The computational complexity of updating parameters by relation (2.13) is bounded by the complexity of matrix multiplications of dimension p which is $\mathcal{O}(p^3)$. Moreover, it should be noted that, regarding the symmetry in the matrix of parameters P, this equation updates $L = (p^2 + p)/2$ number of parameters which correspond to the number of bases used in the value function. Therefore, in terms of the number of parameters, the complexity of the proposed technique is $\mathcal{O}(L^{3/2})$. However, for instance, if recursive least squares technique were employed with the same number of parameters, the computations are bounded by $O(L^3)$. As a result, the proposed parameter update scheme can be done considerably faster than similar model-based techniques, such as, [82, 17]. In another effort, [81] decreased the number of bases used to improve the computational efficiency, while the complexity still remained as $O(L^3)$.*

2.4.2 Identified Model Update

We considered a given structured nonlinear system as in Assumption 1. Therefore, having the control and state samples of the system, we need an algorithm that updates the estimation of system weights. As studied in [25, 78], SINDy is a data-efficient tool to extract the underlying sparse dynamics of the sampled data. Hence, we use SINDy to update the weights of the system to be learned. In this approach, along with the identification, the sparsity is also promoted in the weights by minimizing

$$[\hat{W} \quad \hat{W}_1 \ldots \hat{W}_m]_k = \arg\min_{\bar{W}} \quad \|\dot{X}_k - \bar{W}\Theta_k\|_2^2 + \lambda\|\bar{W}\|_1, \tag{2.29}$$

where k is the time step, $\lambda > 0$, and Θ_k includes a matrix of samples with the columns of

$$\Theta_k{}^s = [\Phi^T(x^s) \quad \Phi^T(x^s)u_1{}^s \quad \ldots \quad \Phi^T(x^s)u_m{}^s]_k^T,$$

for sth sample. In the same order, $\dot{X}_k$ keeps a table of sampled state derivatives.

Updating $\hat{W}_k$ based on a history of samples may not be favored as the number of samples needed tends to be large. Especially, real-time implementations may not be possible because of the latency caused by the computations. There exist other techniques that can be alternatively used in different situations, such as neural networks, nonlinear regression, or any other function approximation and system identification methods. For real-time control applications, considering the linear dependence on the system weights in (2.4), one may choose the RLS update rule that only uses the latest sample of the system and $\hat{W}_{k-1}$, hence will run considerably faster.

2.4.3 Database Update

For using SINDy algorithm, a database of samples is required to recursively perform regressions at each time step. These weights correspond to a library of functions given in Φ. Any sample of the system at time k, to be stored in the database, includes ${\Theta_k}^s$ and the derivatives of the states approximated by $\hat{\dot{x}}_k = (x_k - x_{k-1})/h$.

For better results, higher order approximations of the state derivative can be employed that may also include future samples of the states, for instance, x_{k+1}, x_{k+2}, etc. To make this possible in the implementations, the identification should lag a few step behind the controller update. The effect of lagging for a few steps is minor and can be safely neglected in the long run.

We adopt SINDy to do an online learning task, meaning that the database has to be gradually built along with the exploration and control. Different approaches can be employed in the choice of samples and building a database online. A comparison of these techniques can be found in [87, 161].

In the implementations of SOL done in this chapter, we assume a given maximum size of database N_d, then we keep adding the samples with larger prediction errors to the database. Therefore, at any step we compare the prediction error $\dot{e}_k = \|\dot{x}_k - \hat{\dot{x}}_k\|$ with the average $\bar{\dot{e}}_k = \sum_{i=1}^{k} \dot{e}_k/k$. Hence, if the condition $\dot{e}_k > \eta\bar{\dot{e}}_k$ holds we add the sample to the database, where the constant $\eta > 0$ adjusts the threshold. Choosing smaller values of η will increase the rate of adding samples to the database.

This procedure is done in a loop together with updating the control until a bound of the average prediction error is obtained that allows the controller to regulate the system to the given reference state. If the maximum number of samples in database is reached, we forget the oldest sample and replace it with the recent one. Hence, η should not be set too low to avoid fast forgetting of the older useful samples.

2.4.4 Limitations and Implementation Considerations

In this section, we discuss some considerations should be taken into account prior to running the algorithm:

- It should be noted that although the learning approach is validated only in the simulation environment here, it is proposed to be implemented on real world problems. Hence, the training is meant to take place in real-time on real systems that requires the computational and the data efficiency of the algorithm.

- It is assumed that the environment can be made safe in a region of interest, and there exists a reseting mechanism if reached to the boundary of the region. This allows trials and errors with no considerable damage to the system within a limited number of episodes until the stability can be preserved.

- Considering the control problem formulation, the control and state spaces, and the time horizon cannot be hardly constrained. However, they can be tuned by using the parameters specified by the user in the objective (2.2) including R, Q, and γ, respectively.

- Depending on the system identification technique implemented, there usually exist some tuning parameters. Having an initial knowledge of the system can help greatly in setting these parameters, as well as in choosing the set of bases.

2.4.5 Asymptotic Convergence with Approximate Dynamics

Consider the system structured as

$$\dot{x} = W\Phi + \sum_{j=1}^{m} W_j \Phi \hat{u}_j + \epsilon. \tag{2.30}$$

where $\hat{u}_j = -\Phi^T R^{-1} \hat{P} \Phi_x \hat{W}_j \Phi$ is the feedback control rule obtained based on the estimation of the system $(\hat{W}, \hat{W}_j)$. Moreover ϵ is the bounded approximation error in D. By assuming $W = \hat{W} + \tilde{W}$ and $W_j = \hat{W}_j + \tilde{W}_j$, this can be rewritten as

$$\dot{x} = \hat{W}\Phi + \sum_{j=1}^{m} \hat{W}_j \Phi \hat{u}_j + \Delta(t), \tag{2.31}$$

where unidentified dynamics are lumped together as $\Delta(t)$. By the assumption that the feedback control u_j is bounded in D, we have $\|\Delta(t)\| \leq \bar{\Delta}$. For asymptotic convergence, and also promote the robustness of the controller, the effect of the uncertainty should be taken into account. Hence, we use an auxiliary vector ρ to get

$$\begin{aligned}\dot{x} &= \hat{W}\Phi + \sum_{j=1}^{m} \hat{W}_j \Phi \hat{u}_j + \Delta(t) + \rho - \rho \\ &= \hat{W}_\rho \Phi + \sum_{j=1}^{m} \hat{W}_j \Phi \hat{u}_j + \Delta(t) - \rho,\end{aligned}$$

where assuming that Φ also includes the constant basis, we adjusted the corresponding column in the system matrix to get $\hat{W}_\rho$. In the case $\bar{\Delta} = 0$, by using Theorem 1, the controller $\hat{u}$ can be obtained such that the closed system is locally asymptotically stable. For the case $\bar{\Delta} > 0$, although the system will stay stable for small enough $\bar{\Delta}$, it may not asymptotically converge to zero. Then, similar to [168, 136], we obtain ρ as below to help sliding the system state to zero

$$\rho = \int_0^t [k_1 x(\tau) + k_2 \text{sign}\big(x(\tau)\big)] \mathrm{d}\tau,$$

where k_1 and k_2 are positive scalars. It can be shown that over time $\|\Delta(t) - \rho\| \to 0$, and hence the system will asymptotically converge to the origin.

2.5 Simulation Results

We have implemented the proposed approach on four examples which are presented in two categories considering Assumption 1, 1) the dynamics can be written exactly in terms of some choice of basis functions, and 2) the dynamics include some terms that are required to be approximated in the space of some given bases.

As mentioned, in these numerical examples we have exploited the SINDy algorithm for the identification purpose, however, clearly the focus of the simulations here is on the properties of the proposed control scheme rather than the identification part, regarding that SINDy already has been extensively studied in [25, 78] as an offline identification algorithm. The SINDy algorithm adopted here is a powerful tool to obtain the dynamics of the system with a good precision. However, this depends greatly on how efficiently we can approximate the derivatives of the states. Hence, in different implementations, higher sampling rates or higher order approximation of the derivatives may be needed. For the same reason, in the proposed examples, the number of samples used and the system obtained may be further tuned to match the level of quality reported in [25, 78].

The simulations are done in Python, where we used the Vpython module ([146]) to generate the graphics. We have set the sampling rate to 200Hz ($h = 5$ms) for all the examples, unless explicitly mentioned otherwise. The control input value is updated at every other time step meaning that the update rate is 100Hz. The simulation is stopped if the trajectory reaches to the boundary of D or a timeout is reached without satisfying the objective. Moreover, if the regulation objective is to reach a point other than the origin ($x \equiv 0$), we consider the cost (2.2), the value (2.8), and the obtained differential equation of the value parameters (2.13) by redefining $x := x - x_{\text{ref}}$.

2.5.1 Systems Identifiable in Terms of a Given Set of Bases

In the following two examples, we assume that the bases constituting the system dynamics exist in Φ. The system identified, after running the proposed learning algorithm and obtaining the value function, clearly depends on the identification algorithm used and its tuning parameters.

In Table 2.1, we illustrate the variations of the identified system and the corresponding value function by implementing the presented SOL algorithm with the exact $\dot{x}$ and with the first order approximation of the derivative. It can be observed that, in the pendulum example, both of the obtained equations match the exact system (2.32) with a good precision. On the other hand, the Lorenz system is a more challenging system. Hence, by the first-order approximation of $\dot{x}$ with $h = 5$ms, only an approximation of the dynamics can be obtained, while the exact system (2.33) is identified if we use the exact $\dot{x}$. As shown in Fig. 2.1 and Fig. 2.2, although the model obtained for Lorenz system by using the approximate state variables does not closely match the exact dynamics, the obtained controller can successfully solve the regulation problem as long as the prediction errors remain bounded.

Example 1 (Pendulum)

The state space description of the system is given as

$$\begin{aligned} \dot{x_1} &= -x_2, \\ \dot{x_2} &= -\frac{g}{l}\sin(x_1) - \frac{k}{m}x_2 + \frac{1}{ml^2}u, \end{aligned} \tag{2.32}$$

where $m = 0.1kg$, $l = 0.5m$, $k = 0.1$, and $g = 9.8m/s^2$. The performance criteria are defined by the choices of $Q = \text{diag}([1, 1])$, $R = 2$.

Objective: The system is regulated to the unstable equilibrium point given by $x_{\text{ref}} \equiv 0$.

In table 2.1, the learned dynamics and value function are listed for the exact and the approximated $\dot{x}$.

Example 2 (Chaotic Lorenz System)

The system dynamics are defined by

$$\begin{aligned} \dot{x_1} &= \sigma(x_2 - x_1) + u, \\ \dot{x_2} &= -x_2 + x_1(\rho - x_3), \\ \dot{x_3} &= x_1 x_2 - \beta x_3, \end{aligned} \tag{2.33}$$

where $\sigma = 10$, $\rho = 28$, and $\beta = 8/3$. Furthermore, we set the performance criteria to $Q = \text{diag}([160, 160, 12])$, $R = 1$. This system has two unstable equilibrium points $(\pm\sqrt{72}, \pm\sqrt{72}, 27)$, where the trajectories of the system oscillate around these points.

Objective: By randomly setting the initial state $x_0 \in \{x| -40 \leq x_i \leq 40, i = 1, 2, 3\}$, we regulate the system to the unstable equilibrium $(-\sqrt{72}, -\sqrt{72}, 27)$.

2.5.2 Systems to Be Approximated by a Given Set of Basis

In what follows, we apply the presented learning scheme on two benchmark examples. Unlike the previous examples, the dynamics of these systems includes some rational terms that cannot be written in terms of some basis functions, however, an approximation can be obtained locally that is shown to be sufficient to successfully solve the regulation problem, as shown in Fig. 2.4-2.7.

Moreover, as shown in Fig. 2.3, a video of the graphical simulation of the following benchmark examples is included.

Example 3 (Cartpole Swing up)

The dynamics are given as

$$\dot{x_1} = x_2,$$

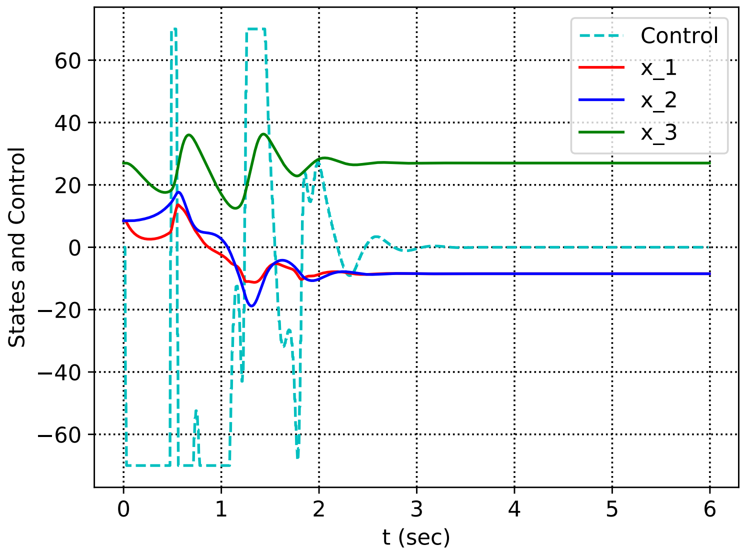


Figure 2.1: Responses of the Lorenz system while learning by using the approximated state derivatives as in Table 2.1, where starting from one equilibrium point, we regulated the system to another unstable equilibrium.

$$\dot{x_2} = \frac{-u\cos(x_1) - mLx_2^2\sin(x_1)\cos(x_1) + (M+m)g\sin(x_1)}{L(M+m\sin(x_1)^2)},$$

$$\dot{x_3} = x_4,$$

$$\dot{x_4} = \frac{u + m\sin(x_1)(Lx_2^2 - g\cos(x_1))}{M + m\sin(x_1)^2}, \tag{2.34}$$

where the state vector is composed of the angle of the pendulum from upright position, the angular velocity, and the position and velocity of the cart, with $m = 0.1kg$, $M = 1kg$, $L = 0.8m$, and $g = 9.8m/s^2$. Moreover, we choose $Q = \text{diag}([60, 1.5, 180, 45])$, $R = 1$.

Objective: By starting from some initial angles close to the stable angle of the pendulum ($\pm\pi$), the cart swings up the pendulum to reach to and stay at the unstable state given as $x_{\text{ref}} \equiv 0$.

By running the learning scheme, an approximation of the system is identified as

$$\dot{x}_1 = 1.000x_2,$$

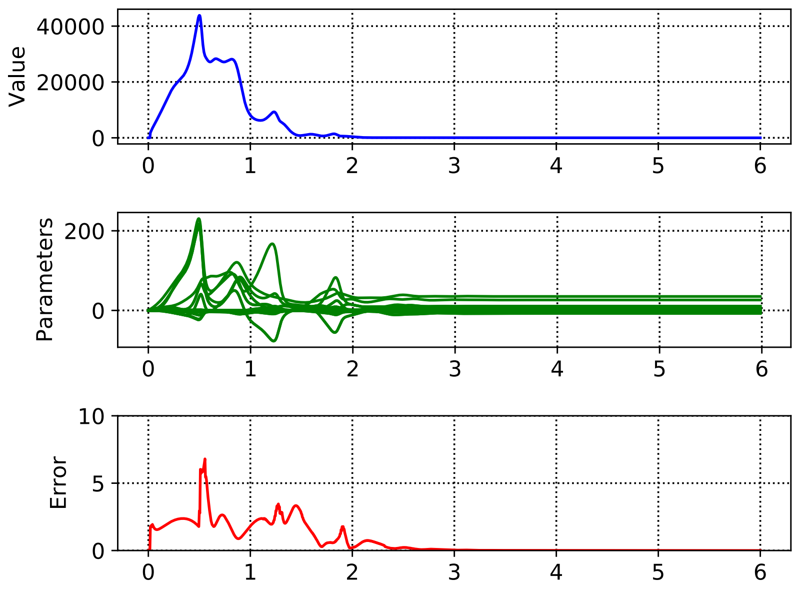


Figure 2.2: The value, components of P, and prediction error corresponding to Fig. 2.1, respectively.

$$\begin{aligned}
\dot{x}_2 &= 12.934\sin(x_1) + 0.230\sin(x_3) - 1.234\cos(x_1)u, \\
\dot{x}_3 &= 0.995x_4, \\
\dot{x}_4 &= 0.926\sin(x_1) + 0.953u,
\end{aligned} \tag{2.35}$$

where $\Phi = \{1, x, x^2, x^3, \sin x, \cos x\}$. Moreover, considering the assumed bases, we obtained the optimal value function as below.

$$\begin{aligned}
V(x) &= 59.712x_1^2 + 9.855x_2x_1 + 134.855x_2^2 + 9.587x_3x_1 \\
&+ 241.295x_3x_2 + 223.389x_3^2 + 4.418x_4x_1 + 222.022x_4x_2 \\
&+ 226.646x_4x_3 + 100.417x_4^2 - 63.050\sin(x_1)x_1 \\
&+ 1098.765\sin(x_1)x_2 + 2294.259\sin^2(x_1) \\
&+ 984.786\sin(x_1)x_3 + 909.030\sin(x_1)x_4 - 1.712\sin(x_3)x_1 \\
&+ 18.102\sin(x_3)x_2 + 15.812\sin(x_3)x_3 + 0.806\sin^2(x_3) \\
&+ 75.231\sin(x_3)\sin(x_1) + 15.072\sin(x_3)x_4.
\end{aligned} \tag{2.36}$$

Figure 2.3: A view of the graphical simulations of the benchmark cartpole and double inverted pendulum examples. The video can be accessed in: https://youtu.be/-j0vaHE9MZY .

Example 4 (Double Inverted Pendulum on a Cart)

By defining $y := [q \quad \theta_1 \quad \theta_2]^T$ to be a vector of the cart position and angles of the double pendulum from the top equilibrium point, the system dynamics can be written in the following form.

$$\dot{x} = \begin{bmatrix} \dot{y} \\ M^{-1} f(y, \dot{y}) \end{bmatrix}, \tag{2.37}$$

where

$$M = \begin{bmatrix} m + m_1 + m_2 & l_1(m_1 + m_2)\cos(\theta_1) & m_2 l_2 \cos(\theta_2) \\ l_1(m_1 + m_2)\cos(\theta_1) & l_1^2(m_1 + m_2) & l_1 l_2 m_2 \cos(\theta_1 - \theta_2) \\ l_2 m_2 \cos(\theta_2) & l_1 l_2 m_2 \cos(\theta_1 - \theta_2) & l_2^2 m_2 \end{bmatrix},$$

$$f(y, \dot{y}) = \begin{bmatrix} l_1(m_1 + m_2)\dot{\theta}_1^2 \sin(\theta_1) + m_2 l_2 \dot{\theta}_2^2 \sin(\theta_2) - d_1 \dot{q} + u \\ -l_1 l_2 m_2 \dot{\theta}_2^2 \sin(\theta_1 - \theta_2) + g(m_1 + m_2) l_1 \sin(\theta_1) - d_2 \theta_1 \\ l_1 l_2 m_2 \dot{\theta}_1^2 \sin(\theta_1 - \theta_2) + g l_2 m_2 \sin(\theta_2) - d_3 \theta_2 \end{bmatrix},$$

$m = 6kg$, $m_1 = 3kg$, $m_2 = 1kg$, $l_1 = 1m$, $l_2 = 2m$,$d_1 = 10$, $d_2 = 1$, and $d_3 = 0.5$.

Objective: We run the system from random angles around the top unstable equilibria of the pendulums given by $\theta_1 = 0$ and $\theta_2 = 0$, where the controller has to learn to regulate the system to $x_{\text{ref}} \equiv 0$.

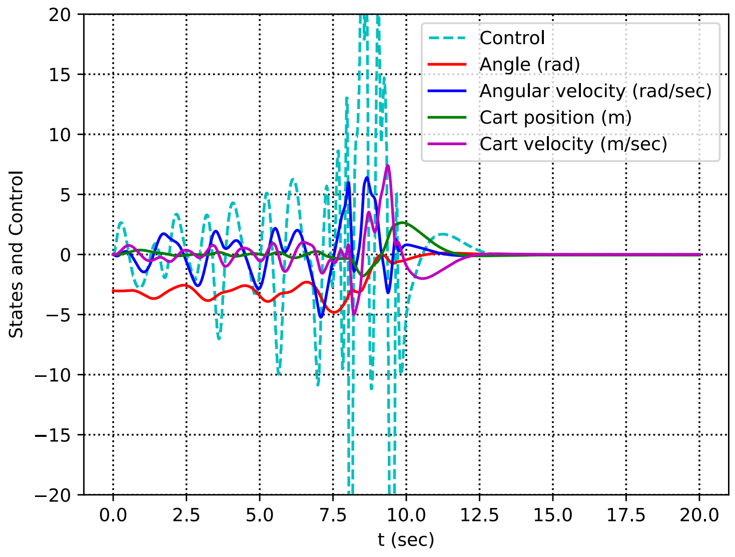


Figure 2.4: Responses of the cartpole system while learning by using the approximated state derivatives.

We choose the bases as $\Phi = \{1, x, x^2\}$. Moreover the performance criteria is given by $Q = \text{diag}([15, 15, 15, 1, 1, 1])$, $R = 1$. A sample of the obtained approximate dynamics is

$$\begin{aligned}
\dot{x}_1 &= 0.998x_4, \quad \dot{x}_2 = 0.997x_5, \quad \dot{x}_3 = 0.996x_6,\\
\dot{x}_4 &= 0.238x_1 - 4.569x_2 1.245x_3 - 1.891x_4 - 0.908x_6\\
&\quad - 0.105x_2^2 + 5.0131u - 2.824x_2^2u,\\
\dot{x}_5 &= 16.718x_2 - 2.328x_3 + 1.558x_4 - 0.598x_5 + 0.130x_6\\
&\quad - 0.114x_2^2 - 4.9911u5.777x_2^2u - 0.690x_3^2u,\\
\dot{x}_6 &= 0.123x_1 - 6.721x_2 + 9.032x_3 + 0.191x_5 - 0.358x_6\\
&\quad + 0.969x_3^2 + 0.184x_6^2 - 1.898x_2^2u + 1.431x_3^2u. \qquad (2.38)
\end{aligned}$$

It should be noted that, because of the random initial conditions and different samples in the database, a different approximation of the system may be obtained in any learning procedure. Furthermore, considering the dimension of the system and the number of terms in the identified system (2.38), the obtained value function includes many terms of polynomials as expected. Therefore, for the sake of brevity, the obtained optimal value

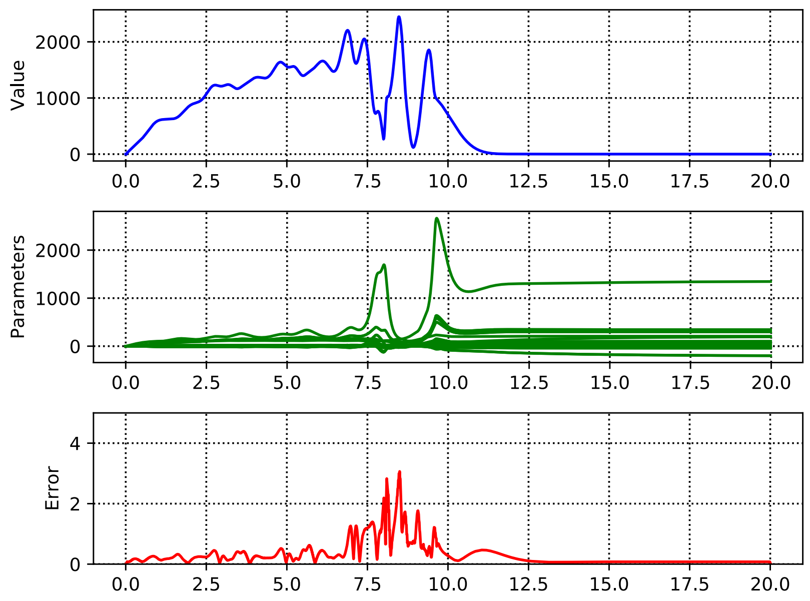


Figure 2.5: The value, components of P, and prediction error corresponding to Fig. 2.4, respectively.

function is omitted.

2.6 Conclusion

Considering the online model-based regulation problem, the structured dynamics helped us in analytically computing an iterative update rule to improve the optimal value function according to the latest update on the identified system. Based on the computational complexity and the performance observed in the numerical and graphical simulations, we showed some potential opportunities in employing the SOL algorithm as an online model-based learning technique. Our future research will follow on the stability analysis and further applications of this approach.

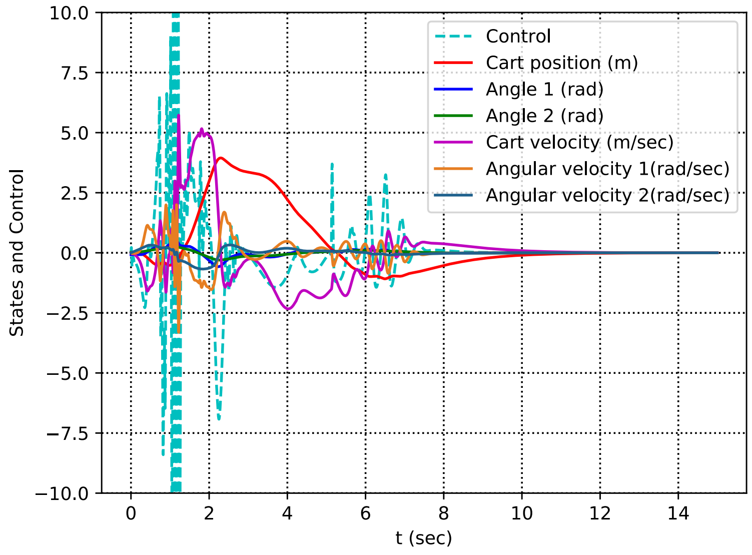


Figure 2.6: Responses of the double-inverted pendulum system while learning by using the approximated state derivatives.

Table 2.1: The system dynamics and the corresponding value function obtained by the proposed method, where the exact and the approximated derivatives of the state variables are used in different scenarios

Exact $\dot{x}$	$\dot{x} \approx (x_{k+1} - x_k)/h, h = 5ms$
Pendulum ($\Phi = \{1, x, \sin x\}$)	
$\dot{x_1} = -1.000x_2$ $\dot{x_2} = -1.000x_2 - 19.600\sin(x_1) + 40.000u$	$\dot{x_1} = -1.011x_2$ $\dot{x_2} = -0.995x_2 - 19.665 sin(x_1) + 40.098u$
$V(x) = 1.974x_1^2 - 0.058x_2x_1 + 0.036x_2^2 - 2.2\sin(x_1)x_1 - 0.077\sin(x_1)x_2 + 1.548\sin^2(x_1)$	$V(x) = 2.049x_1^2 - 0.058x_2x_1 + 0.036x_2^2 - 2.371\sin(x_1)x_1 - 0.077\sin(x_1)x_2 + 1.630\sin^2(x_1)$
Chaotic Lorenz System ($\Phi = \{1, x, x^2, x^3, x_ix_j\}, \quad i, j \in \{1, \ldots, n\}, i \neq j$)	
$\dot{x_1} = -10.000x_1 + 10.000x_2 + 1.000u$ $\dot{x_2} = 28.000x_1 - 1.000x_2 - 1.000x_1x_3$ $\dot{x_3} = -2.667x_3 + 1.000x_1x_2$	$\dot{x_1} = -10.070x_1 + 9.973x_2 + 0.989u$ $\dot{x_2} = 0.993x_1 - 0.997x_2 + 8.483x_3 - 1.000x_1x_3$ $\dot{x_3} = -8.483x_1 - 8.483x_2 - 2.666x_3 + 1.000x_1x_2$
$V(x) = 30.377x_1^2 + 48.939x_2x_1 + 25.311x_2^2 + 1.500x_3^2 - 1.873x_1x_2x_3 + 4.719x_1^2x_2^2 - 3.291x_1^2x_3 + 1.469x_1x_3^2 - 0.012x_1^2x_2x_3$	$V(x) = 11.193x_1^2 + 8.389x_2x_1 + 42.855x_2^2 - 20.950x_3x_1 + 28.441x_3x_2 + 32.045x_3^2 - 1.899x_1^2x_2 - 4.777x_1x_2^2 - 0.456x_1x_2x_3 + 5.064x_1^2x_2^2 + 2.953x_1^2x_3 - 8.168x_1x_3^2 - 2.633x_1^2x_3x_2 + 1.353x_1^2x_3^2$

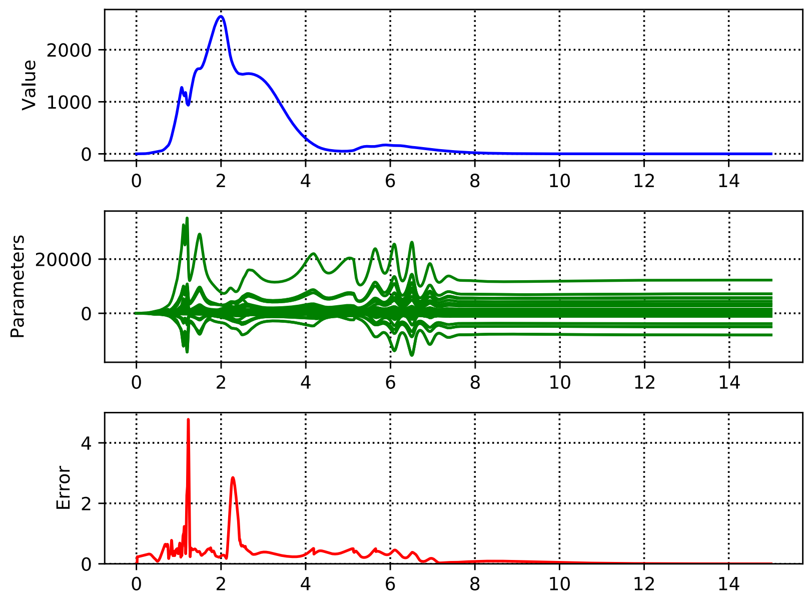


Figure 2.7: The value, components of P, and prediction error corresponding to Fig. 2.6, respectively.

Chapter 3

A Structured Online Learning Approach to Nonlinear Tracking with Unknown Dynamics

The results presented in this chapter are published in [53].

3.1 Introduction

Tracking a desired reference trajectory is one of the most classical objectives in the control of dynamical systems, and is commonly encountered in many real-world applications. However, the design of an effective tracking controller via conventional approaches often requires sufficient knowledge of the model, and a lot of calculations and considerations are involved for any particular application. On the other hand, RL techniques suggest a more adaptable framework that requires less knowledge about the system dynamics.

In this chapter, we extend the results obtained in the previous chapter to the tracking problem of unknown continuous dynamical systems. In Section 3.2, we propose an approximate optimal tracking control framework based on a particular structure of nonlinear dynamics, where a linear quadratic discounted cost is assumed. Section 3.3 provides the details of implementation of the obtained framework as a learning-based approach. In Section 3.4, two numerical results illustrating the proposed approach are reported on two benchmark examples.

3.2 A Structured Online Learning for Tracking Control

Consider the nonlinear affine system

$$\dot{x} = f(x) + g(x)u, \tag{3.1}$$

where $x \in D \subset \mathbb{R}^n$, $u \in \Omega \subset \mathbb{R}^m$, $f : D \to \mathbb{R}^n$, and $g : D \to \mathbb{R}^{n\times m}$.

Assumption 2. *f and g can be identified or effectively approximated within the compact domain of interest by a linear combination of some bases functions $\phi_i \in C^1 : D \to \mathbb{R}$ for $i = 1, 2, \ldots, p$.*

Accordingly, (3.1) is rewritten as

$$\dot{x} = W\Phi(x) + \sum_{j=1}^{m} W_j \Phi(x) u_j, \tag{3.2}$$

where W and $W_j \in \mathbb{R}^{n\times p}$ are the matrices of the coefficients obtained for $j = 1, 2, \ldots, m$, and

$$\Phi(x) = [x^T \; \phi_{n+1}(x) \; \ldots \; \phi_p(x)]^T.$$

Remark 5. *The structure employed in (3.2) is motivated by the fact that the linear combination of the bases provides an opportunity of obtaining an analytical control approach. At the same time, we can profit from the variety of the applicable identification techniques.*

Remark 6. *Regarding Assumption 2, in this chapter, we will not perform convergence analysis of any particular identification approach. Instead, we will focus on the controller design procedure, in a way that the problem formulation and the design allow exploiting different identification methods alternatively. Hence, this section presents the control technique for given W and W_j. However, in the implementations, an estimation of these weights will be used, which is discussed in the next section.*

Assumption 3. *The given reference trajectory $y_{ref}(t) : \mathbb{R} \to \mathbb{R}^d$ is a particular solution of a dynamical system of the form*

$$\dot{y}_{ref} = M\Psi(y_{ref}), \tag{3.3}$$

where

$$\Psi(y_{ref}) = [y_{ref}^T \, \psi_{d+1}(y_{ref}) \; \ldots \; \psi_q(y_{ref})]^T$$

is a set of bases and $M \in \mathbb{R}^{d\times q}$ is the matrix of coefficients. This system can be seen as the virtual command generator, which with the variety of the chosen bases can supports a wide range of signals, from simple ramp or sinusoidal signals to more complex ones.

In the optimal tracking problem, a cost functional is assumed to measure the performance. Starting from the initial condition $x_0 = x(0)$, the following discounted linear quadratic cost is minimized along the trajectory:

$$\begin{aligned} &J(x_0, u) = \\ &\lim_{T\to\infty} \int_0^T \mathrm{e}^{-\gamma t} \left((Cx - y_{\mathrm{ref}})^T Q (Cx - y_{\mathrm{ref}}) + u^T R u\right) dt, \end{aligned} \tag{3.4}$$

where $Q \in \mathbb{R}^{d\times d}$ is positive semi-definite, $\gamma \geq 0$ is the discount factor, and $R \in \mathbb{R}^{m\times m}$ is a diagonal matrix with only positive values, given by design criteria. Moreover, corresponding to the dimension of y_{ref}, a subset of the states is chosen by using $C \in \mathbb{R}^{d\times n}$, that includes entries one corresponding to the measured states and zero elsewhere.

For the closed-loop system, by assuming a feedback control law

$$u = \omega(x(t), y_{\mathrm{ref}}(t))$$

for $t \in [0, \infty)$, the optimal control is given by

$$\omega^* = \underset{u(\cdot)\in\Gamma(x_0)}{\arg\min}\, J(x_0, u(\cdot)), \tag{3.5}$$

where $\Gamma(x_0)$ is the set of admissible control signals.

Lemma 3. *The optimal tracking control obtained by minimizing*

$$J(x_0, u) = \lim_{T\to\infty} \int_0^T \mathrm{e}^{-\gamma t} \left(\bar{\Phi}^T \bar{Q} \bar{\Phi} + u^T R u\right) dt, \tag{3.6}$$

is equivalent to the solution of (3.5) assuming (3.4), where

$$\begin{aligned} \bar{\Phi}(x, y_{ref}) = \big[(Cx)^T \; {y_{ref}}^T \; \phi_{(d+1)}(x) \; \ldots \; \phi_{(p)}(x) \\ \psi_{(d+1)}(y_{ref}) \; \ldots \; \psi_{(q)}(y_{ref})\big]^T, \end{aligned} \tag{3.7}$$

and

$$\bar{Q} = diag\left(\begin{bmatrix} Q & -Q \\ -Q & Q \end{bmatrix}, [\mathbf{0}_{(p+q-2d)\times(p+q-2d)}]\right) \tag{3.8}$$

is a block diagonal matrix that contains all zeros except the first block which correspond to the linear bases Cx and y_{ref}.

Proof. It is straightforward to rewrite the performance measure (3.4) in terms of the vector of bases $[(Cx)^T \; {y_{\text{ref}}}^T]$ with a positive semi-definite matrix defined as the none-zero block in (3.8). Later, the obtained cost is again transformed to the space of bases $\bar{\Phi}$ to take the form (3.6), where we assume (3.8). □

Remark 7. *In some particular application, while tracking a given trajectory, we might need to penalize at the same time the growth in some other states which are not in the list of the tracked states y_{ref}. Such conditions can be still handled by the block-diagonal matrix (3.8), that is to assign a non-zero value, corresponding to that particular state, in the diagonal of the second block of (3.8).*

Now, consider the system dynamics (3.2) and the command generator (3.3). We define the augmented system as

$$\dot{z} = \begin{bmatrix} \dot{x} \\ \dot{y}_{\text{ref}} \end{bmatrix} = F\bar{\Phi} + \sum_{j=1}^{m} G_j \bar{\Phi} u_j, \tag{3.9}$$

where the system matrices F and G_j are obtained by rearranging the block entries of the coefficient matrices of (3.2) and (3.3) according to the ordering of entries in $\bar{\Phi}$.

By defining the Hamiltonian, the corresponding HJB equation of (3.6) is given as

$$\begin{aligned} -\frac{\partial}{\partial t}(\mathrm{e}^{-\gamma t} V) = & \min_{u(\cdot)\in\Gamma(x_0)} \{ H = \mathrm{e}^{-\gamma t}\big(\bar{\Phi}^T \bar{Q} \bar{\Phi} + u^T R u\big) \\ & + \mathrm{e}^{-\gamma t} \frac{\partial V}{\partial z}^T (F\bar{\Phi} + \sum_{j=1}^{m} G_j \bar{\Phi} u_j)\}. \end{aligned} \tag{3.10}$$

Then, we employ an approximation scheme to estimate a parameterized value function $V : D \times [0, \infty) \to \mathbb{R}$ satisfying the above partial differential equation.

Unlike other approximate optimal control approaches in the literature, such as [176, 17, 81, 179], we use a quadratic form to parameterize the value function as follows:

$$V = \bar{\Phi}^T P \bar{\Phi}, \tag{3.11}$$

where P is a symmetric matrix.

As suggested in [52], defining V in the product space $\Lambda := \bar{\Phi} \times \bar{\Phi}$, provides a compact form of parameterizing the value function, and a variety of bases can be produced in the product space Λ by only including a limited number of bases in $\bar{\Phi}$. It should be noted that updating the parameters in the matrix form in SOL will considerably decrease the computations required to updated the parameters, where the matrix multiplications involved will remain as cheap as the dimension of $\bar{\Phi}$. On the other hand, in alternative parameter update methods, such as gradient descent, matrix multiplications of the dimension of the numbers of elements in set Λ is involved, which is a considerably larger set compared to $\bar{\Phi}$.

Assuming (3.11), the Hamiltonian is written as

$$\begin{aligned} H =& \mathrm{e}^{-\gamma t}(\bar{\Phi}^T \bar{Q} \bar{\Phi} + u^T R u) \\ &+ \mathrm{e}^{-\gamma t} \left(\frac{\partial \bar{\Phi}}{\partial z}^T P \bar{\Phi} \right)^T \left(F\bar{\Phi} + \sum_{j=1}^{m} G_j \bar{\Phi} u_j \right) \\ &+ \mathrm{e}^{-\gamma t} \left(F\bar{\Phi} + \sum_{j=1}^{m} G_j \bar{\Phi} u_j \right)^T \left(\frac{\partial \bar{\Phi}}{\partial z}^T P \bar{\Phi} \right) \end{aligned}$$

In the following, we rewrite the quadratic term of u based on its components, assuming that $r_j \neq 0$ is the jth component on the diagonal of R.

$$\begin{aligned} H = \mathrm{e}^{-\gamma t} \Bigg(& \bar{\Phi}^T \bar{Q} \bar{\Phi} + \sum_{j=1}^{m} r_j u_j^2 + \bar{\Phi}^T P \frac{\partial \bar{\Phi}}{\partial z} F \bar{\Phi} + \\ & \bar{\Phi}^T P \frac{\partial \bar{\Phi}}{\partial z} \left(\sum_{j=1}^{m} G_j \bar{\Phi} u_j \right) + \bar{\Phi}^T F^T \frac{\partial \bar{\Phi}}{\partial z}^T P \bar{\Phi} \\ & + \left(\sum_{j=1}^{m} u_j \bar{\Phi}^T G_j^T \right) \frac{\partial \bar{\Phi}}{\partial z}^T P \bar{\Phi} \Bigg). \end{aligned} \tag{3.12}$$

The obtained Hamiltonian is minimized if, for the jth system input, we have

$$\begin{aligned} \frac{\partial H}{\partial u_j} &= 2 r_j u_j + 2 \bar{\Phi}^T P \frac{\partial \bar{\Phi}}{\partial z} G_j \bar{\Phi} \\ &= 0, \qquad j = 1, 2, \ldots, m. \end{aligned} \tag{3.13}$$

Accordingly, the optimal control input is calculated as

$$u_j^* = -\bar{\Phi}^T r_j^{-1} P \frac{\partial \bar{\Phi}}{\partial z} G_j \bar{\Phi}. \tag{3.14}$$

Then, we substitute the obtained optimal feedback control law and the value function to (3.10), which yields

$$
\begin{aligned}
&-\mathrm{e}^{-\gamma t}\bar{\Phi}^T\dot{P}\bar{\Phi} + \gamma \mathrm{e}^{-\gamma t}\bar{\Phi}^T P\bar{\Phi} \\
&\quad = \mathrm{e}^{-\gamma t}\Bigg(\bar{\Phi}^T\bar{Q}\bar{\Phi} + +\bar{\Phi}^T P\frac{\partial\bar{\Phi}}{\partial z}F\bar{\Phi} + \bar{\Phi}^T F^T\frac{\partial\bar{\Phi}}{\partial z}^T P\bar{\Phi} \\
&\quad + \bar{\Phi}^T P\frac{\partial\bar{\Phi}}{\partial z}\left(\sum_{j=1}^{m} G_j\bar{\Phi}r_j^{-1}\bar{\Phi}^T G_j^T\right)\frac{\partial\bar{\Phi}}{\partial z}^T P\bar{\Phi} \\
&\quad - 2\bar{\Phi}^T P\frac{\partial\bar{\Phi}}{\partial z}\left(\sum_{j=1}^{m} G_j\bar{\Phi}r_j^{-1}\bar{\Phi}^T G_j^T\right)\frac{\partial\bar{\Phi}}{\partial z}^T P\bar{\Phi}\Bigg).
\end{aligned}
$$

By some manipulation we get

$$
\begin{aligned}
&\bar{\Phi}^T\dot{P}\bar{\Phi} + \gamma\bar{\Phi}^T P\bar{\Phi} = \bar{\Phi}^T\bar{Q}\bar{\Phi} + \\
&\quad - \bar{\Phi}^T P\frac{\partial\bar{\Phi}}{\partial z}\left(\sum_{j=1}^{m} G_j\bar{\Phi}r_j^{-1}\bar{\Phi}^T G_j^T\right)\frac{\partial\bar{\Phi}}{\partial z}^T P\bar{\Phi} \\
&\quad + \bar{\Phi}^T P\frac{\partial\bar{\Phi}}{\partial z}F\bar{\Phi} + \bar{\Phi}^T F^T\frac{\partial\bar{\Phi}}{\partial z}^T P\bar{\Phi}.
\end{aligned}
$$

Finally, a sufficient condition for satisfying this equation is obtained as

$$
\begin{aligned}
-\dot{P} =& \bar{Q} + P\frac{\partial\bar{\Phi}}{\partial z}F + F^T\frac{\partial\bar{\Phi}}{\partial z}^T P - \gamma P \\
&- P\frac{\partial\bar{\Phi}}{\partial z}\left(\sum_{j=1}^{m} G_j\bar{\Phi}r_j^{-1}\bar{\Phi}^T G_j^T\right)\frac{\partial\bar{\Phi}}{\partial z}^T P.
\end{aligned}
\tag{3.15}
$$

The standard way for solving such an equation is by integrating in the backward direction, where it requires full knowledge of the system including the weights F and G_j for all time horizon. As a result, a value of P is obtained, which realizes the optimal value function (3.11), and leads to the optimal control (3.14). Recalling Remark 6, one can employ the presented method to design a tracking control of a known system. However, in this chapter, we focus on the learning problem where the accurate system model may not be known at first place. Therefore, we propagate the obtained differential equation in forward direction. This will provide an opportunity to update our estimation of the system dynamics online at any step together with the control rule.

3.2.1 Stability and Optimality in the Linear Case

In this section, the proposed control framework is analyzed in a special case where both the virtual target system and the tracker system are linear.

In fact, the optimality and stability of optimal control approaches based on forward propagation of the Riccati differential equation is still an open problem, where even in the linear case, their properties are not fully known yet. For instance, in [165], for time-variant linear systems, it has been shown that if the closed-loop dynamics are symmetric, and also for some systems with sufficiently fast dynamics, the stability can be guaranteed. In a more general result, [131] guarantees stability employing the forward solution of differential Riccati equation for $t > T$, where T is some positive time. Hence a full Lyapunov stability analysis for the linear case remains open. The situation becomes even more complex, when we consider the more general framework where we include arbitrary choices of nonlinear bases, or encounter the issue of incomplete knowledge of the system dynamics.

Therefore, in the followings, by taking linear systems as an example, and by making a connection to the classical LQR formulation, we will demonstrate that, in this special case, the presented approach becomes equivalent to the LQR framework. This allows us to provide guarantees for the presented tracking control approach, by exploiting the existing result in the literature, at least for the linear case.

Proposition 1. *Consider the controllable linear system $\dot{x} = Ax + Bu_1$ together with the linear command generator $\dot{y}_{ref} = Ay_{ref} + Bu_d$, where u_d is the given desired input. Furthermore, the tracking performance measure is given by (3.4) with $C = I_{n\times n}$. Then, as $\gamma \to 0$, the optimal feedback control (3.14) constructed by the solution of (3.15), and the optimal value (3.11) with the choice of $\bar{\Phi} = [x \quad y_{ref} \quad 1]^T$, approaches the LQR feedback control with the gain $k = r_1^{-1} S\tilde{B}$ of the generalized error system*

$$\frac{d}{dt}\begin{bmatrix} x \\ e \\ 1 \end{bmatrix} = \tilde{A}\begin{bmatrix} x \\ e \\ 1 \end{bmatrix} + \tilde{B}u_1, \tag{3.16}$$

where $\tilde{A} = \begin{bmatrix} A & 0 & 0 \\ 0 & A & Bu_d \\ 0 & 0 & 0 \end{bmatrix}$, $\tilde{B} = \begin{bmatrix} B & -B & 0 \end{bmatrix}^T$, *$e$ is the tacking error, and S is given at any $t \in [0, \infty)$ by the solution of the well-known continues-time differential Riccati equation*

$$\dot{S} = \tilde{Q} - S\tilde{B}r_1^{-1}\tilde{B}^T S + S\tilde{A} + \tilde{A}^T S. \tag{3.17}$$

Proof. In this case, the linear bases of each tracker and the target dynamics, together with a constant basis 1 will suffice to implement the presented approach. Hence, we proceed with the choice of $\bar{\Phi} = [x \quad y_{ref} \quad 1]^T$. Note that this will not affect the generality, and including extra basis will only add more columns of zeros in the computations. Then the augmented system (3.9) becomes

$$\dot{z} = \begin{bmatrix} \dot{x} \\ \dot{y}_{\text{ref}} \end{bmatrix} = \underbrace{\begin{bmatrix} A & 0 & 0 \\ 0 & A & Bu_d \end{bmatrix}}_{F} \bar{\Phi} + \underbrace{\begin{bmatrix} 0 & 0 & B \\ 0 & 0 & 0 \end{bmatrix}}_{G_1} \bar{\Phi} u_1. \tag{3.18}$$

Similar to the procedure presented in the previous section, the optimal control should satisfy the HJB equation

$$\begin{aligned} \bar{\Phi}^T \dot{P} \bar{\Phi} + \gamma \bar{\Phi}^T P \bar{\Phi} &= \bar{\Phi}^T \begin{bmatrix} Q & -Q & 0 \\ -Q & Q & 0 \\ 0 & 0 & 0 \end{bmatrix} \bar{\Phi} + u_1^{*T} r_1 u_1^* \\ &+ \left(\bar{\Phi}^T P \frac{\partial \bar{\Phi}}{\partial z} \right) \dot{z} + \dot{z}^T \left(\frac{\partial \bar{\Phi}}{\partial z}^T P \bar{\Phi} \right), \end{aligned} \tag{3.19}$$

where

$$P = \begin{bmatrix} P_1 & P_2 & P_3 \\ P_2^T & P_4 & P_5 \\ P_3^T & P_5^T & P_6 \end{bmatrix}.$$

In this step, we substitute $y_{ref} = x + e$ in (3.19). Accordingly, we redefine $z := \begin{bmatrix} x & e \end{bmatrix}^T$ and $\bar{\Phi} := \begin{bmatrix} x & e & 1 \end{bmatrix}$. Hence, the augmented system (3.18) can be rewritten with the new definitions, where augmented system matrix become

$$G_1 := \begin{bmatrix} 0 & 0 & B \\ 0 & 0 & -B \end{bmatrix},$$

and F remain the same.

Now, by substituting the optimal control (3.14) and considering the change of variables, (3.19) is equivalent to

$$\begin{aligned} \bar{\Phi}^T \dot{S} \bar{\Phi} + \gamma \bar{\Phi}^T S \bar{\Phi} &= \bar{\Phi}^T \tilde{Q} \bar{\Phi} + \\ - \bar{\Phi}^T S \frac{\partial \bar{\Phi}}{\partial z} G_1 \bar{\Phi} r_1^{-1} \bar{\Phi}^T G_1^T \frac{\partial \bar{\Phi}}{\partial z}^T S \bar{\Phi} \end{aligned}$$

$$+ \bar{\Phi}^T S \frac{\partial \bar{\Phi}}{\partial z} F \bar{\Phi} + \bar{\Phi}^T F^T \frac{\partial \bar{\Phi}}{\partial z}^T S \bar{\Phi}, \tag{3.20}$$

where $\tilde{Q} = diag([0, Q, 0])$. For brevity, we omitted the detailed computations, while one can check with some effort the equivalency by using Lemma 3, and

$$S = \begin{bmatrix} P_1 + P_4 + P_2 + P_2^T & P_2 + P_4 & P_3 + P_5 \\ P_2^T + P_4^T & P_4 & P_5 \\ P_3^T + P_5^T & P_5^T & P_6 \end{bmatrix}.$$

By plugging in

$$\frac{\partial \bar{\Phi}}{\partial z} = \begin{bmatrix} I & 0 \\ 0 & I \\ 0 & 0 \end{bmatrix},$$

and defining

$$\tilde{A} = \frac{\partial \bar{\Phi}}{\partial z} F = \begin{bmatrix} A & 0 & 0 \\ 0 & A & Bu_d \\ 0 & 0 & 0 \end{bmatrix},$$

$$\tilde{B} = \frac{\partial \bar{\Phi}}{\partial z} G_1 \bar{\Phi} = \begin{bmatrix} B & -B & 0 \end{bmatrix}^T$$

we can conclude from (3.20):

$$\dot{S} = \tilde{Q} - S\tilde{B} r_1^{-1} \tilde{B}^T S + S\tilde{A} + \tilde{A}^T S - \gamma S. \tag{3.21}$$

As $\gamma \to 0$, this yields the well-known continuous-time differential Riccati equation (3.17) for the generalized error system (3.16). Moreover, the LQR feedback gain $k = r_1^{-1} S \tilde{B}$ can be obtained by using the definition of $\tilde{B}$. □

For linear time-invariant systems, the steady-state solution of (3.17) realizes the optimal feedback control that guarantees asymptotic stability of the error, and optimality of the solution. However, in the reinforcement learning setting, as well as in time-variant systems, we can only rely on the presence knowledge of the system and hence on the continuous evolutions of the differential Riccati equation. In the linear case of this type, the stability is guaranteed only for $t > T$. For more details, we refer the readers to Theorem 4 of [131].

The next section will discuss the details of implementing the nonlinear optimal tracking control as a model-based reinforcement learning approach.

3.3 Learning-based Tracking Control Using SOL

Initially, the SOL approach was proposed for solving stabilization and regulation problems [52]. However, in the previous section, we demonstrated that a tracking controller can be obtained by extending the underlying idea of SOL , that is somehow unifying the dynamics and the optimal control objective to gain some flexibility in solving the optimal control problem. As a result, we obtained a state-dependent matrix differential equation (3.15), whose solution provides the parameters of the nonlinear optimal tracking control in terms of the reference and state trajectories. In the followings, we will briefly review the model-based learning framework.

We run the system from some $x_0 \in D$ for a time step of length h. Then, by sampling the input, system state, and reference trajectories, we evaluate $\bar{\Phi}$, which together with the approximation of the augmented state derivative $\dot{z}$ is used to update our estimation of the augmented system, including the system and the commander dynamics coefficients. Later, the estimated weights and the current measured state is used to integrate (3.15) for some time step. This is immediately followed by updating the control value for the next iteration in the control loop by employing (3.14).

Consider the system weight update as

$$[\hat{F}\ \hat{G}_1\ \dots\ \hat{G}_j]_k = \underset{[F,G_1,G_j]_k}{\arg\min} \quad E(\dot{z}_k, [F, G_1, \dots G_j]_k, \Theta(z_k, u_k)),$$

where k is the time step and $E(\cdot)$ is the defined cost in the identification technique employed. Moreover, we construct

$$\Theta(z_k, u_k) = [\bar{\Phi}^T(z_k)\ \bar{\Phi}^T(z_k)u_{1k}\ \dots\ \bar{\Phi}^T(z_k)u_{mk}]^T \tag{3.22}$$

by using the measurements obtained from the system and reference trajectories at time t_k. To solve (3.22), different approaches can be alternatively exploited in various applications, such as SINDy [25], recursive least-squares, neural networks, each of which has its own benefits and drawbacks.

Remark 8. *It should be noted that estimating the true dynamics by (3.22) is not generally a trivial task. However, this is a well-studied topic in the literature of system identification for any estimation technique specifically. The well-know persistent excitation of the input signal is one of the common requirements that can be satisfied by adding an exploring signal to the control.*

Remark 9. *Depending on the application, the commander dynamics (3.3) may be given or unknown. In the case the dynamics are provided, we initialize the associated coefficients when considering the augmented system. This will accelerate the identification process.*

In the control update procedure, a recommended choice for initial condition P_0 is a zero square matrix of appropriate dimension. Then, off-the-shelf solvers can be used to effectively integrate (3.15). This also requires evaluations of $\partial\bar{\Phi}/\partial z_k$ at any time step. Since the bases $\bar{\Phi}$ are chosen beforehand, the partial derivatives can be analytically calculated and stored as functions. Hence, they can be evaluated for any z_k in a similar way as $\bar{\Phi}$ itself.

3.4 Simulation Results

To illustrate the effectiveness of the proposed SOL approach for tracking, we implement it on two benchmark nonlinear systems. As shown in [52], SOL can be employed to solve the regulation problem of nonlinear systems with unknown dynamics, including the benchmark pendulum and Lorenz system examples. In what follows we borrow these two examples to investigate the characteristics of the tracking control approach derived based on the SOL framework. For the simulation results, we use a Runge–Kutta solver to integrate the dynamics of the system. These data are treated as measurements in lieu of physical experiments.

In these benchmark examples, we update the model with the most recent measurement by using SINDy [25]. However, considering that this identification technique is already studied and introduced as a data-efficient and robust method ([25, 78]), we will focus on the properties of the proposed control scheme rather than the identification process within the simulations.

Moreover, it is observed that the accuracy of the obtained model depends directly on the precision in measuring the derivatives of the states, which may be a source of noise in the real-world implementations, resulting in an ineffective controller. Through these simulations, we assume full access to the states. We then obtained the state derivatives using a one-step backward approximation, which can potentially be improved by considering more steps.

We performed the simulations in Python, including the 3D graphics generated via the Vpython module ([146]). The sampling rate is 200Hz (h = 5ms) for all the simulations. Accordingly, the control input value is calculated with the frequency of 100Hz. In the learning process, to simulate the real behaviors of the given system, we integrate the differential equations with a sufficient precision from initial conditions randomly chosen within the domain of interest. However, we only allow measurements at time steps conforming to the sampling rate.

Furthermore, in the following examples, we use the candidate bases $\{1, x, x^2, x^3, \sin x, \cos x, x_i x_j\}$, where $i, j \in \{1, \ldots, n\}, i \neq j$, and the operations on vector x is assumed to be component-wise and defines a sub-category of bases, e.g. $x^2 = \{x_1^2, \ldots, x_n^2\}$. Hence, Assumption 2 holds.

3.4.1 Tracking Control of Pendulum

The state space model used for the simulation of the pendulum system is given by

$$\begin{aligned} \dot{x_1} &= -x_2, \\ \dot{x_2} &= -\frac{g}{l}\sin(x_1) - \frac{k}{m}x_2 + \frac{1}{ml^2}u, \end{aligned} \tag{3.23}$$

where $m = 0.1kg$, $l = 0.5m$, $k = 0.1$, and $g = 9.8m/s^2$. A desired performance is characterized by matrices $Q = \text{diag}([2, 7])$, $R = 1$, $\gamma = 1$.

The simulations for this system are done in two different scenarios. In the first scenario, we assume having the full state reference trajectory for the angle and angular velocity. Thus, we try, for instance, sinusoidal and ramp reference signals as below respectively.

$$(a) : \begin{cases} y_{1_{\text{ref}}} = -\sin(t), \\ y_{2_{\text{ref}}} = \cos(t). \end{cases} \quad , \quad (b) : \begin{cases} y_{1_{\text{ref}}} = -t, \\ y_{2_{\text{ref}}} = 1, \end{cases} \tag{3.24}$$

where the corresponding states are measured by choosing $C = \text{diag}([1, 1])$ in (3.1).

Fig. 3.1 and Fig. 3.2 illustrate system responses for sinusoidal and ramp references starting from a random initial condition, respectively. As seen in these figures, although the learning process is run with zero prior experience, it can efficiently track the reference by quickly learning the dynamics and the optimal tracking controller.

In the second scenario, to better examine the tracking control scheme presented in the main results, we assume only the trajectory of the angular position is provided as the reference, where $C = \text{diag}([1, 0])$and $Q = 2$. Hence, the control objective is defined based on only the angular position error. For this reason, one should not expect the tracking results to be as smooth as the previous case. However, as shown in Fig. 3.3, the objective is achieved by perfect tracking of the angular position, where in addition, the other state still resembles the non-given target trajectory, to an acceptable extent.

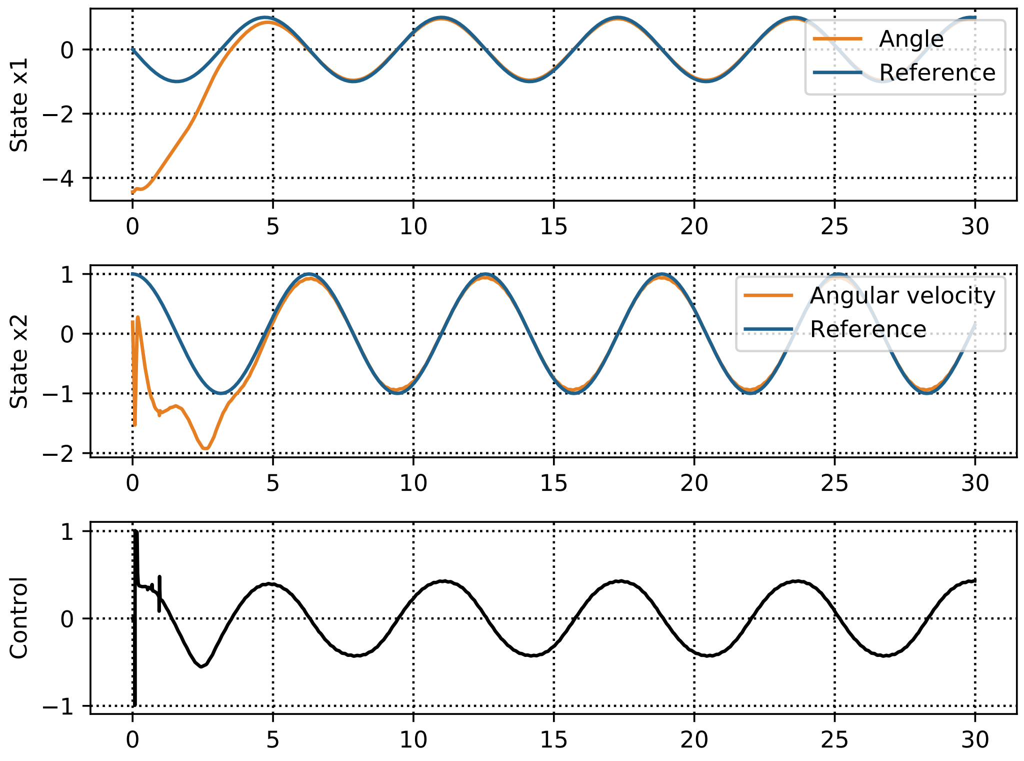


Figure 3.1: The control and states of the pendulum system within a run of the implemented learning approach from a randomly chosen initial condition for tracking a full state sinusoidal reference signal provided as in (3.24a).

3.4.2 Synchronization of Chaotic Lorenz System

The Lorenz system is well-known for its chaotic behavior around its unstable equilibrium points. As an illustrative example (Fig. 3.4), we aim to synchronize two Lorenz systems starting from different initial conditions. This is done by measuring the states of one as the target system, then controlling the other system to track these target states over time by the proposed learning-based tracking control technique. We assume no prior knowledge of the system dynamics and parameters. Hence, the dynamics of the target and the tracker are learned together with a tracking controller on the fly. The system dynamics used in the simulation are described as

$$\dot{x_1} = \sigma(x_2 - x_1) + u,$$

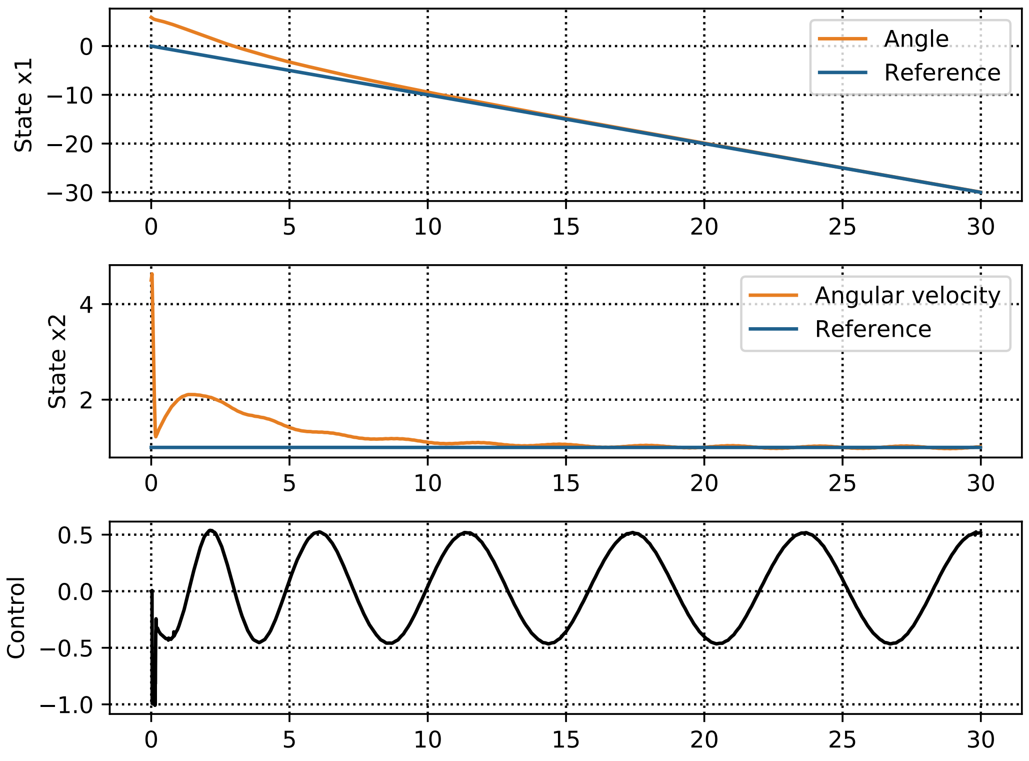


Figure 3.2: The control and states of the pendulum system within a run of the implemented learning approach from a randomly chosen initial condition for tracking a full state ramp reference signal provided as in (3.24b).

$$\dot{x_2} = -x_2 + x_1(\rho - x_3),$$
$$\dot{x_3} = x_1 x_2 - \beta x_3, \tag{3.25}$$

where $\sigma = 10$, $\rho = 28$, and $\beta = 8/3$. Furthermore, we set the performance criteria to $Q = \text{diag}([280, 280, 210])$, $R = 0.05$, and $\gamma = 200$, with choosing C as an identity matrix to match with the provided full state reference by the target system. Fig. 3.5 illustrates the evolution of the controlled system and the target trajectories together with the control while learning process. In Fig. 3.6, the tracking value and its parameters are shown. A video of the simulation showing the synchronization details is uploaded on https://youtu.be/1SnvDyb_7Os.

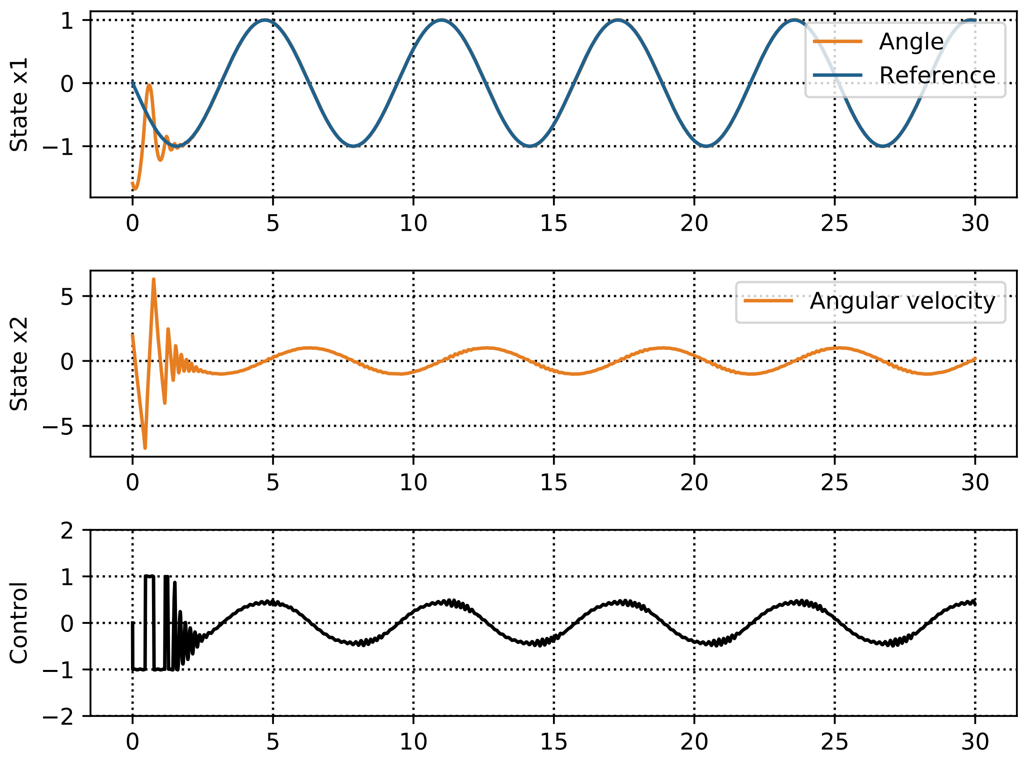


Figure 3.3: The control and states of the pendulum system within a run of the implemented learning approach from a randomly chosen initial condition where only the angle trajectory is provided as reference to be tracked.

3.5 Conclusion

This chapter introduces an online learning-based method for nonlinear tracking with unknown dynamics. We assumed nonlinear affine dynamics structured in terms of a set of bases functions. Accordingly, we formulated an optimal tracking control, where the objective function was redefined to conform with the structured system. This performance measure and a value function parameterized in a quadratic form were approximately minimized by solving a derived matrix differential equation. Hence, the formulation allows us to compose a learning-based tracking control framework, which relies only on the online measurements of the system states and the reference trajectory. In the simulation results, the proposed learning approach demonstrated satisfactory tracking of fully or partially

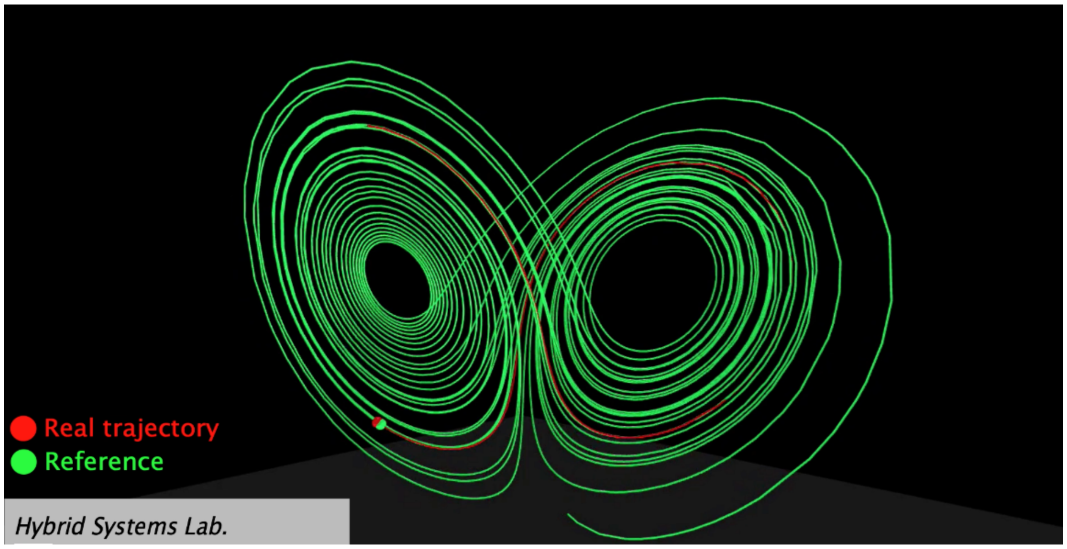


Figure 3.4: A view of the 3D simulation done for synchronizing the chaotic Lorenz system. The video can be accessed at: `https://youtu.be/1SnvDyb_70s`.

provided reference trajectories. Considering the improved computational complexity of the obtained update rule for the value parameters, future research will include addressing practical limitations, and accordingly obtain an end-to-end platform for real-world applications.

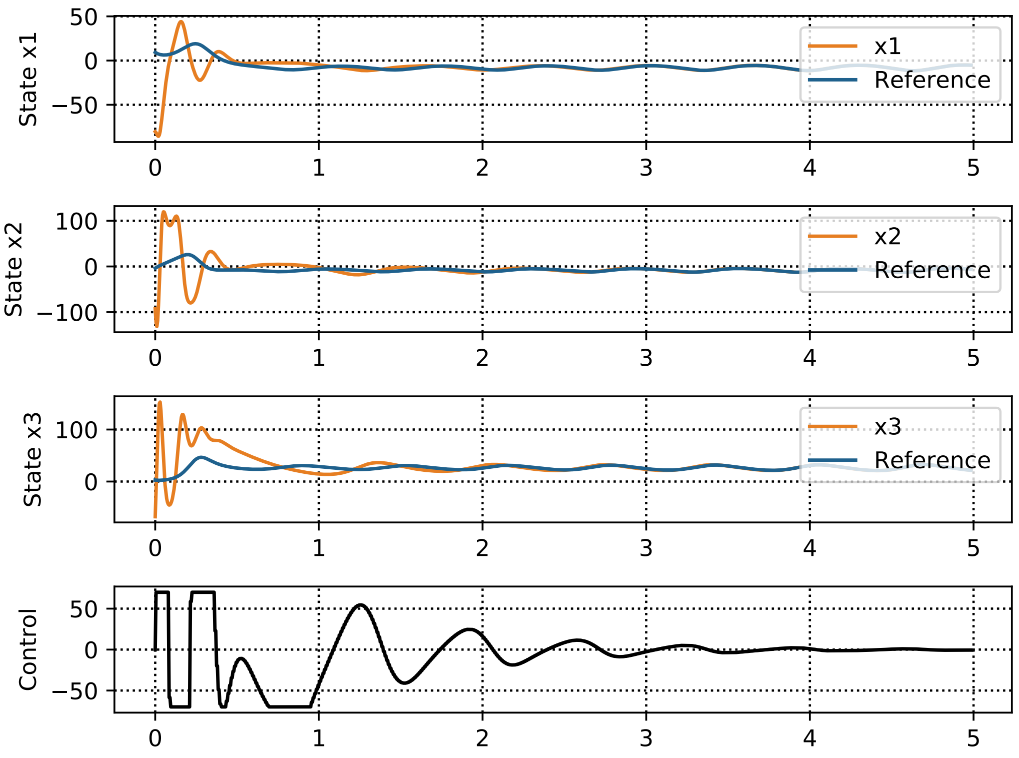


Figure 3.5: The states and the obtained control of the Lorenz system while learning to synchronize with the given reference trajectories, starting from a random initial condition.

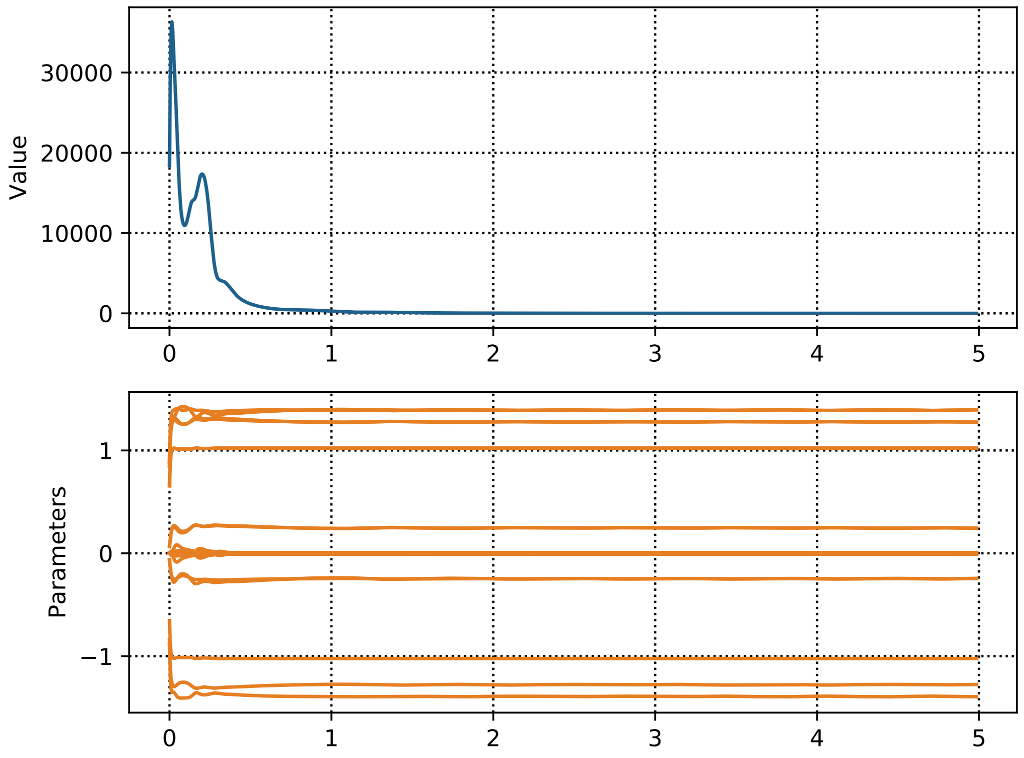


Figure 3.6: The evolutions of the value and parameters while learning the tracking controller of the Lorenz system, corresponding to Fig. 3.5.

Chapter 4

Piecewise Learning and Control with Stability Guarantees

The results presented in this chapter are submitted to [50].

4.1 Introduction

In chapter 2, we employed a set of bases to parameterize the system model. For this purpose, a set of polynomial or trigonometric bases is proven to be effective in approximating different functions with any arbitrary accuracy over a compact domain. Even though a set of polynomial bases, for instance, is known to be sufficient as a universal approximator, the number of bases required for a tight approximation of the dynamics over a given domain may be exceedingly high. The number of the bases, in fact, depends on the domain of interest, where a larger domain may exhibits nonlinearites that requires a larger set of bases. This highly impedes implementations, especially in an online learning and control setting.

In an alternative approach, instead of adding many bases to cover a large domain of interest, we divide the domain into pieces where each can be handled independently with a limited number of bases. Employing a piecewise model will improve learning greatly by keeping the online computations needed for updating the model in a tractable size.

Despite the improvement in the computations, data efficiency of learning may be diminished if a large number of the pieces is chosen. Considering that the total number of the model parameters is relative to the number of pieces, a piecewise model may involve more parameters compared to learning in terms of bases. In fact, there exist a trade off between the data efficiency and computational efficiency that can be controlled by the number of pieces employed.

The rest of the chapter is presented in the following order. Section 4.2 formulates the problem. In Section 4.3, we propose a piecewise learning and control framework, where we first obtain an estimation of the system and then solve an approximate optimal control in a closed-loop form. In Section 4.4, we provide an upper bound for the uncertainty in the identified piecewise model based on the observations. In Section 4.5, the obtained uncertainty bounds are implemented to synthesize a Lyapunov function for the closed-loop system. In Section 4.6, two benchmark examples are discussed to numerically validate the approach.

4.2 Problem Formulation

Consider the nonlinear system in control-affine form

$$\dot{x} = F(x,u) = f(x) + g(x)u = f(x) + \sum_{j=1}^{m} g_j(x)u_j, \tag{4.1}$$

where $x \in D \subset \mathbb{R}^n$, $u \in \Omega \subset \mathbb{R}^m$, $f : D \to \mathbb{R}^n$, and $g : D \to \mathbb{R}^{n\times m}$.

The cost functional to be minimized along the trajectory, started from the initial condition $x(0) = x_0$, is considered to be in the following linear quadratic form

$$J(x_0, u) = \lim_{T\to\infty} \int_0^T \mathrm{e}^{-\gamma t} \left(x^T Q x + u^T R u\right) \mathrm{d}t, \tag{4.2}$$

where $Q \in \mathbb{R}^{n\times n}$ is positive semi-definite, $\gamma \geq 0$ is the discount factor, and $R \in \mathbb{R}^{m\times m}$ is a diagonal matrix with only positive values, given by the design criteria.

4.3 The Piecewise Learning and Control Framework

We approximate the nonlinear system (4.1) by a piecewise model with a bounded uncertainty

$$\dot{x} = W_\sigma \Phi(x) + \sum_{j=1}^{m} W_{j\sigma} \Phi(x) u_j + d_\sigma, \tag{4.3}$$

where $d_\sigma \in \mathbb{R}^n$ is a time-varying uncertainty, W_σ and $W_{j\sigma} \in \mathbb{R}^{n\times p}$ are the matrices of the coefficients for $\sigma \in \{1, 2, \ldots, n_\sigma\}$ and $j \in \{1, 2, \ldots, m\}$, with a set of differentiable bases $\Phi(x) = [\phi_1(x) \quad \ldots \quad \phi_p(x)]^T$, and n_σ denoting the total number of pieces. Moreover, any piece of the system is defined over a convex set given by a set of linear inequalities as $\Upsilon_\sigma = \{x \in D | Z_\sigma x \leq z_\sigma\}$, where $\sigma \in \{1, \ldots, n_\sigma\}$ and Z_σ and z_σ are a matrix and a vector of appropriate dimensions.

We assume that the set $\{\Upsilon_\sigma\}$ forms a partition of the domain and its elements do not share any interior points, i.e. $\bigcup_{\sigma=1}^{n_\sigma} \Upsilon_\sigma = D$ and $\text{int}[\Upsilon_\sigma] \bigcap \text{int}[\Upsilon_l] = \emptyset$ for $\sigma \neq l$ and σ, $l \in \{1, 2, \ldots, n_\sigma\}$. Furthermore, the piecewise model is assumed to be continuous across the boundaries of $\{\Upsilon_\sigma\}$ that will be discussed later in detail. The control input and the uncertainty are assumed to be bounded and lie in the sets $\Omega = \{u \in \mathbb{R}^m | |u_j| \leq \bar{u}_j, \forall j \in \{1, 2, \ldots, m\}\}$ and $\Delta_\sigma = \{d_\sigma \in \mathbb{R}^n | |d_{\sigma i}| \leq \bar{d}_{\sigma i}, \forall i \in \{1, 2, \ldots, n\}\}$, respectively. The uncertainty upper bound $\bar{d}_\sigma = (\bar{d}_{\sigma 1}, \cdots, \bar{d}_{\sigma n})$ is to be determined.

4.3.1 System Identification

Having defined the parameterized model of the system, we employ a system identification approach to update the system parameters. For each pair of samples obtained from the input and state of the system, i.e., (x^s, u^s), we first locate the element in the partition $\{\Upsilon_\sigma\}$ that contains the sampled state x^s. Then, we locally update the system coefficients of the particular piece from which the state is sampled. The weights are updated according to

$$[\hat{W}_\sigma \quad \hat{W}_{1\sigma} \ \ldots \ \hat{W}_{m\sigma}]_k = \underset{\bar{W}}{\arg\min} \quad \|\dot{X}_{k\sigma} - \bar{W}\Theta_{k\sigma}\|_2^2, \tag{4.4}$$

where k is the time step, and $\Theta_{k\sigma}$ includes a matrix of samples with

$$\Theta_k{}^s = [\Phi^T(x^s) \quad \Phi^T(x^s)u_1^s \quad \ldots \quad \Phi^T(x^s)u_m^s]_k^T,$$

for the sth sample in the σth partition. Correspondingly, $\dot{X}_{k\sigma}$ contains the sampled state derivatives. While in principle any identification technique can be used, e.g., [26, 175], the linearity with respect to the coefficients allows us to employ least-square techniques. In this chapter, since an online application is intended, we implement the recursive least-square technique that provides a more computationally efficient way to update the parameters.

Continuity of the Identified Model

Considering that differentiable bases are assumed, the model identified is differentiable within the interior of Υ_σ for $\sigma \in \{1, 2, \ldots, n_\sigma\}$. However, the pieces of the model may not meet in the boundaries of Υ_σ where $x \in \Upsilon_\sigma \bigcap \Upsilon_l$ for any $\sigma \neq l$ and σ, $l \in \{1, 2, \ldots, n_\sigma\}$.

Based on our knowledge of system (4.1) from which we collect samples the continuity holds for the original system. Hence, in theory, if many pieces are chosen, and enough samples are collected, the edges of pieces will converge together to yield a continuous model. However, choosing arbitrarily small pieces is not practical.

There exist different techniques to efficiently choose the partitions on D and best fit a continuous piecewise model, see e.g. [158, 24, 57, 4, 139]. Such techniques usually involve global adjustments of the model weights and the partitions for which the computations can be considerably expensive. Therefore, we choose to locally deal with the gaps among the pieces. This can be done by a post-processing routine performed on the identified model.

A rather straightforward technique is to define extra partitions in the margins of each Υ_σ to fill the gaps among pieces. The weights of the corresponding pieces added can be chosen according to the weights of the adjacent pieces that is given by the identification. This is done in a way that they help to connect all the pieces together to make a continuous piecewise model. Fig. 4.1 illustrates the process of constructing extra partitions for a two-dimensional case, where we choose them to be in triangular shapes. A similar approach can be taken for generalizing to the n-dimensional case.

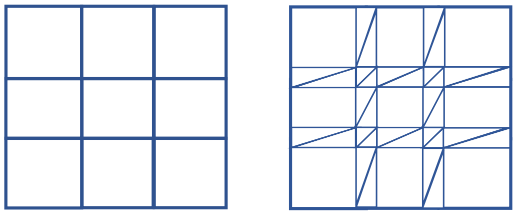


Figure 4.1: A scheme of obtaining a continuous piecewise model is illustrated. On the left, partitions on two-dimensional domain is shown for which the pieces of the model may not be connected in the borders. On the right, some extra triangular pieces are constructed to allow filling the possible gaps in the model.

4.3.2 Database

Although an online technique is used to update the piece-wise model along trajectories, we still need to collect a number of samples for each piece of the system. The set of samples recorded will be used later to obtain an estimation of the uncertainty bounds for each mode of the system. For this purpose, we, over time, hand pick and save samples that best describe the dynamics in any mode of the piecewise system.

It should be noted that, the database will be processed offline to extract the uncertainty bounds. Hence, it does not affect the online learning procedure and its computational cost. Any sample of the system, to be stored in the database, includes $({\Theta_k}^s, \hat{\dot{x}}_k)$, where the state derivative is approximated by $\hat{\dot{x}}_k = (x_k - x_{k-1})/h$ and $\dot{e}_k$. For better results, higher order approximations of the state derivative can be employed.

Different techniques can be employed to obtain a summary of the samples collected. We assume a given maximum size of database N_d. Then, for any mode of the piecewise model, we keep adding the samples with larger prediction errors to the database. Therefore, at any step, we compare the prediction error $\dot{e}_k = \|\dot{x}_k - \hat{\dot{x}}_k\|$ with the most recent average error $\bar{\dot{e}}_{k\sigma}$ obtained for the active piece. Hence, if the condition $\dot{e}_k > \eta\bar{\dot{e}}_{k\sigma}$ holds we add the sample to the database, where the constant $\eta > 0$ adjusts the threshold. If the maximum number of samples in database is reached, we replace the oldest sample with the recent one.

4.3.3 Feedback Control

In Chapter 2, a matrix differential equation is proposed using a quadratic parametrization in terms of the basis functions to obtain a feedback control. Here, we adopt a similar learning framework, but consider a family of n_σ differential equations, each of which corresponds to one particular mode of the system in the piecewise model. We integrate the following state-dependent Riccati differential equation in forward time:

$$\begin{aligned}-\dot{P}_\sigma =&\bar{Q} + P_\sigma \frac{\partial \Phi(x)}{\partial x} W_\sigma + W_\sigma^T \frac{\partial \Phi(x)}{\partial x}^T P_\sigma - \gamma P_\sigma \\ &- P_\sigma \frac{\partial \Phi(x)}{\partial x} \left(\sum_{j=1}^{m} W_{j\sigma} \Phi(x) r_j^{-1} \Phi(x)^T W_{j\sigma}^T \right) \frac{\partial \Phi(x)}{\partial x}^T P_\sigma.\end{aligned} \tag{4.5}$$

The solution to the differential equation (4.5) characterizes the value function defined by

$$V_\sigma = \Phi^T P_\sigma \Phi, \tag{4.6}$$

based on which we obtain a piecewise control

$$u_j = -r_j^{-1} \frac{\partial V_\sigma}{\partial x}^T g_j(x) = -\Phi(x)^T r_j^{-1} P_\sigma \frac{\partial \Phi(x)}{\partial x} W_{j\sigma} \Phi(x). \tag{4.7}$$

4.4 Analysis of Uncertainty Bounds

We use the uncertainty in the piecewise system (4.3) to capture approximation errors in identification. In this section, we analyze the worst-case bounds to provide guarantees for the proposed framework.

There exist two sources of uncertainty that affect the accuracy of the identified model. The first is the mismatch between the identified model and the observations made. The latter may also be affected by the measurement noise. The second is due to unsampled areas in the domain. We can estimate the uncertainty bound for any piece of the model by combining these two bounds. In what follow, we discuss the procedure of obtaining these bounds in more detail.

Assumption 4. *For any given (x^s, u^s), let $F_i(x^s, u^s)$ be the ith element of $F(x^s, u^s)$. We assume that $F_i(x^s, u^s)$ can be measured with some tolerance as $\tilde{F}_i(x^s, u^s)$, where $|\tilde{F}_i(x^s, u^s) - F_i(x^s, u^s)| \leq \varrho_e |\tilde{F}_i(x^s, u^s)|$ with $0 \leq \varrho_e < 1$ for all $i \in \{1, \cdots, n\}$.*

We make predictions $\hat{F}_i(x^s, u^s)$ of the state derivatives for any sample using the identified model. Hence, we can easily compute the distance between the prediction and the approximate evaluation of the system by using the samples collected for any piece. This gives the loss $|\hat{F}_i(x^s, u^s) - \tilde{F}_i(x^s, u^s)|$.

Theorem 3. *Let Assumption 4 hold, and S_{Υ_σ} denote the set of indices for sample pairs (x^s, u^s) such that $x^s \in \Upsilon_\sigma$. Then, an upper bound of the prediction error, regarding any sample (x^s, u^s) for $s \in \{1, \ldots, N_s\}$, is given by*

$$|\hat{F}_i(x^s, u^s) - F_i(x^s, u^s)| \leq \bar{d}_{e\sigma i} := \max_{s \in S_{\Upsilon_\sigma}} \left(|\hat{F}_i(x^s, u^s) - \tilde{F}_i(x^s, u^s)| + \varrho_e |\tilde{F}_i(x^s, u^s)|\right),$$

where $\sigma \in \{1, \ldots, n_\sigma\}$, and $i \in \{1, \ldots, n\}$.

Proof. According to Assumption 4, it is straight forward to show that the prediction error can be bound for any σ by using the samples in partition σ as

$$\begin{aligned}
|\hat{F}_i(x^s, u^s) - F_i(x^s, u^s)| &\leq |\hat{F}_i(x^s, u^s) - \tilde{F}_i(x^s, u^s)| + |\tilde{F}_i(x^s, u^s) - F_i(x^s, u^s)| \\
&\leq |\hat{F}_i(x^s, u^s) - \tilde{F}_i(x^s, u^s)| + \varrho_e |\tilde{F}_i(x^s, u^s)| \\
&\leq \max_{s \in S_{\Upsilon_\sigma}} \left(|\hat{F}_i(x^s, u^s) - \tilde{F}_i(x^s, u^s)| + \varrho_e |\tilde{F}_i(x^s, u^s)|\right) \\
&= \bar{d}_{e\sigma i}.
\end{aligned}$$

□

4.4.1 Quadratic Programs for Bounding Errors

The samples may not be uniformly obtained from the domain. Depending on how smooth the dynamics are, there might be unpredictable behavior of the system in the gaps among the samples. Hence, the predictions made by the identified model may be misleading in the areas we have not visited yet. To take this into account, we assume a Lipschitz constant is given for the system. More specifically, we let $\varrho_x \in \mathbb{R}_+^n$ and $\varrho_u \in \mathbb{R}_+^n$ denote the Lipschitz constants of $F(x, u)$ with respect to x and u on $D \times \Omega$, respectively. We use this to bound the uncertainty for the unsampled areas.

We need to compute the worst case of the prediction error within any piece that is given by $|\hat{F}_i(x, u) - F_i(x, u)|$, where $\hat{F}(\cdot, \cdot)$ denotes an evaluation of the identified model. However, according to Assumption 4, we do not have access to the original system to exactly evaluate $F(\cdot, \cdot)$. Therefore, we obtain the bound in terms of the approximate value instead.

Assumption 5. *For system (4.1), $\exists \varrho_x \in \mathbb{R}_+^n$ such that we have*

$$|F_i(x_0, u) - F_i(y_0, u)| \leq \varrho_{xi} \|x_0 - y_0\|,$$

for any $x_0, y_0 \in D$, and $u \in \Omega$, where $i \in \{1, \dots, n\}$.

Assumption 6. *For system (4.1), $\exists \varrho_u \in \mathbb{R}_+^n$ such that we have*

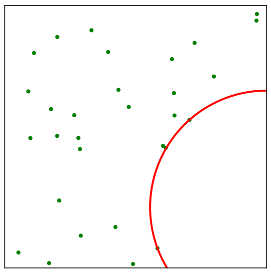
$$|F_i(x, u_0) - F_i(x, w_0)| \leq \varrho_{ui} \|u_0 - w_0\|,$$

for any $x \in D$, and $u_0, w_0 \in \Omega$, where $i \in \{1, \dots, n\}$.

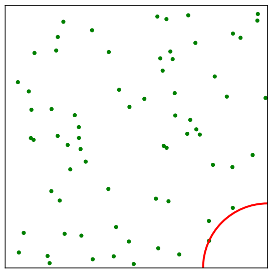
Assumption 7. *An initial estimation of ϱ_e and Lipschitz constants ϱ_{xi} and ϱ_{ui} is known.*

The following results and the bounds will directly depend on the choice of ϱ_x, and ϱ_u. However, this is the least we can assume that allows us to carry out the computations. Moreover, making such assumptions is not restrictive in practice since we often have a general knowledge of the application. Moreover, the learning may be first started with an initial guess of the continuity constants. Later, if the samples collected override the assumption made, we can update these values.

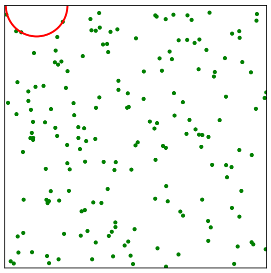
To calculate the uncertainty bound for any piece, we first look for the largest gap existing among the samples within each piece. The procedure starts with searching for the largest gaps in the state and control spaces that do not contain any samples as show in Fig. 4.2. Let (x^{s*}, u^{s*}) be the closest sample indexed in $S_{\Upsilon\sigma}$ to the center point $(c^*_{x\sigma}, c^*_{u\sigma})$



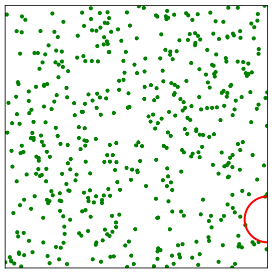
(a)



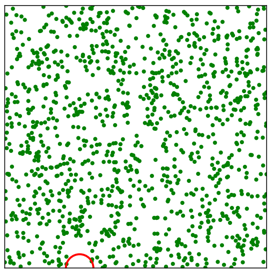
(b)



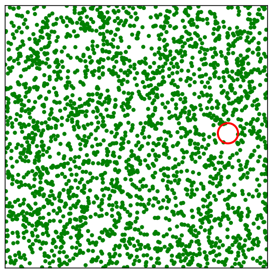
(c)



(d)



(e)



(f)

Figure 4.2: Sub-figures (a)-(f) denote the sample gaps located for different number of samples. It is observed that the radius of the gap decreases by increasing the number of the samples.

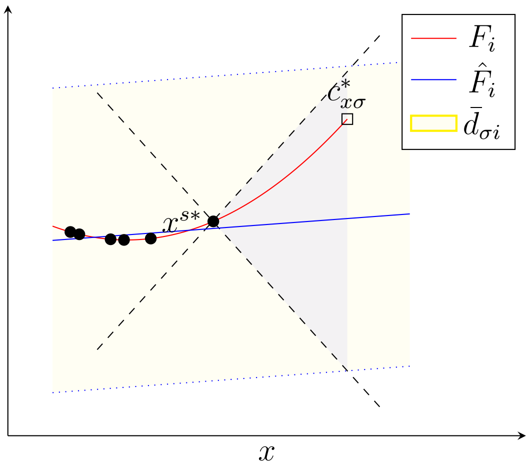


Figure 4.3: The scheme for obtaining the uncertainty bound according to the sample gap. Black dots denote the measurements.

of the sample gap (as a Euclidean ball) with radius $(r^*_{x\sigma}, r^*_{u\sigma})$. For this purpose, we solve a quadratic programming (QP) problem for each piece. The solution to the following QP returns the centre $c^*_{x\sigma}$ at which an n-dimensional ball of the largest radius $r^*_{x\sigma}$ can be found in the σth piece such that no samples x^s are contained in this ball:

$$\begin{aligned} \underset{c_{x\sigma}, r_{x\sigma}}{\arg\max} \quad & r_{x\sigma} \\ \text{subject to} \quad & c_{x\sigma} \in \Upsilon_\sigma \\ & \text{for} \quad s \in S_{\Upsilon\sigma}: \quad \|x^s - c_{x\sigma}\| \geq r_{x\sigma} \end{aligned} \tag{4.8}$$

Similarly, we can obtain the centre $c^*_{u\sigma}$ and radius $r^*_{u\sigma}$ to represent the sample gap as an m-dimensional ball in the control space by solving

$$\begin{aligned} \underset{c_{u\sigma}, r_{u\sigma}}{\arg\max} \quad & r_{u\sigma} \\ \text{subject to} \quad & c_{u\sigma} \in \Omega \\ & \text{for} \quad s \in S_{\Upsilon\sigma}: \quad \|u^s - c_{u\sigma}\| \geq r_{u\sigma}. \end{aligned} \tag{4.9}$$

Theorem 4. *Let Assumptions 4-7 hold and $(r^*_{x\sigma}, r^*_{u\sigma})$ is given by the solutions of (4.8) and (4.9) (details given in Appendix 4.4.1). Then, an upper bound for the prediction error can be obtained regarding all unvisited points $x \in \Upsilon_\sigma$ and $u \in \Omega$ as below*

$$|F_i(x,u) - \hat{F}_i(x,u)| \leq \bar{d}_{\sigma i} = \varrho_{ui} r^*_{u\sigma} + \varrho_{xi} r^*_{x\sigma} + \bar{d}_{e\sigma i} + \hat{\varrho}_{ui} r^*_{u\sigma} + \hat{\varrho}_{xi} r^*_{x\sigma}. \tag{4.10}$$

Proof. According to the Lipschitz condition, the following holds for any $(x, u) \in \Upsilon_\sigma$

$$\begin{aligned}|F_i(x,u)-&F_i(x^{s*},u^{s*})| \\ &\leq |F_i(x,u)-F_i(x,u^{s*})| + |F_i(x,u^{s*})-F_i(x^{s*},u^{s*})| \\ &\leq \varrho_{ui}\|u-u^{s*}\| + \varrho_{xi}\|x-x^{s*}\|. \end{aligned} \tag{4.11}$$

Moreover, we have the estimation $\hat{F}(x, u)$ of the system. Then, the difference is bounded by

$$\begin{aligned}|F_i(x,u)-\hat{F}_i(x,u)| &\leq |F_i(x,u)-F_i(x^{s*},u^{s*})| + |F_i(x^{s*},u^{s*})-\hat{F}_i(x,u)|, \\ &\leq |F_i(x,u)-F_i(x^{s*},u^{s*})| + |F_i(x^{s*},u^{s*})-\hat{F}_i(x^{s*},u^{s*})|, \\ &\quad + |\hat{F}_i(x^{s*},u^{s*})-\hat{F}_i(x,u)|, \\ &\leq \varrho_{ui}\|u-u^{s*}\| + \varrho_{xi}\|x-x^{s*}\| + \bar{d}_{e\sigma i} + |\hat{F}_i(x^{s*},u^{s*})-\hat{F}_i(x,u)|, \\ &\leq \varrho_{ui}\|u-u^{s*}\| + \varrho_{xi}\|x-x^{s*}\| + \bar{d}_{e\sigma i} + |\hat{F}_i(x^{s*},u^{s*})-\hat{F}_i(x^{s*},u)|, \\ &\quad + |\hat{F}_i(x^{s},u)-\hat{F}_i(x,u)|, \\ &\leq \varrho_{ui}\|u-u^{s*}\| + \varrho_{xi}\|x-x^{s*}\| + \bar{d}_{e\sigma i} + \hat{\varrho}_{ui}\|u-u^{s*}\| + \hat{\varrho}_{xi}\|x-x^{s*}\|, \end{aligned}$$

where in the last step, we used inequality (4.11) and the bound obtained in Theorem 3 according to the samples. Then, considering that $\hat{F}_i(x, u)$ is known, we can easily compute the corresponding Lipschitz constants $\hat{\varrho}_{ui}$ and $\hat{\varrho}_{xi}$. The largest distance with the closest sample (x^{s*}, u^{s*}) happens in the sample gap given with the radius $r^*_{x\sigma}$, and $r^*_{u\sigma}$. This yields the total bound of the error as

$$|F_i(x,u)-\hat{F}_i(x,u)| \leq \varrho_{ui}r^*_{u\sigma} + \varrho_{xi}r^*_{x\sigma} + \bar{d}_{e\sigma i} + \hat{\varrho}_{ui}r^*_{u\sigma} + \hat{\varrho}_{xi}r^*_{x\sigma}$$

□

4.5 Stability Verification for Piecewise-Affine Learning and Control

4.5.1 Piecewise Affine Models

A special case of system (4.3) can be obtained when we choose $\Phi(x) = \begin{bmatrix}1 & x^T\end{bmatrix}$. We consider system coefficients in the form of $W_\sigma = \begin{bmatrix}C_\sigma & A_\sigma\end{bmatrix}$ and $W_{j\sigma} = \begin{bmatrix}B_{j\sigma} & 0\end{bmatrix}$. Clearly,

A_σ, $B_{j\sigma}$, and C_σ can be used to rewrite the PWA system in the standard form

$$\dot{x} = A_\sigma x + \sum_{j=1}^{m} B_{j\sigma} u_j + C_\sigma + d_\sigma, \tag{4.12}$$

4.5.2 MIQP-based Stability Verification of PWA Systems

In this section, we adopt an MIQP-based verification technique based on the approach presented in [33]. In this framework, by considering a few steps ahead, we verify that the Lyapunov function is decreasing. However, it may not be necessarily monotonic, meaning that it may be increasing in some steps and then be decreasing greatly in some other steps to compensate. Regarding the fact that this approach is inherently a discrete technique, we need to consider a discretization of (4.12). By Euler approximation we have

$$x_{k+1} = \check{F}_d(x_k, u_k) = \check{A}_\sigma x_k + \sum_{j=1}^{m} \check{B}_{j\sigma} u_{jk} + \check{C}_\sigma + d_\sigma \tag{4.13}$$

where $\check{A}_\sigma$, $\check{B}_{j\sigma}$, and $\check{C}_\sigma$ are the discrete system matrices of the same dimension as (4.12). Moreover, we re-adjust the uncertainty bound as $\bar{d}_\sigma := h\bar{d}_\sigma$, where h denotes the time step.

We refer the uncertain closed loop system with the control $u_{jk} = \omega_j(x_k)$ as

$$x_{k+1} = \check{F}_{d,cl}(x_k). \tag{4.14}$$

For this system, let the convex set $\bar{D} = \{x \in D | Z_{\bar{D}} x \leq z_{\bar{D}}\}$ be a user-defined region of interest, within which obtaining a region of attraction (ROA) is desirable.

Searching for a Lyapunov Function

We summarize an altered version of the technique for obtaining a Lyapunov function that is first presented in [33] for a deterministic closed-loop system with the neural network controller. Hence, we modify the algorithm to allow the uncertainty together with the feedback control (4.7).

The procedure includes two stages that are performed iteratively until a Lyapunov function is obtained and verified, or it is concluded that there exist no Lyapunov function in the given set of candidates.

In the first stage, we assume an initial set of Lyapunov candidates in the form of (4.15). Then, the learner searches for a subset for which the negativity of the Lyapunov difference can be guaranteed with respect to a set of samples collected from the system. If such subset exists, one element in this subset is proposed as the Lyapunov candidate by the learner.

In the second stage, the proposed Lyapunov candidate is verified on the original system. Noting that the learner only uses finite number of samples for suggesting a Lyapunov candidate, it may not be valid for all the evolutions of the uncertain system. Accordingly, the verifier either certifies the Lyapunov candidate, or finds a point as the counter-example for which the Lyapunov candidate fails. This sample is added to the set of samples collected from the system. Then, we again proceed to the learner stage with the updated set of samples.

The algorithm is run in a loop, where we start with an empty set of samples in the learner. Then, we continue with proposing a Lyapunov candidate, and adding one counter-example in each iteration of the loop. While growing the set of samples, the set of Lyapunov candidates shrinks in every iteration until it is either validated, or no element is left in the set meaning that no such Lyapunov exists.

Learning and Verification of A Lyapunov Function

Assuming $u_j = -r_j^{-1} B_{j\sigma}^T P_{3\sigma} x_k$, and defining $\check{A}_{cl,\sigma} = \check{A}_\sigma - \sum_{j=1}^m r_j^{-1} \check{B}_{j\sigma} B_{j\sigma}^T P_\sigma$, the discrete closed-loop system becomes $x_{k+1} = \check{A}_{cl,\sigma} x_k + \check{C}_\sigma + d_\sigma$.

Now, consider the Lyapunov function

$$V(x_k, \hat{P}) = \begin{bmatrix} x_k \\ x_{k+1} \end{bmatrix}^T \hat{P} \begin{bmatrix} x_k \\ x_{k+1} \end{bmatrix} \tag{4.15}$$

characterized by $\hat{P} \in \mathscr{F}$, where

$$\mathscr{F} = \{\hat{P} \in \mathbb{R}^{2n \times 2n} | 0 \leq \hat{P} \leq I, V(x_{k+1}, \hat{P}) - V(x_k, \hat{P}) < 0, \forall x_k \in \bar{D} \backslash \{0\}, d_\sigma \in \Delta_\sigma\}.$$

The structure of the Lyapunov function is suggested by [33] that employs a piecewise quadratic function to parameterize the Lyapunov function. This approach combines the non-monotonic Lyapunov function [3] and finite-step Lyapunov function [20, 2] techniques to provide a guarantee by looking at the next few steps. It should be noted that the Lyapunov function may not be necessarily decreasing within any single step, while it must be decreasing within the finite steps taken into account.

The Learner: To realize a Lyapunov function, one needs a mechanism to look for the appropriate values of $\hat{P}$ within $\mathscr{F}$. For this purpose, we obtain an over-approximation of $\mathscr{F}$ by considering only finite number of elements in $(\bar{D}, \Delta)$. Let us first define the increment on the Lyapunov function as

$$\Delta V(x, \hat{P}) = V(\check{F}_{d,cl}(x), \hat{P}) - V(x, \hat{P}) = \begin{bmatrix} \check{F}_{d,cl}(x) \\ \check{F}^{(2)}_{d,cl}(x) \end{bmatrix}^T \hat{P} \begin{bmatrix} \check{F}_{d,cl}(x) \\ \check{F}^{(2)}_{d,cl}(x) \end{bmatrix} - \begin{bmatrix} x \\ \check{F}_{d,cl}(x) \end{bmatrix}^T \hat{P} \begin{bmatrix} x \\ \check{F}_{cl}(x) \end{bmatrix},$$

where $\check{F}^{(2)}_{d,cl}(x) = \check{F}_{d,cl}(\check{F}_{d,cl}(x))$.

Furthermore, assume that the set of N_s number of samples are given as below

$$\mathscr{S} = \{(x, \check{F}_{d,cl}(x), \check{F}^{(2)}_{d,cl}(x))_1, \dots, (x, \check{F}_{d,cl}(x), \check{F}^{(2)}_{d,cl}(x))_{N_s}\}.$$

Note that $\mathscr{S}$ implicitly includes samples of the disturbance input and the state.

Now, using $\mathscr{S}$ we obtain the over-approximation

$$\tilde{\mathscr{F}} = \{\hat{P} \in \mathbb{R}^{2n \times 2n} | 0 \leq \hat{P} \leq I, \Delta V(x, \hat{P}) \leq 0, \forall x \in \mathscr{S}, d_\sigma \in \Delta_\sigma\}.$$

To find an element in $\tilde{\mathscr{F}}$, there exist efficient iterative techniques that are well-known as cutting-plane approaches. See e.g. [10, 47, 23]. In [33], the ACCPM [64, 120, 22] is employed in an optimization problem:

$$\hat{P}^{(i)} = \arg\min_{\hat{P}} \quad -\sum_{x \in \mathscr{S}_i} \log(-\Delta V(x, \hat{P})) - \log\det(I - \hat{P}) - \log\det(\hat{P}) \tag{4.16}$$

where i is the iteration index. If feasible, the log-barrier function in the first term guarantees the solution within $\tilde{\mathscr{F}}$ for which the negativity of the Lyapunov difference holds. The other two terms ensure $0 \leq \hat{P}^{(i)} \leq I$. The solution gives a Lyapunov function V based on the set of the samples $\mathscr{S}_i$ in the ith stage. On the other hand, if a solution does not exist, the set $\mathscr{F}$ is concluded to be empty.

The Verifier: The Lyapunov function candidate suggested by (4.16) may not guarantee asymptotic stability for all $x \in \bar{D}$ and $d_\sigma \in \Delta_\sigma$ since only the sampled space was considered. Therefore, in the next step, we need to verify the Lyapunov function candidate for the uncertain system. To do so, a mixed-integer quadratic program is solved based on the convex hull formulation of the PWA:

$$\max_{\mathrm{x}^j, \mathrm{u}^j, \mathrm{d}^j, \mu^j} \qquad \begin{bmatrix} \mathrm{x}^1 \\ \mathrm{x}^2 \end{bmatrix}^T \hat{P}^{(i)} \begin{bmatrix} \mathrm{x}^1 \\ \mathrm{x}^2 \end{bmatrix} - \begin{bmatrix} \mathrm{x}^0 \\ \mathrm{x}^1 \end{bmatrix}^T \hat{P}^{(i)} \begin{bmatrix} \mathrm{x}^0 \\ \mathrm{x}^1 \end{bmatrix} \tag{4.17}$$

subject to

$$Z_{\bar{D}}\mathrm{x}^0 \leq z_{\bar{D}}, \|\mathrm{x}^0\|_\infty \geq \epsilon \tag{4.18}$$

$$\mathrm{u}^j = \omega(\mathrm{x}^j) \tag{4.19}$$

$$Z_\sigma \mathrm{x}_\sigma^j \leq \mu_\sigma^j z_\sigma, Z_u \mathrm{u}_\sigma \leq \mu_\sigma^j z_u, |\mathrm{d}_{\sigma i}^j| \leq \mu_\sigma^j \bar{d}_{\sigma i}, \tag{4.20}$$

$$(1, \mathrm{x}^j, \mathrm{u}^j, \mathrm{d}^j, \mathrm{x}^{j+1}) = \sum_{\sigma=1}^{N_\sigma} (\mu_\sigma^j, \mathrm{x}_\sigma^j, \mathrm{u}_\sigma^j, \mathrm{d}_\sigma^j, A_\sigma \mathrm{x}_\sigma^j + B_\sigma \mathrm{u}_\sigma^j + \mu_\sigma^j c_\sigma + \mathrm{d}_\sigma^j) \tag{4.21}$$

$$\mu_\sigma \in \{0,1\}, \forall \sigma \in \{1, \ldots, N_\sigma\}, i \in \{1, \ldots, n\}, j \in \{0,1\}, \tag{4.22}$$

where a ball of radius ϵ around the origin is excluded from the set of states, and ϵ is chosen small enough in (4.18). This is due to Remark 10 and the fact that the numerical value of the objective becomes considerably small when approaching the origin. This makes the negativity of the objective too hard to verify around the origin. For more details in the implementation of the algorithm, we refer the reader to [33].

The system is given by (4.21) and (4.22). To define the piecewise system in a mixed-integer problem, similar to [33], we use the convex-hull formulation of piecewise model that is presented in [108]. However, to consider the uncertainty, we compose a slightly different system where we define extra variables to model the disturbance input.

Constraints (4.18), and (4.20) define the sets of the initial condition, the state, the control, and the disturbance inputs, respectively. Furthermore, the feedback control is implemented by (4.19).

To certify the closed-loop system as asymptotically stable, the optimal value returned by the MIQP (4.17) is required to be negative. Otherwise, the argument $(\mathrm{x}^{0*}, \mathrm{x}^{1*}, \mathrm{x}^{2*})$ of the optimal solution is added to the set of samples $\mathscr{S}$ as a counter-example.

4.5.3 Convergence of ACCPM

The convergence and complexity of the ACCPM for searching a quadratic Lyapunov function is discussed in [151, 33], where an upper bound is obtained for the number of steps taken until the algorithm exits.

Lemma 4. *Let $\mathscr{F}$ be a convex subset of $\mathbb{R}^{n\times n}$. Moreover, there exists $P_{center} \in \mathbb{R}^{n\times n}$ such that $\{P \in \mathbb{R}^{n\times n} | \|P - P_{center}\|_F \leq \epsilon\} \subset \mathscr{F}$, where Frobenius norm is used, and $\mathscr{F} \subset \{P \in \mathbb{R}^{n\times n} | 0 \leq P \leq I\}$. Then, the center cutting-plane algorithm concludes in at most $O(n^3/\epsilon^2)$ steps.*

Proof. See [151, 33] for the proof. □

Stability Analysis

Combining the uncertainty bounds in Section 4.4 and the Lyapunov-based verification results of this section, we are able to prove the following practical stability results of the closed-loop system.

Theorem 5. *Suppose that the MIQP (4.17) yields a negative optimal value. Let B_ϵ denote the set $\{x \in \mathbb{R}^n | \|x\|_\infty \leq \epsilon\}$, i.e., the ball of radius ϵ in infinity norm around the origin. Then the set B_ϵ is asymptotically stable for the closed-loop system (4.14). The largest sub-level set of V, i.e., $\{x \in \mathbb{R}^n \,|\, V(x) \leq c\}$ for some c, contained in $\bar{D}$ is a verified under-approximation of the real ROA.*

Proof. According to the conditions of the verifier, if the optimal value returned by the MIQP (4.17) is negative, we have effectively verified the following Lyapunov conditions:

$$V(0) = 0, \quad V(x) > 0, \quad \forall x \in \bar{D}\backslash\{0\}, \tag{4.23}$$

$$V(\check{F}_{d,cl}(x)) - V(x) < 0, \quad \forall x \in \bar{D}\backslash B_\epsilon, d \in \Delta_\sigma, \tag{4.24}$$

for the uncertain closed-loop system (4.14). By standard Lyapunov analysis for set stability [68, 76], the set B_ϵ, which is the ball of radius ϵ in infinity norm around the origin, is asymptotically stable for system (4.14). Furthermore, any sub-level set of $V(x)$, i.e., $\{x \in \mathbb{R}^n \,|\, V(x) \leq c\}$ for some c, contained in $\bar{D}$ is contained in the ROA of B_ϵ. □

Remark 10. *Due to the existence of a non-zero additive uncertainty bound, one cannot expect convergence to the origin precisely. This issue is addressed by providing convergence guarantee to a small neighborhood of the origin, i.e., B_ϵ. By collecting enough samples around the origin, a local approximation of the system is obtained by the mode $\sigma = 0$ of the identified system, whose domain includes the origin, while d_σ can be made arbitrarily small as $x_k \to 0$. By doing so, we can make ϵ in Theorem 5 arbitrarily small and the stability result is practically equivalent to the asymptotic stability of the origin. Alternatively, one can assume that there exists a local stabilizing controller that one can switch to when entering a small neighborhood of the origin. In this case, asymptotic stability can be achieved.*

4.6 Numerical Results

To validate the proposed piecewise learning and verification technique we implemented the approach on the pendulum system as (2.32) and the dynamical vehicle system [125]. Moreover, we compared the results with other techniques presented in the literature. To make a fair comparison, we have taken the parameters of the system from [31]. We performed all the simulations in Python 3.7 on a 2.6 GHz Intel Core i5 CPU.

4.6.1 Pendulum System

For the pendulum system, we discuss the simulation results in three sections. In the first section, we will explain the procedure of identifying the uncertain PWA model with a piecewise feedback control. In the second section, we verify the closed-loop uncertain system and obtain an ROA in $\bar{D}$. In the third section, we will present the comparison results.

Identify and Control

Control objective is to stabilize the pendulum at the top equilibrium point given by $x_{\text{eq}} = (0, 0)$. First, we start with learning a piecewise model together with the uncertainty bounds, and the feedback control. For this purpose, we sample the system, and update our model as discussed in section 4.3.1. We set the sampling time as $h = 5\text{ms}$. Accordingly, the value function and the control rule are updated online as in section 4.3.3. Then, to verify the value to be decreasing within each mode, it only remains to calculate the uncertainty bounds using the results obtained in section 4.4.

To make a visualization of the nonlinearity in the pendulum system (2.32) possible, we portray the second dynamic assuming $u = 0$ in Fig. 4.4, where the first dynamic is only linear. The procedure of learning is illustrated through several stages in Fig. 4.5. In the first column from the left, we illustrated the estimations only for the second dynamic with $u = 0$ to be comparable to Fig. 4.4. Accordingly, it can be observed that the system identifier is able to closely approximate the nonlinearity with a piecewise model.

It should be noted, the learning is started from the mode containing the origin in its domain, that we label by $\sigma = 0$. As we collect more random samples in Υ_0, we can effectively decrease the uncertainty of the model around the origin, and obtain a local controller. Then, we gradually expand the areas sampled to train the rest of the pieces in the PWA model.

Remark 11. *It is worth mentioning that the model obtained and the uncertainty bounds can be further improved by continuing the sampling. In this implementation, we perform sampling only until the uncertainty bound obtained allows us to verify a decreasing value function for each piece of the PWA system.*

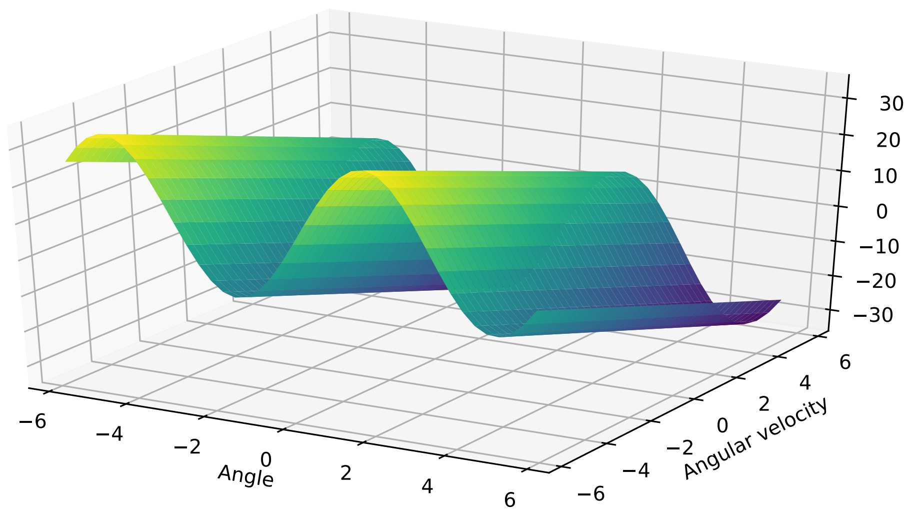


Figure 4.4: A view of the second dynamic of pendulum system (2.32) assuming $u = 0$ that is $f_2(x_1, x_2)$.

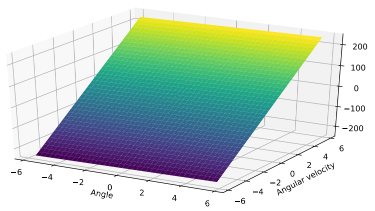
Verification

Having the system identified and the feedback control, we can apply the verification algorithm based on MIQP problem. As done in [33], we implemented the learner in CVXpy [42] with MOSEK [6] solver, and the verifier in Gurobi 9.1.2 [67].

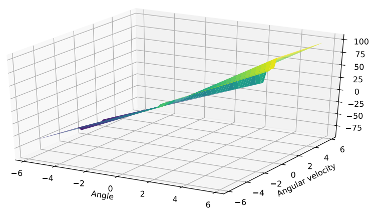
We choose $\bar{D}$ such that x_1 and $x_2 \in [-6, 6]$. To verify the system, we ran the algorithm and obtained a matrix $\hat{P}$ that characterizes the Lyapunov function as in (4.15).

$$\hat{P} = \begin{bmatrix} 0.69371067 & 0.02892586 & 0.1944487 & 0.05196313 \\ 0.02892586 & 0.26941371 & 0.02718769 & -0.21348358 \\ 0.1944487 & 0.02718769 & 0.69518109 & 0.05041737 \\ 0.05196313 & -0.21348358 & 0.05041737 & 0.33469316 \end{bmatrix}.$$

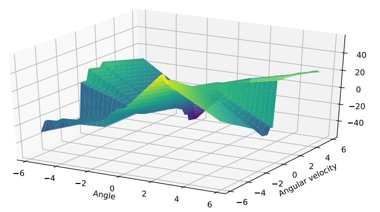
The largest level set of the associated Lyapunov function in $\bar{D}$ is pictured in Fig. 4.8a as the the ROA of the closed-loop system. Moreover, we illustrate different trajectories of the controlled system that confirms the verified Lyapunov function by constructing an ROA around the origin.



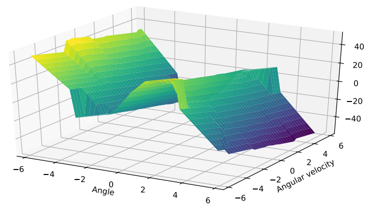
(a)



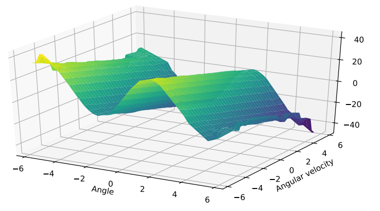
(b)



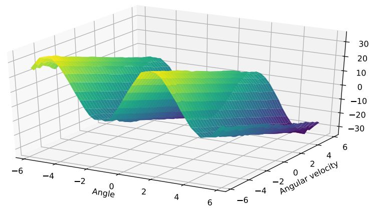
(c)



(d)



(e)



(f)

Figure 4.5: The procedure for learning the dynamics by the PWA model is illustrate step-by-step that shows the convergence of the identifier. Subfigures (a)-(f) show the improvement of estimations of $f_2(x_1, x_2)$, as the number of samples increases.

Comparison Results

To highlight the merits of the proposed piecewise learning approach, we compare the ROA obtained by different approaches in the literature. [31] proposed a Neural Network (NN) Lyapunov function for stability verification. According to [31], the comparison done on the pendulum system with LQR and Sum of Squares (SOS) showed noticeable superiority of the NN-based Lyapunov approach. Following the comparison results from [31], we compare the ROA obtained by our approach with NN, SOS, and LQR techniques in Fig. 4.8b. Clearly, the ROA obtained by the piecewise controller with the non-monotonic Lyapunov function is considerably larger than the ones obtained by NN, SOL, and LQR algorithms as shown in [31].

4.6.2 Dynamic Vehicle System with Skidding

In this section, to better demonstrate the merits of the algorithm proposed, we implemented the approach on a more complex system. The kinematic model of the vehicle system does not consider the real behavior of the system at high speed where skidding is possible. Therefore, [125] proposed a more realistic dynamical model of the vehicle which is implemented in this chapter.

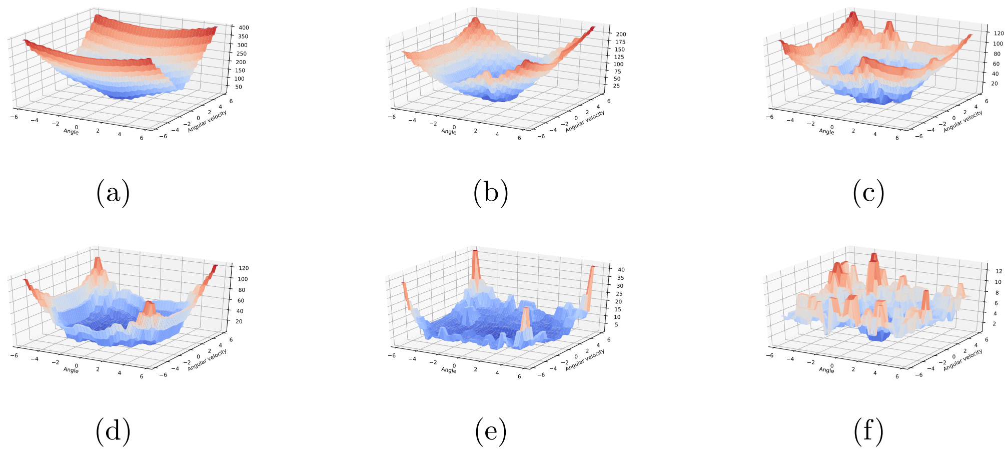
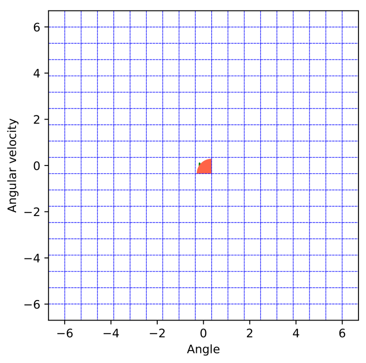


Figure 4.6: To better illustrate the learning procedure, the step-by-step results of the uncertainty bound corresponding to the results in Fig. 4.5 are provided. It is evident that error bound is improved in every step.

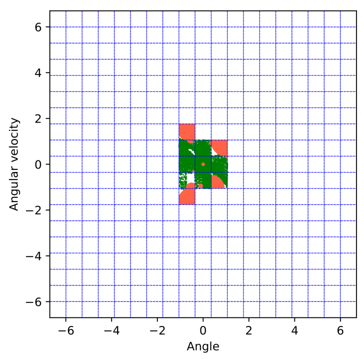
According to [125], we present the dynamic model of the vehicle implemented. Let us define the states x and y as the coordinate of the center of gravity in the 2D space, θ as the orientation of the vehicle, v_y as the lateral velocity, and r as the rate of the orientation. Moreover, the input of the system is given by the front-wheel angle δ_f. Then, by assuming a constant longitudinal velocity v_x, the dynamical model of the vehicle can be written as

$$\begin{aligned}
\dot{v}_y &= -\frac{C_{\alpha f}\cos\delta_f + C_{\alpha r}}{m v_x} v_y + \frac{-L_f C_{\alpha f}\cos\delta_f + L_r C_{\alpha r}}{I_z v_x} r + \frac{C_{\alpha f}\cos\delta_f}{m}\delta_f,\\
\dot{r} &= \left(\frac{-L_f C_{\alpha f}\cos\delta_f + L_r C_{\alpha r}}{m v_x} - v_x\right) v_y - \frac{L_f^2 C_{\alpha f}\cos\delta_f + L_r^2 C_{\alpha r}}{I_z v_x} r + \frac{L_f C_{\alpha f}\cos\delta_f}{I_z}\delta_f,\\
\dot{x} &= v_x\cos\theta - v_y\sin\theta,\\
\dot{y} &= v_x\sin\theta + v_y\cos\theta,\\
\dot{\theta} &= r,
\end{aligned}$$

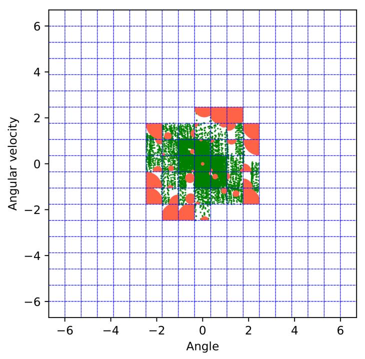
where C_{xf}, and C_{xr} denote the cornering stiffness coefficients of the front and rear wheels. Moreover, the distance of the center of gravity from the front and rear wheels are given by L_f, and L_r.



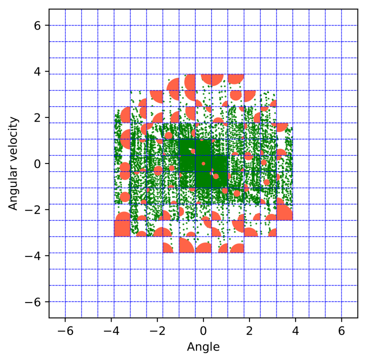
(a)



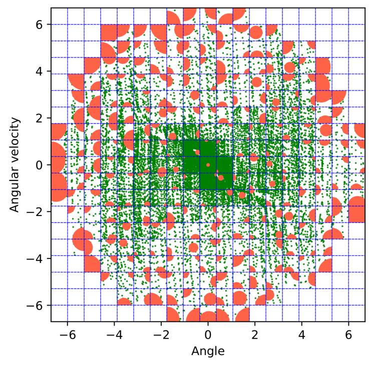
(b)



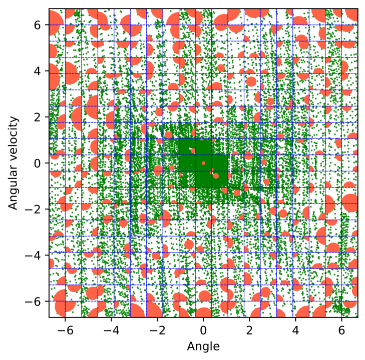
(c)



(d)



(e)

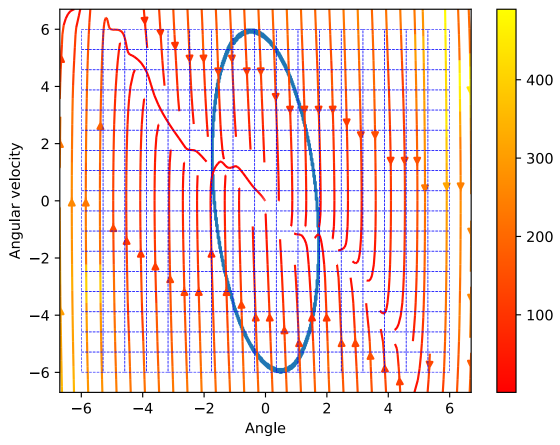


(f)

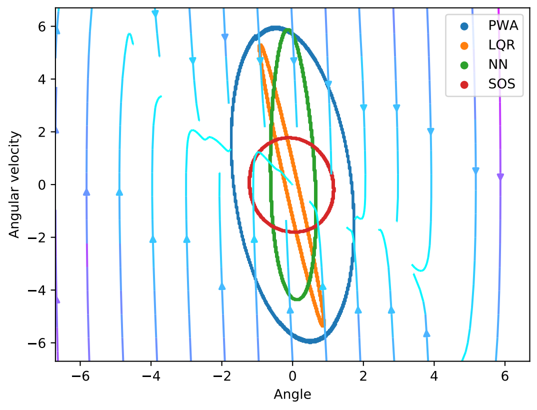
Figure 4.7: The step-by-step results of the sampling procedure and the sample gap obtained are provided that correspond to the results in Fig. 4.5 and 4.6. It is evident that by acquiring more samples over different steps and expanding the learning area, the sample gaps are decreased effectively.

Identify and Control

Control objective is to minimize the distance of the vehicle from the goal point $(x, y)_{\text{goal}} = (70, 70)$ in the 2D map. To achieve the objective, we run the vehicle from some random initial position and yaw values. Then, identification and control procedures are done in a loop through different episodes. The longitudinal velocity of the vehicle is assumed to be constant in this system similar to [125]. Therefore, to minimize the cost given by the control objective, the vehicle converges to some circular path around the goal point, which is indeed the optimal path for the problem defined. Fig. 4.5, contains the simulation results within an episode of learning, including the state and control signals, prediction error for



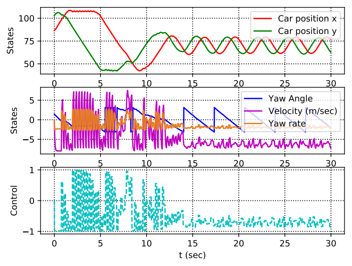
(a)

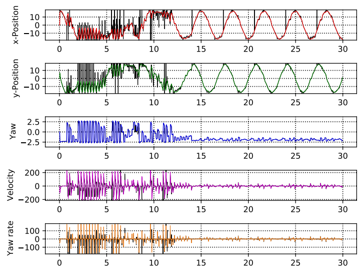


(b)

Figure 4.8: (a). The obtained ROA of the closed loop PWA system is illustrated for x_1 and $x_2 \in [-6, 6]$. The uniform grids denote the modes of the PWA system. Multiple trajectories of the system are shown in a phase portrait where colormap represents the magnitude in the vector field.
(b). The comparison results for ROA of the closed loop system is illustrated for x_1 and $x_2 \in [-6, 6]$, together with the trajectories of the system. The comparison results for LQR, NN, and SOS are taken from [31].

each state, the value function and the modes.


(a)


(b)

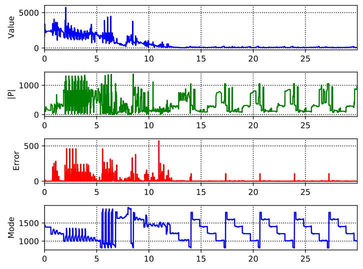

(c)

Figure 4.9: (a). The state and control signals are illustrated within an episode of learning. It can be clearly seen from the position signals that the vehicle is able to minimize the distance from the goal point and converge to a circular path around the goal point after some time of learning.
(b). The graph denotes the evolutions of the value function, the norm of the control parameters for the active mode, the prediction error, and the active mode of the piecewise model that correspond to the results in Fig 4.9a. It can be seen that the value function learned is minimized.
(c). Corresponding to Fig. 4.9a and 4.9c, the prediction results of the learned model is compared with the original system within an episode of learning. It can be observed that the prediction signals shown by the black lines can match the ones obtained from the original dynamic.

4.6.3 Comparison of Runtime Results

To analyze the computational aspects of the proposed technique, we provide the runtime results while learning the dynamics and obtaining the control for both examples implemented. The proposed framework is considered as an online technique. Hence, in the applications, the computational complexity of the realtime identification and control becomes more important. Therefore, we here focus in the complexity of the online learning procedure rather than the verification technique which can be done offline. Fig. 4.10 includes the runtime results separately for the identification and control units. Accordingly, the identifier and the controller can be updated in at most $1ms$ and $20ms$ respectively. Accordingly, it can be observed that for the higher dimensional system with also larger number of partitions the computations still remain in a tractable size that can allow real-time applications, considering the nature of the systems.

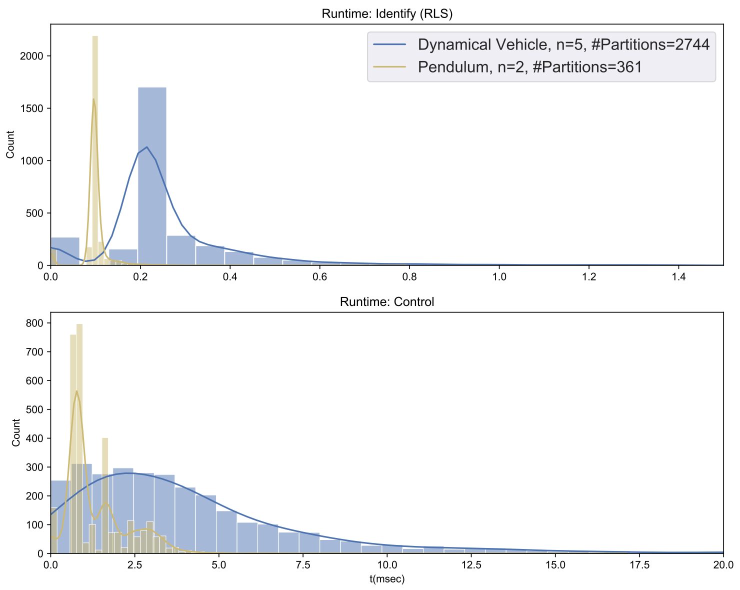


Figure 4.10: A comparison of the runtime results for the identification and control procedures separately is given for the implemented examples.

4.7 Conclusion

For regulating nonlinear systems with uncertain dynamics, a piecewise nonlinear affine framework was proposed in which each piece is responsible for learning and controlling over a partition of the domain locally. Then, in a particular case of the proposed framework, we focused on learning in the form of the well-known PWA systems, for which we presented an optimization-based verification approach that takes into account the estimated uncertainty bounds. We used the pendulum system as a benchmark example for the numerical results. Accordingly, an ROA resulting from the level set of a learned Lyapunov function is obtained. Furthermore, the comparison with other control approaches in the literature illustrates a considerable improvement in the ROA using the proposed framework. As another example, we implemented the presented approach on a dynamical vehicle system with considerably higher number of partitions and dimension. The results demonstrated that the approach can scale efficiently, hence, can be potentially implemented on more complex real-world problems in realtime.

Chapter 5

Structured Online Learning for Low-Level Control of Quadrotors

The results presented in this chapter are submitted to [55].

5.1 Introduction

Quadrotors have received enormous attention because of their efficacy in various applications. Nowadays, quadrotors are produced at a reasonable cost and in different sizes that justify their increasing deployment in new environments. Their applications can range from industry to everyday life and they can reach and operate in situations that may be expensive or dangerous for humans to enter. In response to the high demand, researchers have developed extensive approaches over the past decade to control quadrotors for complicated tasks and high-precision acrobats.

For an effective fly of the quadrotor, there are two levels of controls involved: the low-level control that is required for the stability of the hovering position, and the high-level control that provides a sequence of setpoints as commands to achieve a particular objective. Even though effective controllers have been already implemented on quadrotors for various objectives, the design and tuning procedure of such controllers still requires a considerable amount of expert knowledge about the governing dynamics and experiments to determine the exact system parameters. This becomes even more challenging considering the underactuated and rapid dynamics of the quadrotor. Hence, having successful learning approaches are extremely desired, because it can automate and accelerate the procedure of reaching the flying state with minimum knowledge of the dynamics.

In [45], the main challenges in the application of RL approaches are highlighted. Among RL approaches, MBRL techniques are sometimes preferred over model-free methods, because of their effectiveness in learning from limited data and their computationally tractable properties, which allow real-time inference of the policy. In contrast, direct RL approaches, on the other hand, usually require a large amount of data and a long time of training [44]. Therefore, in this chapter, we will only discuss MBRL strategies that provide an opportunity for learning in a few tries.

There are plenty of MBRL approaches implemented on aerial vehicles that use demonstrations led by humans to collect data and realize a model that can best predict the future state for a given action [39, 12]. Moreover, MBRL techniques are effectively implemented on quadrotors that assume an on-board stable low-level controller to learn the high-level control [101, 1, 15], which is not in the scope of this chapter. Hence, in this chapter, we are interested in obtaining a low-level control that requires no initial controller and no knowledge of the system parameters.

Recently, the authors of [91] suggested an interesting approach in learning a low-level controller by running an MBRL approach on a real nano-quadrotor, best known as Crazyflie. In this approach, they train a neural network by collecting experimental data, which is then used to run a random-shooter MPC on a graphic processing unit (GPU) to establish a real-time controller. The approach can successfully reach a hovering position in a few tries. However, the need for GPU to learn a low-level control can be seen as a limiting factor of its implementation.

In this chapter, similar to [91], we use the flight data obtained along random open-loop trajectories to establish an initial model, with no need for any expert demonstrations. However, in a different approach than [91], once an initial model together with the corresponding controller is obtained, we switch to learning in a closed-loop form to refine the model and the performance achieved. To verify the method, we acquire data from the nonlinear model of the quadrotor, treated as a black box.

For a practical framework, the MBRL approach has to be data-efficient while being fast enough to allow real-time implementation. Unlike [91], our approach does not demand a lot of computational efforts considering that we learn the system in terms of a limited number of bases and accordingly obtain a feedback control rule. Hence, it can be used as a lightweight alternative in implementations. Moreover, in [91], the objective is to reach the hovering position, whereas in this chapter, in addition to the attitude control, we also control the position. This means the quadrotor simultaneously learns to reach and stay at a given point in the 3D space. This will also minimize the instances where the quadrotor slides out of the training environment.

For learning purpose, we implemented the SOL approach proposed in [52] with a RLS algorithm that is well known for its high efficiency in online applications. Successful applications of RLS can be found in [103, 167, 170, 147]. In this chapter, as an alternative to the neural network approach, we used a system structured in terms of a library of bases. Accordingly, by sampling the input and state, we employ RLS to update the system model. Then, by exploiting the structure assumed in the model and a quadratic parametrization of the value function in terms of the same set of bases, we obtain a matrix differential equation to update the controller that can be efficiently integrated online. An extension of SOL is presented in [53] for tracking unknown systems that can be also implemented on the quadrotor. However, in this chapter, we will only focus on the attitude-position control.

The rest of the chapter is organized as follows. In section 5.2, we will introduce the nonlinear model of the quadrotor. In Section 5.3, we will highlight the SOL framework, together with the practical considerations required for a real-time implementation on the quadrotor. Section 5.4 contains the simulation results.

5.2 Quadrotor Model

The nonlinear dynamics of the quadrotor can be written as

$$\begin{aligned}
\dot{y} &= v, \\
\dot{v} &= \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{1}{m} R \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix}, \\
\dot{R} &= RQ(\omega), \\
\dot{\omega} &= J^{-1}(-\omega \times J\omega + \tau),
\end{aligned} \tag{5.1}$$

where the states include the 3D position y, the linear velocity v of center of gravity in the inertial frame, the rotation matrix R, and the angular velocity ω in the body frame with respect to the inertial frame. It should be noted that the third equation is written in a matrix form, where R takes value in the special orthogonal group $\mathrm{SO}(3) = \{R \in \mathbb{R}^{3\times 3} | R^{-1} = R^T, \det(R) = 1\}$. Accordingly, the attitude of the quadrotor $\chi = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}$ can be extracted from R at any time instance, which contains the roll, pitch, and yaw angles, respectively.

Moreover, the gravity acceleration, the body mass, and the inertia matrix are given by g, m, and

$$J = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix}.$$

The skew-matrix Q is composed by the angular velocity $\omega = \begin{bmatrix} p & q & r \end{bmatrix}^T$ as

$$Q(\omega) = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}.$$

The inputs of this system are given by the moments in the body frame

$$\tau = \begin{bmatrix} C_T d(-\bar{\omega}_2^2 - \bar{\omega}_4^2 + \bar{\omega}_1^2 + \bar{\omega}_3^2) \\ C_T d(-\bar{\omega}_1^2 + \bar{\omega}_2^2 + \bar{\omega}_3^2 - \bar{\omega}_4^2) \\ C_D(\bar{\omega}_2^2 + \bar{\omega}_4^2 - \bar{\omega}_1^2 - \bar{\omega}_3^2) \end{bmatrix},$$

and the thrust

$$T = C_T(\bar{\omega}_1^2 + \bar{\omega}_2^2 + \bar{\omega}_3^2 + \bar{\omega}_4^2)$$

generated in the body frame by the rotors, where $\bar{\omega}_i$, d, C_T, and C_D denote the rotational speed of each rotor, the arm length, the lift, and the drag coefficients of the propellers, respectively.

The system is usually actuated by four DC motors that are controlled by Pulse-Width Modulation (PWM) signals. To convert the PWM values to the rotational speed RPM, we use

$$\bar{\omega}_i = \eta_1 u_i + \eta_2,$$

where η_1 and η_2 are the coefficients specified for any motor, and $i \in \{1, 2, 3, 4\}$. Accordingly, in the learning process we will consider u_i as the inputs of the quadrotor system.

5.3 Structured Online Learning with RLS Identifier on Quadrotor

In what follows, we discuss different stages of the proposed learning procedure in detail. Moreover, we present considerations to be taken into account in the implementation together with the computational properties of the proposed framework.

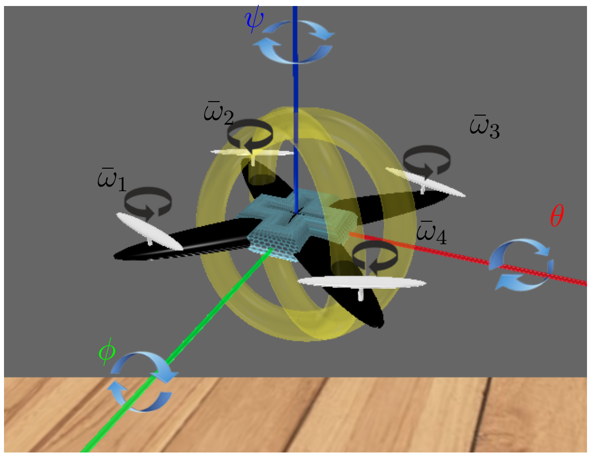


Figure 5.1: A video of the training procedure can be found at `https://youtu.be/Q08Q183qKFM`, where the objective is to learn to fly and reach to the reference position and yaw.

5.3.1 Learning Procedure

The learning procedure is done in the following order: In the first stage (pre-run), we run the quadrotor in an open-loop form with almost equal PWM values for every motor. These values are perturbed slightly in a random way that provides a probing input signal to collect diverse samples from the system. The design and importance of such probing signals are well studied in MBRL techniques, and in system identification algorithms in particular. See, for instance, [126, 83]. The data is used to establish an initial model to start online learning.

In the second stage, we implement the learning in a closed-loop form by using the initial model obtained in the pre-run. In the control loop, at any time step t_k, the samples of the states are acquired and a set of bases are evaluated accordingly. Next, by using RLS algorithm, the system model is updated. Then, the measurements and the most recent model coefficients are used to update a value function, which is required to calculate the control value for the next step t_{k+1}.

Remark 12. *Considering that we assume no knowledge about the system coefficients, initial learning is done through several unsafe tries similar to [91]. Hence, the first runs of the system may show poor control performance or instability, which requires a safe training environment and/or a resetting mechanism that allows safe crashes.*

Two blocks are responsible for the control and model update in a loop that will be discussed next.

Control Update

Consider the quadrotor model (5.1), where we compose the state vector by using the position, the velocity, the attitude, and the angular velocity as

$$x := \begin{bmatrix} y & v & \chi & \omega \end{bmatrix},$$

where $x \in D \subset \mathbb{R}^n$, and $u \in \Omega \subset \mathbb{R}^m$ with $n = 12$ and $m = 4$. As mentioned, angles χ can be obtained using the state R.

The cost function to be minimized along the trajectory started from the initial condition $x_0 = x(0)$ is considered in the following linear quadratic form

$$J(x_0, u) = \lim_{T \to \infty} \int_0^T \mathrm{e}^{-\gamma t} \left(x^T Q_0 x + u^T R_0 u \right) \mathrm{dt}, \tag{5.2}$$

where $Q_0 \in \mathbb{R}^{n \times n}$ is positive semi-definite, $\gamma \geq 0$ is the discount factor, and $R_0 \in \mathbb{R}^{m \times m}$ is a diagonal matrix with only positive values, given by design criteria.

For the closed-loop system, by assuming a feedback control law $u = \nu(x(t))$ for $t \in [0, \infty)$, the optimal control is given by

$$\nu^* = \quad \arg \min_{u(\cdot) \in \Gamma(x_0)} J(x_0, u(\cdot)), \tag{5.3}$$

where Γ is the set of admissible controls.

For now, we assume that we can approximate the dynamics of (5.1) in terms of some differentiable bases such as polynomial and trigonometric functions, where the system identification approach will be discussed later. Accordingly, (5.1) is rewritten as

$$\dot{x} = W\Phi(x) + \sum_{j=1}^{m} W_j \Phi(x) u_j, \tag{5.4}$$

where W and $W_j \in \mathbb{R}^{n \times p}$ are the matrices of the coefficients obtained for $j = 1, 2, \ldots, m$, and $\Phi(x) = [\phi_1(x) \quad \ldots \quad \phi_p(x)]^T$ is the set of chosen bases.

In what follows, without loss of generality, the cost defined in (5.2) is transformed to the space of bases $\Phi(x)$, that is

$$J(x_0, u) = \lim_{T \to \infty} \int_0^T \mathrm{e}^{-\gamma t} \left(\Phi(x)^T \bar{Q}_0 \Phi(x) + u^T R_0 u \right) \mathrm{dt}, \tag{5.5}$$

where $\bar{Q}_0 = \mathrm{diag}\left([Q_0], [\mathbf{0}_{(p-n)\times(p-n)}]\right)$ is a block diagonal matrix that contains all zeros except the first block Q_0, which corresponds to the linear basis x.

Then the corresponding HJB equation can be written by the Hamiltonian defined as

$$\begin{aligned}
&-\frac{\partial}{\partial t}(\mathrm{e}^{-\gamma t}V) \\
&\quad = \min_{u(\cdot)\in\Gamma(x_0)} \{H = \mathrm{e}^{-\gamma t}\big(\Phi(x)^T\bar{Q}_0\Phi(x) + u^T R_0 u\big) \\
&\qquad\qquad\qquad + \mathrm{e}^{-\gamma t}\frac{\partial V}{\partial x}^T (W\Phi(x) + \sum_{j=1}^{m} W_j\Phi(x)u_j)\}.
\end{aligned} \tag{5.6}$$

In general, there exists no analytical approach that can solve such a partial differential equation and obtain the optimal value function. However, it has been shown in the literature that approximate solutions can be computed by numerical techniques.

Assume the optimal value function in the following form

$$V = \Phi(x)^T P\Phi(x), \tag{5.7}$$

where P is symmetric. Then the Hamiltonian is given by

$$\begin{aligned}
H =& \mathrm{e}^{-\gamma t}(\Phi(x)^T\bar{Q}_0\Phi(x) + u^T R_0 u) \\
&+ \mathrm{e}^{-\gamma t}\Phi(x)^T P\frac{\partial\Phi(x)}{\partial x}\left(W\Phi(x) + \sum_{j=1}^{m} W_j\Phi(x)u_j\right) \\
&+ \mathrm{e}^{-\gamma t}\left(\Phi(x)^T W^T + \sum_{j=1}^{m} u_j^T\Phi(x)^T W_j^T\right)\frac{\partial\Phi(x)}{\partial x}^T P\Phi(x).
\end{aligned}$$

Moreover, based on the diagonal structure of R_0, the quadratic term of u is rewritten in terms of its components, where $r_{0j} \neq 0$ is the jth component on the diagonal of matrix R_0. To minimize the resulting Hamiltonian we need

$$\begin{aligned}
\frac{\partial H}{\partial u_j} &= 2r_{0j}u_j + 2\Phi(x)^T P\frac{\partial\Phi(x)}{\partial x}W_j\Phi(x) \\
&= 0, \qquad j = 1, 2, \ldots, m.
\end{aligned} \tag{5.8}$$

Hence, the jth optimal control input is obtained as

$$u_j^* = -\Phi(x)^T r_{0j}^{-1} P\frac{\partial\Phi(x)}{\partial x}W_j\Phi(x). \tag{5.9}$$

Based on [52], the following update rule can be obtained by plugging in the optimal control in the Hamiltonian.

$$\begin{aligned}-\dot{P} =&\bar{Q}_0 + P\frac{\partial \Phi(x)}{\partial x}W + W^T\frac{\partial \Phi(x)}{\partial x}^T P - \gamma P \\ &- P\frac{\partial \Phi(x)}{\partial x}\left(\sum_{j=1}^{m} W_j\Phi(x)r_{0j}^{-1}\Phi(x)^T W_j^T\right)\frac{\partial \Phi(x)}{\partial x}^T P.\end{aligned} \tag{5.10}$$

In a standard optimal control approach, this equation has to be solved backward in time, which assumes complete knowledge of the system, i.e., W and W_j along the time horizon. However, in this chapter, we are interested in the learning problem, where the system model may not be known initially. Therefore, we propagate the obtained differential equation in the forward direction. This will provide an opportunity to update our estimation of the system dynamics online at any step together with the control update.

In this approach, we run the system from some $x_0 \in D$, then solve the matrix differential equation (5.10) along the trajectories of the system. Different solvers are already developed that can efficiently integrate differential equations. Although the solver may take smaller steps, we only allow the measurements and control update at time steps $t_k = kh$, where h is the sampling time and $k = 0, 1, 2, \dots$. For solving (5.10) in continuous time, we use the LSODA solver [74], where the weights and the states in this equation are updated by a system identification algorithm and the measurements x_k at each iteration of the control loop, respectively. A recommended choice for P_0 is a matrix with components of zero or very small values.

The differential equation (5.10) also requires evaluations of $\partial\Phi/\partial x_k$ at any time step. Since the bases Φ are chosen beforehand, the partial derivatives can be analytically calculated and stored as functions. Hence, they can be evaluated for any x_k in a similar way as Φ itself. By solving (5.10), we can calculate the control update at any time step t_k according to (5.9). Although at the very first steps of learning, control is not expected to take effective steps toward the control objective, it can help in the exploration of the state space and gradually improve by learning more about the dynamics.

Model Update

In the previous step, we considered a given structured nonlinear system as in (5.4). Therefore, having the control and state samples of the system, we need an algorithm that updates the estimation of system weights. As studied in [25, 78], SINDy is a data-efficient tool to

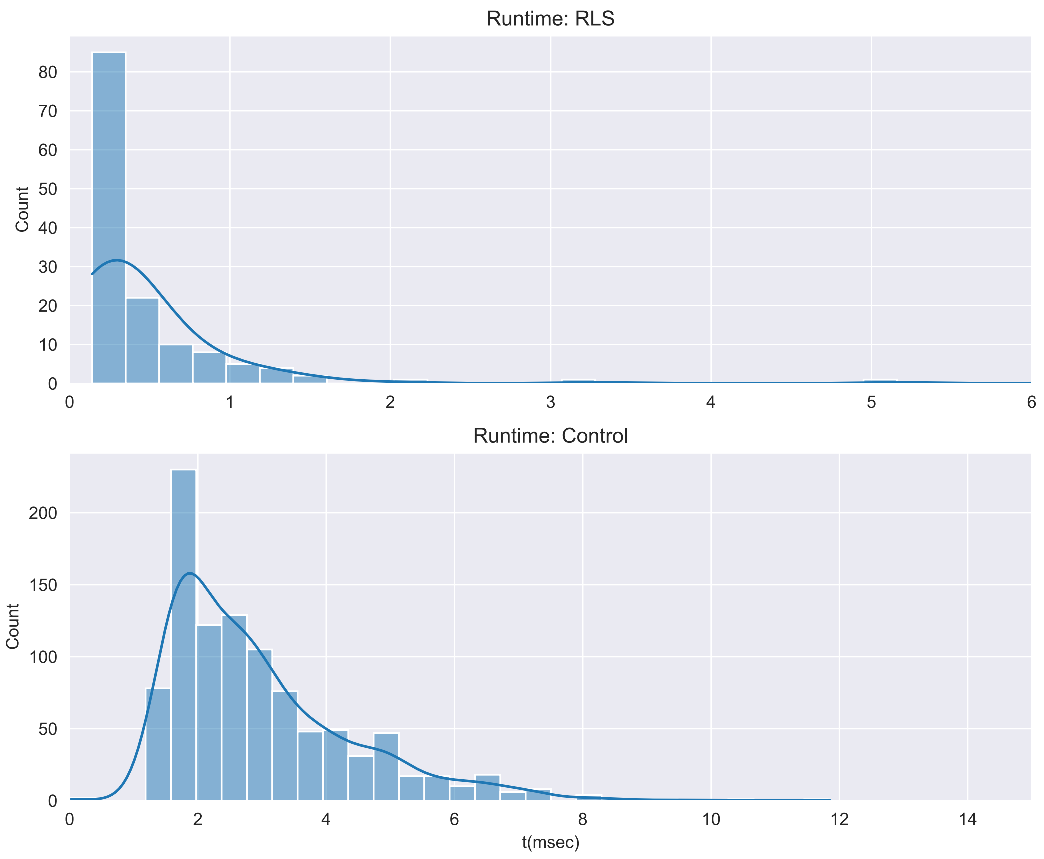


Figure 5.2: The histogram of the runtime of the identification and the control algorithms.

extract the underlying sparse dynamics of the sampled data. In this approach, along with the identification, the sparsity is also promoted in the weights by minimizing

$$[\hat{W} \quad \hat{W}_1 \dots \hat{W}_m]_k = \arg\min_{\bar{W}} \quad \|\dot{X}_k - \bar{W}\Theta_k\|_2^2 + \lambda \|\bar{W}\|_1, \tag{5.11}$$

where k is the time step, $\lambda > 0$, and Θ_k includes a matrix of samples with the columns of

$$\Theta_k{}^s = [\Phi^T(x^s) \quad \Phi^T(x^s)u_1{}^s \quad \dots \quad \Phi^T(x^s)u_m{}^s]_k^T,$$

for sth sample. In the same order, $\dot{X}$ keeps a table of sampled state derivatives.

Updating $\hat{W}_k$ based on a history of samples may not be favored as the number of samples needed tends to be large. Especially, real-time implementations may not be possible

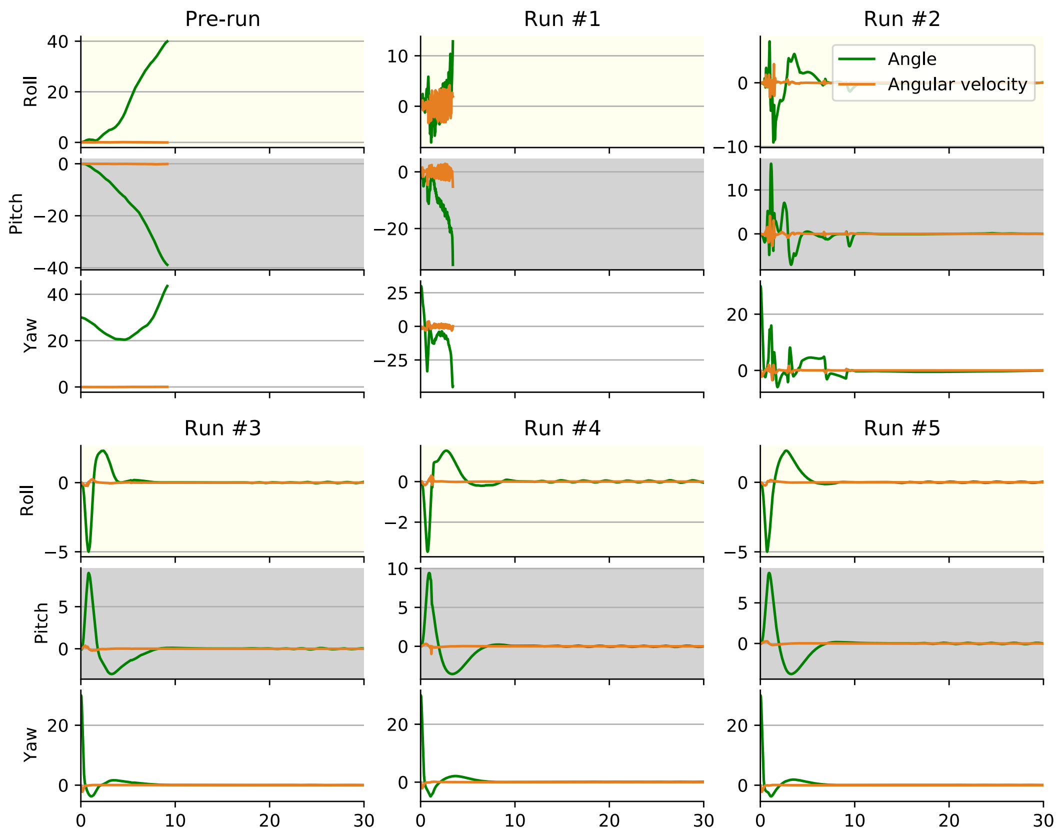


Figure 5.3: The attitude control results in the learning procedure illustrated in different runs.

because of the latency caused by the computations. There exist other techniques that can be alternatively used in different situations, such as neural networks, nonlinear regression, or any other function approximation and system identification methods. For real-time control applications, considering the linear dependence on the system weights in (5.4), one may choose RLS algorithm that only uses the latest sample of the system and $\hat{W}_{k-1}$, hence will run considerably faster.

For this reason, in the application of SOL on the quadrotor we employ the RLS algorithm. Moreover, to improve the runtime of the identification we only choose the linear and the constant bases. Considering that the quadrotor usually operates around the hovering situation such approximation will still be able to preserve the required properties of the system. However, for better results, one can add higher-order polynomial in the library as shown in [52].

In the implementations of SOL done in this chapter, we compare the prediction error $\dot{e}_k = \|\dot{x}_k - \hat{\dot{x}}_k\|$ with the average $\bar{\dot{e}}_k = \sum_{i=1}^{k} \dot{e}_k/k$. Hence, if the condition $\dot{e}_k > \eta\bar{\dot{e}}_k$ holds we use that sample to update the model, where the constant $0 < \eta < 1$ adjusts the threshold. Choosing smaller values of η will increase the rate of adding samples to the database.

5.3.2 Asymptotic Convergence with Uncertain Dynamics

Having arranged the identifier and controller, it only remains us to consider the model uncertainty's effect on the asymptotic convergence to the equilibrium point. Assume the system structured as

$$\dot{x} = W\Phi + \sum_{j=1}^{m} W_j \Phi \hat{u}_j + \epsilon, \tag{5.12}$$

where $\hat{u}_j = -\Phi^T R^{-1} \hat{P} \Phi_x \hat{W}_j \Phi$ is the feedback control rule obtained based on the estimation of the system $(\hat{W}, \hat{W}_j)$. Moreover, ϵ is the bounded approximation error in D. By assuming $W = \hat{W} + \tilde{W}$ and $W_j = \hat{W}_j + \tilde{W}_j$, this can be rewritten as

$$\dot{x} = \hat{W}\Phi + \sum_{j=1}^{m} \hat{W}_j \Phi \hat{u}_j + \Delta(t), \tag{5.13}$$

where unidentified dynamics are lumped together as $\Delta(t)$. By the assumption that the feedback control u_j is bounded in D, we have $\|\Delta(t)\| \leq \bar{\Delta}$. For asymptotic convergence, and also promote the robustness of the controller, the effect of the uncertainty should be taken into account. Hence, we use an auxiliary vector ρ to get

$$\begin{aligned}\dot{x} &= \hat{W}\Phi + \sum_{j=1}^{m} \hat{W}_j \Phi \hat{u}_j + \Delta(t) + \rho - \rho \\ &= \hat{W}_\rho \Phi + \sum_{j=1}^{m} \hat{W}_j \Phi \hat{u}_j + \Delta(t) - \rho,\end{aligned}$$

where assuming that Φ also includes the constant basis, we adjusted the corresponding column in the system matrix to get $\hat{W}_\rho$. In the case $\bar{\Delta} = 0$, the controller $\hat{u}$ can be obtained such that the closed system is locally asymptotically stable. For the case $\bar{\Delta} > 0$, although the system will stay stable for small enough $\bar{\Delta}$, it may not asymptotically converge to zero. Then, similar to [168, 136], we obtain ρ as below to help to slide the system state to zero

$$\rho = \int_0^t [k_1 x(\tau) + k_2 \text{sign}\big(x(\tau)\big)]\mathrm{d}\tau,$$

where k_1 and k_2 are positive scalars. It can be shown that over time $\|\Delta(t) - \rho\| \to 0$, and hence the system will asymptotically converge to the origin.

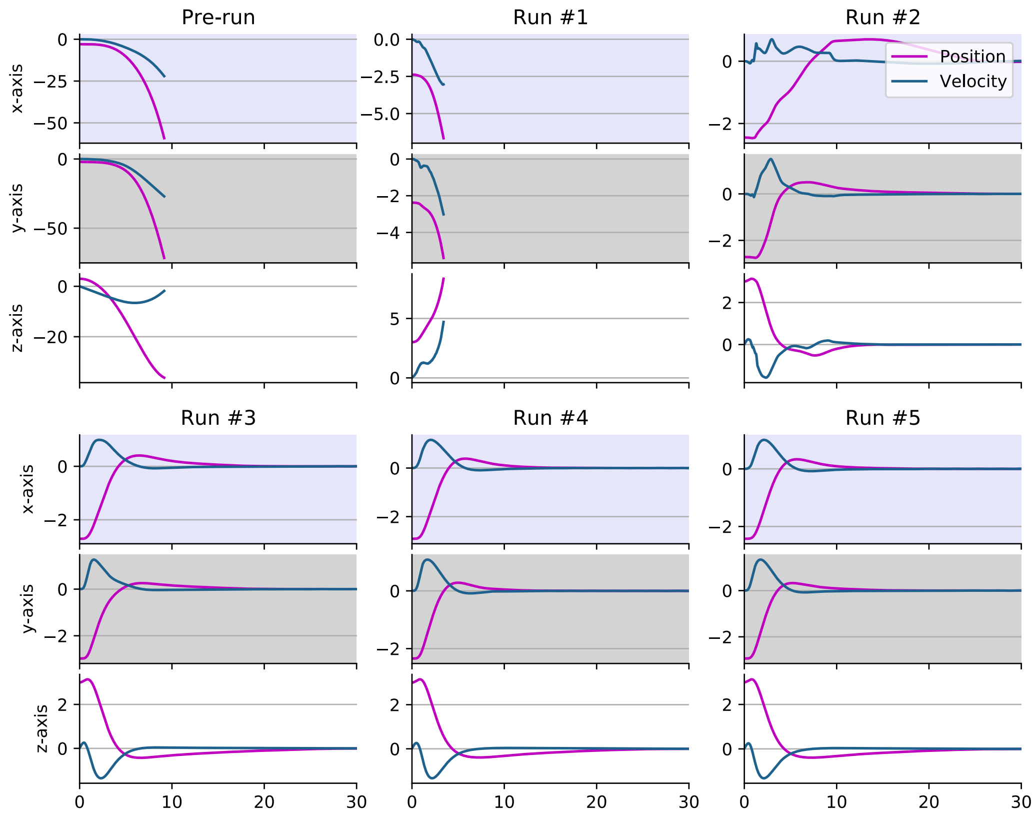


Figure 5.4: The position control results in the learning procedure illustrated in different runs, starting from random initial positions

5.3.3 Computational Properties

The computational complexity of updating parameters by relation (5.10) is bounded by the complexity of matrix multiplications of dimension p which is $\mathcal{O}(p^3)$. Moreover, it should be noted that, regarding the symmetry in the matrix of parameters P, this equation updates $L = (p^2 + p)/2$ number of parameters which correspond to the number of bases used in the value function. Therefore, in terms of the number of parameters, the complexity is $\mathcal{O}(L^{3/2})$. It is discussed in [52] that this can be done considerably faster than similar MBRL techniques, such as [82, 17, 81]. If only linear and constant bases are chosen, we will require matrix multiplications of dimension 13 to update the controller for the quadrotor. The runtime results reported in Fig. 5.2 for the quadrotor indicates a maximum process time of 8ms for calculating the control.

Moreover, as mentioned RLS is already implemented in many online identification techniques [103, 167, 170, 147]. Similarly, the runtime results in Fig. 5.2 confirm that RLS updates can be efficiently done under 2ms.

Accordingly, the total latency added by the computations in the control loop will be at most about 10ms. Therefore, the control frequency of 100Hz is achievable, which is enough for controlling a wide range of quadrotors.

5.4 Numerical Results

In the simulation, we consider the nonlinear model (5.1) for the Crazyflie, where the parameters are taken from [105] as listed in Table 5.1. The model is treated as a black box to simulate the real-world implementation. This model is integrated by using a Runge–Kutta solver.

m	0.33 [Kg]	d	39.73×10^{-3}[m]
I_{xx}	1.395×10^{-5}[Kg×m^2]	C_T	0.2025
I_{yy}	1.436×10^{-5}[Kg×m^2]	C_D	0.11
I_{zz}	2.173×10^{-5} [Kg×m^2]	g	0.98[m/s^2]

Table 5.1: The coefficients of the simulated Crazyflie

Through these simulations, we assume full access to the states. We then obtained the state derivatives using a one-step backward approximation.

We performed the simulations in Python on a 2.6 GHz Intel Core i5, including the 3D graphics generated via the Vpython module [146], as shown in Fig.5.1. The sampling rate is 66.6Hz ($h = 15$ms) for all the simulations. Accordingly, the control input value is calculated with the same rate. In the learning process, to simulate the exact behavior of Crazyflie, we integrate the continuous differential equations in sufficiently high precision. However, we only allow measurements at time steps conforming to the sampling rate. Moreover, the initial position is randomly chosen while the rest of states are set to zero.

The controller algorithm is run with the following settings defining the objective functions and the bases

$$\begin{aligned} &Q_0 = \text{diag}([20, 20, 20, 0.8, 0.8, 0.8, 0, 0, 0, 0.4, 0.4, 0.04]),\\ &R_0 = 3.5 \times 10^{-7} I_{4\times 4}, \Phi = \{1, x\}, \gamma = 0.4. \end{aligned}$$

We choose $x_{ref} = [0, 0, 3]$ meters and $\psi_{ref} = 30$ degrees. In Fig. 5.3 and Fig. 5.4, the attitude and position error of the quadrotor with respect to the reference are illustrated within different runs, where we employed 3 pre-runs before starting the closed-loop learning. A simulation video of the training is also uploaded at `https://youtu.be/QO8Ql83qKFM`. Although the quadrotor shows unstable behavior in its very first runs as expected according to Remark 12, it can achieve and preserve a stable hovering mode soon after. Fig. 5.5 denotes the model coefficients identified using RLS. In addition, it can successfully reach the goal point after collecting 634 samples within 68 seconds of flying until the end of Run #2.

5.5 Conclusion

In this chapter, by focusing on Crazyflie, we implemented the SOL learning algorithm to learn the low-level control for quadrotors to fly and keep the hovering state in a goal position point. To improve the runtime results of the learning, we implemented SOL with the RLS algorithm that is more suitable for online applications. The simulation results, illustrated rapid and efficient learning, where an initial model is obtained by random pre-runs then the model was improved within different runs in a closed-loop form. Based on the flight data and runtime results the approach can be employed to automate the control of the quadrotor. In future work, we will employ the obtained results and the tracking extension of SOL combined with a higher-level approach to achieve more complex objectives with no knowledge of the dynamics.

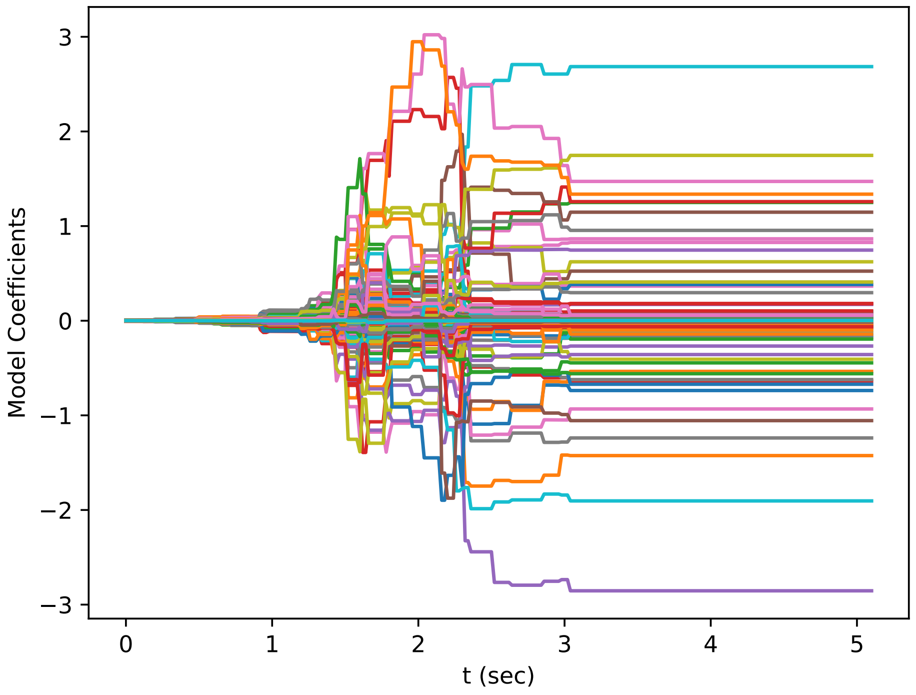


Figure 5.5: The model coefficients, identified by RLS, are shown within a run of the system.

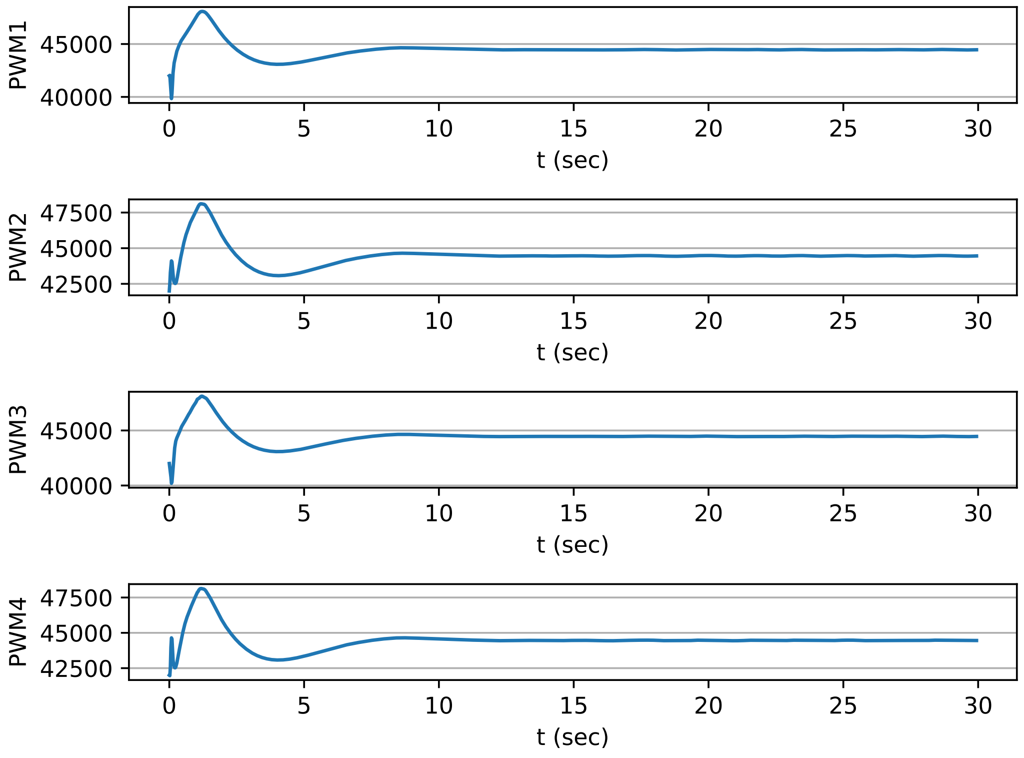


Figure 5.6: The PWM inputs of the quadrotor generated by the learned control.

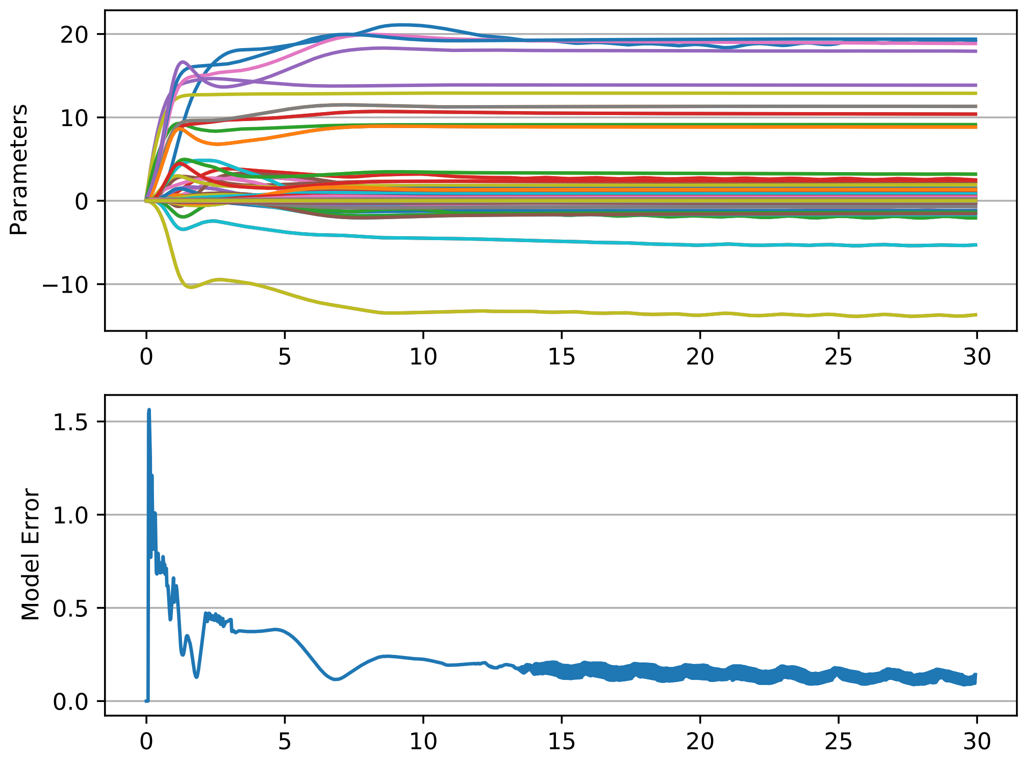


Figure 5.7: The parameters of the value function within a sample run together with the prediction error of the learned model.

Chapter 6

Applications to Solar Photo-voltaic Systems

The results presented in this chapter are published in [51].

6.1 Introduction

Solar power as a renewable source of energy has attracted worldwide attention in recent years. Discussions regarding the maximization of solar energy accumulation have dominated research in this field, and many efforts have been made on the development of PV devices and their applications. Designing an effective control algorithm plays an important role in developing an efficient solar PV system. To this objective, various algorithms, known as Maximum Power Point Tracking (MPPT) methods, have already been presented in the literature of power electronics.

Among the conventional MPPT methods, perturb and observe (P&O)[56], incremental conductance algorithm [93, 148], and hill climbing algorithm (HC)[169] are the most favorable techniques. It has been shown that some of these approaches have advantages over others in terms of implementation complexity or performance. Furthermore, they are all easy to apply and hence, become more suitable for low-cost applications. Comparative results can be found in [49, 116, 138]. In spite of the simplicity of the conventional methods, they have shown a slow response to changes in the ambient temperature and solar radiation power [138]. Consequently, the deviation from the MPP of the system results in an extent of power loss which is proportional to the size of the implemented solar array. Hence, for relatively large solar arrays, in the trade-off between simplicity and performance, we tend to give priority to the latter since the amount of energy saved by the implementation of some elaborate techniques is appreciable enough to justify the extra cost brought.

The performance of MPPT methods can be analyzed by observing the operating point behavior in two phases: the convergence phase and the steady-state phase. In the convergence phase, the implemented control needs to be fast enough in its responses to immediately lead the operating point to the MPP of the system. This can save a considerable amount of energy when the system is exposed to a large-amplitude disturbance by the surrounding environment. On the other hand, in the steady-state phase, the operating point is constantly disturbed due to noises, model imperfections, and the structure of the controller circuit. Hence, the operating point needs to be continuously supervised by a high-performance control scheme to keep it within a desirable bound around the MPP. Although the oscillations caused by inefficient controllers usually take place in a small scale, they potentially waste a huge amount of energy in high-power implementations by failing to tightly track the ideal MPP, and also by draining considerable power on non-ideal switching components.

Soft computing techniques, such as fuzzy logic control, artificial neural networks, and genetic algorithms are effective tools in dealing with nonlinear problems. Hence, as a remedy of certain inherent drawbacks in conventional methods, numerous soft computing-based algorithms have been implemented on solar PV systems[34, 157, 110]. These approaches generally show improved results on the performance of MPPT control while each algorithm has its own constraints. For instance, despite that fuzzy logic is easy to implement and can provide a flexible design, it is widely accepted that designing fuzzy rules to satisfy a particular performance measure normally needs a considerable amount of knowledge and training; otherwise, using an inadequate number of membership functions will encourage oscillations around MPP [138].

From a control systems point of view, solar PV systems can be modeled as a dynamical system for which more complicated and efficient control techniques can be exploited in comparison with the conventional MPPT methods. This provides many advantages in the design and analysis of solar PV systems since there already exists a vast literature of control systems that can be exploited to develop more efficient techniques (see, e.g., [99, 18, 36]).

In [141], a Sliding Mode Control (SMC) approach with a saturation functions has been presented that guarantees stability of the MPP of the system. In this approach, a design coefficient defining the sliding layer is chosen by trying different values and observing the convergence and steady-state results to find the optimum value. Although it illustrates improved results in the performance of the system in the nominal condition, the chattering problem still remains a major drawback since the saturation function and the controller gain chosen by trial and error cannot widely guarantee the performance for various settings of the solar PV system and the surrounding environment.

The double integral of the tracking error term can be used for constructing the sliding surface to eliminate steady-state error as well as to provide robust control responses against uncertainties [156]. While this contributes to the improved performance of the controller, it induces slow transient responses in the system. Hence, for tracking MPP of PV system, [132] develops an altered double integral SMC to speed up the convergence phase and alleviate the chattering effect. It stands to reason that, as an alternative method, the second-order SMC has advantages over the classical SMC methods in dealing with nonlinear systems. This justifies employing the second-order SMC in [145] by implementing a so-called super-twisting algorithm. The approach is developed in [84] for further moderating the chattering effect with the difference that, compared to [145], only one-loop control is used in [84]. The simulation results presented in [84] illustrate improved responses almost everywhere along the control signal. However, chattering effects can still be easily observed in the output power signal.

While there exist various approaches presented in the literature for improving the performance of the control in tracking the MPP, there is still no clear connection made between the configuration of the controller implemented and the performance obtained. Hence, performance analysis and defining the problem of maximizing the output power of PV systems in optimal control framework still remain as a challenging problem. To preserve a uniform quality in the performance of the system, a performance measure needs to be guaranteed.

In this chapter, we formulate and solve an optimal feedback control problem of solar PV systems that potentially brings many benefits in the applications. To obtain the optimal feedback control law, we consider a nonlinear affine model with a performance measure including a cross-weighting term. Hence, in contrast with the previous approaches, the performance analysis is additionally done by satisfying optimality conditions, which are adapted for a set of equilibrium points based on the incremental conductance approach. The obtained feedback controller, due to its suitable responses around the MPP, significantly decreases the undesired oscillations. Considering that the performance of the solar PV system is affected by the changing weather condition, we demonstrate the merits of the proposed controller under changing ambient temperature and solar radiation power.

The remaining of the chapter is organized as follows. In the next section, we will investigate the model details and parameters involved in the solar PV system and the boost converter considered in this chapter. Furthermore, we will see how they are paired together to let the control scheme regulate the operating point of the system as required.

Sections III and IV present the main results of this chapter. In Section III, we first introduce an optimal control problem to minimize the deviation from the MPP of the system. Then, we show that the optimal control law exists with respect to the defined cost functional. Moreover, by modifying the formulated optimal control problem, we derive an optimal voltage control for solar PV systems. Section IV addresses two main challenges in controlling real-world solar PV systems. In the first subsection, we propose an algorithm to realize the obtained control law without complete knowledge of the system parameters and only by samples obtained from the output voltage and current of the PV system. In the second subsection, we discuss the considerations for implementing the approach under non-uniform insolation. In Section V, the simulation results are provided for both model-based and model-free approaches, under uniform and non-uniform insolation.

A list of parameters used is given in Table 6.1.

Table 6.1: Nomenclature

Symbol	Description
I_{ph}	Light-generated current
I_s	Reverse saturation current
n	Ideality factor of PV cell
V_T	Thermal voltage
N_p, N_s	Parallel and series branches in PV module
R_{sh}, R_s	Shunt and series resistance of PV module
V_{oc}, I_{sc}	Open circuit voltage and short circuit current
L, C	Inductor and capacitor of DC-DC converter
R_L	Parasitic resistance of the inductor
R_0	Applied resistive load
P_a, V_a, i_a	Power, voltage, and current of solar array
v_C	Output voltage of DC-DC converter
V_o	Desired output voltage of DC-DC converter
k_{PID}	A vector of PID parameters

6.2 Problem Statement

This section formulates the control system and provides sufficient details on the model for later use in the controller design procedure. As shown in Fig. 6.2, in order to control the operating point of the system, a DC-DC boost converter can be used to couple the PV array with the load. The following two subsections present the model for describing the PV array and the boost converter, respectively.

6.2.1 PV Array Model

A solar PV array contains a number of PV modules. Suppose all the PV modules used in the array have identical electrical characteristics. Likewise, it is assumed that the surrounding environment affecting the solar panels, such as the ambient temperature, power density of the solar radiation, and wind speed, do not change considerably from one module to another.

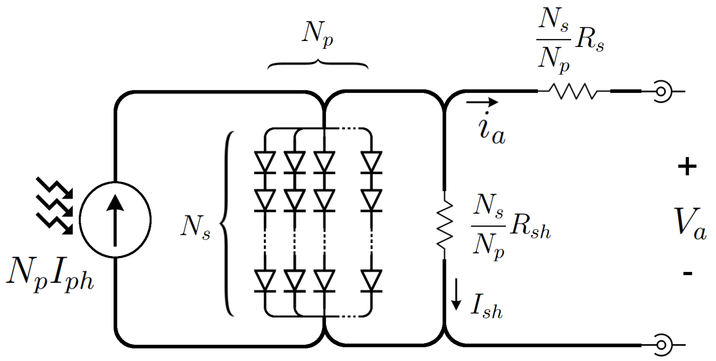


Figure 6.1: The equivalent electrical model of the solar array

Fig. 6.1 illustrates the equivalent circuit used to model an array of the PV modules arranged in N_p parallel branches, where each branch contains N_s modules in series. Applying Kirchhoff's current law at the top node results in

$$i_a = N_pI_{ph} - N_pI_s\left(\exp(\frac{\frac{R_si_a}{N_p}+\frac{V_a}{N_s}}{nV_T})-1\right) - I_{sh}, \tag{6.1}$$

wherein I_{ph}, I_s, n and V_T are the light-generated current, reverse saturation current, ideality factor, and thermal voltage of the PV module, respectively. Moreover, considering the equivalent shunt resistor R_{sh} and the series resistor R_s of the PV module in the circuit, we can write the current flowing through the shunt resistor as

$$I_{sh} = \frac{R_s i_a + \frac{N_p}{N_s} V_a}{R_{sh}}. \tag{6.2}$$

Hence, the output voltage V_a of the PV array can be calculated in terms of the output current i_a as

$$V_a = N_s n V_T \ln\left(\frac{N_p I_{ph} + N_p I_s - i_a - I_{sh}}{N_p I_s}\right) - \frac{N_s}{N_p} R_s i_a. \tag{6.3}$$

Then the first order and the second-order derivative of PV array voltage with respect to the output current can be computed as follows:

$$\begin{aligned} \frac{\partial V_a}{\partial i_a} &= \frac{-N_s n V_T}{N_p I_{ph} + N_p I_s - i_a - I_{sh}} - \frac{N_s}{N_p} R_s, \\ \frac{\partial^2 V_a}{\partial {i_a}^2} &= \frac{-N_s n V_T}{(N_p I_{ph} + N_p I_s - i_a - I_{sh})^2}. \end{aligned} \tag{6.4}$$

Note that, since the output power of the PV array is less sensitive to the changes in R_{sh}, the dependence of I_{sh} to V_a is disregarded to simplify the computations of the derivatives in (6.4). More details on the model of the PV array are provided in [159].

6.2.2 DC-DC Boost Converter

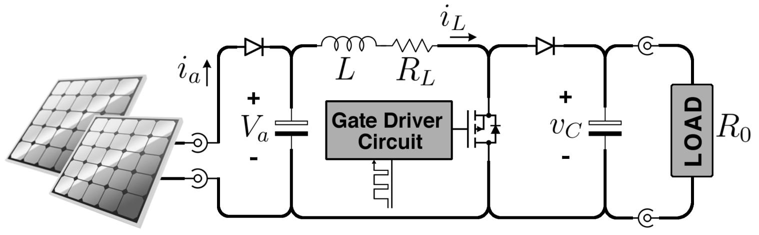


Figure 6.2: DC-DC boost converter used to interface the load to the solar array

Dynamical systems with a finite number of subsystems are generally known as switched systems. In such systems, a switching strategy orchestrates the switching action among these subsystems to ensure stability and performance. Switched affine systems introduce an important class of switched systems with a constant input that brings much convenience to design and application.

Consider the following model of the switched system

$$\begin{aligned}\dot{\mathbf{x}} &= A_\sigma \mathbf{x} + B_\sigma V_a, \\ \mathbf{y} &= C_\sigma \mathbf{x},\end{aligned} \tag{6.5}$$

where $A_\sigma \in \mathbb{R}^{n\times n}$, $B_\sigma \in \mathbb{R}^n$ and $C_\sigma^T \in \mathbb{R}^n$ denote the system matrices. The active subsystem is given by the piecewise-constant signal $\sigma : [0, \infty) \to \{0, 1\}$. By choosing the current of the inductor i_L and the voltage of the capacitor v_C as the states of the system, the state vector becomes $\mathbf{x} = [i_L \quad v_C]^T$. The input V_a plays the role of the power source for the system (6.5). As seen in the model of the PV array (6.3), this input depends on the state since i_a equals i_L for the coupled DC-DC converter and the solar PV array. Moreover, V_a is affected by the changes in the light-generated current I_{ph} and operating temperature T, considering that V_T is related to the temperature in (6.3). However, since the input irradiance and temperature are slowly changing parameters compared to the switched current flowing through the inductor, V_a is considered as a function of the state only. Henceforth, we regard $V_a : D \to \Xi$, where $D \subset \mathbb{R}^2_+$ and $\Xi \subset \mathbb{R}_+$ are domains of interest with $\mathbb{R}^n_+$ denoting the n-dimensional positive-real space.

In this chapter, a boost converter is used to formulate the problem, while a similar approach can be applied to different configurations of converters. Details on the design and switched system model of DC-DC converters can be found in [123, 41, 122]. For a typical boost converter, the switched affine model can be constructed by the following system matrices:

$$A_0 = \begin{bmatrix} -R_L/L & 0 \\ 0 & -1/R_0C \end{bmatrix},\ A_1 = \begin{bmatrix} -R_L/L & -1/L \\ 1/C & -1/R_0C \end{bmatrix},$$

$$B_1 = B_0 = \begin{bmatrix} 1/L \\ 0 \end{bmatrix},\ C_1 = C_0 = \begin{bmatrix} 1 & 1 \end{bmatrix}.$$

The load applied to the system is denoted by $R_0 \in \Omega \subseteq \mathbb{R}_+$. Moreover, L and C are positive constants that denote the values of the inductor and the capacitor, respectively. In this model, the inductor is supposed to be non-ideal, for which a parasitic resistance, indicated by R_L, is considered in series. Likewise, the leakage current of the output capacitor can be modeled as a resistor in parallel combination with the output load; however, it is disregarded in this model since the structure of the system matrices is not affected by that.

To obtain the PWM-controlled model of the system, the switched affine system is over-approximated by the convex hull of the subsystems

$$\dot{\mathbf{x}} = (1 - u(t))(A_0\mathbf{x} + B_0V_a) + u(t)(A_1\mathbf{x} + B_1V_a),$$

where $u(t) : [0, \infty) \to [0, 1]$ gives the duty-cycle values. This results in the average model of the system as

$$\dot{\mathbf{x}} = u(t)g(\mathbf{x}) + f(\mathbf{x}, V_a, R_0), \tag{6.6}$$

with $\mathbf{x} = [i_L \quad v_C]^T$ and

$$g(\mathbf{x}) = \begin{bmatrix} -\frac{1}{L}v_C \\ \frac{1}{C}i_L \end{bmatrix}, \quad f(\mathbf{x}, V_a, R_0) = \begin{bmatrix} -\frac{R_L}{L}i_L + \frac{1}{L}V_a \\ -\frac{1}{R_0C}v_C \end{bmatrix},$$

where $\mathbf{x} := \mathbf{x}(t) \in D \subseteq \mathbb{R}^2_+$ for $t \in [0, \infty)$ and $\mathbf{x}(0) = x_0$. Moreover, $f : D \times \Xi \times \Omega \to \mathbb{R}^2$ and $g : D \to \mathbb{R}^2$ are vector-valued functions. It should be noted that, the problem is formulated for inverted PWM generators, where, for instance, the sampled control $u(t^k) = 0$ at kth time step will keep the transistor in ON mode within $t \in [t^k, t^{k+1})$ for $k \in \mathbb{N}$. This can be simply adapted for non-inverted PWM devices by again inverting the duty-cycle value generated by the obtained control signal.

Furthermore, by the following substitution:

$$u(t) = \frac{V_a - R_Li_L}{v_C} + \omega_c(t), \tag{6.7}$$

the system dynamics can be rewritten as

$$\frac{\mathrm{d}}{\mathrm{dt}} \begin{bmatrix} i_L \\ v_C \end{bmatrix} = \omega_c(t) \underbrace{\begin{bmatrix} -\frac{1}{L}v_C \\ \frac{1}{C}i_L \end{bmatrix}}_{g(\mathbf{x})} + \underbrace{\begin{bmatrix} 0 \\ \frac{V_ai_L - R_Li_L^2}{v_CC} - \frac{v_C}{R_0C} \end{bmatrix}}_{f(\mathbf{x}, V_a, R_0)}, \tag{6.8}$$

where the control $\omega_c(t) \in W \subseteq \mathbb{R}$ for $t \in [0, \infty)$ is chosen from the set of all admissible controls Γ. The first part of control (6.7) can be seen as an equivalent control that requires $\frac{di_L}{dt} = 0$ for (6.6). Furthermore, the simple linear dependence of $\frac{di_L}{dt}$ on the control input $\omega_c(t)$ in (8) facilitates the stability analysis done later in the main results.

After modeling the switched system using a nonlinear affine system, in the next step, we exploit an inverse optimal control approach (see e.g. [118, 16, 70, 69]) to pose and solve an optimal control problem for tracking the MPP of the solar PV system.

6.3 Optimal Control of PV Array

In this section, we consider a particular form of the cost functional to regulate the performance of the solar PV system. We first present conditions needed for optimality and stability of (6.8) with respect to a set of equilibrium points and a given performance measure. By formulating an optimal MPPT problem, we then confirm that the stability and optimality conditions obtained indeed hold for the obtained control law.

Lemma 5. *Consider the system (6.8) with the cost functional*

$$J(x_0, \omega_c(\cdot)) = \lim_{T\to\infty} \int_0^T \mathbf{L}\big(\xi\big(\boldsymbol{x}, V_a(\boldsymbol{x})\big), \omega_c(t)\big)\, dt, \tag{6.9}$$

where $\boldsymbol{x}$ is the solution starting from $x_0 \in D$, and $\mathbf{L} : \mathbb{R} \times W \to \mathbb{R}$ is the running cost. Moreover $\xi : D \times \varXi \to \mathbb{R}$ defines the equilibrium set as

$$E = \{x \in D : \xi(x, V_a(x)) = 0\}. \tag{6.10}$$

Suppose there exists a C^1 function $\boldsymbol{V} : D \times \varXi \to \mathbb{R}$ and a control law $\omega_c^* = \phi(x, V_a(x))$ with $\phi : D \to W$ such that

$$\boldsymbol{V}(x, V_a(x)) = 0 \qquad \text{for} \quad x \in E, \tag{6.11}$$

$$\boldsymbol{V}(x, V_a(x)) > 0 \qquad \text{for} \quad x \in D, \qquad x \notin E, \tag{6.12}$$

$$\phi(x, V_a(x)) = 0 \qquad \text{for} \quad x \in E, \tag{6.13}$$

$$\begin{aligned}\boldsymbol{V}_x^T[\phi(x, V_a(x))g(x) + f(x, V_a(x), R_0)] < 0 \\ \text{for} \quad x \in D, \quad x \notin E, \quad R_0 \in \Omega,\end{aligned} \tag{6.14}$$

$$H\big(x, V_a(x), \phi(x, V_a(x))\big) = 0 \quad \text{for} \quad x \in D, \quad R_0 \in \Omega, \tag{6.15}$$

$$H\big(x, V_a(x), \omega_c\big) \geq 0 \qquad \text{for} \quad x \in D, \ \ \omega_c \in W, \quad R_0 \in \Omega, \tag{6.16}$$

where $\boldsymbol{V}_x$ and H denote, respectively, the partial derivatives with respect to the state as

$$\boldsymbol{V}_x := \frac{\partial \boldsymbol{V}}{\partial x} + \frac{\partial \boldsymbol{V}}{\partial V_a}\frac{\partial V_a}{\partial x},$$

and the Hamiltonian defined by

$$\begin{aligned}H\big(x, v, \omega\big) =& \mathbf{L}\big(\xi(x, v), \omega\big) \\ &+ \boldsymbol{V}_x^T\big(\omega g(x) + f(x, v, R_0)\big).\end{aligned}$$

Then, with the feedback control rule, the solutions of (6.8) converge to the set E. Moreover, the feedback control rule minimizes the performance functional in the sense that

$$J(x_0, \omega_c^*(\cdot)) = \min_{\omega_c \in \Gamma} J(x_0, \omega_c(\cdot)), \tag{6.17}$$

where

$$J\big(x_0, \omega_c^*(\cdot)\big) = \boldsymbol{V}(x_0, V_a(x_0)). \tag{6.18}$$

Proof. Conditions (6.11) to (6.14) guarantee attractivity of the set E since $\mathbf{V}$ is a Lyapunov function of the system (6.8).

The derivative of the Lyapunov function is given by

$$\dot{\mathbf{V}}(\mathbf{x}, V_a(\mathbf{x})) = \mathbf{V}_x^T\big(\omega_c g(\mathbf{x}) + f(\mathbf{x}, V_a(\mathbf{x}), R_0)\big), \tag{6.19}$$

then we add the running cost to both sides of (6.19) to obtain

$$\begin{aligned}\mathbf{L}\big(\xi(\mathbf{x}, V_a(\mathbf{x})), \omega_c(t)\big) =& \mathbf{L}\big(\xi(\mathbf{x}, V_a(\mathbf{x})), \omega_c(t)\big) - \\ & \dot{\mathbf{V}}(\mathbf{x}, V_a(\mathbf{x})) + \mathbf{V}_x^T\left(\omega_c(t) g(\mathbf{x}) + f(\mathbf{x}, V_a(\mathbf{x}), R_0)\right).\end{aligned} \tag{6.20}$$

By integrating both sides from 0 to T and letting $T \to \infty$, we obtain

$$\begin{aligned}J(x_0, \omega_c(\cdot)) =& \lim_{T\to\infty} \int_0^T [\mathbf{L}\big(\xi(\mathbf{x}, V_a(\mathbf{x})), \omega_c(t)\big) - \dot{\mathbf{V}}(\mathbf{x}, V_a(\mathbf{x})) \\ & + \mathbf{V}_x^T\big(\omega_c(t) g(\mathbf{x}) + f(\mathbf{x}, V_a(\mathbf{x}), R_0)\big)]\mathrm{dt} \\ =& \lim_{T\to\infty} \int_0^T [-\dot{\mathbf{V}}(\mathbf{x}, V_a(\mathbf{x})) \\ & \qquad + H\big(\mathbf{x}, V_a(\mathbf{x}), \omega_c(t)\big)]\mathrm{dt} \\ =& \mathbf{V}(x_0, V_a(x_0)) - \lim_{T\to\infty} \mathbf{V}\big(x_T, V_a(x_T)\big) + \\ & \lim_{T\to\infty} \int_0^T H\big(\mathbf{x}, V_a(\mathbf{x}), \omega_c(t)\big)\mathrm{dt} \\ \geq & \ \mathbf{V}(x_0, V_a(x_0)),\end{aligned} \tag{6.21}$$

where this concludes (6.17) by defining the Hamiltonian and using (6.16) and (6.18). □

6.3.1 Maximum Power Point Tracking Control

The goal of MPPT techniques is to maximize the output power of the PV array which is measured as below

$$P_a = V_a i_a.$$

The output power is obviously a function of the state and the state-dependent input of the switched system. Looking at (6.3), the input V_a is only related to the first state, where by considering the average model, we have $i_a = i_L$. Hence, according to the incremental conductance approach [93, 148], the stationary point of the output power with respect to the inductor current, i.e. points such that

$$\frac{\partial P_a}{\partial i_L} = \frac{\partial V_a}{\partial i_L} i_L + V_a = 0, \tag{6.22}$$

defines a set of equilibrium points that addresses the MPP of the solar array. By this knowledge, one can observe that choosing (6.9) in the following form

$$\begin{aligned} J(x_0, \omega_c(\cdot)) = \\ \lim_{T\to\infty} \int_0^T [\mathbf{L}_1(\xi) + \omega_c(t)\mathbf{L}_2(\xi) + S\omega_c(t)^2]\mathrm{d}t, \end{aligned} \tag{6.23}$$

along with the appropriate choices of functions $\mathbf{L}_1$, $\mathbf{L}_2 : \mathbb{R} \to \mathbb{R}$ penalize the deviation of the operating point of PV array from $\xi = \frac{\partial P_a}{\partial i_L} = 0$, which guarantees the maximum power point, and simultaneously regulates the control input. In (6.23), $\mathbf{L}_1$ and $\mathbf{L}_2$ can be written in terms of x and V_a by using (6.22), and S is a positive constant given by design considerations.

As declared in [70], it is evident that involving the cross-weighting term, with $\mathbf{L}_2 \neq 0$, not only brings extra flexibility into the design, but also illustrates better transient performance in terms of peak overshoot over the case $\mathbf{L}_2 = 0$.

Remark 13. *The uniform irradiance only leads to one maximum point in the power-voltage (P-V) characteristic curves that can be located with minimizing the defined cost functional and it is known to be the global MPP of the system. However, under non-uniform irradiance, only reaching a local MPP is guaranteed. To overcome the resulting non-convexity, more actions are needed that will be discussed in detail later in Section IV.*

Having defined the cost functional, we need to solve the minimization problem (6.17) to achieve the control law, which optimally leads the operating point of the system to MPP.

Theorem 6. *Consider the nonlinear affine dynamical system (6.8) and performance measure (6.23) with functions* $\mathbf{L}_1$, $\mathbf{L}_2 : \mathbb{R} \to \mathbb{R}$ *chosen respectively as*

$$\mathbf{L}_1(\xi) = \frac{p^2}{4S}(\bar{\xi} - 1)^2 \xi^2, \quad \mathbf{L}_2(\xi) = p\xi, \quad p > 0, \tag{6.24}$$

where

$$\xi = \frac{\partial P_a}{\partial i_L}, \quad \bar{\xi} = (\frac{\partial^2 V_a}{\partial i_L^2} i_L + 2\frac{\partial V_a}{\partial i_L})(\frac{v_C}{L}). \tag{6.25}$$

Then, by the choice of the inductor L *with* $0 < R_L < \lim_{i_L \to 0}\left\{|\frac{\partial V_a}{\partial i_L}|\right\}$ *and the feedback control rule (6.7) constructed by*

$$\omega_c^* = \frac{1}{2S} p\xi(\bar{\xi} - 1), \tag{6.26}$$

the solutions of the system (6.8) converge to the set E *defined by (6.10) with (6.25), and the performance measure (6.23) is minimized, for a given positive constant* S.

Proof. As stated, the maximum power point is given by the stationary point of the output power with respect to the state i_L as in (6.22). Thus, for tracking the MPP, we need to guarantee that the set E defined by choosing $\xi = \frac{\partial P_a}{\partial i_L}$ is attractive. Therefore, regarding the structure of the running cost, the optimal value function is supposed to possess the following form

$$\mathbf{V} = \frac{1}{2}p(\frac{\partial P_a}{\partial i_L})^2 = \frac{1}{2}p(\frac{\partial V_a}{\partial i_L} i_L + V_a)^2. \tag{6.27}$$

According to the HJB equation and using the optimal value function defined in (6.27),

$$0 = \inf_{\omega_c(\cdot)\in\Gamma}\Big\{ H(x, V_a, \omega_c) = \tag{6.28}$$

$$\mathbf{L}(\xi, \omega_c) + \mathbf{V}_x^T(\omega_c g(x) + f(x, V_a, R_0))\Big\},$$

holds along the optimal path given by the optimal control law chosen from the set of all admissible controls Γ, where $L(\xi, w_c)$ is the running cost chosen as in (6.23). With regard to (6.8) and (6.27), the optimal value function is independent of v_C. Also, v_C does not appear in (6.3) and (6.4). Therefore, $\mathbf{V}$ only changes along i_L, i.e.,

$$\mathbf{V_x} = \begin{bmatrix} \frac{\partial \mathbf{V}}{\partial i_L} \\ \\ \frac{\partial \mathbf{V}}{\partial v_C} \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{V}}{\partial i_L} + \frac{\partial \mathbf{V}}{\partial V_a}\frac{\partial V_a}{\partial i_L} + \frac{\partial \mathbf{V}}{\partial(\frac{\partial V_a}{\partial i_L})}\frac{\partial}{\partial i_L}(\frac{\partial V_a}{\partial i_L}) \\ \\ 0 \end{bmatrix} \tag{6.29}$$

$$= \begin{bmatrix} p(\frac{\partial V_a}{\partial i_L} i_L + V_a)(\frac{\partial^2 V_a}{\partial i_L^2} i_L + 2\frac{\partial V_a}{\partial i_L}) \\ 0 \end{bmatrix}.$$

Now, by substitution of the optimal value and the running cost, the Hamiltonian is obtained as

$$\begin{aligned} H(x, V_a, \omega_c) &= \mathbf{L}_1(\xi) + \omega_c \mathbf{L}_2(\xi) + S{\omega_c}^2 + \\ &\quad \mathbf{V}_{\mathbf{x}}^T (w_c g(x) + f(x, V_a, R_0)) \\ &= \mathbf{L}_1(\xi) + \omega_c \mathbf{L}_2(\xi) + S{\omega_c}^2 + \frac{\partial V}{\partial i_L}(\frac{-v_C \omega_c}{L}) + \\ &\quad \frac{\partial V}{\partial v_C}(\frac{1}{C} i_L \omega_c + \frac{V_a i_L - R_L i_L^2}{v_C C} - \frac{v_C}{R_0 C}) \\ &= \mathbf{L}_1(\xi) + \omega_c \mathbf{L}_2(\xi) + S{\omega_c}^2 + \\ &\quad p(\frac{\partial V_a}{\partial i_L} i_L + V_a)(\frac{\partial^2 V_a}{\partial i_L^2} i_L + 2\frac{\partial V_a}{\partial i_L})(\frac{-v_C \omega_c}{L}). \end{aligned} \tag{6.30}$$

Then, the optimal control law minimizing the given Hamiltonian is obtained by solving the following equation

$$\begin{aligned} \frac{\partial H(x, V_a, \omega_c)}{\partial \omega_c} &= \mathbf{L}_2(\xi) + 2\omega_c S + \\ &\quad p(\frac{\partial V_a}{\partial i_L} i_L + V_a)(\frac{\partial^2 V_a}{\partial i_L^2} i_L + 2\frac{\partial V_a}{\partial i_L})(\frac{-v_C}{L}) = 0. \end{aligned}$$

As a result, the optimal control law is given by

$$\begin{aligned} \omega_c^* = \frac{-1}{2S} \Big[&\mathbf{L}_2(\xi) + \\ &p(\frac{\partial V_a}{\partial i_L} i_L + V_a)(\frac{\partial^2 V_a}{\partial i_L^2} i_L + 2\frac{\partial V_a}{\partial i_L})(\frac{-v_C}{L}) \Big]. \end{aligned} \tag{6.31}$$

To shorten the computations, an intermediary function $\bar{\xi}$ is defined as (6.25), by which equations (6.30) and (6.31) become

$$H(x, V_a, \omega_c) = \mathbf{L}_1(\xi) + \omega_c(\mathbf{L}_2(\xi) - p\xi\bar{\xi}) + S\omega_c^2,$$

$$\omega_c^* = \frac{-1}{2S}(\mathbf{L}_2(\xi) - p\xi\bar{\xi}), \tag{6.32}$$

respectively. Furthermore, the substitution of ω_c^* in the Hamilton-Jacobi-Bellman (HJB) equation yields

$$\begin{aligned}\inf_{\omega_c(\cdot)\in\Gamma}\{H(x,V_a,\omega_c)\} &= H(x,V_a,\omega_c^*) \\ &= \mathbf{L}_1(\xi) - \frac{1}{2S}(\mathbf{L}_2(\xi) - p\xi\bar{\xi})^2 + \\ &\quad \frac{1}{4S}(\mathbf{L}_2(\xi) - p\xi\bar{\xi})^2 \\ &= \mathbf{L}_1(\xi) - \frac{1}{4S}(\mathbf{L}_2(\xi) - p\xi\bar{\xi})^2.\end{aligned} \tag{6.33}$$

To determine the optimal control input, it only remains to choose $\mathbf{L}_1$ and $\mathbf{L}_2$ functions such that the optimality and stability conditions are satisfied as required in Lemma 5. Regarding the structure of the value function in (6.27), conditions (6.11) and (6.12) hold. Furthermore, to guarantee the asymptotic stability, we need to verify (6.14) for the obtained feedback control. Using (6.8) and (6.29), $\dot{\mathbf{V}}$ is obtained as the following:

$$\begin{aligned}\dot{\mathbf{V}} &= \mathbf{V}_{\mathbf{x}}^T(w_c g(x) + f(x,V_a,R_0)) \\ &= \begin{bmatrix} p(\frac{\partial V_a}{\partial i_L} i_L + V_a)(\frac{\partial^2 V_a}{\partial i_L^2} i_L + 2\frac{\partial V_a}{\partial i_L}) \\ 0 \end{bmatrix}^T \\ &\quad \left(\omega_c \begin{bmatrix} -\frac{1}{L} v_C \\ \frac{1}{C} i_L \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{V_a i_L - R_L i_L^2}{v_C C} - \frac{v_C}{R_0 C} \end{bmatrix}\right) \\ &= p(\frac{\partial V_a}{\partial i_L} i_L + V_a)(\frac{\partial^2 V_a}{\partial i_L^2} i_L + 2\frac{\partial V_a}{\partial i_L})(\frac{-v_C\omega_c}{L}).\end{aligned} \tag{6.34}$$

With $\bar{\xi}$ defined in (6.25) and the optimal control law (6.32) this results in

$$\begin{aligned}\dot{\mathbf{V}}|_{\omega_c^*} &= -p\xi\bar{\xi}\left(\frac{-1}{2S}(\mathbf{L}_2(\xi) - p\xi\bar{\xi})\right) \\ &= \frac{1}{2S}p\xi\bar{\xi}\mathbf{L}_2(\xi) - \frac{1}{2S}(p\xi\bar{\xi})^2,\end{aligned} \tag{6.35}$$

wherein p and S are positive constants. Moreover, according to (6.25), the sign of $\bar{\xi}$ depends on $\frac{\partial V_a}{\partial i_L}$ and $\frac{\partial^2 V_a}{\partial i_L^2}$ which both are known to be negative values, considering electrical characteristics of solar PV cells. See, for instance, the current-voltage characteristic curves of a solar array in Fig. 6.3 and Fig. 6.4, wherein the voltage is strictly decreasing and concave-downward with respect to the current. As a result, $\bar{\xi}$ only takes negative values.

In (6.35), the second term is obviously negative definite; however, to decide the sign of $\dot{V}$, we still need to inspect the values taken by ξ and $\mathbf{L}_2$ that can be any positive or negative real values. To deal with this undetermined situation, let $\mathbf{L}_2$ as in (6.24), then,

$$\dot{\mathbf{V}}|_{\omega_c^*} = \frac{1}{2S}(p\xi)^2\bar{\xi} - \frac{1}{2S}(p\xi\bar{\xi})^2 < 0$$

is obtained that is clearly negative definite with $\bar{\xi} < 0$ for $x \notin E$, and satisfies (6.14). Moreover, this completes the optimal control input as in (6.26) by which (6.13) is also verified for the set E defined in (6.10).

Now, we choose $\mathbf{L}_1$ as in (6.24) to satisfy the HJB equation. With the substitution of $\mathbf{L}_1$ and $\mathbf{L}_2$, it follows from (6.33) that

$$\begin{aligned} H(x, V_a, \omega_c^*) &= \frac{p^2}{4S}\xi^2(\bar{\xi}-1)^2 + \frac{1}{2S}p\xi(\bar{\xi}-1)(p\xi - p\xi\bar{\xi}) + \\ &\quad S\left(\frac{1}{2S}p\xi(\bar{\xi}-1)\right)^2 \\ &= \frac{p^2}{4S}\xi^2(\bar{\xi}-1)^2 - \frac{p^2}{2S}\xi^2(\bar{\xi}-1)^2 + \frac{p^2}{4S}\xi^2(\bar{\xi}-1)^2 \\ &= 0. \end{aligned} \tag{6.36}$$

Thus, the HJB condition of optimality holds as in (6.15). In the last step, we need to inspect the Hamiltonian given by (6.32) to assure positivity for all admissible control $\omega_c \in W$, as required in (6.16),

$$\begin{aligned} H(x, V_a, \omega_c) &= \mathbf{L}_1(\xi) + \omega_c(\mathbf{L}_2(\xi) - p\xi\bar{\xi}) + S{\omega_c}^2 \\ &= \frac{p^2}{4S}\xi^2(\bar{\xi}-1)^2 + \omega_c(p\xi - p\xi\bar{\xi}) + S{\omega_c}^2 \\ &= \left(\frac{p}{2\sqrt{S}}\xi(\bar{\xi}-1)\right)^2 - \omega_c p\xi(\bar{\xi}-1) + (\sqrt{S}\omega_c)^2 \\ &= \left(\frac{p}{2\sqrt{S}}\xi(\bar{\xi}-1) - \sqrt{S}\omega_c\right)^2 \geq 0. \end{aligned} \tag{6.37}$$

This completes the conditions needed to be verified in Lemma 5. Hence, the obtained feedback control also satisfies optimality conditions and regulates the performance of MPPT controller with the performance measure given by (6.23).

Furthermore, the equilibrium currents of the closed-loop system (6.8), with control law (6.26), can be obtained by using the first state equation of the system (6.8):

$$i_L^{eq} = \begin{cases} -V_a/\frac{\partial V_a}{\partial i_L}, \\ \left(\frac{L}{v_C} - 2\frac{\partial V_a}{\partial i_L}\right)(1/\frac{\partial^2 V_a}{\partial i_L^2}), \end{cases} \tag{6.38}$$

where the latter is obviously negative and out of D, by negativity of the partial derivatives. Hence, only the first equilibrium current belongs to D. Plugging in the valid equilibrium current in the second state equation of the system (6.8) yields a relation for the equilibrium voltage of the capacitor as

$$v_C^{eq} = \pm\frac{\sqrt{-R_0(R_L + \frac{\partial V_a}{\partial i_L})}}{\frac{\partial V_a}{\partial i_L}} V_a. \tag{6.39}$$

Similarly, one of the equilibrium voltage relations is always negative and out of D. Thus, for a fixed R_0, the operating point of the system converges to the only valid equilibrium point

$$(i_L^{eq}, v_C^{eq}) = (-V_a/\frac{\partial V_a}{\partial i_L}, -\frac{\sqrt{-R_0(R_L + \frac{\partial V_a}{\partial i_L})}}{\frac{\partial V_a}{\partial i_L}} V_a), \tag{6.40}$$

which takes values in E for different $R_0 \in \Omega$.

Accordingly, to achieve a positive real equilibrium voltage, we also need, the design specification of the inductor satisfy $0 < R_L < |\frac{\partial V_a}{\partial i_L}|$, for any $x \in D$. Hence, a practical and safe choice of the upper bound for R_L is $\inf_{x\in D}\left\{|\frac{\partial V_a}{\partial i_L}|\right\} = \lim_{i_L\to 0}\left\{|\frac{\partial V_a}{\partial i_L}|\right\}$, which can be estimated from the voltage-current characteristic curve or experiment results of the solar PV array near open-circuit state.

□

Remark 14. *Considering relation (6.40), the equilibrium current does not directly depend on the output load. Hence, the proposed control law can independently follow the MPP regardless of the applied load, while the equilibrium voltage can be chosen by only regulating R_0. In the solar PV array connected to the AC grid, the load is usually controlled by a separate PI controller to regulate the output voltage v_C at a fixed level which is vital for correct power injection to the grid. Moreover, to ensure the valid operation of the boost converter, we need $v_C > V_a$. This suggests a lower bound to the applied load as*

$$R_0 > -\left(\frac{\partial V_a}{\partial i_L}\right)^2/(R_L + \frac{\partial V_a}{\partial i_L}), \tag{6.41}$$

for any $x \in D$, which can be used to make an estimation of Ω.

6.3.2 Reference Voltage Tracking Control

In the previous subsection, we proposed a framework to optimally control the solar PV array to gain the maximum power. As a second application of the proposed framework, we design an optimal feedback control rule to regulate the output voltage of the solar PV array to a reference value.

Corollary 1. *Consider the nonlinear affine dynamical system (6.8) and performance measure (6.23) with $\mathbf{L}_1$ and $\mathbf{L}_2$ chosen respectively as (6.24), where*

$$\xi = V_a - V_{ref}, \quad \bar{\xi} = \frac{\partial V_a}{\partial i_L}\frac{v_C}{L}. \tag{6.42}$$

Then, by the feedback control rule (6.26) constructed by (6.42) the solutions of the system (6.8) converge to the set E defined by (6.10), and the performance measure (6.23) is minimized.

Proof. Regarding the structure of the running cost, the optimal value function is supposed to possess a quadratic form as

$$\mathbf{V} = \frac{1}{2}p(V_a - V_{ref})^2. \tag{6.43}$$

By taking the derivative, we obtain

$$\begin{aligned}
\dot{\mathbf{V}} &= \mathbf{V}_{\mathbf{x}}^T(w_c g(x) + f(x, V_a, R_0)) \\
&= \begin{bmatrix} p(V_a - V_{ref})(\frac{\partial V_a}{\partial i_L}) \\ 0 \end{bmatrix}^T \left(\omega_c \begin{bmatrix} -\frac{1}{L}v_C \\ \frac{1}{C}i_L \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{V_a i_L - R_L i_L^2}{v_C C} - \frac{v_C}{R_0 C} \end{bmatrix} \right) \\
&= p(V_a - V_{ref})(\frac{\partial V_a}{\partial i_L})(\frac{-v_C \omega_c}{L}).
\end{aligned} \tag{6.44}$$

For the closed loop system, by substituting the control rule (6.26) constructed by (6.42) the following can be concluded

$$\dot{\mathbf{V}} = \underbrace{\frac{p^2}{2S}(V_a - V_{ref})^2(\frac{-\partial V_a}{\partial i_L}\frac{v_C}{L})}_{(+)}\underbrace{(\frac{\partial V_a}{\partial i_L}\frac{v_C}{L} - 1)}_{(-)} < 0.$$

This is obtained by the fact that the first partial derivative is always negative that also makes the last term negative while the other terms multiplied are all positive. Hence, the set E defined by choosing ξ as in (6.42) is attractive. The rest is followed in the similar way as the proof of Theorem 6 by defining Hamiltonian as (6.30) with the choices of ξ and $\bar{\xi}$ as in (6.42), by which, in addition, conditions (6.15) and (6.16) of the Lemma 5 are satisfied. □

While the control rule obtained can be employed to regulate the output voltage of the solar PV array in various applications, in the next section and later in the simulation results, we will see how it is particularly useful in partial shading condition.

Remark 15. *It should be noted that the control objective is to control the output voltage of the PV array, which is different than the goal of MPPT controller obtained in the last subsection. Hence, both parameters p and S may be independently adjusted for each case.*

Remark 16. *Similar to [84] and [141], the proposed approach can be considered as a model-based control approach since the partial derivatives appeared in the control rule depend on the model parameters as in (6.4). In the following section, we introduce a procedure, according to the obtained optimal scheme, to develop a model-free control by approximating the partial derivatives. In addition, we combine the results of Theorem 6 and Corollary 1 to deal with the partial shading phenomenon as a well-known imperfection in the operation of real-world PV arrays.*

6.3.3 Piecewise Learning Control

Consider the following system

$$\frac{d}{dt}\begin{bmatrix} i_L \\ v_C \\ \xi \end{bmatrix} = \begin{bmatrix} -\frac{1}{L}v_C u(t) - \frac{R_L}{L}i_L + \frac{1}{L}V_a \\ \frac{1}{C}i_L u(t) - \frac{1}{R_0 C}v_C \\ \frac{d}{dt}(\frac{\partial V_a}{\partial i_L}i_L + V_a) \end{bmatrix} \tag{6.45}$$

that is constructed by the average model of the solar PV system (6.6), augmented with ξ as the third state. It should be noted that, the states i_L and v_C can be directly measured. In addition, since we can measure V_a, then ξ can be approximated. The only concern is obtaining the partial derivative term that can be also obtained by measurements done. This will be discussed in detail in the next section.

Now, this nonlinear system contains the DC-DC converter's dynamics, the PV array model through function V_a, and ξ which can be used to reach the MPP. Let us assume that we do not have access to the parameters of this system that is indeed a common scenario in real-world applications. Hence, we need to approximate such system by only our observations of $\begin{bmatrix} i_L & v_C & \xi \end{bmatrix}^T$. The problem formulated fits best in the piecewise learning framework presented in Chapter 4. Accordingly, we use the following model to approximate this unknown system.

$$\dot{x} = W_\sigma \Phi(x) + \sum_{j=1}^{m} W_{j\sigma} \Phi(x) u_j + d_\sigma, \tag{6.46}$$

where W_σ and $W_{j\sigma} \in \mathbb{R}^{n \times p}$ are the matrices of the coefficients for $\sigma \in \{1, 2, \ldots, n_\sigma\}$ and $j \in \{1, 2, \ldots, m\}$, with a set of differentiable bases $\Phi(x) = [\phi_1(x) \quad \ldots \quad \phi_p(x)]^T$, and n_σ denoting the total number of pieces.

Considering that the MPP is given by $\xi = 0$, to set up a learning MPPT controller, it only remains to choose Q in the control objective (2.2) in a way that the third component of the state is penalized.

6.4 Application Considerations

In this section, we address two main challenges of establishing maximum power point tracking control in real-world applications.

6.4.1 Partial Derivative Approximation Procedure

In the optimal approach proposed in the previous section, it is supposed that the exact values of partial derivatives, given by (6.4), are available at any $t \in \mathbb{R}_+$. However, in the real-world implementation of PV arrays, there exist numerous parameters affecting the output power characteristics of the PV array that cannot be directly measured or estimated. In the literature, some effort has been made on online parameter identification of solar PV arrays, while application of the MPPT methods using only the output voltage and current measurements of the PV array is often preferred. This is because of their simplicity and robustness while using no extra knowledge of the surrounding environment and electrical characteristics of the solar cell, which make them less expensive for applications as well.

To set up the presented optimal control approach, we only need the partial derivatives in (6.4), which can be obtained approximately by using the sampled output voltage and output current of the PV array. Consider the output current i_a, the light-generated current

I_{ph} and the ambient temperature T as three major parameters affecting the output voltage of the PV array, given in (6.3). Compared to the output current, the solar irradiation and ambient temperature are changing slowly. Hence, it is assumed $|di_a| \gg |dI_{ph}|$ and $|di_a| \gg |dT|$. Then the rate of changes of the output voltage of the array is approximated for sufficiently small $|dt|$ as

$$\frac{dV_a}{dt} \simeq \frac{\partial V_a}{\partial i_a}\frac{di_a}{dt},$$

and this yields an estimation of the partial derivative as

$$\frac{\partial V_a}{\partial i_a} \simeq \frac{dV_a}{dt} / \frac{di_a}{dt}. \tag{6.47}$$

Remark 17. *This can be considered as a strong assumption applied to the problem that might adversely affect the performance of the system when the rate of changes in the solar irradiation and the ambient temperature is relatively high. However, since in the realistic weather condition, the irradiation and temperature inputs will eventually reach a stable condition with a tolerable rate of changes, the controller will also be able to survive from sudden disturbances and retrieve the track of the MPP in a short time.*

Moreover, the second-order derivative can be similarly written as:

$$\begin{aligned}\frac{d^2V_a}{dt^2} &= \frac{d}{dt}(\frac{dV_a}{dt})\\ &\simeq \frac{\partial}{\partial i_a}\left(\frac{\partial V_a}{\partial i_a}\frac{di_a}{dt}\right)\frac{di_a}{dt}\\ &= \left(\frac{\partial^2 V_a}{\partial {i_a}^2}\frac{di_a}{dt} + \frac{\partial V_a}{\partial i_a}\frac{\partial}{\partial i_a}(\frac{di_a}{dt})\right)\frac{di_a}{dt}\\ &= \frac{\partial^2 V_a}{\partial {i_a}^2}\left(\frac{di_a}{dt}\right)^2. \end{aligned} \tag{6.48}$$

This is obtained using the fact that the derivative of the output current is independent of the output current, according to the first state equation of the system (6.6). Then the second-order partial derivative becomes

$$\frac{\partial^2 V_a}{\partial {i_a}^2} \simeq \frac{d^2V_a}{dt^2} / \left(\frac{di_a}{dt}\right)^2, \tag{6.49}$$

as $dt \to 0$. For the implementation of the proposed approach, the values given by (6.47) and (6.49) are needed to be calculated at each time sample. Thus, for a sufficiently small sampling time $\tau = t^k - t^{k-1}$, (6.47) and (6.49) can be measured at $t = t^k$ respectively as

$$\left.\frac{\partial V_a}{\partial i_a}\right|_{t=t^k} \simeq \left(\frac{V_a^k - V_a^{k-1}}{\tau}\right) / \left(\frac{i_a^k - i_a^{k-1}}{\tau}\right) = \frac{\Delta V_a^k}{\Delta i_a^k},$$

$$\left.\frac{\partial^2 V_a}{\partial {i_a}^2}\right|_{t=t^k} \simeq \left(\frac{\Delta^2 V_a^k}{\tau^2}\right) / \left(\frac{\Delta i_a^k}{\tau}\right)^2 = \frac{\Delta V_a^k - \Delta V_a^{k-1}}{(\Delta i_a^k)^2}, \tag{6.50}$$

where

$$\begin{aligned} &\Delta i_a^k = i_a^k - i_a^{k-1}, \\ &\Delta V_a^k = V_a^k - V_a^{k-1}, \\ &\Delta V_a^k - \Delta V_a^{k-1} = V_a^k - 2V_a^{k-1} + V_a^{k-2}. \end{aligned} \tag{6.51}$$

Hence, while the division by Δi_a is allowed, the relations in (6.50) approximate the partial derivatives required for constructing the feedback control. It is worth noting that the signals obtained in (6.51) may be prone to high-frequency noises in applications. However, considering the smooth properties of (6.4), we can safely employ low-pass filters as long as they do not induce slow responses.

In the convergence phase, since the inductor current is strictly increasing or decreasing toward its steady-state value within a period of time, the chance to meet $\Delta i_a \to 0$ is sufficiently low. Therefore, the approximations done by the divisions in relations (6.50) are expected to be valid until the steady state is achieved. In other words, the feedback controller supplied by these approximations will be able to converge to the MPP.

Furthermore, the performance of the controller highly depends on the steady-state responses as well, where oscillations of i_L around its steady-state value result in some stationary points of i_L that make the denominator $\Delta i_a \to 0$. Let $\kappa > 0$ be the minimum value allows dividing by, that is chosen by design considerations. Then, based on the second derivative in (6.51), $|\Delta i_a|$ can take at least $\sqrt{\kappa}$ to yield a valid division. Hence, relations (6.50) are used as long as Δi_a is greater than $\sqrt{\kappa}$. Otherwise, the samples are accumulated by $\Sigma_{\Delta i}$, $\Sigma_{\Delta V}$, and $\Sigma_{\Delta^2 V}$, without any update done in the approximation values. Afterwards, once the condition $|\Sigma_{\Delta i}| > \sqrt{\kappa}$ is met, the divisions are done and the approximation values are updated by the accumulated samples.

As stated, the technique established for the approximation of the partial derivatives relies on the magnitude of changes in i_L. If the controller chooses to let this current rest for a while, there will be no longer updates on the approximation values as well. Thus, it gives the operating point the opportunity to diverge from MPP as far as possible. In other words, a minimum perturbation is always required on i_L to perform a valid approximation and to decide a suitable control input as quickly as possible. Therefore, when the amplitude of changes on i_L do not satisfy the condition $|\Delta i_a| > \sqrt{\kappa}$, a constant positive or negative value, in accordance with the sign of changes on i_L, is added to the control input to encourage the perturbations. This procedure is illustrated in Algorithm 1, where the approximated values are only updated when the condition is satisfied; otherwise, the samples are accumulated for future use, and a small perturbation is continuously added to the PWM value in each iteration.

In the simulation results, the proposed algorithm will be implemented to control a sample solar PV array under uniform and non-uniform insolation.

6.4.2 Partial Shading Effect

Partial shading phenomenon is widely studied in the literature as one of the factors resulting in current-voltage characteristic curve mismatching among PV modules. Although non-uniform insolation caused by partially shaded PV modules is known as the most likely scenario, there exist other possible imperfections, such as the production tolerance, accumulated dust, and ageing ([124, 149]), that can promote the mismatching effect.

As shown in Fig. 6.12 and Fig. 6.13, the mismatching in current-voltage (I-V) curves caused by partial shading effect leads to some local maxima in the P-V characteristic curve of the solar array. Consequently, conventional MPPT methods, such as P&O, incremental conductance, and HC algorithms, as well as control system approaches, such as SMC, second-order SMC, and double integrator control, possibly fail in tracking the global maximum since they search locally by following the direction that increases the output power. Hence, a higher level control is required to systematically [19] or randomly [153] switch among the local areas to search for the greatest MPP of the solar array. In this regard, some approaches are already presented in the literature, such as power increment technique [90], load-line MPPT [88, 75], and instantaneous operating power optimization approach [30], that exploit one of the conventional methods at some stage to identify the local MPP corresponding to the current area of interest (for more details see [19]). Therefore, to boost the performance of these algorithms, the control rule suggested by Theorem 6 can be implemented as an alternative to the conventional methods and previously presented control system approaches.

The proposed optimal control framework can be combined by algorithms presented in the literature to tackle the mismatching effect appeared in non-uniform insolation. In

Algorithm 1

```
1: procedure APPROXIMATION OF PARTIAL DERIVATIVES
   Input:
2: Samples obtained from $i_a$ and $V_a$;
   Output:
3: $\frac{\partial V_a}{\partial i_a}\Big|_{t^k}$, $\frac{\partial^2 V_a}{\partial {i_a}^2}\Big|_{t^k}$;
   Initialization:
4: Choose $\varepsilon$, $\kappa > 0$;
5: Set $\Sigma_{\Delta i}, \Sigma_{\Delta V}, \Sigma_{\Delta^2 V} = 0$;
6:     while ($true$) do
7:         Read samples $i_a^k$, $V_a^k$;
8:         Update $\Delta i_a^k$, $\Delta V_a^k$ and $\Delta^2 V_a^k$ by (6.51);
9:         if $|\Delta i_a| > \sqrt{\kappa}$ then
10:            Update $\frac{\partial V_a}{\partial i_a}\Big|_{t^k}$, $\frac{\partial^2 V_a}{\partial {i_a}^2}\Big|_{t^k}$ using (6.50);
11:        else
12:            $\Sigma_{\Delta i} := \Sigma_{\Delta i} + \Delta i_a^k$;
13:            $\Sigma_{\Delta V} := \Sigma_{\Delta V} + \Delta V_a^k$;
14:            $\Sigma_{\Delta^2 V} := \Sigma_{\Delta^2 V} + \Delta^2 V_a^k$;
15:            if  $|\Sigma_{\Delta i}| > \sqrt{\kappa}$ then
16:                Update:
17:                $\frac{\partial V_a}{\partial i_a}\Big|_{t^k} \simeq \frac{\Sigma_{\Delta V}}{\Sigma_{\Delta i}}$;
18:                $\frac{\partial^2 V_a}{\partial {i_a}^2}\Big|_{t^k} \simeq \frac{\Sigma_{\Delta^2 V}}{(\Sigma_{\Delta i})^2}$;
19:                Reset $\Sigma_{\Delta i}, \Sigma_{\Delta V}, \Sigma_{\Delta^2 V} = 0$;
20:            else
21:                Add perturbation:
22:                $\omega_c := \omega_c + \varepsilon \mathrm{Sign}(\Sigma_{\Delta i})$;
23:            end if
24:        end if
25:    end while
26: end procedure
```

this chapter, we exploit the load line technique together with the optimal voltage control obtained in Corollary 1 to relocate the operating point after the partial shading event. In contrast with [75], to implement [88], additional circuits are required to measure V_{oc} and I_{sc} online. Hence, we compose the main controller by the following control rules:

$$\begin{aligned} &\text{Controller\_One} : (6.7) \text{ with } (6.26) \text{ using } (6.25),\\ &\text{Controller\_Two} : (6.7) \text{ with } (6.26) \text{ using } (6.42), \end{aligned} \tag{6.52}$$

where the controller choice is governed by Algorithm 2 that exploits the load-line technique in [75]. A graphical representation of the approach together with the simulation results will be provided in the next section.

Algorithm 2

```
1:  procedure CONTROL SCHEME IN PARTIAL SHADING
    Input:
2:  Samples obtained from $i_a$ and $V_a$;
    Output: Sub-controller chosen from (6.52).
3:  Initialization:
4:  Choose $\varsigma > 0$;
5:  Define $V_{ref} = 0, Controller =$ Controller_One;
6:      while ($true$) do
7:          Read samples $i_a^k$, $V_a^k$;
8:          if ($partial\_shading\_condition$) then
9:              $V_{ref} = \frac{N_s V_{oc}}{N_p I_{sc}} i_a^k$;
10:             $Controller =$ Controller_Two;
11:             while ($|V_a^k - V_{ref}| > \varsigma$) do
12:                 Read sample $V_a^k$;
13:             end while
14:             $Controller =$ Controller_One;
15:         end if
16:     end while
17: end procedure
```

6.5 Simulation Results

To assess the proposed approach under a realistic condition, the Canadian Solar CS6X-335M-FG module [29] has been simulated in MATLAB/Simulink® with the obtained control rule to generate the maximum power in different weather conditions. This module contains 72 solar cells and the electrical characteristics were listed in Table 6.2. In this simulation, the PV array is composed by 12 modules placed in 6 parallel branches where there exist 2 modules in series at each branch. As illustrated in Fig. 6.2, a DC-DC boost converter is used to regulate the load applied to the PV array that provides the control over the operating point of the solar array.

Control input (6.7) is constructed by the optimal control law obtained in (6.26). According to the cost functional (6.23) defined by (6.24), increasing S penalizes the control effort. Moreover, by looking at (6.26), one can observe that only the proportion of S and p appeared in the control rule. Hence, by arbitrarily fixing p, which scales the value function, and by relatively changing S, the control parameters corresponding to a desirable performance can be obtained. Regarding Remark 15, for each of Controllers (6.52), the parameters are chosen as $[S, p] = [5.88,\ 1 \times 10^{-5}]$ and $[S, p] = [1,\ 1 \times 10^{-2}]$, respectively.

Since the average model of the converter was considered, the continuous value given by the control law needs to be converted back into a quantized signal that complies with the number of subsystems. Therefore, any value given by $u(t) \in [0, 1]$ is considered as a duty cycle to constantly generate a pulsed signal to drive the switch within a particular period of time which depends on the frequency chosen for PWM. Moreover, the output load is controlled by a PI controller to regulate the output voltage v_C to a fixed DC voltage level V_o. For PV arrays connected to AC grid, this can be replaced by a DC-AC inverter. A schematic diagram of the system is given in Fig. 6.15.

In this simulation, the parameters of the boost converter are set to have the following values: $L = 0.2mH$, $R_L = 1\Omega$, and $C = 2500\mu F$. Also, switching components are chosen such that for the diode we have $V_{On} = 0.6$V, on resistance= 0.3Ω, and off conductance= $10^{-8}\Omega^{-1}$, and for the switch, on resistance $= 10^{-2}\Omega$, and off conductance= $10^{-6}\Omega^{-1}$. The PWM frequency is set to be 65kHz, where the duty cycle value is updated by the controller every $0.1ms$. Moreover, we set $k_{PID} = [6, 10, 0]$. The desired output voltage level is assumed to be $V_o = 120V$ for any part of the simulations, unless explicitly mentioned otherwise.

6.5.1 Model and Control Verification

Since the function of solar PV systems is mainly affected by the changes in the input irradiance and temperature, the performance of the proposed nonlinear optimal control (NOC) approach is investigated by changing these parameters. Fig. 6.3 denotes the output results of the simulated solar array by changing the irradiance power from 400 to 1000

Table 6.2: Electrical data of the CS6X-335M-FG module[29]

Under STC (1000 W/m^2 and 25°C)	Value
Nominal Max. Power (Pmax)	335 W
Open Circuit Voltage (Voc)	46.1 V
Short Circuit Current (Isc)	9.41 A
Temperature Coefficient (Pmax)	-0.41 % / °C
Temperature Coefficient (Voc)	-0.31 % / °C
Temperature Coefficient (Isc)	0.053 % / °C
Nominal Module Operating Temperature (NMOT)	43 ± 2 °C

W/m^2. Similarly, the next set of graphs in Fig. 6.4 is obtained by changing the ambient temperature from 5°C to 65°C. In both figures, the system starts up from the rest situation, i.e. $x = 0$. Then, the operating point continues moving on the I-V and P-V characteristic curves determined by the input irradiance and temperature, until it reaches its MPP. The system operates around the MPP while the input does not change. Once the input starts moving to the next value on a ramp, the controller also regulates the operating point to the corresponding MPP of the system. Hence, it is evident in these figures that the proposed controller can successfully lead the operating point to the MPP and track it in the presence of a disturbance applied on the temperature and irradiance.

6.5.2 Comparative Results

In the comparison results of the system, the obtained control law is compared with two recent approaches targeting on the performance improvement of the generated control signal: sliding mode control [141] and second-order sliding mode control[84]. For a fair comparison, the boost converter and solar PV system were chosen to remain the same for all simulated approaches.

As the first scenario, we generated a random fast-changing signal as the input irradiance of the system while the operating temperature is fixed at 25°C. This signal is shown in Fig. 6.7(d). By running the simulation with these inputs, the output power is captured for the optimal control and the other two approaches in Fig. 6.8. In the second scenario, the same was done by fixing the input irradiance at 800 W/m^2 and applying a changing temperature, as shown in Fig. 6.9(d).

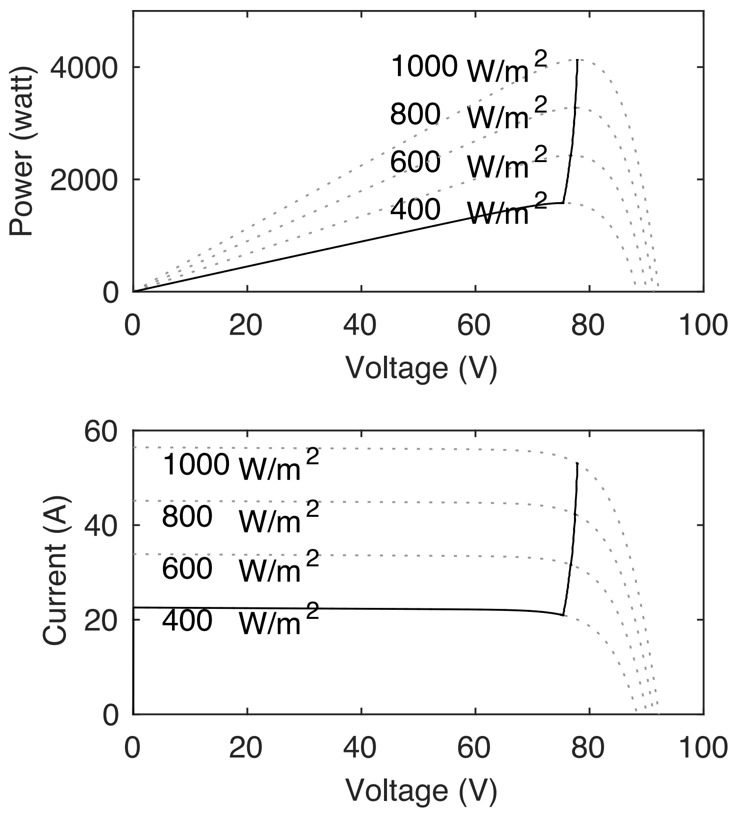


Figure 6.3: Output results of the simulated solar array by changing the irradiance power from 400 to 1000 W/m^2, where the solid lines represent the track of the MPP obtained by using the proposed NOC approach.

By the output power comparison results, in Fig. 6.8 and Fig. 6.10, we can observe the SMC control reaches to the MPP faster than the other approaches while it shows a constant chattering effect almost everywhere. The second-order SMC shows improved results on the chattering effect; however, it fails to demonstrate a robust performance within the simulation. See, for instance, Fig. 6.8(a) and Fig.6.10(b), wherein the second-order SMC manages to efficiently decrease the chattering effect, whereas, in Fig. 6.8(c) and Fig. 6.10(c), the amplitude of oscillations around MPP is almost as large as the SMC control. In addition, the second-order SMC illustrates slow responses under fast changes of the input, i.e. in the convergence phase, hence looses the track of MPP in fast-changing weather conditions. For instance, see Fig. 6.8(a) and Fig. 6.10(a), respectively. In contrast with the second-order SMC control, the obtained control with a guaranteed cost is able

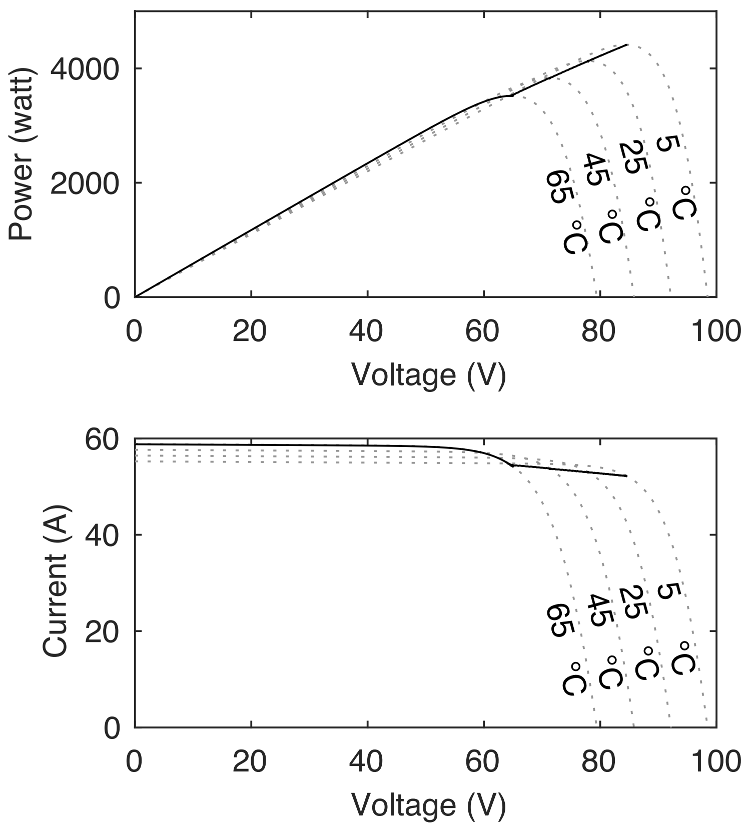


Figure 6.4: Output results of the simulated solar array by changing the ambient temperature from 5°C to 65 °C, where the solid lines represent the track of the MPP by using the proposed NOC.

to maintain a constant performance for different weather conditions. Moreover, from the comparative error results given in Fig. 6.8 and Fig. 6.10, it can be clearly seen that the proposed control law outperforms the other two methods, where the error graph, for any control approach, is obtained by the absolute difference of the corresponding output power with the maximum of all three approaches at any time instant $t \in [0, 1]$. This is also evident in the corresponding control signals given in Fig. 6.7 and Fig. 6.9, that the optimal control signal illustrates the smoothest response with almost no chattering effect.

6.5.3 Model-free Approach Results

To verify the model-free controller using the proposed approximation procedure, Algorithm 1 was implemented with a changing irradiance as the input. As mentioned, κ is a sufficiently small value that yields a valid division. This can be chosen by starting from large values and decreasing until a minimum amplitude of oscillations is observed in the steady state. Moreover, ε is a small perturbation recursively added to control that can affect output signal of the PWM generator within a few time steps. Hence, it is chosen with respect to the PWM resolution of the device. A possible choice is a fraction of $1/PWM\_resolution$. In this simulation, the parameters are chosen as $\kappa = 1\times10^{-3}$ and $\varepsilon = 5\times10^{-3}$. Fig. 6.16 shows the approximated partial derivatives, perturbation, and output power signals corresponding to the input irradiance. It is evident by Fig. 6.16(e) that NOC operating based on the approximation of partial derivatives are able to tightly track the ideal maximum power curve. Although, as expected, the output power corresponding to the model-free controller is not as s illustrate
satisfacto

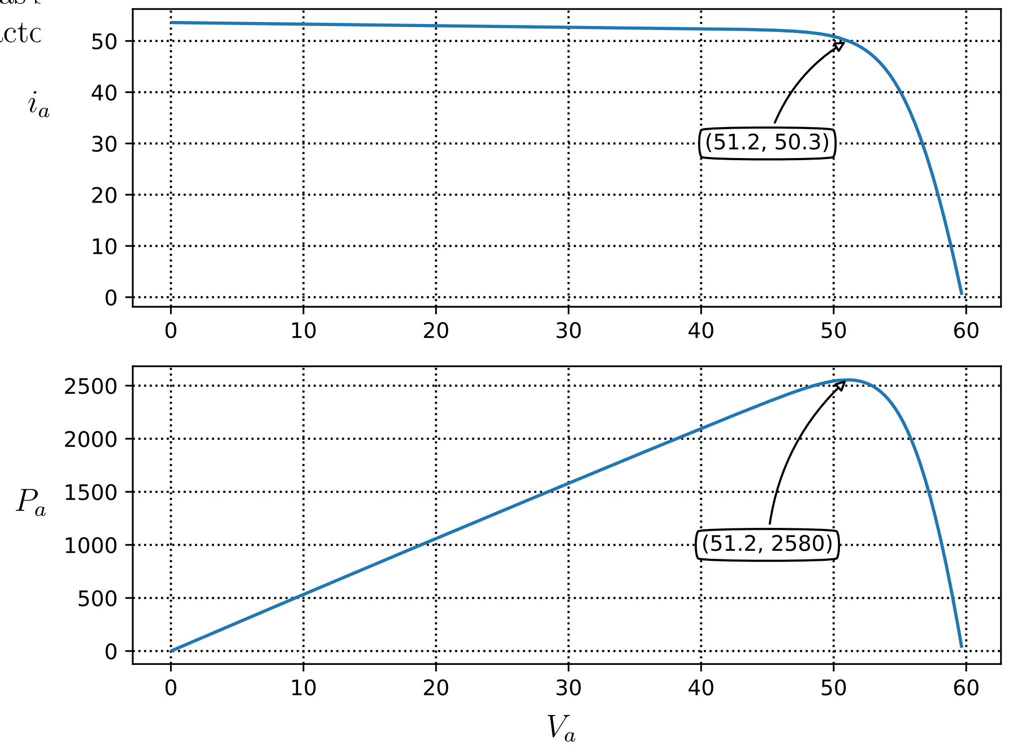


Figure 6.5: I-V and P-V graphs are shown that characterize the solar PV system used for piecewise learning control as an example. Moreover, the MPP is denoted.

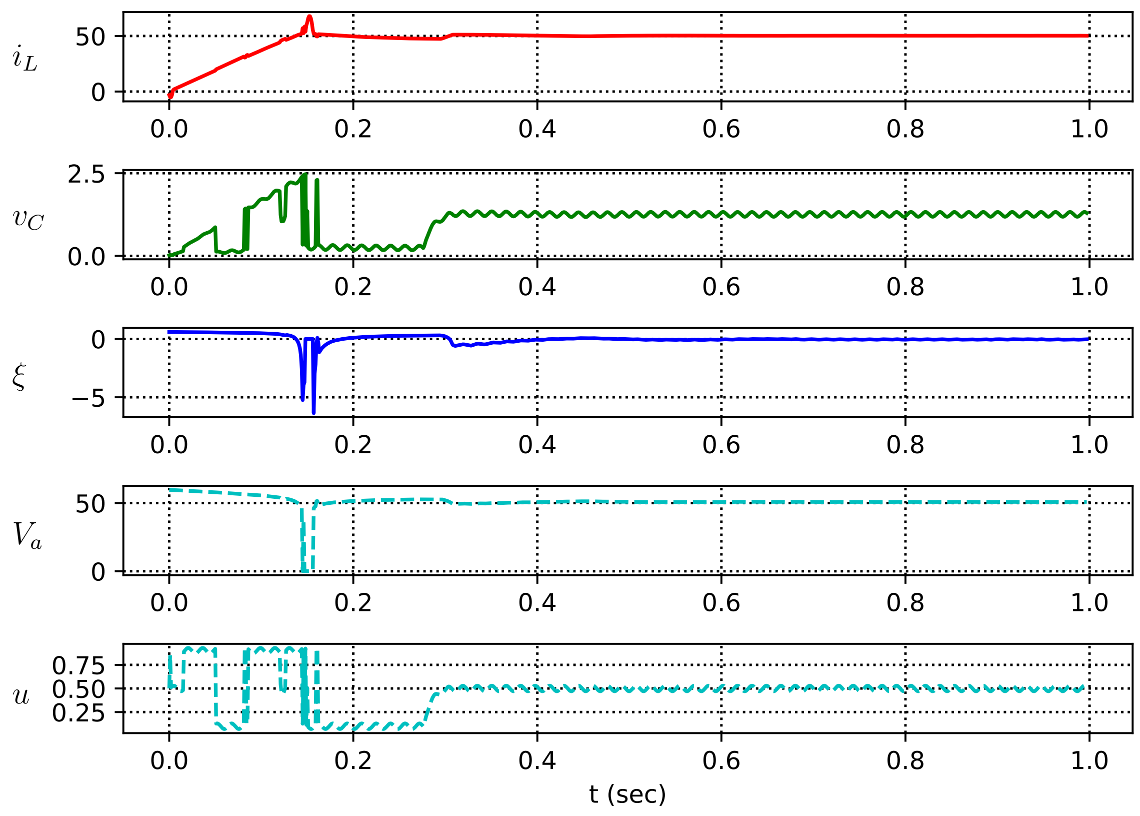


Figure 6.6: The learning result of the solar PV system, given by Fig. 6.5, are shown. It can be observed that after 0.4 sec, ξ converges to zero, that guarantees operating in the MPP. It is also evident by the V_a signal that the PV array voltage can track the MPP voltage 51.2V given by Fig. 6.5.

6.5.4 Piecewise Learning Results

In this section, we implement the piecewise learning controller on a sample PV system. To do so, we use a PV system as a black box for which the characterizing P-V and I-V graphs are given in Fig. 6.5 together with the MPP. Targeting the system (6.45), we choose the observation vector as $\begin{bmatrix} i_L & v_c & \xi \end{bmatrix}$. To start learning, we need to define the partitions of the piecewise model. Therefore, we uniformly grid the state space with the number of points given by the vector $[11, 11, 11]$, which corresponds to the state vector. We choose only the linear and constant bases that yields a linear PWA model. Regarding the discussion on approximating the partial derivatives, and the convergence of the piecewise model learner, the system needs to be persistently stimulated with some input signal. Hence, a sinusoidal probing signal with the amplitude 0.03 and frequency 76Hz is added to the control. Moreover, we define the objective function with $Q = \text{diag}([0, 0, 10^2])$ and $R = [10^2]$.

According to the objective defined, it is expected that after enough time of learning the controller drives ξ to zero. According to the definition of the MPP this guarantees operating in the MPP of the system. It can be observed from Fig. 6.6 that ξ indeed converges to zero after about 0.4 sec of training. Hence, the learning control can achieves the MPPT objective. This is also shown by the PV array voltage V_a that can track the MPP voltage given by Fig. 6.5.

6.5.5 Partial Shading Results

Considering the partial shading effect as shown in Fig. 6.11, we run a simulation of the PV array in both uniform and non-uniform insolation. In this regard, we implemented Algorithm 2 with $\varsigma = 1$ and the dc link voltage is set to be $V_o = 160$. Furthermore, we used Algorithm 1 to estimate the partial derivatives. Evolutions of the operating point on I-V and P-V curves are given in Fig. 6.12 and Fig. 6.13, respectively. The system is run from zero initial state assuming uniform insolation. As seen in P-V curve of the system Fig. 6.13, Controller One (6.52) is used to reach the MPP of the system. Once the system is exposed to the non-uniform insolation, some local maxima appear in the P-V curve. Consequently, the partial shading condition is detected by the algorithm and the reference voltage is calculated as $V_{ref} = 49.5V$. This voltage is tracked by Controller Two (6.52) until V_a is ς-close to the reference voltage, where ς can be chosen with respect to the open circuit voltage of the PV array so that remaining in the neighborhood of the global MPP is assured. Once the output voltage is arrived in the neighborhood of the calculated V_{ref}, Controller One (6.52) is again activated to track the MPP in that local area which is expected to be the global MPP. In Fig. 6.14, output power, voltage, and current are illustrated that correspond to Fig. 6.12 and Fig. 6.13. Furthermore, Fig. 6.14 denotes the control signal and switchings between the sub-controllers given by (6.52).

6.6 Conclusion

Motivated by the lack of performance analysis of the solar PV system in the literature, we developed an optimal feedback control approach to improve the convergence and steady-state responses of the system. The nonlinear non-quadratic cost as a performance measure, which involves the cross-weighting term, introduced a degree of freedom in the stability and optimality analysis. Then, we obtained a nonlinear optimal feedback controller by minimizing the corresponding Hamiltonian. The idea is exploited to establish two controllers for tracking the MPP and a given reference voltage that is separately activated according to an algorithm to deal with the partial shading condition. Moreover, the resulting optimal control rule involved partial derivatives of the output voltage of the solar PV array with respect to the inductor current. Hence, an exact implementation of the proposed scheme depends on the solar PV model details. This motivated us to obtain a model-free based control scheme. Moreover, we proposed a piecewise learning-based control that relaxes the need for the exact dynamics. The simulation results, obtained by the implementation of the proposed algorithm on a realistic solar PV model, demonstrated the applicability of the approach in uniform and non-uniform insolation. Furthermore, in the comparison results, the nonlinear optimal control illustrated a suitable convergence response with the minimum oscillations around the MPP. Compared to the second-order SMC, the chattering phenomenon that emerges in the SMC was further decreased by employing the optimal approach with a guaranteed performance measure.

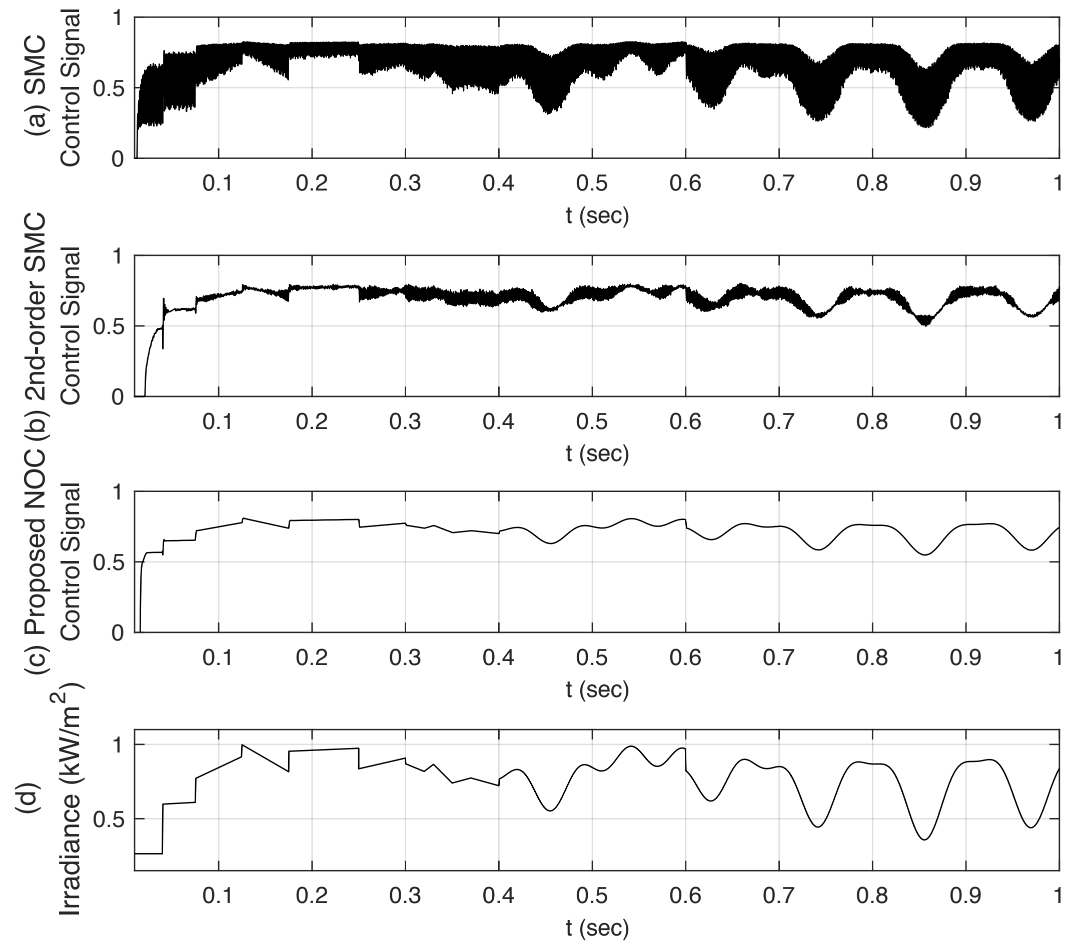


Figure 6.7: The obtained control signal (NOC) compared to SMC and second-order SMC under the changing irradiance shown in (d).

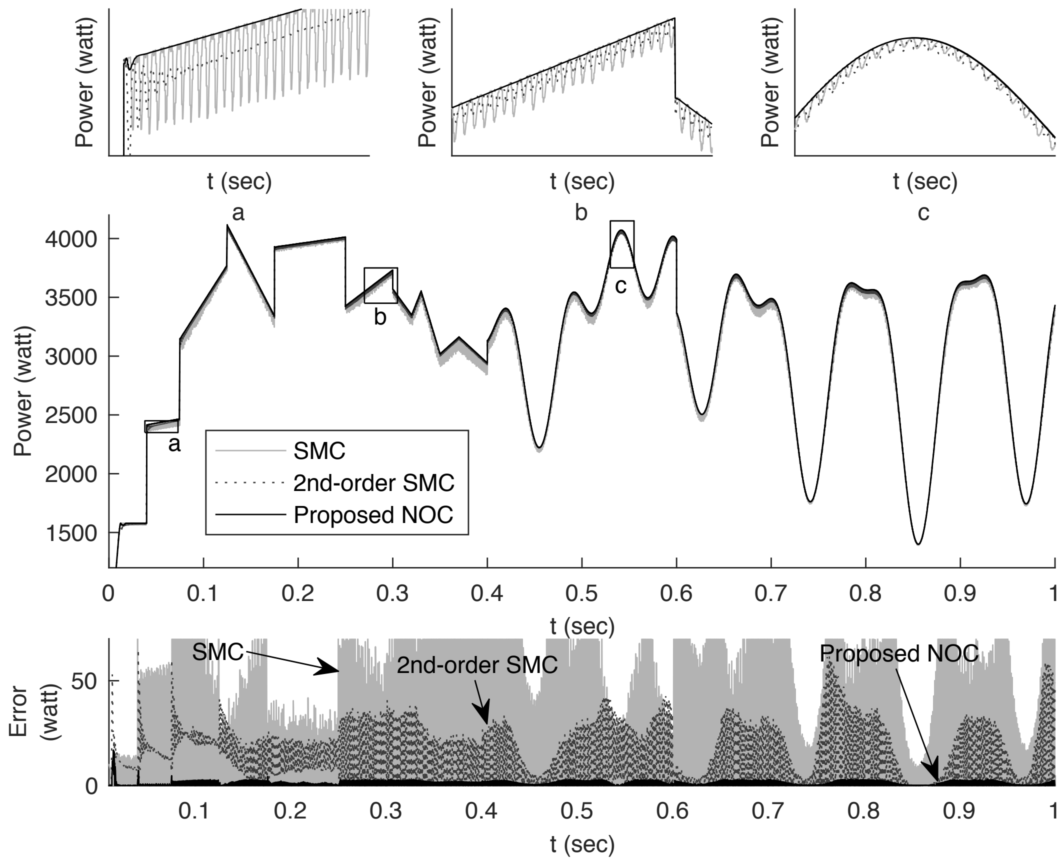


Figure 6.8: Comparison results of the output power under the changing irradiance shown in Fig. 5(d), where some parts of the graph are magnified in sub-figures (a-c). The error graph denotes the comparative error for the proposed NOC compared to SMC and second-order SMC (see the text for details).

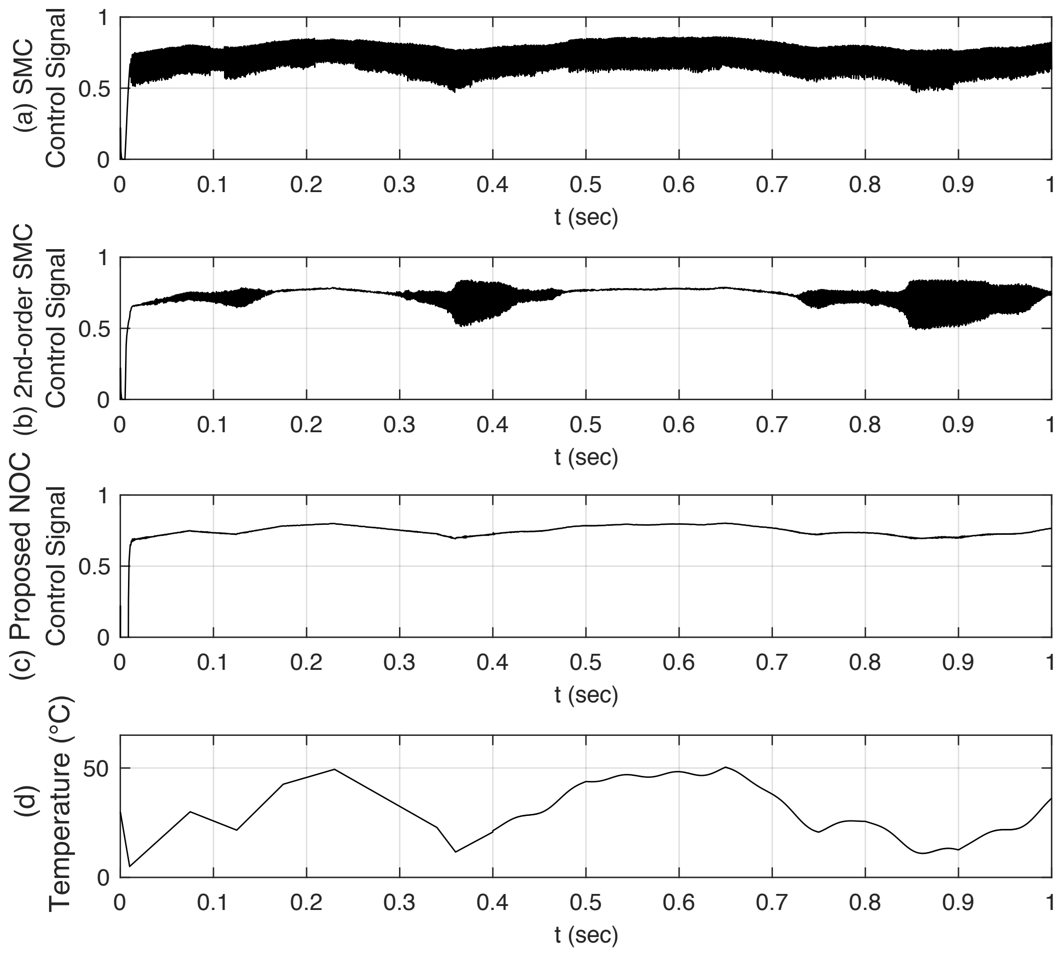


Figure 6.9: The obtained control signal (NOC) compared to SMC and second-order SMC under the changing ambient temperature shown in (d).

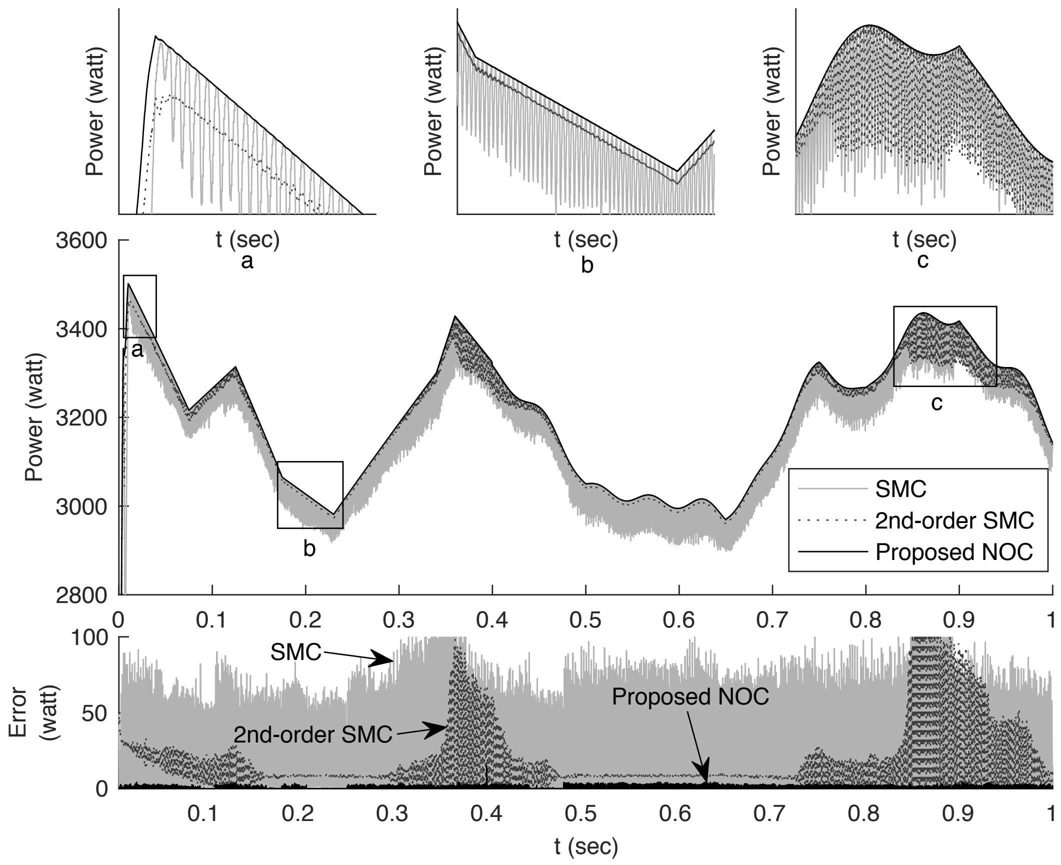


Figure 6.10: Comparison results of the output power under the changing ambient temperature shown in Fig. 7(d), where some parts of the graph are magnified in sub-figures (a-c). The error graph denotes the comparative error for the proposed NOC compared to SMC and second-order SMC (see the text for details).

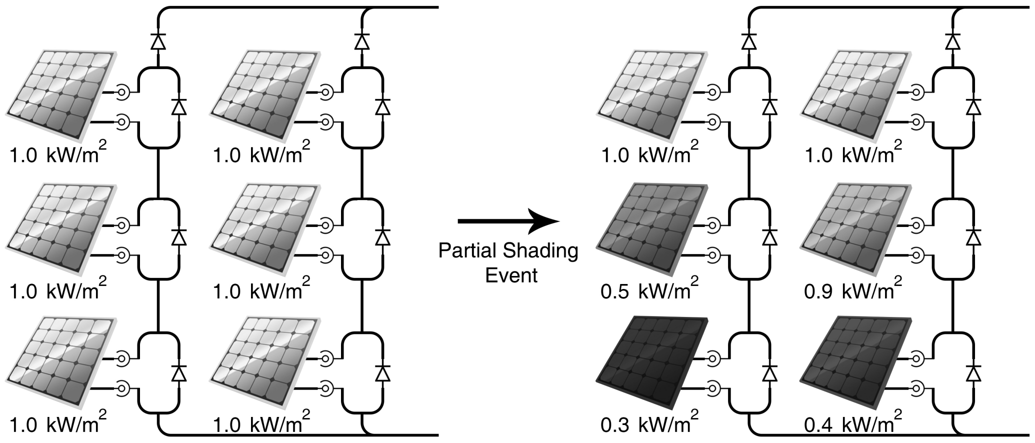


Figure 6.11: The solar PV array illustrating the partial shading condition considered in the simulation results.

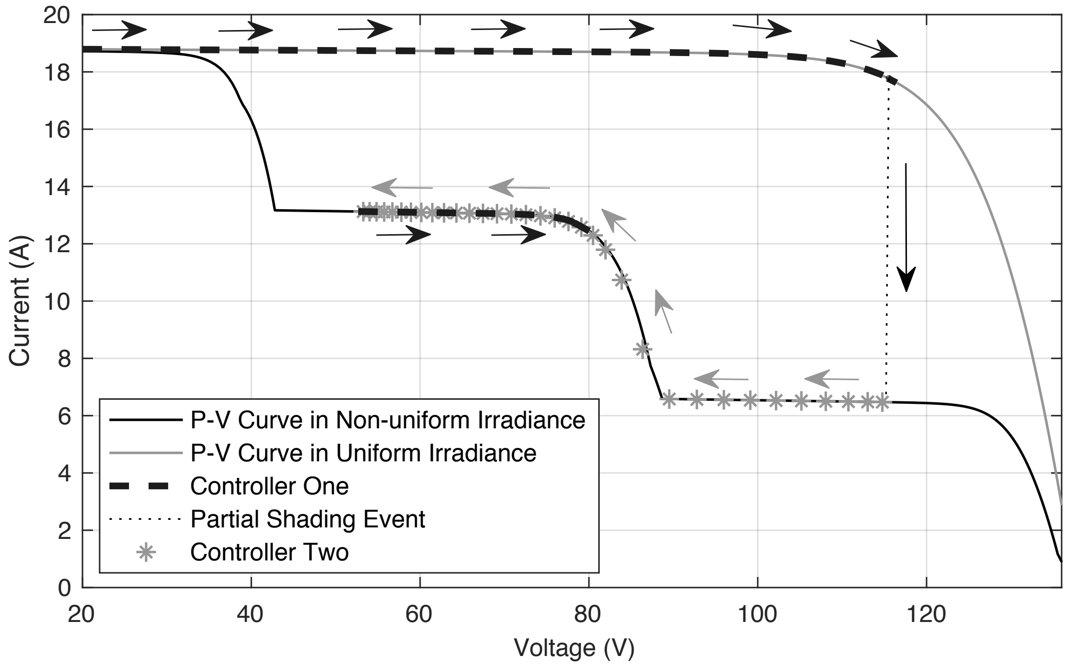


Figure 6.12: Evolutions of the operating point of system on I-V curve before and after partial shading event, where controllers are defined by (6.52).

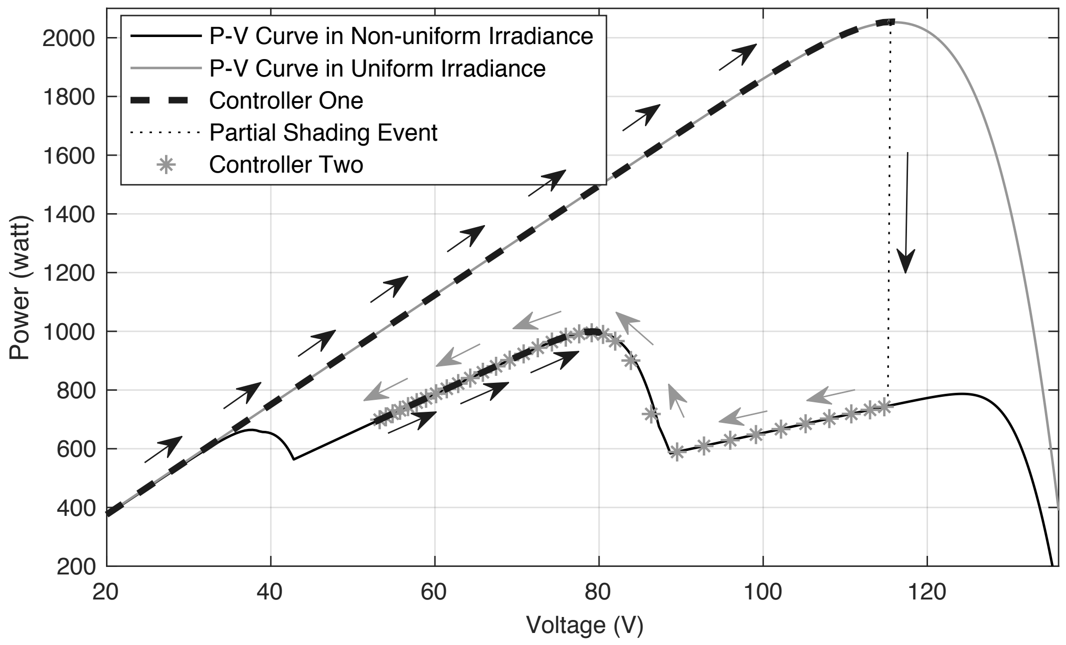


Figure 6.13: Evolutions of the operating point of system on P-V curve before and after partial shading event, where controllers are defined by (6.52).

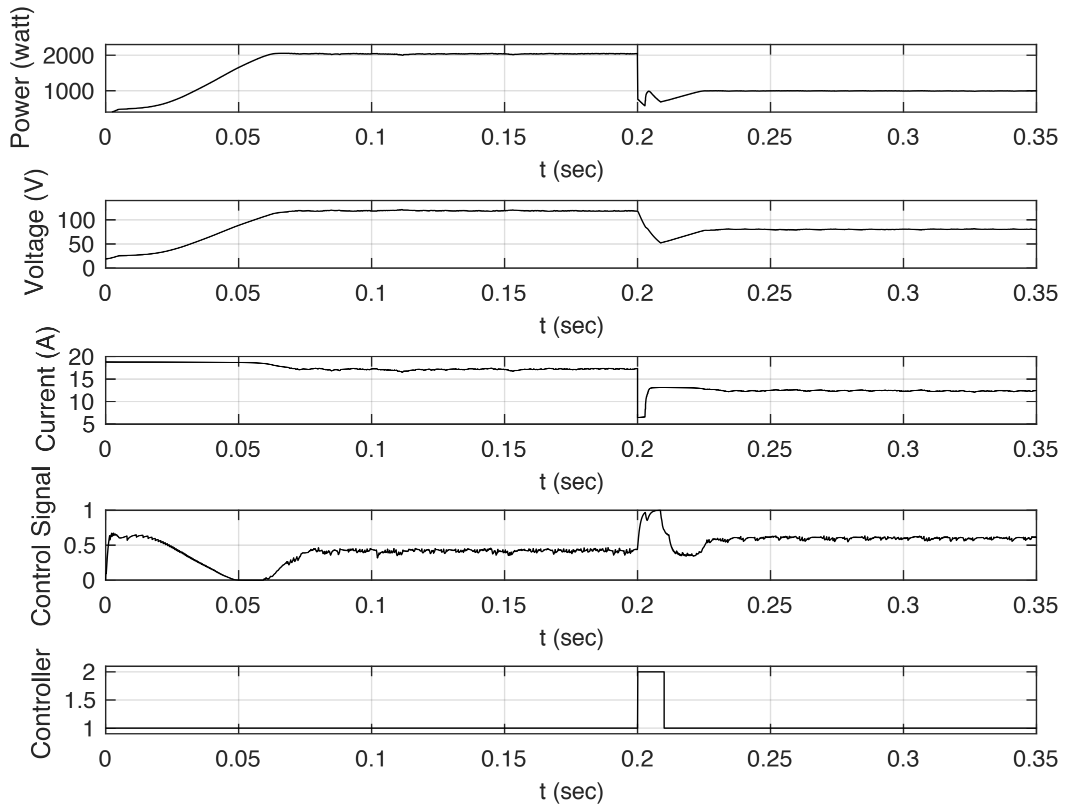


Figure 6.14: Output voltage and current signals of the solar PV array together with the control signal, where the shading event, corresponding to Fig. 9 and Fig. 10, is detected at 0.2 sec.

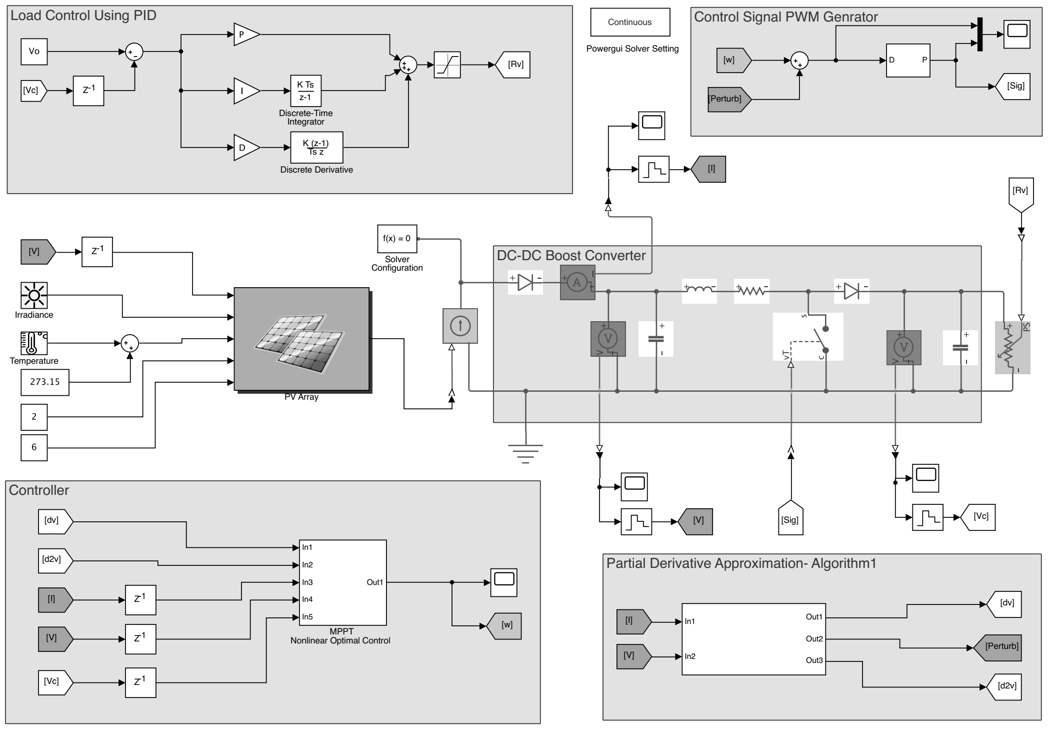


Figure 6.15: A sketch of the simulated solar PV system together with the proposed control approach in Matlab Simulink.

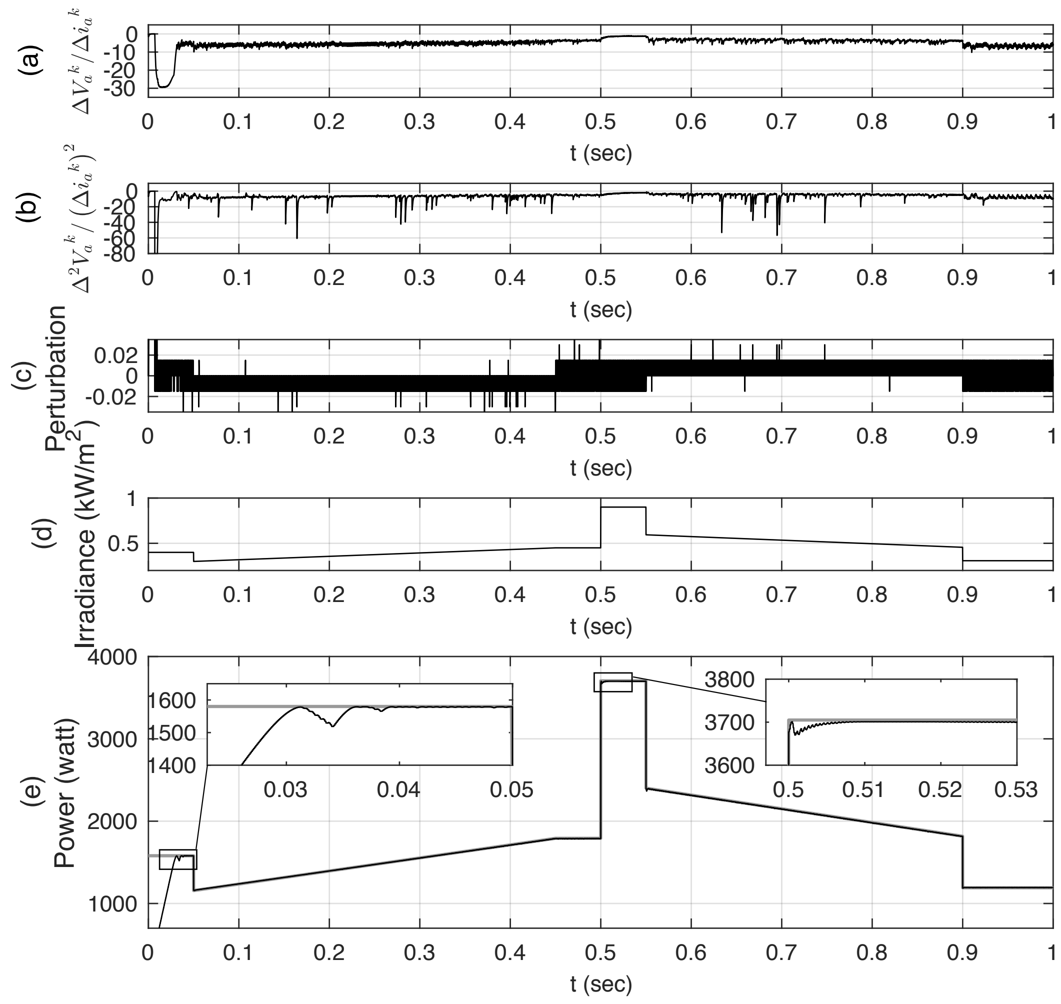


Figure 6.16: The results obtained by simulating the system with the proposed Algorithm 2, which illustrate respectively: (a)-(b) First and second-order partial derivatives. (c) Perturbation signal added to improve the estimation.(d) The variable input irradiance applied to the solar PV array. (e) The output power.

Chapter 7

Conclusions and Future Works

7.1 Conclusions

In this dissertation, we aimed on solving the control regulation problem for unknown nonlinear dynamics. We assumed nonlinear affine dynamics in terms of a collection of basis functions as a structured system. Then, it assisted us in analytically computing an iterative update method to obtain the optimal value function according to the most recent update on the identified system. We demonstrated some possible options for using the SOL algorithm as an online model-based learning tool based on the computational cost and performance seen in numerical and graphical simulations. Accordingly, we proposed a nonlinear tracking control technique with unknown dynamics. As a result, we devised an optimal tracking control, with the performance measure rewritten to fit the structured system. A matrix differential equation was developed to approximate the solution of the optimal control problem with a value function parameterized in quadratic form. As a result of this formulation, we accomplished a learning-based tracking control framework that only uses online measurements of the system states and the reference trajectory. Moreover, we proposed a piecewise nonlinear affine learning framework with guarantee for regulating nonlinear systems with uncertain dynamics. In this framework, each piece was intended for locally learning and controlling over a partition of the domain. Then, as a specific instance of the proposed framework, we focused on learning in the form of linear PWA systems. Due to the linearity, we achieved an optimization-based verification approach that takes into account the estimated uncertainty bounds of the model. Hence, this verification technique allowed us to validate the stability globally among all the pieces. We used the pendulum system as a benchmark example in the numerical results that demonstrated the superiority of the proposed technique by achieving a larger ROA.

Simulation results on benchmark nonlinear systems were presented to illustrate the applicability of the established frameworks. We applied the SOL approach on four distinct nonlinear systems for this purpose. Through these implementations, the model was identified and the control problem was satisfactorily addressed. On unknown nonlinear dynamics, we successfully implemented the trajectory tracking extension. Furthermore, for the quadrotor and solar PV systems, detailed numerical results of SOL and the piecewise learning technique were reported that demonstrated the applicability of these approaches. Regarding the runtime results on the higher dimensional quadrotor system, we observed that the proposed MBRL can be efficiently implemented in realtime. Moreover, we used a nonlinear optimal feedback control strategy for the MPPT problem to deal with the oscillations caused by the chattering phenomena in the control. The comparison results confirmed the advantages of the proposed technique over the ones in the literature.

7.2 Future Work

Currently, there exist two directions in which we follow the research based on the MBRL techniques presented in this dissertation.

Python Toolbox Considering that most of the implementations are done in the Python environment, we develop a Python-based toolbox. This toolbox allows learning in terms of different basis function can be chosen from a library of bases. Regarding the system identification, one may choose different well-known techniques already implemented, such as RLS, GD, SINDy, etc. Accordingly the SOL and the piecewise learning technique are employed for MBRL of different nonlinear benchmark examples including quadrotor and vehicle system, robot arm, etc. Then, for the PWA, we provide the user with the option to verify the learned controller for the uncertain system based on the technique presented in Chapter 4.

Real-world Implementation Given the improved computational complexity of the resulting update algorithm for the value parameters, as well as the simulation results, future research in this direction focuses on overcoming practical constraints and developing a complete platform for real-world applications, such as vehicle and quadrotor systems.

References

[1] Mahyar Abdolhosseini, YM Zhang, and Camille Alain Rabbath. An efficient model predictive control scheme for an unmanned quadrotor helicopter. *Journal of intelligent & robotic systems*, 70(1-4):27–38, 2013.

[2] Dirk Aeyels and Joan Peuteman. A new asymptotic stability criterion for nonlinear time-variant differential equations. *IEEE Transactions on automatic control*, 43(7):968–971, 1998.

[3] Amir Ali Ahmadi and Pablo A Parrilo. Non-monotonic lyapunov functions for stability of discrete time nonlinear and switched systems. In *2008 47th IEEE conference on decision and control*, pages 614–621. IEEE, 2008.

[4] Edoardo Amaldi, Stefano Coniglio, and Leonardo Taccari. Discrete optimization methods to fit piecewise affine models to data points. *Computers & Operations Research*, 75:214–230, 2016.

[5] George Andrikopoulos, George Nikolakopoulos, Ioannis Arvanitakis, and Stamatis Manesis. Piecewise affine modeling and constrained optimal control for a pneumatic artificial muscle. *IEEE Transactions on Industrial Electronics*, 61(2):904–916, 2013.

[6] MOSEK ApS. The mosek optimization toolbox for python manual, 2020.

[7] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[8] Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Corporation, 2013.

[9] Christopher G Atkeson and Juan Carlos Santamaria. A comparison of direct and model-based reinforcement learning. In *Proceedings of International Conference on Robotics and Automation*, volume 4, pages 3557–3564. IEEE, 1997.

[10] David S Atkinson and Pravin M Vaidya. A cutting plane algorithm for convex programming that uses analytic centers. *Mathematical Programming*, 69(1):1–43, 1995.

[11] SN Balakrishnan, Jie Ding, and Frank L Lewis. Issues on stability of adp feedback controllers for dynamical systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(4):913–917, 2008.

[12] Somil Bansal, Anayo K Akametalu, Frank J Jiang, Forrest Laine, and Claire J Tomlin. Learning quadrotor dynamics using neural network for flight control. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 4653–4660. IEEE, 2016.

[13] Mato Baotic. *Optimal control of piecewise affine systems: A multi-parametric approach.* PhD thesis, ETH Zurich, 2005.

[14] Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, SMC-13(5):834–846, 1983.

[15] Philip Becker-Ehmck, Maximilian Karl, Jan Peters, and Patrick van der Smagt. Learning to fly via deep model-based reinforcement learning. *arXiv preprint arXiv:2003.08876*, 2020.

[16] Dennis S Bernstein. Nonquadratic cost and nonlinear feedback control. *Int. J. Robust Nonlin.*, 3(3):211–229, 1993.

[17] Shubhendu Bhasin, Rushikesh Kamalapurkar, Marcus Johnson, Kyriakos G Vamvoudakis, Frank L Lewis, and Warren E Dixon. A novel actor–critic–identifier architecture for approximate optimal control of uncertain nonlinear systems. *Automatica*, 49(1):82–92, 2013.

[18] Enrico Bianconi, Javier Calvente, Roberto Giral, Emilio Mamarelis, Giovanni Petrone, Carlos Andrés Ramos-Paja, Giovanni Spagnuolo, and Massimo Vitelli. A fast current-based MPPT technique employing sliding mode control. *IEEE Trans. Ind. Electron.*, 60(3):1168–1178, 2013.

[19] Ali Bidram, Ali Davoudi, and Robert S Balog. Control and circuit techniques to mitigate partial shading effects in photovoltaic arrays. *IEEE Journal of Photovoltaics*, 2(4):532–546, 2012.

[20] Ruxandra Bobiti and Mircea Lazar. A sampling approach to finding lyapunov functions for nonlinear discrete-time systems. In *2016 European Control Conference (ECC)*, pages 561–566. IEEE, 2016.

[21] Francesco Borrelli, Alberto Bemporad, Michael Fodor, and Davor Hrovat. An mpc/hybrid system approach to traction control. *IEEE Transactions on Control Systems Technology*, 14(3):541–552, 2006.

[22] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[23] Stephen Boyd and Lieven Vandenberghe. Localization and cutting-plane methods. *From Stanford EE 364b lecture notes*, 2007.

[24] Valentina Breschi, Dario Piga, and Alberto Bemporad. Piecewise affine regression via recursive multiple least squares and multicategory discrimination. *Automatica*, 73:155–162, 2016.

[25] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[26] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[27] Lucian Busoniu, Robert Babuska, Bart De Schutter, and Damien Ernst. *Reinforcement learning and dynamic programming using function approximators*. CRC press, 2017.

[28] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer science & business media, 2013.

[29] CanadianSolar. Datasheet - dymond-cs6x-m-fg-v5.51en, 2016.

[30] Giuseppe Carannante, Ciro Fraddanno, Mario Pagano, and Luigi Piegari. Experimental performance of mppt algorithm for photovoltaic sources subject to inhomogeneous insolation. *IEEE transactions on industrial electronics*, 56(11):4374–4380, 2009.

[31] Ya-Chien Chang, Nima Roohi, and Sicun Gao. Neural lyapunov control. *arXiv preprint arXiv:2005.00611*, 2020.

[32] M-S CHEN and C-Y KAO. Control of linear time-varying systems using forward riccati equation. *Journal of dynamic systems, measurement, and control*, 119(3):536–540, 1997.

[33] Shaoru Chen, Mahyar Fazlyab, Manfred Morari, George J. Pappas, and Victor M. Preciado. Learning lyapunov functions for piecewise affine systems with neural network controllers, 2020.

[34] Chian-Song Chiu and Ya-Lun Ouyang. Robust maximum power tracking control of uncertain photovoltaic systems: A unified TS fuzzy model-based approach. *IEEE Trans. Control Syst. Technol.*, 19(6):1516–1526, 2011.

[35] Frank J Christophersen, Mato Baotić, and Manfred Morari. Optimal control of piecewise affine systems: A dynamic programming approach. In *Control and Observer Design for Nonlinear Finite and Infinite Dimensional Systems*, pages 183–198. Springer, 2005.

[36] Chen-Chi Chu and Chieh-Li Chen. Robust maximum power point tracking method for photovoltaic cells: A sliding mode control approach. *Sol. Energy*, 83(8):1370–1378, 2009.

[37] Tayfun Çimen. State-dependent riccati equation (sdre) control: a survey. *IFAC Proceedings Volumes*, 41(2):3761–3775, 2008.

[38] James R Cloutier. State-dependent riccati equation techniques: an overview. In *Proceedings of the 1997 American control conference (Cat. No. 97CH36041)*, volume 2, pages 932–936. IEEE, 1997.

[39] Adam Coates, Pieter Abbeel, and Andrew Y Ng. Apprenticeship learning for helicopter control. *Communications of the ACM*, 52(7):97–105, 2009.

[40] Felipe Leno Da Silva and Anna Helena Reali Costa. A survey on transfer learning for multiagent reinforcement learning systems. *Journal of Artificial Intelligence Research*, 64:645–703, 2019.

[41] G Silva Deaecto, José Cláudio Geromel, FS Garcia, and JA Pomilio. Switched affine systems control design with application to DC-DC converters. *IET Control Theory & A.*, 4(7):1201–1210, 2010.

[42] Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

[43] Yingwei Du, Fangzhou Liu, Jianbin Qiu, and Martin Buss. Online identification of piecewise affine systems using integral concurrent learning. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 68(10):4324–4336, 2021.

[44] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

[45] Gabriel Dulac-Arnold et al. Challenges of real-world reinforcement learning. *arXiv:1904.12901*, 2019.

[46] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019.

[47] Jack Elzinga and Thomas G Moore. A central cutting plane algorithm for the convex programming problem. *Mathematical Programming*, 8(1):134–145, 1975.

[48] Evrin B Erdem and Andrew G Alleyne. Design of a class of nonlinear controllers via state dependent riccati equations. *IEEE Transactions on Control Systems Technology*, 12(1):133–137, 2004.

[49] Trishan Esram and Patrick L Chapman. Comparison of photovoltaic array maximum power point tracking techniques. *IEEE Trans. Energy Convers.*, 22(2):439–449, 2007.

[50] Milad Farsi, Yinan Li, Ye Yuan, and Jun Liu. A piecewise learning framework for control of nonlinear systems with stability guarantees. In *Learning for Dynamics and Control (Submitted)*. PMLR, 2022.

[51] Milad Farsi and Jun Liu. Nonlinear optimal feedback control and stability analysis of solar photovoltaic systems. *IEEE Transactions on Control Systems Technology*, 28(6):2104–2119, 2019.

[52] Milad Farsi and Jun Liu. Structured online learning-based control of continuous-time nonlinear systems. *IFAC-PapersOnLine*, 53(2):8142–8149, 2020.

[53] Milad Farsi and Jun Liu. A structured online learning approach to nonlinear tracking with unknown dynamics. In *2021 American Control Conference (ACC)*, pages 2205–2211. IEEE, 2021.

[54] Milad Farsi and Jun Liu. Structured online learning-based control of continuous-time nonlinear systems. *Automatica (Submitted)*, 2022.

[55] Milad Farsi and Jun Liu. Structured online learning for low-level control of quadrotors. In *2022 American Control Conference (ACC) (Submitted)*. IEEE, 2022.

[56] Nicola Femia, Giovanni Petrone, Giovanni Spagnuolo, and Massimo Vitelli. Optimization of perturb and observe maximum power point tracking method. *IEEE Trans. Power Electron.*, 20(4):963–973, 2005.

[57] Giancarlo Ferrari-Trecate, Marco Muselli, Diego Liberati, and Manfred Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2):205–217, 2003.

[58] Vladimir Gaitsgory, Lars Grüne, and Neil Thatcher. Stabilization with discounted optimal control. *Systems & Control Letters*, 82:91–98, 2015.

[59] Claudio Gambella, Bissan Ghaddar, and Joe Naoum-Sawaya. Optimization problems for machine learning: A survey. *European Journal of Operational Research*, 290(3):807–828, 2021.

[60] Carlos E Garcia, David M Prett, and Manfred Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335–348, 1989.

[61] Andrea Garulli, Simone Paoletti, and Antonio Vicino. A survey on switched and piecewise affine system identification. *IFAC Proceedings Volumes*, 45(16):344–355, 2012.

[62] Chris Gaskett, David Wettergreen, and Alexander Zelinsky. Q-learning in continuous state and action spaces. In *Australasian joint conference on artificial intelligence*, pages 417–428. Springer, 1999.

[63] Tobias Geyer, Georgios Papafotiou, and Manfred Morari. Hybrid model predictive control of the step-down dc–dc converter. *IEEE Transactions on Control Systems Technology*, 16(6):1112–1124, 2008.

[64] Jean-Louis Goffin and Jean-Philippe Vial. On the computation of weighted analytic centers and dual ellipsoids with the projective algorithm. *Mathematical Programming*, 60(1):81–92, 1993.

[65] S Khatiry Goharoodi, Kevin Dekemele, Luc Dupre, Mia Loccufier, and Guillaume Crevecoeur. Sparse identification of nonlinear duffing oscillator from measurement data. *IFAC-PapersOnLine*, 51(33):162–167, 2018.

[66] Lars Grüne and Jürgen Pannek. Nonlinear model predictive control. In *Nonlinear model predictive control*, pages 45–69. Springer, 2017.

[67] Gurobi Optimizer Gurobi. Reference manual, gurobi optimization, 2020.

[68] Wassim M Haddad and VijaySekhar Chellaboina. *Nonlinear dynamical systems and control*. Princeton university press, 2011.

[69] Wassim M Haddad and Andrea L'Afflitto. Finite-time stabilization and optimal feedback control. *IEEE Trans. Autom. Control*, 61(4):1069–1074, 2016.

[70] W.M. Haddad and V. Chellaboina. *Nonlinear Dynamical Systems and Control: A Lyapunov-based Approach*. Princeton University Press, 2008.

[71] Dongchen Han and SN Balakrishnan. State-constrained agile missile control with adaptive-critic-based neural networks. *IEEE Transactions on Control Systems Technology*, 10(4):481–489, 2002.

[72] Ammar Haydari and Yasin Yilmaz. Deep reinforcement learning for intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[73] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797, 2019.

[74] A. C. Hindmarsh and L. R. Petzold. LSODA, ordinary differential equation solver for stiff or non-stiff system, 2005.

[75] Young-Hyok Ji, Doo-Yong Jung, Jun-Gu Kim, Jae-Hyung Kim, Tae-Won Lee, and Chung-Yuen Won. A real maximum power point tracking method for mismatching compensation in pv array under partially shaded conditions. *IEEE Transactions on power electronics*, 26(4):1001–1009, 2011.

[76] Zhong-Ping Jiang and Yuan Wang. Input-to-state stability for discrete-time nonlinear systems. *Automatica*, 37(6):857–869, 2001.

[77] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[78] Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proceedings of the Royal Society A*, 474(2219):20180335, 2018.

[79] Rudolf Emil Kalman et al. Contributions to the theory of optimal control. *Bol. soc. mat. mexicana*, 5(2):102–119, 1960.

[80] Shivaram Kalyanakrishnan and Peter Stone. An empirical analysis of value function-based and policy search reinforcement learning. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 749–756. Citeseer, 2009.

[81] Rushikesh Kamalapurkar, Joel A Rosenfeld, and Warren E Dixon. Efficient model-based reinforcement learning for approximate online optimal control. *Automatica*, 74:247–258, 2016.

[82] Rushikesh Kamalapurkar, Patrick Walters, and Warren E Dixon. Model-based reinforcement learning for approximate optimal regulation. *Automatica (Journal of IFAC)*, 64(C):94–104, 2016.

[83] Rushikesh Kamalapurkar, Patrick Walters, Joel Rosenfeld, and Warren Dixon. Model-based reinforcement learning for approximate optimal control. In *Reinforcement Learning for Optimal Feedback Control*, pages 99–148. Springer, 2018.

[84] A Kchaou, A Naamane, Y Koubaa, and N M'sirdi. Second order sliding mode-based MPPT control for photovoltaic applications. *Sol. Energy*, 155:758–769, 2017.

[85] Karel J Keesman and Karel J Keesman. *System identification: an introduction*, volume 2. Springer, 2011.

[86] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[87] Jyrki Kivinen, Alexander J Smola, and Robert C Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52(8):2165–2176, 2004.

[88] Kenji Kobayashi, Ichiro Takano, and Yoshio Sawada. A study on a two stage maximum power point tracking control of a photovoltaic system under partially shaded insolation conditions. In *2003 IEEE Power Engineering Society General Meeting*, volume 4, pages 2612–2617. IEEE, 2003.

[89] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[90] Eftichios Koutroulis and Frede Blaabjerg. A new technique for tracking the global maximum power point of pv arrays operating under partial-shading conditions. *IEEE Journal of Photovoltaics*, 2(2):184–190, 2012.

[91] Nathan O Lambert, Daniel S Drew, Joseph Yaconelli, Sergey Levine, Roberto Calandra, and Kristofer SJ Pister. Low-level control of a quadrotor with deep model-based reinforcement learning. *IEEE Robotics and Automation Letters*, 4(4):4224–4230, 2019.

[92] Fabien Lauer. On the complexity of piecewise affine system identification. *Automatica*, 62:148–153, 2015.

[93] Jae Ho Lee, HyunSu Bae, and Bo Hyung Cho. Advanced incremental conductance MPPT algorithm with a variable step size. In *12th International Power Electronics and Motion Control Conference*, pages 603–607. IEEE, 2006.

[94] George G Lendaris, Larry Schultz, and Thaddeus Shannon. Adaptive critic design for intelligent steering and speed control of a 2-axle vehicle. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 3, pages 73–78. IEEE, 2000.

[95] Frank L Lewis and Derong Liu. *Reinforcement learning and approximate dynamic programming for feedback control*, volume 17. John Wiley & Sons, 2013.

[96] Frank L Lewis and Draguna Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE circuits and systems magazine*, 9(3):32–50, 2009.

[97] Frank L Lewis and Draguna Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 9(3):32–50, 2009.

[98] Frank L Lewis, Draguna Vrabie, and Kyriakos G Vamvoudakis. Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers. *IEEE Control Systems Magazine*, 32(6):76–105, 2012.

[99] Xiao Li, Yaoyu Li, John E Seem, and Peng Lei. Detection of internal resistance change for photovoltaic arrays using extremum-seeking control MPPT signals. *IEEE Trans. Control Syst. Technol.*, 24(1):325–333, 2016.

[100] Daniel Liberzon. *Calculus of variations and optimal control theory.* Princeton university press, 2011.

[101] Hao Liu, Danjun Li, Jianxiang Xi, and Yisheng Zhong. Robust attitude controller design for miniature quadrotors. *International Journal of Robust and Nonlinear Control*, 26(4):681–696, 2016.

[102] Xin Liu and SN Balakrishnan. Convergence analysis of adaptive critic based optimal control. In *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No. 00CH36334)*, volume 3, pages 1929–1933. IEEE, 2000.

[103] XY Liu, Stefano Alfi, and Stefano Bruni. An efficient recursive least square-based condition monitoring approach for a rail vehicle suspension system. *Vehicle System Dynamics*, 54(6):814–830, 2016.

[104] Lennart Ljung and Torsten Söderström. *Theory and practice of recursive identification.* MIT press, 1983.

[105] Carlos Luis and Jérôme Le Ny. Design of a trajectory tracking controller for a nanoquadcopter. *arXiv preprint arXiv:1608.05786*, 2016.

[106] Biao Luo et al. Model-free optimal tracking control via critic-only q-learning. *IEEE TNNLS*, 27(10):2134–2144, 2016.

[107] Tobia Marcucci, Robin Deits, Marco Gabiccini, Antonio Bicchi, and Russ Tedrake. Approximate hybrid model predictive control for multi-contact push recovery in complex environments. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 31–38. IEEE, 2017.

[108] Tobia Marcucci and Russ Tedrake. Mixed-integer formulations for optimal control of piecewise-affine systems. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, pages 230–239, 2019.

[109] David Q Mayne and Hannah Michalska. Receding horizon control of nonlinear systems. In *Proceedings of the 27th IEEE Conference on Decision and Control*, pages 464–465. IEEE, 1988.

[110] Abderraouf Messai, Adel Mellit, A Guessoum, and SA Kalogirou. Maximum power point tracking using a GA optimized fuzzy logic controller and its FPGA implementation. *Sol. Energy*, 85(2):265–277, 2011.

[111] José Del R Millán, Daniele Posenato, and Eric Dedieu. Continuous-action q-learning. *Machine Learning*, 49(2):247–265, 2002.

[112] Hamidreza Modares et al. H_∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning. *IEEE TNNLS*, 26(10):2550–2562, 2015.

[113] Hamidreza Modares and Frank L. Lewis. Linear quadratic tracking control of partially-unknown continuous-time systems using reinforcement learning. *IEEE TAC*, 59(11):3051–3056, 2014.

[114] Hamidreza Modares, Frank L Lewis, and Mohammad-Bagher Naghibi-Sistani. Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica*, 50(1):193–202, 2014.

[115] Thomas M Moerland, Joost Broekens, and Catholijn M Jonker. Model-based reinforcement learning: A survey. *arXiv preprint arXiv:2006.16712*, 2020.

[116] Alivarani Mohapatra, Byamakesh Nayak, Priti Das, and Kanungo Barada Mohanty. A review on MPPT techniques of PV system under partial shading condition. *Renew. Sust. Energ. Rev.*, 80:854–867, 2017.

[117] Manfred Morari and Jay H Lee. Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23(4-5):667–682, 1999.

[118] P Moylan and B Anderson. Nonlinear regulator theory and an inverse optimal control problem. *IEEE Transactions on Automatic Control*, 18(5):460–465, 1973.

[119] John J Murray, Chadwick J Cox, George G Lendaris, and Richard Saeks. Adaptive dynamic programming. *IEEE transactions on systems, man, and cybernetics, Part C (Applications and Reviews)*, 32(2):140–153, 2002.

[120] Yu Nesterov. Cutting plane algorithms from analytic centers: efficiency estimates. *Mathematical Programming*, 69(1):149–176, 1995.

[121] Nhan T Nguyen. Model-reference adaptive control. In *Model-Reference Adaptive Control*, pages 83–123. Springer, 2018.

[122] Abdollah Noori, Milad Farsi, and Reza Mahboobi Esfanjani. Robust switching strategy for buck-boost converter. In *2014 4th International Conference on Computer and Knowledge Engineering (ICCKE)*, pages 492–496. IEEE, 2014.

[123] Abdollah Noori, Milad Farsi, and Reza Mahboobi Esfanjani. Design and implementation of a robust switching strategy for DC-DC converters. *IET Power Electron.*, 9(2):316–322, 2016.

[124] Hiren Patel and Vivek Agarwal. Matlab-based modeling to study the effects of partial shading on pv array characteristics. *IEEE transactions on energy conversion*, 23(1):302–310, 2008.

[125] Romain Pepy, Alain Lambert, and Hugues Mounier. Path planning using a dynamic vehicle model. In *2006 2nd International Conference on Information & Communication Technologies*, volume 1, pages 781–786. IEEE, 2006.

[126] John W Pierre, Ning Zhou, Francis K Tuffner, John F Hauer, Daniel J Trudnowski, and William A Mittelstadt. Probing signal design for power system identification. *IEEE Transactions on Power Systems*, 25(2):835–843, 2009.

[127] Athanasios S Polydoros and Lazaros Nalpantidis. Survey of model-based reinforcement learning: Applications on robotics. *Journal of Intelligent & Robotic Systems*, 86(2):153–173, 2017.

[128] L. Pontryagin. *Mathematical Theory of Optimal Processes*. London:Routledge, 1987.

[129] Romain Postoyan, L Buşoniu, D Nešić, and Jamal Daafouz. Stability of infinite-horizon optimal control with discounted cost. In *53rd IEEE Conference on Decision and Control*, pages 3903–3908. IEEE, 2014.

[130] Warren Buckler Powell. *Handbook of learning and approximate dynamic programming*, volume 2. John Wiley & Sons, 2004.

[131] Anna Prach, Ozan Tekinalp, and Dennis S Bernstein. Infinite-horizon linear-quadratic control by forward propagation of the differential riccati equation [lecture notes]. *IEEE Control Systems Magazine*, 35(2):78–93, 2015.

[132] Raseswari Pradhan and Bidyadhar Subudhi. Double integral sliding mode MPPT control of a photovoltaic system. *IEEE Trans. Control Syst. Technol.*, 24(1):285–292, 2016.

[133] Danil Prokhorov. Neural networks in automotive applications. In *Computational intelligence in automotive applications*, pages 101–123. Springer, 2008.

[134] Danil V Prokhorov, Roberto A Santiago, and Donald C Wunsch II. Adaptive critic designs: A case study for neurocontrol. *Neural Networks*, 8(9):1367–1372, 1995.

[135] S Joe Qin and Thomas A Badgwell. A survey of industrial model predictive control technology. *Control engineering practice*, 11(7):733–764, 2003.

[136] Zhihua Qu and Jian-Xin Xu. Model-based learning controls and their comparisons using lyapunov direct method. *Asian Journal of Control*, 4(1):99–110, 2002.

[137] S. E. Dreyfus R. E. Bellman. *Applied dynamic programming.* Princeton university press, 2015.

[138] J Prasanth Ram, T Sudhakar Babu, and N Rajasekar. A comprehensive review on solar PV maximum power point tracking techniques. *Renew. Sust. Energ. Rev.*, 67:826–847, 2017.

[139] Steffen Rebennack and Vitaliy Krasko. Piecewise linear function fitting via mixed-integer linear programming. *INFORMS Journal on Computing*, 32(2):507–530, 2020.

[140] Benjamin Recht. A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2:253–279, 2019.

[141] Miloud Rezkallah, Shailendra Kumar Sharma, Ambrish Chandra, Bhim Singh, and Daniel R Rousse. Lyapunov function and sliding mode control approach for the solar-PV grid interface system. *IEEE Trans. Ind. Electron.*, 64(1):785–795, 2017.

[142] Luis Rodrigues and Stephen Boyd. Piecewise-affine state feedback for piecewise-affine slab systems using convex optimization. *Systems & Control Letters*, 54(9):835–853, 2005.

[143] Luis Rodrigues and Jonathan P How. Observer-based control of piecewise-affine systems. *International Journal of Control*, 76(5):459–477, 2003.

[144] Moonkyung Ryu, Yinlam Chow, Ross Anderson, Christian Tjandraatmadja, and Craig Boutilier. Caql: Continuous action q-learning. *arXiv preprint arXiv:1909.12397*, 2019.

[145] Hamza Sahraoui, Larbi Chrifi, Said Drid, and Pascal Bussy. Second order sliding mode control of DC-DC converter used in the photovoltaic system according an adaptive MPPT. *Int. J. Renew. Energy Res.*, 6(2), 2016.

[146] David Scherer, Paul Dubois, and Bruce Sherwood. Vpython: 3d interactive scientific graphics for students. *Computing in Science & Engineering*, 2(5):56–62, 2000.

[147] Matthias Schreier. Modeling and adaptive control of a quadrotor. In *2012 IEEE international conference on mechatronics and automation*, pages 383–390. IEEE, 2012.

[148] P Sivakumar, Abdullah Abdul Kader, Yogeshraj Kaliavaradhan, and M Arutchelvi. Analysis and enhancement of PV efficiency with incremental conductance MPPT technique under non-linear loading conditions. *Renew. Energ.*, 81:543–550, 2015.

[149] Filippo Spertino and Jean Sumaili Akilimali. Are manufacturing i–v mismatch and reverse currents key factors in large photovoltaic arrays? *IEEE Transactions on Industrial Electronics*, 56(11):4520–4531, 2009.

[150] Nard Strijbosch, Isaac Spiegel, Kira Barton, and Tom Oomen. Monotonically convergent iterative learning control for piecewise affine systems. *IFAC-PapersOnLine*, 53(2):1474–1479, 2020.

[151] Jie Sun, Kim-Chuan Toh, and Gongyun Zhao. An analytic center cutting plane method for semidefinite feasibility problems. *Mathematics of Operations Research*, 27(2):332–346, 2002.

[152] Xiaoqiang Sun, Houzhong Zhang, Yingfeng Cai, Shaohua Wang, and Long Chen. Hybrid modeling and predictive control of intelligent vehicle longitudinal velocity considering nonlinear tire dynamics. *Nonlinear Dynamics*, 97(2):1051–1066, 2019.

[153] Kinattingal Sundareswaran, Sankar Peddapati, and Sankaran Palani. Application of random search method for maximum power point tracking in partially shaded photovoltaic systems. *IET Renewable Power Generation*, 8(6):670–678, 2014.

[154] Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine Learning Proceedings 1990*, pages 216–224. Elsevier, 1990.

[155] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[156] Siew-Chong Tan, YM Lai, and K Tse Chi. Indirect sliding mode control of power converters via double integral sliding surface. *IEEE Trans. Power Electron.*, 23(2):600–611, 2008.

[157] Boutabba Tarek, Drid Said, and MEH Benbouzid. Maximum power point tracking control for photovoltaic system using adaptive neuro-fuzzy ANFIS. In *IEEE 8th EVER Conf.*, pages 1–7, 2013.

[158] Alejandro Toriello and Juan Pablo Vielma. Fitting piecewise linear continuous functions. *European Journal of Operational Research*, 219(1):86–95, 2012.

[159] Huan-Liang Tsai. Insolation-oriented model of photovoltaic module using matlab/simulink. *Sol. Energy*, 84(7):1318–1326, 2010.

[160] Kyriakos G Vamvoudakis, Frank L Lewis, and Greg R Hudas. Multi-agent differential graphical games: Online adaptive learning solution for synchronization with optimality. *Automatica*, 48(8):1598–1611, 2012.

[161] Steven Van Vaerenbergh and Ignacio Santamaría. Online regression with kernels. In *Regularization, Optimization, Kernels, and Support Vector Machines*, pages 495–521. Chapman and Hall/CRC, 2014.

[162] Cristina Vlad, Pedro Rodriguez-Ayerbe, Emmanuel Godoy, and Pierre Lefranc. Explicit model predictive control of buck converter. In *2012 15th International Power Electronics and Motion Control Conference (EPE/PEMC)*, pages DS1e–4. IEEE, 2012.

[163] Fei-Yue Wang, Huaguang Zhang, and Derong Liu. Adaptive dynamic programming: An introduction. *IEEE computational intelligence magazine*, 4(2):39–47, 2009.

[164] Ermo Wei, Drew Wicke, David Freelan, and Sean Luke. Multiagent soft q-learning. In *2018 AAAI Spring Symposium Series*, 2018.

[165] Avishai Weiss, Ilya Kolmanovsky, and Dennis S Bernstein. Forward-integration riccati-based output-feedback control of linear time-varying systems. In *2012 American Control Conference (ACC)*, pages 6708–6714. IEEE, 2012.

[166] Marco A Wiering and Martijn Van Otterlo. Reinforcement learning. *Adaptation, learning, and optimization*, 12(3), 2012.

[167] Lifu Wu, Xiaojun Qiu, Ian S Burnett, and Yecai Guo. A recursive least square algorithm for active control of mixed noise. *Journal of Sound and Vibration*, 339:1–10, 2015.

[168] Bin Xian, Darren M Dawson, Marcio S de Queiroz, and Jian Chen. A continuous asymptotic tracking control strategy for uncertain nonlinear systems. *IEEE Transactions on Automatic Control*, 49(7):1206–1211, 2004.

[169] Weidong Xiao and William G Dunford. A modified adaptive hill climbing MPPT method for photovoltaic power systems. In *IEEE 35th Power Electron. Conf.*, pages 1957–1963. IEEE, 2004.

[170] Jinpeng Yang, Zhihao Cai, Qing Lin, and Yingxun Wang. Self-tuning pid control design for quadrotor uav based on adaptive pole placement control. In *2013 Chinese Automation Congress*, pages 233–237. IEEE, 2013.

[171] Ting Yang, Liyuan Zhao, Wei Li, and Albert Y Zomaya. Reinforcement learning in sustainable energy and electric systems: A survey. *Annual Reviews in Control*, 49:145–163, 2020.

[172] Xiong Yang et al. Guaranteed cost neural tracking control for a class of uncertain nonlinear systems using adaptive dynamic programming. *Neurocomputing*, 198:80–90, 2016.

[173] Chao Yu, Jiming Liu, and Shamim Nemati. Reinforcement learning in healthcare: A survey. *arXiv preprint arXiv:1908.08796*, 2019.

[174] Miao Yu, Chao Lu, and Yongjun Liu. Direct heuristic dynamic programming method for power system stability enhancement. In *2014 American Control Conference*, pages 747–752. IEEE, 2014.

[175] Ye Yuan, Xiuchuan Tang, Wei Zhou, Wei Pan, Xiuting Li, Hai-Tao Zhang, Han Ding, and Jorge Goncalves. Data driven discovery of cyber physical systems. *Nature communications*, 10(1):1–9, 2019.

[176] Huaguang Zhang, Lili Cui, Xin Zhang, and Yanhong Luo. Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method. *IEEE Transactions on Neural Networks*, 22(12):2226–2236, 2011.

[177] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pages 321–384, 2021.

[178] Zidong Zhang, Dongxia Zhang, and Robert C Qiu. Deep reinforcement learning for power system applications: An overview. *CSEE Journal of Power and Energy Systems*, 6(1):213–225, 2019.

[179] Yuanheng Zhu et al. Using reinforcement learning techniques to solve continuous-time non-linear optimal tracking problem without system dynamics. *IET CTA*, 10(12):1339–1347, 2016.

[180] Yuanyuan Zou and Shaoyuan Li. Robust model predictive control for piecewise affine systems. *Circuits, Systems & Signal Processing*, 26(3):393–406, 2007.