Doctoral theses at NTNU, 2023:149

Akhil S Anand

Model-based Reinforcement Learning for Variable Impedance Control

Doctoral thesis

NTNU
Norwegian University of Science and Technology
Thesis for the Degree of Philosophiae Doctor
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of Science and Technology

Akhil S Anand

Model-based Reinforcement Learning for Variable Impedance Control

Thesis for the Degree of Philosophiae Doctor

Trondheim, May 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU

Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the Degree of Philosophiae Doctor

Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

ISBN 978-82-326-7003-1 (printed ver.)
ISBN 978-82-326-7002-4 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

ITK No.: 2023-08-W

Doctoral theses at NTNU, 2023:149

Printed by NTNU Grafisk senter

Summary

This thesis is a collection of research work in the area of reinforcement learning and robotic manipulation. A set of new results on reinforcement learning focusing on model-based approaches and variable impedance control for compliant robotic manipulation are presented. One of the major challenges in robotic research at present is to develop real-world manipulation skills with human-level dexterity while being safe. Humans possess dexterous manipulation skills owing to the compliance properties of human motor control. Inspired by human manipulation, incorporating compliant control skills is understood as a promising way to achieve this goal. Impedance control offers an ideal framework to incorporate such compliant control skills in robotic manipulators. The variable impedance control framework is an extension of the impedance control framework where compliance can be adapted in real-time according to the task requirements. Even though there exist many approaches for impedance adaptation, designing optimal impedance adaptation laws for complex manipulation tasks is highly challenging and demands further research. Machine learning with its flexibility and scalability properties is a promising approach to designing impedance adaption law for a variable impedance controller. Reinforcement learning in particular is interesting as it enables robots to identify optimal stiffness profiles by interacting with their environment. However, reinforcement learning despite its promises has not been successful in real-world robotic applications. Low sample efficiency and a lack of safety guarantees are the most critical factors hindering its success in real-world robotic applications. Model-based reinforcement learning is a promising approach to address these deficiencies in reinforcement learning. This underlines the need for research on evaluating the applicability of model-based reinforcement learning methods and also to address the limitation of current model-based reinforcement learning approaches. This thesis is an attempt in this direction with a focus on

robotic manipulation and compliant robotic control. The core research focus of this thesis is enabling compliant control in robotic manipulation leveraging model-based reinforcement learning approaches while also addressing the deficiencies in model-based reinforcement learning. A part of the contribution of our work addresses the development of new model-based reinforcement learning methods. We explored combining conventional control approaches with reinforcement learning to develop novel model-based reinforcement learning methods. This thesis contributed to advancing the research on compliant control for robotic manipulation by developing and evaluating different model-based variable impedance learning control approaches for real-world applications. Additionally, this thesis addresses planning in robotic manipulation in the context of an uncertain environment and guarantees safety and stability in learning-based approaches relevant to robot control. The results of this have provided promising directions and insights into the area of reinforcement learning and robotic manipulation in general. This research is expected to contribute towards facilitating the use of reinforcement learning and compliant control skills in real-world robotic systems.

Contents

Glossary

AC	Actor-Critic
AdC	Admittance Control
AI	Artificial Intelligence
CBF	Control Barrier Functions
CEM	Cross Entropy Method
CLF	Control Lyapunov Functions
CMA-ES	Covariance Matrix Adaptation
CNN	Convolutional Neural Network
DCEM	Differentiable Cross Entropy Method
DDPG	Deep Deterministic Policy Gradients
DEVILC	Data-Efficient Variable Impedance Learning Controller
DMP	Dynamic Movement Primitive
DNN	Deep Neural Networks
DoF	Degree of Freedom
DPG	Deterministic Policy Gradients
F/M Control	Force/Motion Control
GMM	Gaussian Mixture Model
GMR	Gaussian Mixture Regression
GP	Gaussian Processes
HFMC	Hybrid Force-Motion Controller
HRC	Human-Robot Collaboration
HRI	Human-Robot Interaction
IC	Impedance Control
IFT	Inverse Function Theorem
IL	Imitation Learning
ILC	Iterative learning control

iLQG	iterative Linear-Quadratic-Gaussian
KKT	Karush-Kuhn-Tucker
LfD	Learning from Demonstration
LSTD	Least-Squares Temporal-Difference learning algorithm
MBRL	Model-based Reinforcement Learning
MDP	Markov Decision Process
ML	Machine Learning
MPC	Model Predictive Control
MPIC	Model Predictive Impedance Control
MPVIC	Model Predictive Variable Impedance Control
NN	Neural Network
OCP	Optimal Control Problem
PE	Probabilistic Ensemble NN
PETS	Probabilistic Ensembles with Trajectory Sampling
PI^2	Policy Improvement with Path Integrals
PID	Proportional–Integral–Derivative
PILCO	Probabilistic Inference for Learning Control
PNN	Probabilistic Neural Networks
PPO	Proximal Policy Optimization
ProMP	Probabilistic Movement Primitives
RL	Reinforcement Learning
RLMPC	Reinforcement Learning based Model Predictive Control
SAC	Soft Actor Critic
SEDS	Stable Estimator of Dynamical Systems
SL	Supervised Learning
SPD	Symmetric Positive Definite
SPG	Stochastic Policy Gradient
TD	Temporal Difference
TRPO	Trust Region Policy Optimization
VIC	Variable Impedance Control
VIL	Variable Impedance Learning
VILC	Variable Impedance Learning Control
WLS	Weighted Least-Squares
XAI	Explainable Artificial Intelligence

Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of philosophiae doctor (PhD) at the Norwegian University of Science and Technology (NTNU), Trondheim. The reserach work presented in this thesis has been carried out from August 2019 to January 2023 at the Department of Engineering Cybernetics under the supervision of Professor Jan Tommy Gravdahl (Dept. Engineering Cybernetics, NTNU) and co-supervision of Professor Sebastien Gros (Dept. Engineering Cybernetics, NTNU), Esten Ingar Grøtli (Dept. of Mathematics and Cybernetics, SINTEF Digital, Trondheim, Norway), and Marialena Vagia (Dept. of Mathematics and Cybernetics, SINTEF Digital, Trondheim, Norway). This work is a part of the project "Dynamic Robot Interaction and Motion Compensation" funded by the Research Council of Norway under contract number 270941.

Acknowledgments

Firstly I am would like to thank my supervisor Prof. Jan Tommy Gravdahl for providing this great opportunity to pursue a Ph.D. at the Department of Engineering Cybernetics, NTNU. Tommy has provided me with all the help I asked for throughout this journey both academically and personally. I am grateful to him for giving me the freedom and motivation to explore a wide range of research topics and to collaborate with other researchers. His insights on research work helped throughout to find confidence during the tough phases of the Ph.D. I would like to thank my co-supervisor Prof. Sebastien Gros for his valuable contribution to my research. He helped to discover and create interest in new research directions. I deeply appreciate him for taking his valuable time to discuss the theoretical aspects of our work thoroughly. It was a wonderful experience to be a part of his research group during the last phase of my Ph.D. I would like to thank my co-supervisors Esten Ingar Grøtli and Marialena Vagia for all their help in my research and advice from the very start till this day. Thank you to both for motivating me throughout and providing a very warm and welcoming research atmosphere for me. I remember the first time I explored Norway outside of Trondheim was with both of them on a visit to a research partner at Røros. I would like to thank Andreas Østvik for helping me out a lot with experiments for my first paper. It has been a very fun and interesting experience to work with him, helped me explore many interesting developments and directions for healthcare robotics. I am immensely thankful to Fares J Abu-Dakka for being one of my main academic supervisors even though he wasn't officially. He has worked very closely with me and kept me motivated during the second half of my Ph.D. We had countless hours of online discussions during this time, even during the time he became a father to three sweeties. He has helped me with all aspects of the research from exploring new research topics, and detailed discussions to academic writing.

I was lucky to work with my friends who are also my fellow Ph.D. colleagues Dirk, Shambhu, and Katrine specifically. It has been a very fun experience working with all of them, and I would like to thank them for all their efforts and contribution during my Ph.D. I would like to thank Prof. Anne Håkansson and the entire IDUN team for all the guidance and research activities we carried out as a part of the IDUN project. It was a great experience to travel to a conference and conduct a workshop with all of you despite during the Covid time. I would like to thank Mathias Arbo for his help with setting up the Franka Emika Panda robot in the lab here and also for providing me with all the necessary guidance on conducting experiments. I would like to thank all the master students I supervised during my Ph.D., Herman, Martin, Susanne, Sondre, Jens, Magne, and Shavin for all their efforts and valuable contributions to my research. I would like to thank all the administrative staff at ITK, specifically Tove, for helping with all the administrative issues. I am deeply thankful to all my friends at ITK and in Trondheim for helping to have an amazing time in and out of my Ph.D. life. I am not mentioning the names here, it will be a long list. I am grateful to all of them for being very warm and welcoming and making me a part of their life and creating so many fun and amazing memories together.

Lastly, I would like to thank the pillars of my life, my mother and sister for all their love and care. I am deeply indebted to them for all my achievements.

Publications

The results presented in this thesis are based on the following publications:

Journal Papers

1. (submitted after revision) Anand, A. S., Abu-Dakka, F. J., and Gravdahl, J. T. (2023). Deep Model Predictive Variable Impedance Control, Journal of Robotics and Autonomous Systems.

2. (submitted) Anand, A. S., Abu-Dakka, F. J., and Gravdahl, J. T. (2023). Data-Efficient Variable Impedance Control, IEEE Access.

Conference Papers

1. Anand, A.S., Kveen J.E, Abu-Dakka, F. J., Grøtli, E. I., Gravdahl, J. T., Addressing Sample Efficiency and Model-bias in Model-based Reinforcement Learning. In 21st IEEE International Conference on Machine Learning and Applications (IEEE ICMLA): IEEE ICMLA 2022.

2. Anand, A. S, Seel, K., Gjærum, V., Håkansson, A., Robinson, H., and Saad, A. (2021). Safe learning for control using control Lyapunov functions and control barrier functions: A review. Procedia Computer Science, 192, 3987-3997.

3. Anand, A. S., Østvik, A., Grøtli, E. I., Vagia, M., and Gravdahl, J. T. (2021, December). Real-time temporal adaptation of dynamic movement primitives for moving targets. In 2021 20th International Conference on Advanced Robotics (ICAR) (pp. 261-268). IEEE.

4. Anand, A. S., Myrestrand, M. H., and Gravdahl, J. T. (2022, January). Evaluation of Variable Impedance-and Hybrid Force/MotionControllers for Learning Force Tracking Skills. In 2022 IEEE/SICE International Symposium on System Integration (SII) (pp. 83-89). IEEE.

5. Anand, A.S., Reinhardt, D., Sawant, S., Gravdahl, J. T., and Gros, S., A Painless Deterministic Policy Gradient Method for Learning-based MPC, Submitted to 20th European Control Conference (ECC), 2023.

6. Kordabad A.B., Reinhardt, D, Anand, A.S. and Gros, S, Reinforcement Learning for MPC: Fundamentals and Current Challenges, Submitted to The 22nd World Congress of the International Federation of Automatic Control (IFAC), 2023.

7. (submitted) Sawant, S., Anand, A.S., Reinhardt, D. and Gros, S., Learning-based MPC from big data using reinforcement learning. Submitted to the 62nd IEEE Conference on Decision and Control (CDC), 2023.

Chapter 1

Introduction

This thesis presents a collection of research works which can be broadly classified into RL and robotic manipulation. The thesis is organized into two parts, Part I presents the research works on RL for robotic control focusing on Model-based Reinforcement Learning (MBRL), and Part II presents the research works in the area of robotic manipulation with a particular focus on compliant robotic manipulation using Variable Impedance Learning Control (VILC) approaches. The Part I is motivated on developing RL methods that can be applied to real-world systems and provide safety and stability guarantees. Part II is majorly focused on incorporating human-like compliance properties into robotic manipulation while additionally exploring the planning aspect of robotic manipulation.

This thesis addresses the research problem of *exploring the prospects of Machine Learning (ML), specifically RL for developing compliant control methods control for robotic manipulation*. This research goal is motivated by the possible benefits of safe and reliable robotic solutions in the area of healthcare, rehabilitation, elderly care, and in industrial applications. ML in general has enormous promise in the area of robotics and control, but its progress in real-world robotic and control applications is comparatively slow due to uncertainties posed by real-world physical systems. Nevertheless, the scientific community has been persistent in exploring new directions to fulfill the promise of ML in robotics, this thesis is a contribution to this cause. The works presented in the thesis may not align perfectly with each other, therefore both parts can be seen as disjoint but with the common thread of developing learning-based approaches for robotic control. In that sense, the second part can be seen as an application of the first part.

This chapter will provide a detailed introduction to the research works presented

in this thesis. The rest of this chapter will briefly discuss the importance and challenges of RL for robot control in Section 1.1 and on the promise of model-based approaches to tackle some of these challenges to bring RL to real-world robotic applications in Section 1.2. Section 1.3 will discuss the importance of compliant robotic manipulation and how this thesis tries to advance the research in this area.

1.1 Reinforcement Learning for Robot Control

RL provides a framework for learning from experiences, where robots can learn skills by directly interacting with their environment [204]. In the past decade RL has gained popularity across scientific disciplines by solving various complex control and decision-making problems relying on Deep Neural Networks (DNN) based function approximations [156]. Since the success of AlphaGo [197], owing to the success of deep-RL, RL has grown drastically over a wide range of applications ranging from control, robotics, natural language processing, recommendation systems, gaming, etc [15]. From a control theory perspective, RL is a framework to find an optimal policy or control law given an objective/cost function. In that sense RL is closely related to classical optimal control theory and dynamic programming [25]. One major advantage of RL compared to optimal control methods is that, while optimal control requires a model of the underlying system dynamics RL could find the optimal policy directly from the measured data.

In recent years RL has been widely studied in the context of robot control as it promises to enable robots to autonomously discover optimal control strategies by interacting with the environment [114, 1]. This is motivated by the increasing demand for human-level skills in robots such as robotic locomotion, manipulation, human-robot interaction, and wearable robotics [127]. Encoding complex skills in robots via conventional control approaches is very difficult in high-dimensional and continuous-space real-world robotic applications, especially in unstructured environments [70]. RL offers an alternative by allowing the robots to acquire such skills autonomously by interacting with their environment. RL has become suitable for continuous control problems with the advent of policy-gradient methods [168]. The combination of policy-gradient RL methods with DNN based function approximators forms a powerful RL tool for continuous domain control problems. An enormous amount of research has been carried out in the area of RL across various robotics and control applications in recent years, summarised in [199, 104].

In spite of its promises RL poses major difficulties by itself, and these difficulties are magnified in real-world systems as many assumptions, in theory, are rarely satisfied in practice. Applying RL in robot control has multiple challenges such as partial observability of system states, sample efficiency, reward specification,

lack of safety and stability guarantees, the curse of dimensionality, explainability, etc [57]. Especially in deep-RL, i.e RL with DNN based function approximators, the black-box nature of the DNN based function approximators translates to the RL solution. The first part of the thesis addresses some of these issues to facilitate their use in real-world applications in the future. The specific issues addressed will be summarized in the remainder of this section.

RL has high sample complexity, meaning it requires an enormous amount of interactions to learn a policy, sometimes in the order of millions depending on the complexity of the problem. This makes it difficult to apply in real-world robotic applications directly as it can be very expensive to collect interaction data in real-world robotic systems. Sim-to-real has been a prominent approach recently, where a policy is pre-trained on a simulator is transferred to the real system with minimal learning or fine-tuning on the real system [12]. This has been successfully applied in research settings, but developing high-quality simulators for most robotic applications may not be a reliable solution [231]. In addition to this, RL approaches for robotic applications have to deal with partial observability of system states and uncertainty and delays inherent to the measurements on real-world physical systems, etc. Usually, the RL-based policies are specific to the task and scenarios encountered during the learning phase. Although there are approaches available to tackle this issue, e.g. *distillation* [162] and *meta-learning* [211], this remains a major challenge.

Efficient exploration is a key aspect of RL algorithms, as RL relies on exploration strategies to gather informative experience [148]. But exploration in real-world robotic applications is more difficult owing to reasons such as satisfying mechanical and safety constraints and stability conditions [57]. One major drawback of RL compared to conventional control approaches in real-world applications is the lack of guarantees on safety or constraint satisfaction, and closed-loop stability [7]. When applying RL to real-world systems, it is important to have such guarantees during the exploration phase in addition to the final policy. This is one open area of research in RL, there are recent researches in RL to fill this gap where a lot of them combine RL with conventional control approaches to provide guarantees [33].

Apart from these issues arising from the perspective of RL, in many real-world robotic applications, the systems are equipped with highly reliable low-level controllers ranging from Proportional–Integral–Derivative (PID) control to a Model Predictive Control (MPC). In many practical applications re-engineering such controllers is not possible and it would be ideal if RL could be used to improve the performance by combining it with such control paradigms appropriately. Even though the need for such combinations arises from constraints imposed by the application,

it can be seen as an opportunity to develop reliable learning systems for real-world robotic systems by borrowing the best of control theory and RL. From the control perspective, RL can be seen as a tool to further improve the performance and adaptability of control approaches, thereby enabling the design of complex skills for robots based on control theory. Interestingly from the RL perspective, this can be seen as a way to develop RL methods for real-world applications with the necessary properties of explainability, safety, constraint satisfaction, etc.

1.2 Model-based Reinforcement Learning for Robot Control

Most existing RL approaches are model-free, requiring a large number of interactions in the order of millions in the case of complex systems to learn a policy [78]. This is feasible in simulations, whereas in real-world robotic systems, it is very expensive to collect large amounts of interaction data. Therefore, improving sample efficiency is a key contributing factor to applying RL to real-world robotics and control problems in general. MBRL mitigate this problem within the RL framework by utilizing a system dynamics model to accelerate policy estimation [53] and is found to be useful in real-world robotics applications [170]. MBRL offers multiple benefits such as sample efficiency, better exploration schemes, easier transfer learning, safety, stability, explainability, and even optimality [154].

MBRL can be seen broadly as a RL approach utilizing the model of the system dynamics for finding the optimal policy. One immediate benefit of MBRL is the improved sample efficiency as the model could reduce the amount of data required to learn the policy [203, 17]. There is no unique way to utilize the model in MBRL, the model can be utilized in different ways depending on the objective. For example, while some methods optimize a policy by explicitly planning over a learned model [53, 51, 94], other methods use a mix of model-based and model-free updates [203, 157]. Even though MBRL could improve the sample efficiency of RL, further improvement in sample efficiency is required for real-world application, especially when using DNN models [154]. This partly addresses the issue of further improving the sample efficiency in MBRL as detailed in Chapter 3.

The MBRL framework opens up the possibility to provide safety and stability guarantees to RL leveraging on the dynamics model along with the possibility of having explainable policies. [23, 227, 16] are some examples exploring this possibility. Tools from control theory such as Control Lyapunov Functions (CLF) and Control Barrier Functions (CBF) can be used to verify the safety and stability of model-based control approaches [223, 103]. For real-world safety-critical systems, it can be paramount to ensure safety, either in terms of constraint satisfaction or in terms of stability, or both. This thesis explores the aspect of ensuring safety and stability in learning-based control methods using CBFs and CLFs. Safety and stability can

be expressed as functions over the model dynamics using CLFs and CBFs, which are particularly suited to learning settings. Chapter 5 provides a thorough review of the existing methods using CLFs and CBFs in combination with learning methods including MBRL and discusses possible future directions.

Transfer learning in RL refers to utilizing the learned information for one task to speed up the learning on another task [234]. In robotic applications, even for a specific type of task, the specification can change from time to time or depending on the scenarios. In such cases, it can be expensive to learn policies for each task specification; it is thus important to learn policies that are generalizable across tasks or a set of task specifications. Even though there are approaches in model-free RL to achieve this [209], MBRL could provide an additional advantage here by leveraging on the model of the dynamics. This thesis explores this aspect of MBRL to provide better generalizability of a learned control policy over a range of task specifications in Chapter 3.

Although MBRL offers the flexibility of RL to design complex control policies with better sample efficiency, it lacks asymptotic performance compared to model-free RL [220]. The main constraint in achieving better performance with MBRL is the quality of the model, as the policy is biased toward the imperfect model. In the case of complex dynamics such as robotic systems, multistep forward predictions of such models are prone to errors due to uncertainties in the system and limitations of the model structure [2] even when using Neural Network (NN) models. Chapter 3, Chapter 4, and Appendix B of this thesis address this issue and improve the asymptotic performance of MBRL.

The aspects of MBRL discussed above are mainly centered on using a DNN-based function approximator for the policy. But deep RL in its current form has a lot of limitations because of the black-box nature of DNNs, which is translated into MBRL with DNN-based function approximators. With the motivation of building explainable MBRL approaches, holistically integrating the state-of-the-art control theory approaches in MBRL is an interesting direction. MPC is a well-established model-based control strategy that uses a model of the real system dynamics to generate input sequences that minimize a cost function under given constraints [173]. The MPC problem is solved at every time instant, in a receding-horizon fashion, delivering a control policy for the real system. For most real-world applications, building an accurate model of the real system dynamics can be challenging, especially for stochastic systems. Such inaccuracies in the model can significantly degrade the performance of the MPC scheme on controlling real-world systems. This effect is more pronounced in economics problems (problems with an economic cost/objective) rather than tracking problems where the MPC scheme tries to bring the real system to a specific reference state.

MPC is extremely suitable for tracking applications in robotics even with an inaccurate model of the dynamics. Recently, RL has been used for adjusting the MPC scheme from data to improve its closed-loop performance [67]. RL with MPC forms a unique combination in the field of learning-based MPC in the sense that learning is directly coupled to the closed-loop performance of the resulting MPC policy independent of the accuracy of the model [67]. This is in contrast to conventional learning-based MPC approaches that focus on learning an accurate dynamics model, and expecting that will improve the closed-loop performance of the resulting MPC scheme. In the context of this thesis, this approach is seen as an MBRL approach with a parameterized MPC scheme as the policy. This is unique in MBRL as it opens the door for using all useful properties of MPC such as safety, closed-loop stability, and constraint satisfaction while leveraging on the ability of RL to learn optimal policies directly from data. In the rest of this thesis, the combination of RL with MPC is termed as Reinforcement Learning based Model Predictive Control (RLMPC). In this thesis, we make efforts to advance the area of RLMPC to be suitable for real-world control applications.

An overview of the existing research and current challenges in RLMPC is provided in Appendix A. Even though this approach is highly promising for MBRL, it is not trivial to apply RL methods in RLMPC. One key criterion would be to device efficient RLMPC methods leveraging on the MPC properties for generating highly structured policies. In this direction, Chapter 4 introduces an easy-to-use RLMPC approach leveraging on a well-formulated MPC scheme. In a closer direction towards RL, it would be beneficial to formulate RLMPC schemes directly from data, which is very relevant in this era of big data. But using the existing RLMPC methods on such big data is challenging as it needs to solve an enormous amount of computationally expensive MPC schemes and requires access to the system for on-policy interactions. We address this issue in Appendix B, where a novel method to formulating an RLMPC scheme directly from data is presented.

A detailed introduction to the MBRL and RLMPC background is provided in Chapter 2.

1.3 Compliant Robotic Manipulation

A major focus of current robotic-manipulation research is on robots intelligently interacting with their environment rather than performing carefully pre-planned movements as in earlier days [27]. Robotic manipulation can be seen as a combination of trajectory planning and control problems. Complex manipulation skills from grasping, Human-Robot Interaction (HRI), robotic assembly, robotic surgery, etc., especially in unstructured environments, require both planning and control skills. Developing real-world robots with such manipulation skills remains

Figure 1.1: Example laboratory set-up we use for a robotic ultrasound on a human dummy.

an open area of research. With the growing demand for robotic solutions across areas from manufacturing, healthcare, elderly care, etc., it is vital to advance the research in this area. To that end, this thesis is a part of a bigger research project on *Robotics for Moving Objects within manufacturing and healthcare*, which aims to develop robotic manipulation skills for dynamics and unstructured environments. With the limitations of conventional control approaches for developing complex robotic manipulation skills, learning approaches have been widely adopted as a promising direction to address this challenge [158, 155]. While this thesis majorly focuses on the aspect of control, we begin the Part II by addressing the aspect of trajectory planning in uncertain environments which is integral to real-world robotic manipulation. In Chapter 8, we address the issue of targeting and manipulating a moving object by a robotic arm which is of great importance in numerous industrial applications. Learning from Demonstration (LfD) has been widely deployed in many robotic manipulation tasks, but similarly, it mainly involves static targets. Among the various LfD methods that exist, the Dynamic Movement Primitive (DMP) [89] is a widely used framework in robotic manipulation. In Chapter 8, we present a DMP framework for with real-time capabilities to manipulate a moving object.

Manipulating objects is central to how humans interact with the real world, and even with a limitation of low-frequency biological feedback loops, we possess dexterous manipulation skills. Although the exact motor control mechanisms respons-

ible for such skills remain unknown, the impedance modulation of the arm has been proposed as a key mechanism [29, 84, 100]. While in robotics the feedback control loops can be operated at much higher frequencies, human-level dexterity is seldom achieved in real-world applications. Most real-world applications using robotic manipulators traditionally relied on trajectory planning and position control which is undesirable for dexterity, safety, energy efficiency, and constrained interactions. Human muscle actuators possess impedance properties (stiffness and damping) [80] which can be adapted by the neural control to achieve various manipulation behaviors.

Motivated by the compliance properties of human manipulation, Impedance Control (IC) for robot control was introduced by Hogan in [83]. IC aims to couple the manipulator dynamics with its environment instead of treating it as an isolated system while designing control strategies. Most robotic manipulator tasks involve dynamic interactions with the environment. In [83], it is emphasized that control of position and force alone is inadequate and one needs to additionally consider the control of dynamic behavior of the robot-object interaction and that *"It is impossible to devise a controller which will cause a physical system to present an apparent behavior to its environment which is distinguishable from that of a purely physical system"*. This postulate describes the complete controlled system as an equivalent physical system. A robotic manipulator may exert a force on its environment or impose a displacement/ velocity on it, along any degree of freedom. But not both force and displacement simultaneously [202] considering the environment is rigid.

IC can be used to control the robotic manipulator end-effector while interacting with its environment [5]. IC offers a framework to develop compliant, safe, energy-efficient, and dexterous robotic manipulation skills. Unlike the more conventional control approaches, IC attempts to implement a dynamic relation between the robot's end-effector pose and wrench rather than just controlling these variables independently. The use of IC provides a feasible solution to overcome position uncertainties in order to avoid large impact forces. This is a result of controlling the robots to modulate their motion or compliance according to force feedback [106]. As a result of such properties, IC has been widely researched in the area of robotic manipulation, locomotion, HRI, and Human-Robot Collaboration (HRC) [92, 180]. As adapting the robot compliance properties is necessary to achieve human-like manipulation skills, IC is naturally extended to Variable Impedance Control (VIC) where the impedance parameters are varied during the task [91, 39]. VIC has gained popularity in robotic research as it provides the necessary scalability to IC for complex robotic manipulation tasks.

Formulating a variable impedance law for robotic manipulation tasks, especially in

contact-rich tasks, is very difficult and might need extensive tuning on the real system. With the availability of torque-controlled robots and a wide range of learning algorithms applicable, research interest has been sparked in learning such variable impedance skills for robots. Variable Impedance Learning Control (VILC) is a closed-loop VIC strategy where the variable impedance law is learned from the data generated by the controller. Learning a variable impedance law avoids all the difficulties associated with hand-designing variable impedance laws. VILC essentially relies on an impedance adaptation strategy acquired via ML strategies for an underlying VIC. A wide variety of learning-based approaches can be combined with VIC to achieve a desired VILC [4]. Some examples are Imitation Learning (IL), Iterative learning control (ILC), and RL.

IL has been used in many recent VILC works [40, 105, 134, 185]. IL-based VILC methods are generally some form of LfD methods as they often rely on demonstrations to learn from [87]. It is very suitable in robotics, especially in the case of robotic manipulation tasks, as the user can often provide demonstrations easily using kinesthetic teaching. IL can be useful in developing highly sample efficient VILC [4]. But such learning strategies can be biased to the demonstration which are often suboptimal and potentially limit the performance and generalization of the learned policies. IL is useful for tasks that are easy to demonstrate and which do not have a clear optimal way of execution, whereas RL is well suited for highly-dynamic tasks, where there is a clear measure of the success of the task [122].

Optimizing variable impedance gains/parameters can be done using ILC where the robot improves its performance iteratively. ILC based methods have been used for VILC in a range of works [44, 62, 3, 126]. An ILC improves the performance of a task by repeatedly executing the task and learning a control law using the data from the previous trials [13]. The key difference between ILC and RL is that, in RL, the control law is derived by maximizing a reward function defined by the task requirements. One advantage of ILC compared to RL is its sample efficiency. But even when a model of the dynamics is not available, RL offers better performance and can be applied to a broader range of problems [229].

With the advances made in RL for robotics [114], it has been widely adopted for VILC in recent times [4]. One of the first work in this area was [109], where the Natural Actor-Critic algorithm is used in an episodic way to learn the stiffness matrix for a VIC. Policy Improvement with Path Integrals (PI^2) algorithm was used in [34], to find the optimal impedance parameters. A detailed review of existing RL based VILC approaches are presented in Chapter 7. The issues with RL discussed in Section 1.1 are also reflected in RL-based VILC approaches. We identified data-efficiency and transferability across tasks as two key issues in utilizing the potential of RL in VILC. In the second part of this thesis, we aim to advance the

Part I	
Chapter 2:	Background on MBRL
Chapter 3:	A new improved MBRL framework
Chapter 4:	A new RLMPC algorithm
Chapter 5:	A review on existing safe learning for control
Part II	
Chapter 6:	Background on VIC
Chapter 7:	A review on existing RL-based VILC methods
Chapter 8:	A new improved DMP framework
Chapter 9:	Evaluation of compliant controllers within MBRL framework
Chapter 10:	A new data-efficient VILC method
Chapter 11:	A new scalable VILC framework

Table 1.1: Structure of the thesis mentioning the contribution in all the chapters in Part I and Part II.

area VILC focusing on MBRL approaches for developing human-level compliance in robots. The last three chapters (9 - 10) in Part II of this thesis present different VILC approaches focusing on MBRL.

1.4 Contributions

This thesis is organized into two parts and twelve chapters. Part I consists of Chapter 2 - Chapter 5 presents the research works on RL for robotic control focusing on MBRL. Part II consists of Chapter 6 - Chapter 11 presents the research works in the area of robotic manipulation with a particular focus on compliant robotic manipulation using VILC approaches. Table 1.1 depicts the structure of the thesis.

Chapter 2 provides the necessary background on MBRL and RLMPC for Part I. Chapter 6 provides the background on VIC for Part II. In the following, we summarize the topic and contributions of each chapter in this thesis.

1.4.1 Part I: Reinforcement Learning for Control

Part I contains three main research works focusing on advancing RL for control applications. Two related results in Appendix A and B also belong to this part.

Chapter 3: Addressing Sample Efficiency and Model-Bias in Model-based RL

This chapter aims to address some of the major drawbacks of existing MBRL methods and to develop an improved MBRL framework. MBRL promises to be an effective way to bring RL to real-world robotic systems by offering a sample efficient learning approach compared to model-free RL. However, at the present, MBRL approaches struggle to match the performance of model-free ones. The work presented in this chapter attempts to fill this gap by improving the performance of MBRL while further improving its sample efficiency. To improve the sample efficiency, an exploration strategy is formulated which maximizes the information gain. The asymptotic performance is improved by compensating for the model bias using a model-free critic. The proposed approach has been evaluated empirically on four RL benchmarking tasks in the openAI gym framework. The results of the empirical evaluation demonstrated improved data efficiency, performance and generalizability of the proposed MBRL approach compared to the state-of-the-art methods.

The MBRL framework proposed in this chapter represents the dynamic model using ensembles of NNs and uses a Cross Entropy Method (CEM)-based [31] MPC as the policy. This work focuses on the aspects of sample efficiency, performance, and efficient transfer of policies between tasks within the standard MBRL framework. The main contributions of this work are:

- Improving the sample efficiency and transfer of policies between tasks by designing exploration policies targeted at maximizing the information gain from the region of interest using the uncertainty estimate of the model.
- To compensate for the inaccuracies in the learned model, a critic-value function estimated from the real data is used as a terminal value during the policy optimization, thereby improving the asymptotic performance of the MBRL framework.

This chapter is based on the following publication (accepted):

Akhil S Anand, Jens Erik Kveen, Fares Abu-Dakka, Esten Ingar Grøtli, Jan Tommy Gravdahl (2022, December). Addressing Sample Efficiency and Model-bias in Model-based Reinforcement Learning. In 21st IEEE International Conference on Machine Learning and Applications (IEEE ICMLA): IEEE ICMLA 2022.

Chapter 4: Deterministic Policy Gradient Method for Learning-based MPC

This chapter presents a control theory oriented MBRL approach using the RLMPC framework. The combination of RL and MPC has gained a lot of interest in the

recent literature as a way of computing the optimal policies from MPC schemes based on inaccurate models. In that context, the Deterministic Policy Gradient (DPG) methods are often observed to be the most reliable class of RL methods to improve the MPC closed-loop performance. The DPG methods are fairly easy to formulate when used with compatible function approximation as an advantage function. However, this formulation requires an additional value function approximation, often carried out using DNNs. In this chapter, we propose to estimate the required value function approximation as a first-order expansion of the value function estimate from the MPC scheme providing the policy. The proposed approach drastically simplifies the use of DPG methods for learning-based MPC as no additional structure for approximating the value function needs to be constructed. We illustrate the proposed approach with two numerical examples of varying complexity.

This chapter introduces a novel actor-critic formulation using the DPG theorem [198] with only an MPC scheme as an actor. The proposed method uses the MPC-based actor to approximate the value function, and hence eliminates the need for an additional approximation structure. To this end, we exploit the result proposed by Gros and Zanon [67] which states that, under mild assumptions, the optimal value function can be approximated by MPC and that the resulting policy and value functions satisfy the Bellman equations. A key feature of this approach is that the value function approximator is designed to be compatible with the policy gradient. The main contributions of this work are:

- A novel DPG-based actor-critic method for MPC is formulated where the value function in the critic is approximated using MPC actor itself.
- A simplified DPG algorithm for learning-based MPC is formulated by circumventing the need to construct an additional structure for approximating the value function in the compatible critic function approximation in the DPG formulation.

This chapter is based on the following paper:

Akhil S Anand, Dirk Reinhardt, Shambhuraj Sawant, Jan Tommy Gravdahl, Sbastien Gros, A Painless Deterministic Policy Gradient Method for Learning-based MPC, Submitted to 20th European Control Conference (ECC), 2023.

Chapter 5: A Survey on Safe Learning for Control

This chapter deals with the aspects of safety and stability guarantees in learning-based control methods. Real-world autonomous systems are often controlled using

conventional model-based control methods. But if accurate models of a system are not available, these methods may be unsuitable. For many safety-critical systems, such as robotic systems, a model of the system and a control strategy may be learned using data. When applying learning to safety-critical systems, guaranteeing safety during learning as well as testing/deployment is paramount. A variety of different approaches for ensuring safety exists, but the published works are cluttered and there are few reviews that compare the latest approaches. CBF has been widely employed to guarantee safety for control methods [223]. CBF gained popularity within the conventional control community during recent years and has been utilized more as a safety filter for an existing nominal controller [6]. In a different but slightly more conservative approach, safety can be guaranteed indirectly by stability guarantees of the closed-loop system based on Lyapunov stability verification, utilizing CLF [103]. While CBFs provides an option to incorporate safety in terms of constraint satisfaction, CLF are used to define safety in terms of stability. A combination of CBFs and CLFs can be used to guarantee a safe and stabilizing controller [177], which is also utilized in learning to guarantee stability [95]. These two promising approaches have been widely adopted to guarantee safety for learning-based robust control of uncertain dynamical systems.

This chapter presents a rigorous review of learning methods that incorporate CBFs and CLFs and their combination for safe learning-based control. The review summarizes the existing learning-based methods for safe control of dynamical systems with uncertainty, utilizing CBF and CLF. The relevant references are divided into three main categories, grouped by the learning method that CBFs and CLFs are combined with, namely RL, online and offline Supervised Learning (SL). The similarities and differences between the methods used in the review references are highlighted and their suitability in different scenarios is discussed. It is observed that, despite steady progress, there still exists a large gap between the theory and practical application of the methods. Because using CLFs and CBFs with learning is a rather new approach, a major challenge is demonstrating their capabilities on real-world safety-critical systems. This widens the scope for future research in the area.

This chapter is based on the following publication [8]:

Anand, A., Seel, K., Gjærum, V., Håkansson, A., Robinson, H., and Saad, A. (2021). Safe learning for control using control Lyapunov functions and control barrier functions: A review. Procedia Computer Science, 192, 3987-3997.

1.4.2 Part II: Learning-based Complaint Robotic Manipulation

Part II contains four main research works focusing on advancing RL for control applications. Chapter 7 provides a review of the existing RL-based VILC approaches in robotic manipulation. Chapter 8 focuses on the trajectory planning aspect of robotic manipulation whereas chapters 9 -10 focus on learning-based compliant robotic manipulation.

Chapter 7: Survey on RL-based Variable Impedance Learning Control

This chapter presents a detailed survey on RL-based VILC approaches for robotic manipulation. VILC combines VIC with learning-based control strategies for learning a variable impedance law. The majority of the VILC approaches make use of imitation learning, iterative learning, or reinforcement learning. Among different learning approaches RL offers the flexibility and scalability for VILC on a wider range of robotic problems such as manipulation, locomotion, and HRI. This survey discusses the advantages and disadvantages of different VILC approaches and their suitability for various applications. The survey provides a tabular comparison of the approach on five different criteria: (i) data-efficiency, (ii) transferability, (iii) model-based or model-free, (iv) computational effort, and (v) force-/ position-based VIC. This chapter serves as a detailed review of existing works for the chapters 9 -10.

This chapter is not part of any publication.

Chapter 8: Real-time Dynamic Movement Primitives for Moving Targets

This chapter deals with the problem of learning-based trajectory planning for robotic manipulation in uncertain environments. This chapter is aimed at extending the standard DMP framework to adapt to real-time changes in the task execution time while preserving its style characteristics. We propose an alternative polynomial canonical system and an adaptive law allowing a higher degree of control over the execution time. The extended framework has a potential application in robotic manipulation tasks that involve moving objects demanding real-time control over the task execution time. The existing methods require a computationally expensive forward simulation of DMP at every time step which makes it undesirable for integration in real-time control systems. To address this deficiency, the behavior of the canonical system has been adapted according to the changes in the desired execution time of the task performed. An alternative polynomial canonical system is proposed to provide increased real-time control on the temporal scaling of DMP system compared to the standard exponential canonical system. The developed method was evaluated on scenarios of tracking a moving target where the desired tracking time is varied in real-time. The results presented show that the extended

version of DMP provides better control over the temporal scaling during the execution of the task. We have evaluated our approach on a UR5 robotic manipulator for tracking a moving object.

We define *real-time control of the execution time* as, how the DMP system is adapting to real-time changes in its desired execution time during the task. In order to achieve this, we only manipulate the temporal scaling of DMP system while preserving its spatial properties. We formulate two methods to achieve efficient real-time temporal scaling, (i) a control law to vary the temporal scaling term of the standard exponential canonical system and (ii) an alternate polynomial-based canonical system with a suitable control law for temporal scaling. This is useful in a manipulation task with an underlying DMP planner, where the task execution time needs to be changed during the execution phase. Additionally, in the case of moving targets, velocity feedback of the target and a simple estimate of the goal position based on the current target position and velocity are included in the DMP system.

This chapter is based on the following publication [11]:

Anand, A. S., Østvik, A., Grøtli, E. I., Vagia, M., and Gravdahl, J. T. (2021, December). Real-time temporal adaptation of dynamic movement primitives for moving targets. In 2021 20th International Conference on Advanced Robotics (ICAR) (pp. 261-268). IEEE.

Chapter 9: Evaluation of Compliant Controllers for Learning Force Tracking Skills

Chapter 9 presents an evaluation of two prominent force control methods, VIC and Hybrid Force-Motion Controller (HFMC) in a robot learning framework for contact-rich interaction tasks demanding force and motion tracking. The controllers are evaluated on a Franka Emika Panda robotic manipulator for a robotic interaction task using a MBRL algorithm, Probabilistic Inference for Learning Control (PILCO). The PILCO algorithm is chosen for evaluating the controller considering its high sample efficiency which facilitates learning directly in the experimental set-up in a handful of trails. Utilizing the learning framework to find the optimal controller parameters has significantly improved the performance of the controllers.

It was shown that combining a learning-based approach with force controllers has the ability to improve robotic interaction control. For HFMC, the framework was used to learn direct strategies for its damping- and stiffness parameters. For VIC, strategies were learned for the parameters of an adaptation law. Both controllers showed significant improvement in force tracking ability by introducing model-based learning. Introducing learning led to faster convergence to the desired force

in VIC, and to a significant improvement in the force tracking error in HFMC.

This chapter is based on the following publication [10]:

Anand, A. S., Myrestrand, M. H., and Gravdahl, J. T. (2022, January). Evaluation of Variable Impedance-and Hybrid Force/MotionControllers for Learning Force Tracking Skills. In 2022 IEEE/SICE International Symposium on System Integration (SII) (pp. 83-89). IEEE.

Chapter 10: A Data-Efficient Variable Impedance Learning Control Framework

This chapter addresses the issue of data-efficiency in VILC to facilitate their easy and reliable application to real-world robotic manipulation tasks. In this chapter, we introduce a Reinforcement Learning (RL) based approach called Data-Efficient Variable Impedance Learning Controller (DEVILC) to learn the variable impedance controller through real-world interaction of the robot. More concretely, we use a model-based RL approach where, after every interaction, the robot iteratively learns a probabilistic model of its dynamics using the Gaussian process regression model. The model is then used to optimize a neural-network policy that modulates the impedance of the robot such that the long-term reward for the task is maximized.

Thanks to the model-based RL framework, Data-Efficient Variable Impedance Learning Controller (DEVILC)allows a robot to learn the VIC policy with only a few interactions, making it practical for real-world applications. We evaluate Data-Efficient Variable Impedance Learning Controller (DEVILC) on a Franka Emika Panda robotic manipulator for different manipulation tasks in the Cartesian space in simulations and experiments. The results show that Data-Efficient Variable Impedance Learning Controller (DEVILC) is a promising direction toward autonomously learning compliant manipulation skills directly in the real world through interactions. The results show that this approach is a promising direction toward learning compliant manipulation skills.

This approach presented in this chapter focuses on improving the sample efficiency of VILC approaches for real-world robotic applications demanding real-time impedance adaptation. The main contributions of this chapter are:

- a Model-based VILC framework using Gaussian Processes (GP) models and using the evolution strategy, Covariance Matrix Adaptation (CMA-ES) to optimize a NN policy.
- demonstrates a highly data-efficient approach for learning impedance adaptation strategy for robotic manipulation.

This chapter is based on the following paper under review:

Anand, A. S., Abu-Dakka, F. J., and Gravdahl, J. T. (2023). Data-Efficient Variable Impedance Control, IEEE Access.

Chapter 11: Deep Model Predictive Variable Impedance Control

This chapter addresses crucial issues of data-efficiency, performance, and task transferability in VILC in order to develop a scalable and highly reliable VILC approach for real-world robotic applications. We propose a deep Model Predictive Variable Impedance Controller for compliant robotic manipulation which combines Variable Impedance Control (VIC) with Model Predictive Control (MPC). A generalized Cartesian impedance model of a robot manipulator is learned using an exploration strategy maximizing the information gain. This model is used within an MPC framework to adapt the impedance parameters of a low-level variable impedance controller to achieve the desired compliance behavior for different manipulation tasks without any retraining or fine-tuning. The deep Model Predictive Variable Impedance Control approach is evaluated using a Franka Emika Panda robotic manipulator operating on different manipulation tasks in simulations and real experiments. The proposed approach was compared with model-free and model-based reinforcement approaches in variable impedance control for transferability between tasks and performance.

In this chapter, we propose a deep Model Predictive Variable Impedance Control (MPVIC) framework, where a NN based Cartesian impedance model of the robotic manipulator is used in a CEM-based MPC for online adaptation of the impedance parameters of a VIC. This deep MPVIC framework is utilized to learn impedance adaptation strategy for various robotic manipulation tasks by specifying a suitable cost function. The main contributions of this work are:

- a novel VIC framework, we call it deep MPVIC, which combines a CEM-based MPC with Probabilistic Ensemble NN (PENN) dynamical model for compliant robotic manipulation.
- the deep MPVIC framework learns a generalized Cartesian impedance model of the robot to facilitate the transferability between completely different manipulation tasks without any need of relearning the model.
- an uncertainty-based exploration scheme is integrated into the proposed framework to facilitate learning a generalized model efficiently from fewer samples.
- an extensive evaluation in simulation and real setups, in addition to a comparison between our approach and the state-of-the-art model-free and model-based RL approaches on transferability and performance.

This chapter is based on the following paper under review (publically available on arxiv currently [9]):

Anand, A. S., Abu-Dakka, F. J., and Gravdahl, J. T. (2023). Deep Model Predictive Variable Impedance Control, Journal of Robotics and Autonomous Systems.

Part I

Model-based Reinforcement Learning

Chapter 2

Preliminaries on Model-based Reinforcement Learning

This chapter introduces the preliminary background for Model-based Reinforcement Learning (MBRL) and Reinforcement Learning based Model Predictive Control (RLMPC) relevant to this part of the thesis.

2.1 Model-based Reinforcement Learning

Reinforcement learning methods can be categorized into two main categories; model-based RL (MBRL) and model-free RL. Model-free methods don't use any model of the transition dynamics of the underlying Markov Decision Process (MDP). They learn the optimal policy directly from the interaction data considering the underlying MDP as a black box responding to actions taken. Whereas, model-based methods either use a known model describing the transition dynamics and reward function of the underlying MDP, or they learn a model by sampling transitions in the environment. Model-based methods rely on this model to learn an optimal policy. Since the focus of this thesis is on model-based RL, this section provides the necessary preliminaries on MBRL, assuming the reader has basic knowledge is RL.

RL formulates an optimization problem as Markov Decision Process (MDP) [19], defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho, \gamma, H)$. MDP constitutes the system states, $\mathbf{s}_t \in \mathcal{S}$, actions, $\mathbf{a}_t \in \mathcal{A}$, a transition function, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$. $\mathcal{T}$ represents the probability of transitioning to a new state $\mathbf{s}_{t+1}$ from the current $\mathbf{s}_t$ by applying action $\mathbf{a}_t$; i.e:

$$\mathbf{s}_{t+1} \sim \mathcal{T}(\cdot|\mathbf{s}_t, \mathbf{a}_t) \tag{2.1}$$

and can be either deterministic or stochastic. $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ represents the scalar reward function, where a scalar reward value r_t is received for every action applied to the system such that: $r_t = \mathcal{R}(\mathbf{s}_t, \mathbf{a}_t)$. The initial state distribution is denoted by $\rho(\mathbf{s})$, $\gamma \in [0, 1]$ is the discount factor, and H is the horizon of the process. $\boldsymbol{\pi} : \mathcal{S} \to \mathcal{A}$ is an agent or policy that maps the state to actions.

MBRL utilize a parametrized model $\mathcal{T}_\theta$ predicting the state transition as a proxy for the actual transition function (2.1),

$$\mathbf{s}_{t+1} \sim \mathcal{T}_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t, \mathbf{a}_t)\,. \tag{2.2}$$

MBRL solves for an optimal policy $\boldsymbol{\pi}^\star$ that maximizes the expected total reward $J(\boldsymbol{\pi}) = \mathbb{E}_{\substack{\mathbf{a}_t \sim \boldsymbol{\pi} \\ \mathbf{s}_t \sim \mathcal{T}_\theta}} \left[\sum_{t=0}^{H} \gamma^t \mathcal{R}\left(\mathbf{s}_t, \mathbf{a}_t\right)\right]$, given by

$$\boldsymbol{\pi}^* = \arg\max_{\boldsymbol{\pi}} \mathbb{E}_{\substack{\mathbf{a}_t \sim \boldsymbol{\pi}(\mathbf{s}_t) \\ \mathbf{s}_{t+1} \sim \mathcal{T}_\theta(\mathbf{s}_t, \mathbf{a}_t)}} \left[\sum_{t=0}^{H} \gamma^t \mathcal{R}\left(\mathbf{s}_t, \mathbf{a}_t\right)\right]. \tag{2.3}$$

The state-value function $V^{\boldsymbol{\pi}}(\mathbf{s})$ associated with the policy $\boldsymbol{\pi}$ is defined as the expectation of the cumulative return given by following a policy $\boldsymbol{\pi}$ from state $\mathbf{s}$:

$$V^{\boldsymbol{\pi}}(\mathbf{s}) = \mathbb{E}_{\boldsymbol{\pi}, \mathcal{T}} \left[\sum_{k=0}^{H} \gamma^k r_{t+k} \mid \mathbf{s}_t = \mathbf{s}\right]. \tag{2.4}$$

The action-value function $Q^{\boldsymbol{\pi}}(\mathbf{s}, \mathbf{a})$ associated with the policy $\boldsymbol{\pi}$ is defined as the expectation of the cumulative return given by following a policy $\boldsymbol{\pi}$ from state $\mathbf{s}$ after taking action $\mathbf{a}$:

$$Q^{\boldsymbol{\pi}}(\mathbf{s}, \mathbf{a}) = \mathbb{E}_{\boldsymbol{\pi}, \mathcal{T}} \left[\sum_{k=0}^{H} \gamma^k r_{t+k} \mid \mathbf{s}_t = \mathbf{s}, \mathbf{a}_t = \mathbf{a}\right]. \tag{2.5}$$

Discussing the solution of MDPs is often best done via the Bellman equations defining implicitly the optimal value function $V^\star : \mathcal{S} \to \mathbb{R}$ and the optimal action-value function $Q^\star : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as

$$V^\star\left(\mathbf{s}\right) = \max_{\mathbf{a}} Q^\star\left(\mathbf{s}, \mathbf{a}\right) \tag{2.6a}$$

$$Q^\star\left(\mathbf{s}, \mathbf{a}\right) = r\left(\mathbf{s}, \mathbf{a}\right) + \gamma \mathbb{E}\left[V^\star\left(\mathbf{s}_+\right) \mid \mathbf{s}, \mathbf{a}\right]. \tag{2.6b}$$

The optimal policy $\boldsymbol{\pi}^\star$ can be written as:

$$\boldsymbol{\pi}^\star\left(\mathbf{s}\right) = \arg\min_{\mathbf{a}} Q^\star\left(\mathbf{s}, \mathbf{a}\right). \tag{2.7}$$

MBRL aims to solve for the optimal policy $\boldsymbol{\pi}^\star$ that maximizes our expected return $Q^{\boldsymbol{\pi}}(\mathbf{s}, \mathbf{a})$:

$$\boldsymbol{\pi}^\star = \arg\max_{\boldsymbol{\pi}} Q^{\boldsymbol{\pi}}(\mathbf{s}, \mathbf{a}) = \arg\max_{\boldsymbol{\pi}} \mathbb{E}_{\boldsymbol{\pi}, \mathcal{T}} \left[\sum_{k=0}^{H} \gamma^k r_{t+k} \mid \mathbf{s}_t = \mathbf{s}, \mathbf{a}_t = \mathbf{a} \right]. \quad (2.8)$$

There is at least one optimal policy, denoted by $\boldsymbol{\pi}^\star$, which is better or equal than all other policies [204]. In the planning and search literature, the above problem is typically formulated as a cost minimization problem [182], instead of a reward maximization problem. That formulation is interchangeable with our presentation by negating the reward function.

One way to perform MBRL is by learning the transition dynamics, $\mathcal{T}_\theta$ from observed data. MBRL iterates between model learning and policy optimization. The policy optimization step uses the learned model differently depending on the MBRL algorithm [220, 153, 170]. Additionally, the reward function $\mathcal{R}$ is learned in many MBRL approaches from the data [153]. Data collected by interacting with the environment is used to learn $\mathcal{T}_\theta$. The model and reward function provide reversible access to the MDP.

2.1.1 Model Learning

All MBRL method utilizes some form of model about the environment dynamics. Unlike in conventional model-based control approaches, in MBRL it is often learned from interaction data iteratively while optimizing the policy. There are several ways of representing the dynamic model, and the choice of representation depends on the complexity of the dynamics. In this thesis, we focus on dynamic models $\mathcal{T}_{\boldsymbol{\theta}}$ which attempt to learn the state transition dynamics of the system. Note that the reward function $\mathcal{R}_{\boldsymbol{\phi}}$ can also be learned separately or can be a part of the learned transition model where it predicts the reward along with the next state. We represent them separately for the sake of clarity. Given a batch $\mathbf{s}_t, \mathbf{a}_t, r_t, \mathbf{s}_{t+1}$ of one-step transition data, we consider the dynamic model predicting the forward dynamics of the system and the corresponding reward model as

$$\mathbf{s}_{t+1} \leftarrow \mathcal{T}_{\boldsymbol{\theta}}(\mathbf{s}_t, \mathbf{a}_t), \quad (2.9a)$$

$$\mathbf{r}_t \leftarrow \mathcal{R}_{\boldsymbol{\phi}}(\mathbf{s}_t, \mathbf{a}_t). \quad (2.9b)$$

If the model is known or can be estimated easily then the representation can be often intuitive. Such model representation can be for example in the form of a

Algorithm 1 MBRL

Initialize $\boldsymbol{\pi}, \mathcal{T}_{\boldsymbol{\theta}}$, and an empty dataset $\mathcal{D}$
for $i \leftarrow 1$ **to** N **do**
  for $t \leftarrow 1$ **to** T **do**
    Apply $\mathbf{a}_t \leftarrow \boldsymbol{\pi}(\mathbf{s}_t)$ to the system : $\mathcal{D} = \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, r_t)\}$
  end
  Update $\mathcal{T}_{\boldsymbol{\theta}}$ and $\mathcal{R}_{\boldsymbol{\phi}}$ using on $\mathcal{D}$ via maximum likelihood:
  $\theta \leftarrow \operatorname{argmax}_\theta \mathbb{E}_{\mathcal{D}} \left[\log p_\theta \left(\mathbf{s}_{t+1} \mid \mathbf{s}, \mathbf{a}\right)\right]$
  $\phi \leftarrow \operatorname{argmax}_\phi \mathbb{E}_{\mathcal{D}} \left[\log p_\theta \left(r_t \mid \mathbf{s}, \mathbf{a}\right)\right]$
  Optimize $\boldsymbol{\pi}$ under $\mathcal{T}_{\boldsymbol{\theta}}$ using $\mathcal{R}_{\boldsymbol{\phi}}$ using (2.8)
end

linear parametric model, a differential equation describing the system dynamics, or a set of rules defining a board game, etc. But such models are often not expressive enough to represent complex non-linear and uncertain dynamics which is often the case in robotics. Function approximations such as GP [28] and DNN [76] offer an alternative way to model such dynamics directly from data. The approach presented in this thesis relies on such models and their details are provided in the corresponding chapters. A general form of MBRL algorithm using DNN based models is provided in Algorithm 1

2.1.2 Model Utilization

Utilizing a model of the system dynamics is the key aspect of MBRL differentiating it from model-free RL. It provides various advantages such as optimality, data-efficiency, efficient exploration, easier transfer learning, safety guarantees, and explainability [154]. How the model is used within MBRL to find an optimal policy is the key differentiating factor between most MBRL algorithms. MBRL algorithms can be broadly categorized into two categories based on how the model is utilized to find the optimal policy [220] as follows:

- Dyna-style Algorithms: This class of algorithm iterates between model learning and policy optimization until the optimal policy is obtained. First, they use the data collected under the current policy to learn a model using supervised learning. Secondly, they use the model as a simulator to optimize policy in a model-free setting. Essentially any of the model-free RL algorithms can be coupled with learning a model iteratively as a Dyna-style algorithm. Some examples of Dyna-style algorithms are [141, 129, 52, 94].

- Policy Search algorithms: This class of algorithms also iterate between model learning and policy optimization until the optimal policy is obtained

similar to Dyna-style algorithms. But they exploit the model properties more effectively than Dyna-style algorithms. For example, these algorithms can utilize the analytic gradient of the RL objective with respect to the policy, and improve the policy. Some examples of policy search algorithms using model derivates are [53, 178, 135, 77]. Some approaches utilize the learned model in an MPC setting to optimize the policy [51, 157].

Policy search algorithms are more sample efficient compared to Dyna-style algorithms owing to their efficient use of the model rather than using it to generate simulated data. The MBRL approaches presented in this thesis fall into the category of policy search algorithms.

2.2 Reinforcement Learning based Model Predictive Control

RLMPC is introduced in [67] as a novel learning-based MPC where model-free RL approaches are used to improve the closed-loop performance of an MPC policy. We categorize RLMPC as a class of MBRL approach where an MPC policy is optimized using RL using a dynamics model which is either learned or known beforehand.

Model Predictive Control

Note: we use slightly different notations in the case of MPC as typically used in control theory. The reward, r in the MDP is replaced with a cost ℓ which is generally represented in MPC as the stage cost. State and action/control input are represented using $\mathbf{x}$ and $\mathbf{u}$ respectively. A more control-oriented form of the state transition (2.1) is given by:

$$\mathbf{s}_{k+1} = \mathbf{f}(\mathbf{s}_k, \mathbf{a}_k, \mathbf{d}_k)\,, \tag{2.10}$$

where $\mathbf{d}_k$ is a stochastic disturbance. Setting the disturbance to zero yields the deterministic case.

A solution to a finite horizon MDP can be delivered by MPC under some assumptions as discussed in [67]. MPC builds a sequence of actions that minimizes a cost function over the MPC horizon. In a fully parameterized form, an MPC scheme reads as

$$V_{\boldsymbol{\theta}}(\mathbf{s}_k) = \min_{\mathbf{u},\mathbf{x}} \quad T_{\boldsymbol{\theta}}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \ell_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{u}_k), \tag{2.11a}$$

$$\text{s.t.} \quad \mathbf{x}_0 = \mathbf{s}_k, \tag{2.11b}$$

$$\mathbf{x}_{k+1} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{u}_k), \tag{2.11c}$$

$$\mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_k,\, \mathbf{u}_k) \leq 0, \tag{2.11d}$$

where $\mathbf{s}_k$ is the current state of the system, $\ell_{\boldsymbol{\theta}}$, $T_{\boldsymbol{\theta}}$ denote the respective stage cost and terminal cost. The (deterministic) dynamic model is denoted by $\mathbf{f}_{\boldsymbol{\theta}}$ and the inequality constraints by $\mathbf{h}_{\boldsymbol{\theta}}$. Variables $\mathbf{x}_k$ and $\mathbf{u}_k$ label the predicted state and control input, respectively. For a given system state $\mathbf{s}_k$, problem (2.11) produces a complete profile of control inputs $\mathbf{u}^\star = \{\mathbf{u}_0^\star, \ldots, \mathbf{u}_{N-1}^\star\}$ and corresponding state predictions $\mathbf{x} = \{\mathbf{x}_0^\star, \ldots, \mathbf{x}_N^\star\}$. Only the first element $\mathbf{u}_0^\star$ of the input sequence $\mathbf{u}^\star$ is applied to the system. At the next sampling step, a new state $\mathbf{s}_k$ is received, and problem (2.11) is solved again, producing a new $\mathbf{u}^\star$ and a new $\mathbf{u}_0^\star$. MPC hence yields a policy:

$$\boldsymbol{\pi}_{\mathrm{MPC}}(\mathbf{s}) = \mathbf{u}_0^\star, \tag{2.12}$$

with $\mathbf{u}_0^\star$ solution of (2.11) for $\mathbf{s}$ given. For $\gamma \approx 1$, policy (2.12) can provide a good approximation of the optimal policy $\boldsymbol{\pi}^\star$ for an adequate choice of prediction horizon N, terminal cost T_θ and if the MPC model $\mathbf{f}$ approximates the true dynamics sufficiently well. In that context, the latter is arguably the major weakness as many systems are challenging to model accurately. Furthermore, within a modeling structure, selecting the model $\mathbf{f}$ that yields the best closed-loop performance $J(\boldsymbol{\pi}_{\mathrm{MPC}})$ is very difficult. And there is in general no guarantee that the most accurate model yields the best closed-loop performance.

In addition to delivering a policy, the MPC scheme can also deliver a model of the optimal action-value function, using the modified form [67]

$$\begin{aligned} Q_{\boldsymbol{\theta}}(\mathbf{s}_k, \mathbf{a}_k) = \min_{\mathbf{u},\mathbf{x}} \quad & \text{(2.11a)} && \text{(2.13a)} \\ \text{s.t.} \quad & \text{(2.11b)} - \text{(2.11d)} && \text{(2.13b)} \\ & \mathbf{u}_0 = \mathbf{a}. && \text{(2.13c)} \end{aligned}$$

One can readily observe that (2.11)-(2.13) satisfy the Bellman identities, i.e.

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(s) = \arg\min_{\mathbf{a}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}), \tag{2.14a}$$

$$V_{\boldsymbol{\theta}}(s) = \min_{\mathbf{a}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}). \tag{2.14b}$$

2.2.1 Combining RL and MPC

Here we present how the combination of RL and MPC can address the closed-loop performance of an MPC policy. Firstly we introduce howMPC can be used as a (possibly local) model of the action-value function $Q^\star$. Consider an MPC policy

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}) = \mathbf{u}_0^\star \tag{2.15}$$

where $\mathbf{u}_0^\star$ is part of the solution of:

$$\mathbf{x}^\star, \mathbf{u}^\star = \arg\min_{\mathbf{x},\mathbf{u}} \quad T_{\boldsymbol{\theta}}\left(\mathbf{x}_N\right) + \sum_{k=0}^{N-1} \ell_{\boldsymbol{\theta}}\left(\mathbf{x}_k, \mathbf{u}_k\right), \tag{2.16a}$$

$$\text{s.t.} \quad \mathbf{x}_{k+1} = \mathbf{f}_{\boldsymbol{\theta}}\left(\mathbf{x}_k, \mathbf{u}_k\right), \quad \mathbf{x}_0 = \mathbf{s}, \tag{2.16b}$$

$$\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{x}_k, \mathbf{u}_k\right) \leq 0, \quad \mathbf{u}_k \in \mathcal{A}. \tag{2.16c}$$

This MPC scheme is a parameterized formulation of (2.11), where the parameters $\boldsymbol{\theta}$ are included in the cost, dynamics and constraints. The motivation for not parameterizing $\mathbf{u}_k \in \mathcal{A}$ is to derive an MPC-based model of $Q^\star$ as follows:

$$Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) = \min_{\mathbf{x},\mathbf{u}} \quad \text{(2.16a)}, \tag{2.17a}$$

$$\text{s.t.} \quad \text{(2.16b)} - \text{(2.16c)}, \quad \mathbf{u}_0 = \mathbf{a}\,, \tag{2.17b}$$

where a constraint $\mathbf{u}_0 = \mathbf{a}$ included in (2.17b) is the only difference to (2.16). MPC (2.17) is a valid model of $Q^\star$ in the sense that it satisfies the relationships (2.6) and (2.7), i.e.:

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}\right) = \arg\min_{\mathbf{a}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}), \quad V_{\boldsymbol{\theta}}(\mathbf{s}) = \min_{\mathbf{a}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a})\,. \tag{2.18}$$

Here $V_{\boldsymbol{\theta}}(\mathbf{s})$ is the optimal cost resulting from solving MPC (2.16). One can verify that, if $Q_{\boldsymbol{\theta}} = Q^\star$, then MPC scheme (2.16) delivers the optimal policy $\boldsymbol{\pi}^\star$ through (2.15), i.e. $\boldsymbol{\pi}_{\boldsymbol{\theta}} = \boldsymbol{\pi}^\star$. An important question is how effectively an MPC scheme can approximate $Q^\star$ at least in a neighborhood of $\mathbf{a} = \boldsymbol{\pi}^\star\left(\mathbf{s}\right)$. The main concern here is arguably the MPC model $\mathbf{f}_{\boldsymbol{\theta}}$ for the reasons already raised in Sec. 2.2. In addition, $Q^\star$ is typically built from a discounted sum of the stage costs L, while undiscounted MPC formulations are typically preferred.

The Theorem reported below addresses these concerns and provides the central justification for considering the MPC parametrization (2.16) in RLMPC. It establishes that under some mild conditions, (2.17) is able to provide an exact model of $Q^\star$ even if the predictive model (2.16b) is inaccurate. This in turn entails that MPC (2.16) can achieve optimal closed-loop performance even if the MPC model is inaccurate.

Theorem 1 *[67] Suppose that the parameterized stage cost, terminal cost and constraints in* (2.16) *are universal function approximators with adjustable parameters $\boldsymbol{\theta}$. Further suppose that $\mathbf{x}_k^*$ is an optimal state trajectory generated by the MPC. Then there exist parameters $\boldsymbol{\theta}^\star$ s.t. the following identities hold $\forall\gamma$:*

1. $V_{\boldsymbol{\theta}^\star}(\mathbf{s}) = V^\star(\mathbf{s})$, $\forall \mathbf{s} \in \mathcal{S}$

2. $\boldsymbol{\pi}_{\boldsymbol{\theta}^\star}(\mathbf{s}) = \boldsymbol{\pi}^\star(\mathbf{s}),\ \forall \mathbf{s} \in \mathcal{S}$
3. $Q_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}) = Q^\star(\mathbf{s}, \mathbf{a}),\ \forall \mathbf{s} \in \mathcal{S}$, *for the inputs* $\mathbf{a} \in \mathcal{A}$ *such that* $|V^\star(\mathbf{f}_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}))| < \infty$

if the set

$$\Omega =: \left\{ \mathbf{s} \in \mathcal{S} \;\middle|\; |[V^\star(\mathbf{x}_k^\star)]| < \infty,\ \forall\, k \leq N \right\} \tag{2.19}$$

is non-empty.

Proof: We select the parameters such that the following holds:

$$T_{\boldsymbol{\theta}^\star}(\mathbf{s}) = V^\star(\mathbf{s}) \tag{2.20a}$$

$$L_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}) = \tag{2.20b}$$
$$\begin{cases} Q^\star(\mathbf{s}, \mathbf{a}) - V^\star(\mathbf{f}_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a})) & \text{If } |V^\star(\mathbf{f}_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}))| < \infty \\ \infty & \text{otherwise} \end{cases}$$

The proof then follows from [67, 120].

Theorem 1 states that, for a given MDP, an MPC scheme with an inaccurate model can deliver the optimal value functions and the optimal policy of the original MDP. This can be achieved by selecting the appropriate stage cost, terminal cost, and constraints. Theorem 1 extends to robust MPC, stochastic MPC, and Economic MPC (EMPC), for discounted and undiscounted settings. The assumption in (2.19) can be interpreted as a form of stability condition on $\mathbf{f}_{\boldsymbol{\theta}^\star}$ under the optimal trajectory $\mathbf{x}^\star$. More specifically, this assumption requires the existence of a non-empty set such that the optimal value function $V^\star$ of the predicted optimal trajectories $\mathbf{x}^\star$ on the system model is finite for all initial states starting from this set. The assumption is thus milder than requiring stability of the MPC scheme.

Chapter 3

Addressing Sample Efficiency and Model-Bias in Model-based RL

This chapter is based on the following publication (accepted):

Akhil S Anand, Jens Erik Kveen, Fares Abu-Dakka, Esten Ingar Grøtli, Jan Tommy Gravdahl (2022, December). Addressing Sample Efficiency and Model-bias in Model-based Reinforcement Learning. In 21st IEEE International Conference on Machine Learning and Applications (IEEE ICMLA): IEEE ICMLA 2022.

3.1 Introduction

Although Model-based Reinforcement Learning (MBRL) offers the flexibility of Reinforcement Learning (RL) to design complex control policies with better sample efficiency, it lacks asymptotic performance compared to model-free RL [220]. The main constraint in achieving better performance with MBRL is the quality of the model, as the policy is biased toward the model. In the case of complex dynamics such as robotic systems, multi-step forward predictions of such models are prone to errors due to uncertainties in the system and limitations of the model structure [2] even when using Neural Network (NN) models. The policies learned using such inaccurate models are prone to provide sub-optimal performance. The accuracy of the model can be improved using ensembles [129], where ensembles of deep NNs are utilized to capture the uncertainty of the model given the data collected from the environment. The use of ensembles of NNs was further extended to Probabilistic Ensemble NN (PENN) models to quantify both aleatoric and epistemic

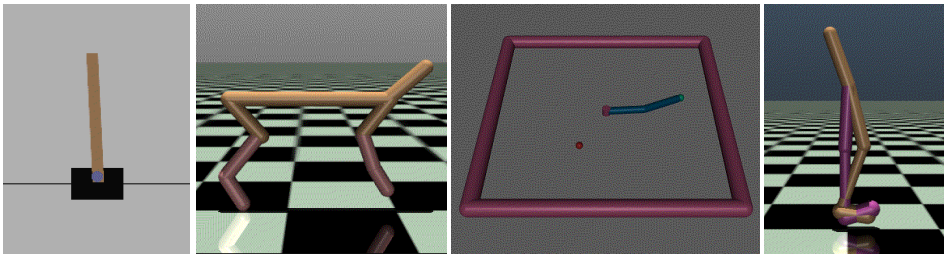


Figure 3.1: Four benchmarking tasks used for evaluation. In the order (from left to right) , Cartpole, Half Cheetah, Reacher, and Walker2D.

uncertainties in the Probabilistic Ensembles with Trajectory Sampling (PETS) algorithm [51].

Learning a model of the dynamics is the key to achieving sample efficiency. It could be beneficial in providing safety guarantees and facilitating learning multiple tasks with minimal to no resampling of a system. In MBRL, it is ideal to learn an accurate model over the entire task space of a system for which the policy is optimized in order to achieve good performance on the real system. This can be achieved using an explorative policy that explicitly tries to improve the quality of the model. A widely used approach in this direction is to use an exploration policy that maximizes the information gain based on the formulation of the expected information gain in [139]. This approach in principle could speed up the model learning and thereby further improve the sample efficiency of MBRL. Model Predictive Control (MPC) approaches are widely used in robotic control where its possible to acquire a model of the dynamics. The use of MPC based policy within MBRL is rewarding in benchmarking tasks [51].

In this chapter, we propose an MBRL framework where the dynamic model is represented using ensembles of NNs and a CEM-based [31] MPC as the policy. For sake of clarity, the main contributions of this chapter are:

- Explore aspects of sample efficiency, performance, and efficient transfer of policies between tasks within the standard MBRL framework.
- Improves the sample efficiency and transfer of policies between tasks by designing exploration policies targeted at maximizing the information gain from the region of interest using the uncertainty estimate of the model.
- To compensate for the inaccuracies in the learned model, a critic-value function estimated from the real data is used as a terminal value during the policy

optimization, thereby improving the asymptotic performance of our MBRL framework.

The rest of the chapter is organized as follows: Related works to this research are discussed in Section 3.2. Section 3.3 briefly introduces the necessary background knowledge and Section 3.4 presents the details of the proposed MBRL framework. Section 3.5 presents the evaluation of the MBRL framework on four RL benchmarking tasks shown in Fig. 3.1. Section 3.6 discusses the results, and the conclusion and future scope are discussed in Section 3.7.

3.2 Related Works

Many recent works focus on improving MBRL in line with the advancements in deep-RL [157, 52, 129, 94, 51]. In [129], authors used model ensembles in a dyna-style algorithm to improve asymptotic performance in MBRL. Authors in [52] utilized the model more efficiently by using its differentiability property by calculating the pathwise derivative of the learned model in future timesteps. In the state-of-the-art of shooting-based algorithms [51, 157] uncertainty-aware deep network dynamics models are combined with sampling-based uncertainty propagation. In [94], a theoretical analysis is formulated to guarantee a monotonic policy improvement in MBRL and demonstrates that a simple procedure of using short model-generated rollouts branched from real data could improve the performance of MBRL. The asymptotic performance of MBRL is largely improved by using ensembles of NNs for modeling the dynamics [51] essentially by improving the model quality.

Efficient exploration-exploitation schemes are proposed based on model uncertainty estimation over ensembles of NN dynamic models [195, 193, 166]. These works addressed the efficient model learning in terms of sample efficiency, uncertainty-based exploration with ensembles of NNs. In [165], termed as *curiosity-driven exploration*, the model uncertainty is evaluated based on the variance of the model predictions on the next state. In [193], a self-supervised exploration strategy is proposed to learn a global model by leveraging the uncertainty of the model and using it to solve the task sample efficiently. In [195], an active model-based active exploration algorithm is proposed for pure exploration. Although our approach utilizes the uncertainty estimates from ensembles of NNs it focuses on improving the sample efficiency and task transferability of MBRL by targeting the exploration of the system state space.

Incorporating H step rollouts from a learned dynamical model into the model-free value estimate can improve sample efficiency of model-free RL [58]. The STEVE algorithm [35] extends this model-based value expansion for the estimation of the

critic-value with ensembles of models and Q functions. They used uncertainty estimates from the ensembles to provide weighting to high-confidence predictions, thereby improving the value estimation. In [94], sample efficiency is improved by querying the model for short rollouts very frequently. In [157], a deep NN policy is initialized using a MPC and fine-tunes it in a model-free RL setting to achieve better performance. The similar problem of model-bias in MPC is tackled by learning an action-value function to improve performance [67]. Authors in [26] tried to address model-bias in MPC by blending a model-free value estimate that achieves performance close to MPC with real dynamics. The TD-MPC approach proposed in [72] uses a learned latent dynamics model and terminal value function to improve the performance of MPC in complex tasks. The progress in improving the performance of MBRL is limited by unavoidable inaccuracies in the learned model. Unlike the aforementioned approaches, we address this issue by reducing model-bias in MBRL by estimating a model-free critic-value. Our approach estimates a critic-value function similar to the approach in [58, 35], while these methods combine it with model rollouts to estimate the target value for the critic, we use it as a terminal value function in an CEM-based MPC policy.

Usually, the RL-based policies are specific to the tasks scenarios encountered during the learning. Therefore, a different task on the same system usually requires learning a different policy, even in the case of MBRL algorithms. Although there are approaches available to tackle this issue, e.g. *distillation* [162] and *meta-learning* [211], this remains a major challenge. In MBRL this process can be made sample efficient by utilizing the learned dynamics model, as in [132], which used to train a warm-up policy for the new task prior the interactions. To facilitate easy transfer between tasks, we utilize a gradient-free MPC policy to reuse the learned model for a different task without relearning the policy.

3.3 Background

3.3.1 Probabilistic Ensemble NN

PENN [51] is a NN based model approach capable of learning uncertainty-aware NN dynamics models including both aleatoric and epistemic uncertainties. Aleatoric uncertainty refers to the inherent stochasticity of the system. Whereas epistemic uncertainty is a systematic uncertainty arising from issues one could in principle avoid but does not in practice, such as inaccurate measurement, lack of data, modeling errors, etc. The output neurons of the probabilistic NN parameterize a probability distribution function, which can capture the aleatoric uncertainty of the model. Using multiple such networks in an ensemble can capture epistemic uncertainty. contrary, an ensemble of deterministic NN can only quantify epi-

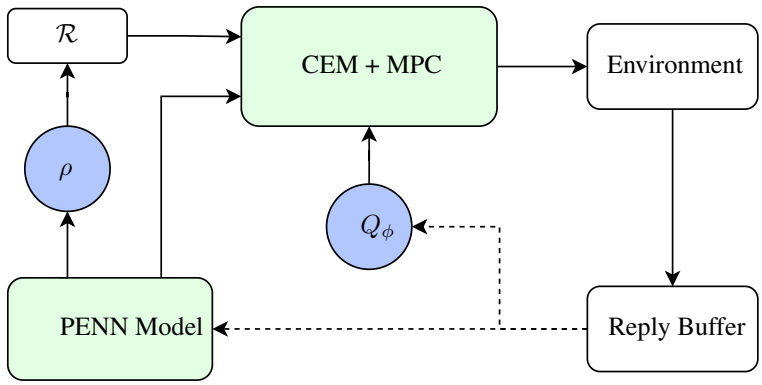


Figure 3.2: Illustration of the MBRL framework with uncertainty-targeted exploration and model-free Q function. Dynamics are modelled using PENN models and a CEM-based MPC is used as the policy.

stemic uncertainty. In [51] a thorough comparison of PENN with an ensemble of deterministic NN is provided, demonstrating the advantages of PENN for modeling dynamics. The predictive PENN model is trained with negative log prediction probability as a loss function, $\text{loss}_\text{P}(\boldsymbol{\theta}) = -\sum_{t=1}^{N} \log \widetilde{f}_{\boldsymbol{\theta}}\left(s_{t+1} \mid s_t, a_t\right)$. Where s_t is the state of the system at time step t, a_t is the applied action, and s_{t+1} is the next state. The PENN model is defined to output a Gaussian distribution with mean μ and diagonal covariances Σ parameterized by θ s.t, $\widetilde{f} = \Pr\left(s_{t+1} \mid s_t, a_t\right) = \mathcal{N}\left(\mu_\theta\left(s_t, a_t\right), \Sigma_\theta\left(s_t, a_t\right)\right)$ [51]. The network output in this fashion parameterizes a Gaussian distribution allowing for modeling the aleatoric uncertainty.

3.3.2 CEM-based MPC

The CEM [31] offers a gradient free optimization scheme, coupling it with an MPC to optimize an action sequence under the current model and executes the first action from the sequence on the environment. CEM samples multiple action sequences from a time-evolving distribution, which is usually modeled as a Gaussian distribution $a_{t:t+H} \sim \mathcal{N}\left(\mu_{t:t+H}, \text{diag}\left(\sigma^2_{t:t+H}\right)\right)$. These action sequences are evaluated on the learned dynamical model with respect to a cost function. The means $\mu_{t:t+H}$ and variances $\sigma^2_{t:t+H}$ of the sampling distribution are then updated based on best N trajectories.

3.4 Model-based Reinforcement Learning framework

We address the performance and sample efficiency of MBRL using model uncertainty-based exploration and model-free value function estimate. An MBRL framework incorporating these two approaches is presented here and its algorithm is summar-

ized in Algorithm 2.

3.4.1 Uncertainty-targeted Exploration for Model Learning

Modeling dynamics using the PENN captures the aleatoric and epistemic uncertainty of the model [51]. We propose using the model uncertainty measurement directly during planning to direct exploration towards parts of the state space where the model is more uncertain. Uncertainty-based exploration used with ensemble settings has been used in previous work [195, 193, 166] to efficiently explore to learn the model. The uncertainty of the model is evaluated based on the variance of the model in predicting the next state. In this chapter, we estimate the uncertainty in the same fashion, but the exploration strategy is devised by quantifying the model uncertainty using a dataset that defines the region of interest for exploration.

For a PENN with B bootstrap models, where $\tilde{f}_b$ are the individual models and $\overline{\tilde{f}}$ represents the mean of the ensemble. The uncertainty ρ_s, of the model prediction at state s for an action a, is calculated based on the disagreement between the individual models in the PENN as,

$$\rho_s(s,a) = \frac{1}{B-1} \sum_{b=1}^{B} \left(\tilde{f}_b(s,a) - \overline{\tilde{f}(s,a)} \right)^2 . \tag{3.1}$$

A weighted exploration-exploitation scheme is proposed by modifying the reward function for the exploration agent to incentivize the exploration in the regions where the model is most uncertain. The exploration component of the reward is the model uncertainty ρ and the exploitation component is the task reward r. The weighting between exploration and exploitation in the reward function is adjusted using ε,

$$\mathcal{R}_{\text{agent}} = \varepsilon\, \rho(s,a) + (1-\varepsilon)\, r(s,a) . \tag{3.2}$$

Exploration is targeted to minimize the uncertainty of the model in a region defined by an evaluation dataset. Given an ensemble model $\tilde{f}$ of B bootstrap models, and a dataset $\mathcal{D}_\rho$, the epistemic uncertainty of the model ρ_m can be estimated by taking the empirical variance between the predictions of each member model in the ensemble and averaging over each data point $\{s,a\}$ in $\mathcal{D}_\rho$:

$$\rho_m = \frac{1}{|\mathcal{D}|} \sum_{s \in \mathcal{D}} \frac{1}{B-1} \sum_{b=1}^{B} \left(\tilde{f}_b(s,a) - \overline{\tilde{f}(s,a)} \right)^2 . \tag{3.3}$$

An adaptive strategy is proposed to modulate the value of ε proportional to ρ_m as $\varepsilon = \rho_m/\rho_{max}$, where ρ_{max} represents the maximum value of the uncertainty of the model. This adaptive law helps to target the exploration of specific regions in the state-space. One way to do this is by using a dataset $\mathcal{D}_\rho$ representing the region of interest to calculate ε. For example, using a a random dataset sampled from the system state space earns a generalized model with low uncertainty over the entire state space. Whereas using a dataset sampled by the current policy π optimized for task performance will guarantee a model with low uncertainty on the task space. This provides a flexible framework to target the exploration across different areas in the state space.

3.4.2 Compensating for Model-bias with Model-free Critic

In dyna-style MBRL algorithms, the policy is optimized solely on the model rollouts. This allows single-step errors in model predictions to propagate along the longer-horizon rollouts and result in a large compound error in the Q value estimates. This results in sub-optimal policy performance and is a key reason for the performance of MBRL policies to be lower than model-free RL policies. As the performance of the optimized policy depends on the quality of the estimates of the Q function, one way to improve the performance of MBRL is to reduce the model-bias in the Q function. A similar approach exists in MPC, where model-bias is compensated for by improving the cost function using model-free RL [26, 67]. Along with learning the model, a model-free estimate of the critic, Q_ϕ,

$$Q(s,a) = \mathbb{E}_\pi \left[\sum_{t=0}^{H-1} \gamma^t \mathcal{R}\left(s_t, a_t\right) + \gamma^H Q_\phi\left(s_H, a_H\right) \right] \tag{3.4}$$

can be estimated from the dataset $\mathcal{D}$. Here, the model rollout on a high confidence horizon H is augmented with a terminal critic-value Q_ϕ estimated from the exploration data. Q_ϕ is estimated using fitted Q-iteration,

$$\phi^{k+1} \leftarrow \arg\min_\phi \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left\| Q_\phi(s,a) - y(s,a) \right\|_2^2, \tag{3.5}$$

where, for the k^{th} trial, y is the target value given by,

$$\mathcal{R}(s_t, a_t) + \gamma \max_{a_{t+1}} Q_\phi\left(s_{t+1}, a_{t+1}\right). \tag{3.6}$$

The model rollout horizon H is adapted based on the uncertainty of the learned PENN dynamical model by setting a threshold for the uncertainty estimate ρ in (3.1) to ρ_t.

Algorithm 2 MBRL with uncertainty-targeted exploration and model-free critic

Initialize dynamics model $\tilde{f}$, ε, π_u Q_ϕ
Populate dataset $\mathcal{D}$ using random controller for n initial trials.
for $k \leftarrow 1$ **to** K *Trials* **do**
  Train dynamics model $\tilde{f}$ on $\mathcal{D}$
  Populate uncertainty data $\mathcal{D}_\rho$ using π_u with some frequency
  $\rho_m \leftarrow$ Evaluate model uncertainty on $\mathcal{D}_\rho$
  $\varepsilon \leftarrow$ Calculate new ε based on ρ_m
  for $t \leftarrow 1$ **to** *TaskHorizon* **do**
    for *Actions sampled* $a_{t:t+H} \sim$*CEM*$(\cdot)$, *1* **to** *NIters* **do**
      Evaluate and sort the actions based on (3.2)
    end
    Execute first action a_t^* from optimal action sequence $a_{t:t+H}^*$
    Record outcome: $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, s_{t+1})$ update the Q_ϕ function
  end
end

3.5 Evaluation

We evaluate the uncertainty-targeted exploration approach and the model-free critic implementation on Standard deep-RL benchmarking simulation tasks provided by Open-AI Gym [32]. The approach is evaluated in four benchmarking tasks, (i) Cartpole, (ii) Half Cheetah, (iii) Reacher, and (iv) Walker2D as shown in Fig. 3.1. The performance is evaluated by comparing with the PETS algorithm [51] as a baseline. The design and optimization's hyper-parameters for the policy and the model are kept constant for the methods compared on a specific benchmarking task.

3.5.1 Uncertainty-targeted Exploration

In order to evaluate the uncertainty-targeted exploration approach three verification criterion: (*i*) the quality of the learned model, (*ii*) the sample efficiency of the policy, and (*iii*) the generalization of the model for different task scenarios. Two different exploration strategies have been considered:

1. *random exploration* : Uncertainty estimate using a random dataset on the state space of the system to adapt the value of ϵ. This targets the exploration over the entire state-space of the model and helps to learn a generalized model.

2. *task exploration* : Uncertainty estimated on a dataset sampled by the current

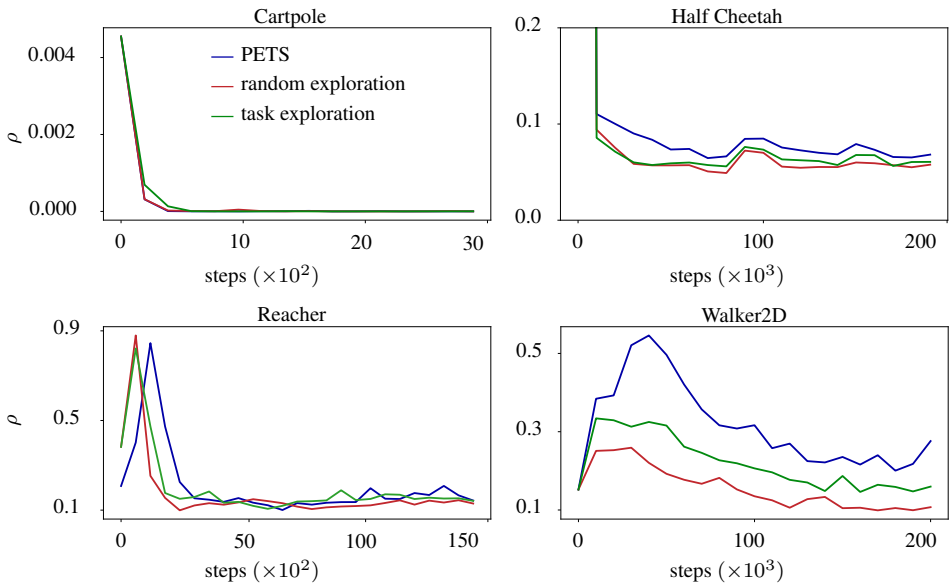


Figure 3.3: Model uncertainty estimates during the training.

task policy is used to adapt ϵ. This guides the exploration to the task space instead of the entire state space of the model.

The model uncertainty of the standard PETS is compared with the ones learned with random and task exploration strategies. Model uncertainty is estimated for all benchmarking tasks at regular intervals during training using a common dataset collected using a random agent from the entire state space of a system. The results in Fig. 3.3 show that the quality of the model is improved by the uncertainty-based exploration and is noticeable for tasks with complex dynamics such as Half Cheetah and Walker2D. The difference is more pronounced in the initial phase with exploration that maximizes the information gained by using larger values of ϵ. The uncertainty values will converge as the training progresses, the results on Cartpole and Reacher with simpler dynamics indicate this.

The sample efficiency is evaluated by comparing the performance of the MPC policies optimized with and without uncertainty-targeted exploration during different phases of the training. Figure 3.4 shows the performance of the policy correlated with the uncertainty of the model in Fig. 3.3. In very low data regimes (early phases of training), the policy optimized using the task exploration-based model provides better performance while in the later phases the policy optimized using the random exploration based model outperforms the rest.

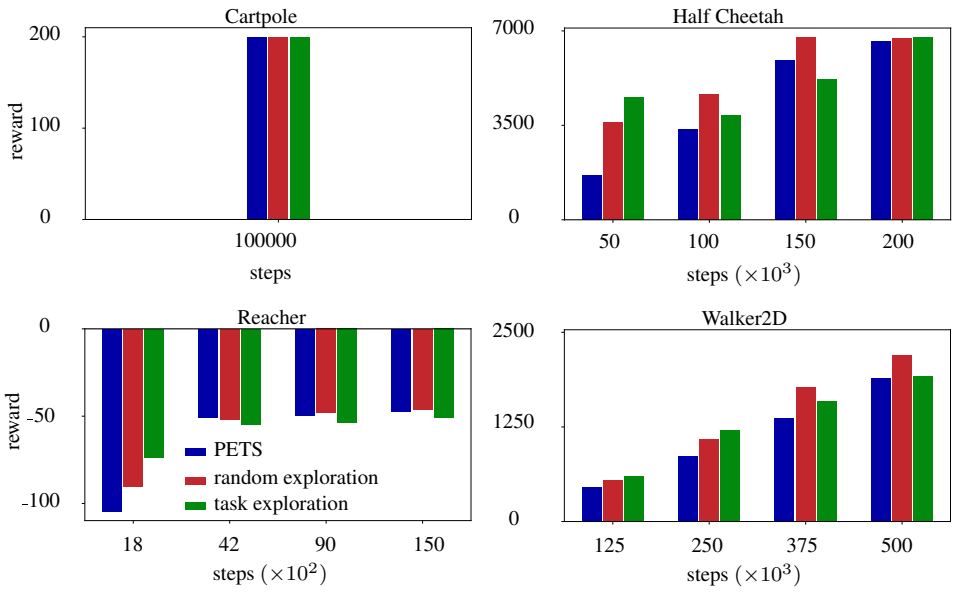


Figure 3.4: Mean model performance estimate on different phases of the learning over 10 trials. For Cartpole, the performance is evaluated only once as there is no significant difference in the model uncertainties.

Generalizability of the model learned by random exploration has been evaluated by changing task scenarios (task transferability). Two different sets of experiments have been conducted (*i*) Half Cheetah tracking a desired velocity, and (*ii*) Walker2D running backward. Figure 3.5 shows that the model learned with the random exploration strategy outperforms the standard PETS model in changing the task scenarios.

3.5.2 Model-free Critic

The policy learned with random exploration and model-free critic is compared with the model learned using the PETS and the random exploration agent. Figure 3.6 shows that the incorporation of the model-free critic improves the asymptotic performance of MBRL in complex tasks. In half cheetah and walker2D a performance increase of around 30% was observed compared to PETS. The advantage of the approach is pronounced in the tasks with complex dynamics. The performance of a low uncertainty model is limited by the model structure, whereas incorporating a model-free critic shows significant improvement in performance which is consistent in the results on Half Cheetah, Reacher, and Walker2D tasks.

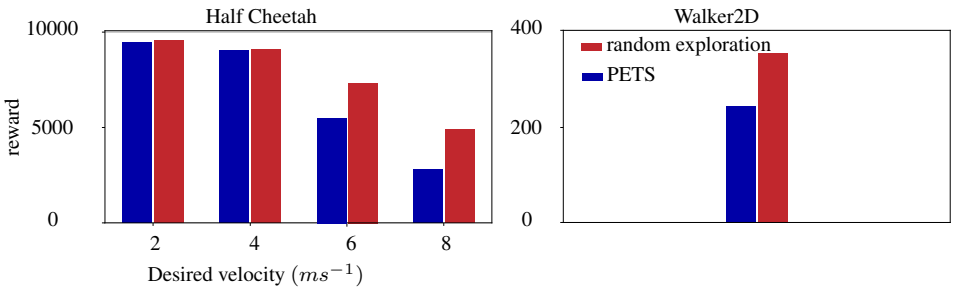


Figure 3.5: Model performance on varying the task scenarios. The trained models in Fig. 3.4 are used to evaluate the generalizability.

3.6 Discussion

The results in Section 3.5 demonstrate improvements in sample efficiency and performance using uncertainty-targeted exploration and model-free critic. Sample efficiency is improved by the targeted exploration of the state space and thereby increasing the information gained from the exploration data. The sample efficiency is an outcome of learning an accurate model while efficiently leveraging the uncertainty-targeted exploration. When the model is trained for longer, the performance ideally converges to the same value as PETS as the uncertainties of the model converge. However, in low-data regimes with complex dynamics, uncertainty-targeted exploration could be an advantage, as it could achieve better performance by efficiently learning the model from fewer trials compared to the standard MBRL approaches. Utilizing a model-free critic to compensate for the model-bias in the optimized policy improves the asymptotic performance of MBRL. This approach is promising for systems with complex dynamics where learning an accurate model is difficult, which is true for most robotic systems. In very low data regimes, the use of model-free critic has a slight adverse effect on the performance, as the critic-values are highly noisy during this phase. The noisy critic can be handled by blending it with the model-based critic to reduce variance. The model learned with random exploration could reduce the uncertainty over its entire state space with fewer samples, which could accelerate the learning of a new task for the same dynamical system. In the case of a nMPC-based policy, this property of generalization to new tasks can be verified by changing the definition of the task without any need to relearn the policy. In RL, transferability of a policy between tasks is a major challenge. Using a generalized model of the system in an MBRL setting as proposed here could be fruitful in mitigating this issue. Transferring the policy between tasks with minimal learning effort could be very beneficial in robotic systems in case the same robotic system performs different tasks.

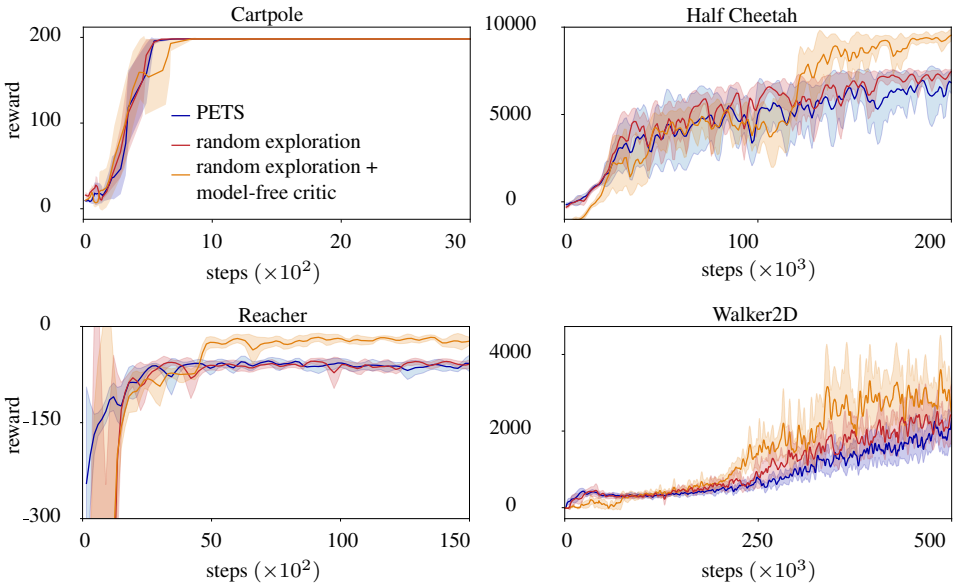


Figure 3.6: Learning curves for the algorithm on the benchmarking tasks. Each algorithm was trained with four random seeds. The results were filtered using Gaussian smoothing, the solid lines represents the mean and the shaded regions represent the variance.

Our proposed approach has few limitations in different aspects and requires further research to develop a generalized framework for robotic applications. For instance, random exploration over the entire state space of the system is usually difficult, especially in the case of safety-critical applications. The handling of safety while learning to control without having an accurate model is not straightforward. An interesting approach in this direction could be to provide probabilistic safety guarantees over the model. Learning a model-free critic is task-specific and the transferability of this critic is limited, but one way to improve this could be to initialize the model-free critic using a model-based critic if we have a generalized model and can fine-tune it for the task. Even though using a model-free critic could improve the performance, it cannot entirely compensate for the inaccuracies in the model. One interesting future direction would be to learn a cost function that can provide an optimal policy given an inaccurate model.

3.7 Conclusions

In this work, we address the improvement in sample efficiency and performance of the MBRL framework using uncertainty-targeted exploration and incorporating a critic-value estimate from the data. The proposed uncertainty-targeted exploration could improve the quality of the model by maximizing the information gain

during the exploration. Speeding up the model learning resulted in improving the sample efficiency of the MBRL thereby achieving the asymptotic performance faster. However, this did not largely improve the asymptotic performance as it is limited by the inaccuracies in the learned model. By introducing a critic-value function estimated from the real system data, we could compensate for the model inaccuracies to some extent. The MBRL algorithm that implements this approach could optimize a policy with better asymptotic performance compared to the baseline. We used a MPC based policy in our MBRL framework, but the approach presented here is generalizable to NN based policies. We aim to extend targeted exploration into more complex robotic manipulation tasks and address the issue of optimizing a policy over an imperfect model in the MBRL framework in detail in future work.

Chapter 4

Deterministic Policy Gradient Method for Learning-based MPC

This chapter is based on the following paper under review:

Akhil S Anand, Dirk Reinhardt, Shambhuraj Sawant, Jan Tommy Gravdahl, Sbastien Gros, A Painless Deterministic Policy Gradient Method for Learning-based MPC, Submitted to 20th European Control Conference (ECC), 2022.

4.1 Introduction

Within the Reinforcement Learning (RL) framework, policy gradient algorithms are most commonly used for dynamical systems with continuous action spaces [204]. Policy gradient methods optimize the policy parameters in the direction suggested by the gradient of the policy performance [205]. Policy gradient methods are typically implemented in an actor-critic setting and are grouped as: a) Stochastic Policy Gradient (SPG) methods for learning stochastic policies and, b) Deterministic Policy Gradient (DPG) methods for deterministic policies [198]. The DPG formulation can be seen as a limiting case of SPG where the variance of the policy tends to zero. The DPG algorithms are more sample-efficient compared to the SPG algorithms, owing to the simple form of its gradient step, which is formed with the expected gradient of the action-value function. Silver et al. [198] demonstrated significant improvement in sample efficiency of using the DPG formulation compared to SPG on a variety of tasks [117]. The DPG formulation has been extended for Deep Neural Networks (DNN) based RL by Lillicrap et al. [138] and is widely popular among various deep RL applications such as autonomous driving with Convolutional Neural Network (CNN) policy [219]. Recently DPG

has been used in various works to improve the close-loop performance of Model Predictive Control (MPC) despite using inaccurate MPC models, e.g., to learn the MPC parameters for a simplified freight mission of autonomous surface vehicles in [38], for battery storage applications in [118, 119] and for peak power management within smart grids in [37]. The DPG methods are fairly easy to handle when used with compatible advantage function approximations. However, this approach still requires an additional value function approximation, often carried out using DNNs. Using DNNs in actor-critic methods can cause issues, such as overestimating the value estimates and resulting in sub-optimal policies [61]. Beyond that, the need of designing an adequate structure for carrying the value function approximation in the DPG formulation is a source of difficulties.

In this chapter, we propose a novel actor-critic formulation using the DPG theorem with only an MPC scheme as an actor. The proposed method uses the MPC-based actor to approximate the value function, and hence eliminates the need for an additional approximation structure. To this end, we exploit the result proposed by Gros and Zanon [67] which states that, under mild assumptions, the optimal value function can be approximated by MPC and that the resulting policy and value functions satisfy the Bellman equations. A key feature of this approach is that the value function approximator is designed to be compatible with the policy gradient. The main contributions of the present chapter are:

- A novel deterministic policy gradient based actor-critic method for MPC is formulated where the value function in the critic is approximated using MPC actor itself.

- A simplified DPG algorithm for Reinforcement Learning based Model Predictive Control (RLMPC) is formulated by circumventing the need to construct an additional structure for approximating the value function in the compatible critic function approximation in the DPG formulation.

The rest of the chapter is organized as follows. Section 4.2 briefly introduces the necessary background knowledge about DPG, in Section 4.3 we presents the details of the proposed actor-critic DPG method for MPC. Section 4.4 assesses our approach in simulations and a brief discussion and conclusions are presented in Section 4.5 and Section 4.6, respectively. For the sake of clarity, throughout the chapter we use the term *system* to represents the true system dynamics and *model* represents the approximated dynamics model used in MPC. a and u are used interchangeably to represent an action or the control input for the system or model.

4.2 Deterministic Policy Gradients

DPG methods use a deterministic policy $\boldsymbol{\pi_\theta}$ parameterized by $\boldsymbol{\theta}$. The deterministic policy gradient theorem [198, Theorem 1] establishes the existence of a deterministic policy gradient of the form

$$\nabla_{\boldsymbol{\theta}} J_{\boldsymbol{\theta}}(\boldsymbol{\pi_\theta}) = \mathbb{E}_{\mathbf{s}\sim\rho^{\boldsymbol{\pi_\theta}}}\left[\nabla_{\boldsymbol{\theta}}\boldsymbol{\pi_\theta}(\mathbf{s})\nabla_{\mathbf{a}}Q^{\boldsymbol{\pi}}(\mathbf{s},\,\mathbf{a})|_{\mathbf{a}=\boldsymbol{\pi_\theta}(\mathbf{s})}\right]. \tag{4.1}$$

Here $Q^{\boldsymbol{\pi}}(\mathbf{s},\,\mathbf{a})$ represents the action-value function under the policy $\boldsymbol{\pi}$. The DPG gradient can be integrated into on-policy and off-policy actor-critic settings. Actor-critic methods are Temporal Difference (TD) learning algorithms that approximate the value function to evaluate the policy. In this context, the actor represents the policy approximation and the critic represents a form of approximation for an advantage function.

Here we consider an on-policy actor-critic version of the DPG method, where the critic is an estimate of the action value function for the policy. The action value function $Q^{\boldsymbol{\pi}}(\mathbf{s},\,\mathbf{a})$ can be approximated using a differentiable function approximation, $Q^{\mathbf{w}}(\mathbf{s},\,\mathbf{a})$ with parameter vector $\mathbf{w}$. The actor updates the deterministic policy parameters $\boldsymbol{\theta}$, using the policy gradient estimated in (4.1). DNNs are often utilized as universal function approximation for the action-value function $Q^{\mathbf{w}}$. The on-policy version of the deterministic actor-critic update rule using the TD error δ_k to update the critic and corresponding policy update is given by,

$$\delta_k = \ell(\mathbf{s}_k,\,\mathbf{a}_k) + \gamma Q^{\mathbf{w}}(\mathbf{s}_{k+1},\,\mathbf{a}_{k+1}) - Q^{\mathbf{w}}(\mathbf{s}_k,\,\mathbf{a}_k), \tag{4.2a}$$

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha_{\vec{w}}\delta_k\nabla_{\mathbf{w}}Q^{\mathbf{w}}(\mathbf{s}_k,\,\mathbf{a}_k), \tag{4.2b}$$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha_{\boldsymbol{\theta}}\delta_k\nabla_{\boldsymbol{\theta}}\boldsymbol{\pi_\theta}(\mathbf{a}_k|\mathbf{s}_k)\nabla_{\mathbf{a}}Q^{\mathbf{w}}(\mathbf{s}_k,\,\mathbf{a}_k)|_{\mathbf{a}_k=\boldsymbol{\pi_\theta}(\mathbf{s}_k)}. \tag{4.2c}$$

4.3 Deterministic Policy Gradient for MPC

DPG uses an approximation $Q^{\boldsymbol{\pi}}(\mathbf{s},\,\mathbf{a}) \approx Q^{\mathbf{w}}(\mathbf{s},\,\mathbf{a})$ under the condition that the gradient $\nabla_{\mathbf{a}}Q^{\mathbf{w}}(\mathbf{s},\,\mathbf{a})$ can replace the true gradient $\nabla_{\mathbf{a}}Q^{\boldsymbol{\pi}}(\mathbf{s},\,\mathbf{a})$ in the policy gradient update (4.1), i.e.,

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi_\theta}) = \mathbb{E}_{s\sim\mathbb{P}}[\nabla_{\boldsymbol{\theta}}\boldsymbol{\pi_\theta}(\mathbf{s})\nabla_{\mathbf{a}}Q^{\mathbf{w}}(\mathbf{s},\mathbf{a})|_{\mathbf{a}=\boldsymbol{\pi_\theta}(\mathbf{s})}]. \tag{4.3}$$

The class of function approximators that satisfy this condition is referred as *compatible* function approximators and a formal verification is given in [198, Theorem 3]. In this context, [198] shows that the function approximator

$$Q^{\mathbf{w}}(\mathbf{s},\mathbf{a}) = (\mathbf{a} - \boldsymbol{\pi_\theta}(\mathbf{s}))^\top\nabla_{\boldsymbol{\theta}}\boldsymbol{\pi_\theta}(\mathbf{s})^\top\mathbf{w} + V^{\mathbf{v}}(\mathbf{s}) \tag{4.4}$$

is compatible for any deterministic policy $\boldsymbol{\pi_\theta}(\mathbf{s})$. The first term in the approximation is the policy gradient and the second term is an approximation of the policy

value function, i.e. $V^{\mathbf{v}}(\mathbf{s}) \approx V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\mathbf{s})$. For a more compact derivation, we use the linear feature vector $\boldsymbol{\phi}(\mathbf{s}, \mathbf{a}) = \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s})(\mathbf{a} - \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}))$ to rewrite the action-value function (4.4) as

$$Q^{\mathbf{w}}(\mathbf{s}, \mathbf{a}) = \boldsymbol{\phi}(\mathbf{s}, \mathbf{a})^\top \mathbf{w} + V^{\mathbf{v}}(\mathbf{s}) \,. \tag{4.5}$$

The DPG method then seeks to compute parameters $\mathbf{v}$ and $\mathbf{w}$ from data such that the least-squares problems:

$$\min_{\mathbf{v}} \quad \mathbb{E}[(V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\mathbf{s}) - V^{\mathbf{v}}(\mathbf{s}))^2] \tag{4.6}$$

and

$$\min_{\mathbf{w}} \quad \mathbb{E}[(Q^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\mathbf{s},\, \mathbf{a}) - Q^{\mathbf{w}}(\mathbf{s},\, \mathbf{a}))^2] \tag{4.7}$$

are approximately solved for $\mathbf{a}$ in a neighborhood of $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s})$. These parameters are then used in (4.3) to build a policy gradient estimation, and subsequently, a gradient step in the MPC parameters $\boldsymbol{\theta}$ that reduces the closed-loop cost $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$.

An important difficulty in the DPG method is then to design an effective structure for $V^{\mathbf{v}}$ to achieve $V^{\mathbf{v}} \approx V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$. Indeed, value function $V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ can be fairly rich and complex and is a priori unknown. Hence, one often needs to rely on universal function approximators, such as DNNs, for $V^{\mathbf{v}}$ to approximate $V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ effectively. This typically requires a very large set of parameters $\mathbf{v}$, which are then difficult to estimate. In order to alleviate this issue, we propose to leverage on the value function $V_{\boldsymbol{\theta}}$ delivered by MPC scheme (2.11) to build $V^{\mathbf{v}}$. The basic intuition behind this idea is that for a well-formulated MPC scheme (2.11), the resulting value function $V_{\boldsymbol{\theta}}$ is likely to already have a structure fairly close to the one needed to approximate $V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$, hence removing the need of building a universal function approximation where a large set of parameters $\mathbf{v}$ is required.

To further motivate the idea, consider the central result of [67, Theorem 1]. It shows that MPC scheme (2.11) can deliver the optimal policy and value functions even if based on an inaccurate model, provided that $\ell_{\boldsymbol{\theta}}$, $T_{\boldsymbol{\theta}}$, $\mathbf{h}_{\boldsymbol{\theta}}$ are richly parameterized. In that ideal context, there is a $\boldsymbol{\theta}$ such that $\boldsymbol{\pi}_{\boldsymbol{\theta}} = \boldsymbol{\pi}^\star$ and

$$V_{\boldsymbol{\theta}}(\mathbf{s}) = V^\star(\mathbf{s}) = V^{\boldsymbol{\pi}^\star}(\mathbf{s}) = V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\mathbf{s}) \,. \tag{4.8}$$

In many control applications, it is reasonable to assume that the value function provided by the MPC scheme (2.11) is reasonably close to the optimal one $V^*(\mathbf{s})$. That is arguably the case if the MPC model $\mathbf{f}_{\boldsymbol{\theta}}$ of the MPC is representing the system dynamics (2.1) reasonably well, and if $\ell_{\boldsymbol{\theta}} = \ell$, and $T_{\boldsymbol{\theta}}$ is an adequate choice of terminal cost. In that context, it is reasonable to assume that $V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\mathbf{s})$ lies in a neighborhood of $V_{\boldsymbol{\theta}}(\mathbf{s})$. Then, assuming that $V_{\boldsymbol{\theta}}$ admits a first-order Taylor expansion in $\boldsymbol{\theta}$, one can consider the approximation:

$$V^{\boldsymbol{\pi}}(\mathbf{s}) \approx V^{\mathbf{v}}(\mathbf{s}) = V_{\boldsymbol{\theta}}(\mathbf{s}) + \nabla_{\boldsymbol{\theta}} V_{\boldsymbol{\theta}}(\mathbf{s})^\top \mathbf{v} \tag{4.9}$$

of $V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$, where $\mathbf{v}$ is a set of parameters to be estimated according to (4.6). The hope with (4.9) is that $V_{\boldsymbol{\theta}}$ and $\nabla_{\boldsymbol{\theta}} V_{\boldsymbol{\theta}}$ provide useful *features* for building $V^{\mathbf{v}}$, by exploiting the prior knowledge embedded in MPC scheme (2.11).

In order to ensure that approximation (4.9) becomes increasingly effective as the learning progresses, it is arguably useful to update the MPC parameters $\boldsymbol{\theta}$ to satisfy $V_{\boldsymbol{\theta}} \approx V^\star$ in addition to following the policy gradient (4.3). To that end, we modify the DPG step by introducing the additional optimization objective of capturing the $Q^\star$ along with $\pi^\star$:

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) + \frac{\beta}{2}\mathbb{E}[(Q^\star(\mathbf{s}_k,\, \mathbf{a}_k) - Q_{\boldsymbol{\theta}}(\mathbf{s}_k,\, \mathbf{a}_k))^2]\,. \tag{4.10}$$

for some weight β. This technique and its benefits were recently presented in [192]. It essentially combines DPG with Q-learning in a single update step, to ensure that both the policy and the value function of MPC scheme (2.11) are driven to their optimal values.

The rest of this section derives the update rules for the parameters of the MPC policy $\boldsymbol{\theta}$ and for the weights $\mathbf{w}$ for the feature vector $\boldsymbol{\phi}(\mathbf{s}, \mathbf{a})$ in (4.5) in a batch learning scheme. This can be coupled with any actor-critic DPG method for MPC without needing an extra function approximator such as a DNN. Here we formulate an actor-critic setting for our DPG method with an MPC actor and a critic estimated using least-squares temporal-difference learning algorithm (LSTD) [130]. We chose LSTD based critic approximation due to its simplicity with the compatible form for the critic. Our approach relaxes the constraint of using a linear value function approximation in the LSTD approach. Making use of the richer structure provided by the MPC-based approximation allows the LSTD approach to be applicable to problems with complex value functions.

Suppose we have a dataset of K samples, $\mathcal{D} = \{(\mathbf{s}_1, \mathbf{a}_1, \mathbf{s}_2), \ldots, (\mathbf{s}_K, \mathbf{a}_K, \mathbf{s}_{K+1})\}$. To learn the parameter $\mathbf{v}$ that minimizes (4.6), we can approximate it by the following LSTD problem:

$$\min_{\mathbf{v}} \sum_{k=1}^{K} \left(\ell(\mathbf{s}_k, \mathbf{a}_k) + \gamma V_{\boldsymbol{\theta}}(\mathbf{s}_{k+1}) - V^{\mathbf{v}}(\mathbf{s}_k)\right)^2 . \tag{4.11}$$

The minimizer $\mathbf{v}^\star$ is then given by the unique solution

$$\begin{aligned} \mathbf{v}^\star = &\left(\sum_{k=1}^{K} \nabla_{\boldsymbol{\theta}} V_{\boldsymbol{\theta}}(\mathbf{s}_k) \nabla_{\boldsymbol{\theta}} V_{\boldsymbol{\theta}}(\mathbf{s}_k)^\top\right)^{-1} \\ &\sum_{k=1}^{K} \nabla_{\boldsymbol{\theta}} V_{\boldsymbol{\theta}}(\mathbf{s}_k)\left(\ell(\mathbf{s}_k,\, \mathbf{a}_k) + \gamma V_{\boldsymbol{\theta}}(\mathbf{s}_{k+1}) - V_{\boldsymbol{\theta}}(\mathbf{s}_k)\right) . \end{aligned} \tag{4.12}$$

The optimal weights $\mathbf{w}^\star$ for the feature vector $\boldsymbol{\phi}(\mathbf{s}, \mathbf{a})$ in (4.5) can be obtained by minimizing

$$\min_{\mathbf{w}} \sum_{k=1}^{K} \left(Q^{\boldsymbol{\pi}}(\mathbf{s}_k,\, \mathbf{a}_k) - Q^{\mathbf{w}}(\mathbf{s}_k)\right)^2 , \tag{4.13}$$

which has the closed-form solution

$$\begin{aligned}\mathbf{w}^\star = &\left(\sum_{k=1}^{K} \boldsymbol{\phi}(\mathbf{s}_k,\, \mathbf{a}_k)\boldsymbol{\phi}(\mathbf{s}_k,\, \mathbf{a}_k)^\top\right)^{-1} \\ &\sum_{k=1}^{K} \boldsymbol{\phi}(\mathbf{s}_k,\, \mathbf{a}_k) \left(\ell(\mathbf{s}_k,\, \mathbf{a}_k) + \gamma V_{\boldsymbol{\theta}}(\mathbf{s}_{k+1}) - V_{\boldsymbol{\theta}}(\mathbf{s}_k) - \nabla_{\boldsymbol{\theta}} V_{\boldsymbol{\theta}}(\mathbf{s}_k)^\top \mathbf{v}\right) .\end{aligned} \tag{4.14}$$

Based on the compatible critic function approximation (4.4), the gradient of the critic has the form,

$$\nabla_{\mathbf{a}} Q^{\boldsymbol{\pi}}(\mathbf{s}, \mathbf{a}) \approx \nabla_{\mathbf{a}} Q^{\mathbf{w}}(\mathbf{s}, \mathbf{a}) = \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s})^\top \mathbf{w} . \tag{4.15}$$

The parameter update given along the policy gradient can be obtained by combining (4.3) and (4.15) into

$$\Delta \boldsymbol{\theta}_J = \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}) \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s})^\top \mathbf{w} . \tag{4.16}$$

The second term of combined the objective function (4.10) can be minimized via temporal-difference learning,

$$\begin{aligned} \delta_k &= \ell\left(s_k, a_k\right) + V_\theta\left(s_{k+1}\right) - Q_\theta\left(s_k, a_k\right) \\ \Delta\theta_Q &= \delta_k \nabla_\theta Q_\theta\left(s_k, a_k\right) . \end{aligned} \tag{4.17}$$

The update step combining the policy gradient and Q-learning is then given by

$$\Delta \boldsymbol{\theta} = \Delta \boldsymbol{\theta}_J + \beta \Delta \boldsymbol{\theta}_Q . \tag{4.18}$$

The parameter update is given by the gradient step

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \Delta \boldsymbol{\theta} . \tag{4.19}$$

Note that the update step in (4.18) can be done using a mixed objective, ensuring the policy gradient updates and Q-learning updates does not disturb each other. The solution is to use Null space projections as described in [192].

Algorithm 3 DPG for MPC

Initialize $\boldsymbol{\theta}, \mathbf{w}$
for $i \leftarrow 1$ **to** N **do**
  for $k \leftarrow 1$ **to** K **do**
    Solve MPC obtain to $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)$, $V_{\boldsymbol{\theta}}(\mathbf{s}_k)$ and the sensitivities $\nabla_{\boldsymbol{\theta}}\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)$, $\nabla_{\boldsymbol{\theta}}V_{\boldsymbol{\theta}}(\mathbf{s}_k)$ and store to $\mathcal{D}_i$.
    Apply the policy to system with exploration, $\mathbf{a}_k = \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k) + \boldsymbol{\epsilon}$, and store the transition data $(\mathbf{s}_k,\, \mathbf{a}_k,\, \mathbf{s}_{k+1},\, \ell(\mathbf{s}_k,\, \mathbf{a}_k))$ to $\mathcal{D}_i$.
  end
  $\mathbf{v} \leftarrow$ evaluate (4.12) with $\mathcal{D}_i$.
  $\mathbf{w} \leftarrow$ evaluate (4.14) with $\mathcal{D}_i$.
  $\Delta\boldsymbol{\theta} \leftarrow$ estimate $\Delta\boldsymbol{\theta}_J$ (4.16) and $\Delta\boldsymbol{\theta}_Q$ (4.17).
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha\Delta\boldsymbol{\theta}$ (4.19).
end

An on-policy version of the proposed algorithm is provided in Algorithm 3. The required sensitivities $\nabla_{\boldsymbol{\theta}}\boldsymbol{\pi}_{\boldsymbol{\theta}}$, $\nabla_{\boldsymbol{\theta}}V_{\boldsymbol{\theta}}$, $\nabla_{\boldsymbol{\theta}}Q_{\boldsymbol{\theta}}$ are computed by building the Karush-Kuhn-Tucker (KKT) conditions for (2.11) and (2.13) and using the Inverse Function Theorem (IFT) [36]. This approach is a standard practice for combinations of RL and MPC and more details can be found in [67, 38].

4.4 Evaluation

In this section, we evaluate the proposed method for two non-linear MPC problems in simulation. In the first experiment, we consider a first-order optimal investment problem with non-quadratic cost to demonstrate the effectiveness of the MPC-based local value function approximation. In the second experiment, we consider the classical non-linear control problem of cartpole swing-up and balancing to demonstrate the effectiveness of the proposed DPG algorithm for learning MPC parameters.

4.4.1 MPC-based Value Function Approximation

In this example, we illustrate the proposed value function approximation for an optimal investment problem [183]. The dynamics of the optimal investment problem is given by,

$$s_{k+1} = u_k, \quad \ell(s,u) = -\ln\left(As^{\alpha} - u\right). \tag{4.20}$$

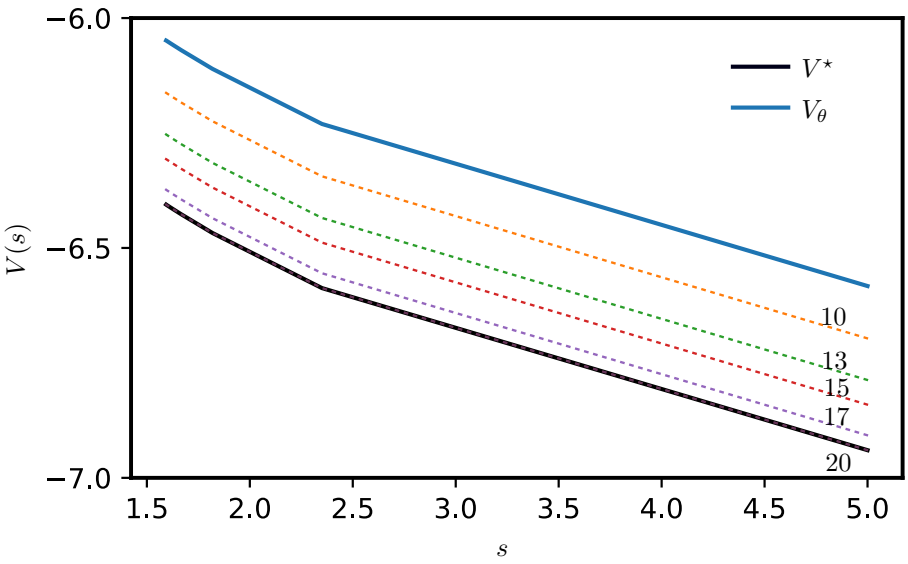


Figure 4.1: Learning value function with an imperfect model: value function approximated using MPC during the learning iterations (iteration number is indexed at the right end of the curves). V_θ is the initial value function of the MPC and $V^\star$ is the value function of the MPC with a perfect model (4.20).

Where s is the state, u is the control input, and A and $0 < \alpha < 1$ are constant parameters. The state s denotes the investment in a company and the term As^α is the return from this investment after one period. Then $As^\alpha - u$ is the amount of money that can be used for consumption in the current time period. The objective is to maximize the sum of the logarithmic utility function in (4.20) for which we consider a finite-horizon discounted MPC with perturbed model $s_{k+1} = \mu u_k$. The prediction horizon and model parameter of the MPC are set to $N = 10$ and $\mu = 0.8$. The cost parameters $A = 5$, $\alpha = 0.34$ and discount factor $\gamma = 0.8$ are the same for baseline cost ℓ and MPC cost ℓ_θ.

The problem has an optimal value function $V^\star$, to which we compare the value function approximations V_θ for different amounts of learning iterations in Fig. 4.1. This result demonstrates that the optimal value function can be learned efficiently with the proposed MPC-based value function approximation (4.9).

4.4.2 Learning MPC Parameters

This example illustrates the successful learning of the MPC parameters using the DPG-based approach.

Cartpole System

The cartpole system, as depicted in Fig. 4.2, consists of a wheeled cart of mass m_c which can freely move on a rail with friction coefficient of μ_f. A pole with mass m_p and length l is hinged to the cart on a friction-less joint. The pole can swing freely around the hinged joint. The control input, u to the system is the force exerted on the cart along the rail. The goal of the control task is to swing the pole to upright position as quickly as possible and balance it there. The parameter values of the true system and the initial MPC are listed in Table 4.1.

The state and input of the cartpole system are given by

$$\mathbf{s} = \begin{bmatrix} x & \dot{x} & \phi & \dot{\phi} \end{bmatrix}^\top, \quad u = F\,, \tag{4.21}$$

where x is the position of the cart along the rail, $\dot{x}$ is the velocity of the cart, ϕ is the angle of the pole with respect to the vertical axis, and $\dot{\phi}$ is the angular velocity of the pole.

The dynamic equations governing the accelerations of the cartpole system are given by

$$\begin{aligned} \ddot{\phi} &= \frac{g\sin\phi + \cos\phi\left(\frac{\mu_f\dot{x} - u - m_p l\dot{\phi}^2\sin\phi}{m_c+m_p}\right)}{l\left(\frac{4}{3} - \frac{m_p\cos^2\phi}{m_c+m_p}\right)} \\ \ddot{x} &= \frac{u - \mu_f\dot{x} + m_p l\left(\dot{\phi}^2\sin\phi - \ddot{\phi}\cos\phi\right)}{m_c+m_p}. \end{aligned} \tag{4.22}$$

The cost function for the cartpole swing-up task is given by

$$\ell(\mathbf{s}, u) = x^2 + \phi^2 + 0.01\dot{x}^2 + 0.01\dot{\phi}^2 + 0.001u^2\,. \tag{4.23}$$

To evaluate the performance after learning, we use the (undiscounted) cumulative cost over the horizon N given by

$$L = \sum_{k=0}^{N-1} \ell(\mathbf{s}_k, u_k). \tag{4.24}$$

Model Predictive Control Formulation

For the MPC formulation, we consider the following state and control input:

$$\mathbf{x} = \begin{bmatrix} x & \dot{x} & \sin\phi & \cos\phi & \dot{\phi} \end{bmatrix}^\top, \quad u = F, \tag{4.25}$$

Table 4.1: Model parameters

system parameter	value	model parameter	value
m_c	$1\,\mathrm{kg}$	$\hat{m}_c$	$(1.5 - r)\mathrm{kg}$
m_p	$0.5\,\mathrm{kg}$	$\hat{m}_p$	$0.5(1 + r)\mathrm{kg}$
l	$0.5\,\mathrm{m}$	$\hat{l}$	$(0.5 - 0.2r)\mathrm{m}$
μ_f	$1.0\,\mathrm{kg\,s^{-1}}$	$\hat{\mu}_f$	$(1 + r)\,\mathrm{kg\,s^{-1}}$

where we use $\sin\phi$ and $\cos\phi$ to represent the pole angle instead of ϕ to avoid problems with repeating solutions at $\phi = \pi + k2\pi$ for $k \in \mathbb{N}$. The accelerations of the system are given by (4.22), with the nominal parameters as summarized in Table 4.1. For learning the model parameters, we use the nominal values as an initial guess and update them during the learning process. The corresponding parameter vector is given by $\boldsymbol{\theta}_m = \begin{bmatrix}\hat{m}_c & \hat{m}_p & \hat{l} & \hat{\mu}_f\end{bmatrix}^\top$.

We define the stage cost as

$$\ell_{\boldsymbol{\theta}_\ell}(\mathbf{x},\, u) = \|\mathbf{x} - \mathbf{x}_r\|^2_{\mathbf{Q}} + \|u\|^2_R, \tag{4.26}$$

with the reference $\mathbf{x}_r$ for the swing-up task where we want to control the pole to an upright position and the cart to the origin given by $\mathbf{x}_r = [0\ \ 0\ \ 0\ \ 1\ \ 0]^\top$. The weighting matrices are parametrized by the parameter vector $\boldsymbol{\theta}_\ell = [(\boldsymbol{\theta}_Q)^\top\ \ \theta_r]^\top$ as

$$\mathbf{Q} = \mathrm{diag}(\boldsymbol{\theta}_Q), \quad R = \theta_r. \tag{4.27}$$

Box constraints ensure the cart stays on the rail and the force exerted on the cart is within the actuator limits. The constraints are given by

$$\begin{aligned} -2.4 \leq x \leq 2.4 \\ -10 \leq u \leq 10. \end{aligned} \tag{4.28}$$

The cartpole system is discretized using the explicit Euler method with a sampling time of 0.05 seconds. The horizon length N is set to 25 and the control input is updated every 0.05 seconds.

Learning MPC Parameters:

Here we use our DPG approach to adapt the parameters of the MPC formulation (2.11) for better closed-loop performance. We use an inaccurate model with an

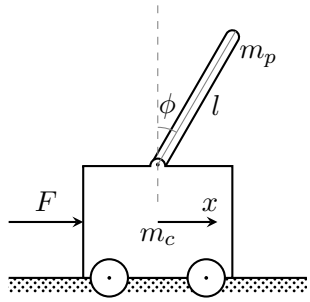


Figure 4.2: Cartpole system.

uncertain estimate of the true system parameters in (4.22) in the MPC formulation (2.11). The uncertain parameters in the MPC model are chosen as

$$\begin{aligned} \hat{m}_c &= m_c + 0.5 - r, \quad \hat{\mu}_f = \mu_f + r \\ \hat{m}_p &= m_p + 0.5r, \quad \hat{l} = l - 0.2r\,, \end{aligned} \tag{4.29}$$

where r denotes a random noise given by $r \sim \mathcal{U}[0, 1]$. The nominal values of the system parameters and their corresponding uncertain/wrong model parameters are shown in Table 4.1. We conducted two sets of experiments with 10 seeded random trials where in each experiments a randomly chosen set of model parameters (4.29) are used. The two sets of experiments are:

1. Learning only the stage cost parameters $\boldsymbol{\theta}_\ell$ to improve the MPC performance.
2. Learning the stage cost parameters $\boldsymbol{\theta}_\ell$ and model parameters $\boldsymbol{\theta}_m$ to improve the MPC performance.

For both the experiments, the $\boldsymbol{\theta}_\ell$ and $\boldsymbol{\theta}_m$ parameters are initialized with the nominal MPC parameters provided in (4.23). The results for both experiments are shown in Fig. 4.3, indicating that with only learning $\boldsymbol{\theta}_\ell$ the MPC performance improves but stagnates at lower performance range compared to the ideal MPC performance given the perfect dynamics model. Whereas learning the model parameters $\boldsymbol{\theta}_m$ in addition to $\boldsymbol{\theta}_\ell$ further improved the performance close to the ideal MPC performance with perfect model. The MPC policies learned in both scenarios were compared with the MPC with imperfect model and the ideal MPC with correct model for successfully swinging up and balancing the pole with the task time of $2.5\,\mathrm{s}$. The comparison was done for 100 trials of the cartpole swing-up task using a set of 100 initial states, randomly sampled from the system state space. The

Table 4.2: Cartpole swing-up success rate

MPC with perturbed model	39%
Learning $\boldsymbol{\theta}_\ell$	56%
Learning $\boldsymbol{\theta}_\ell$ and $\boldsymbol{\theta}_m$	100%
MPC with perfect model	100%

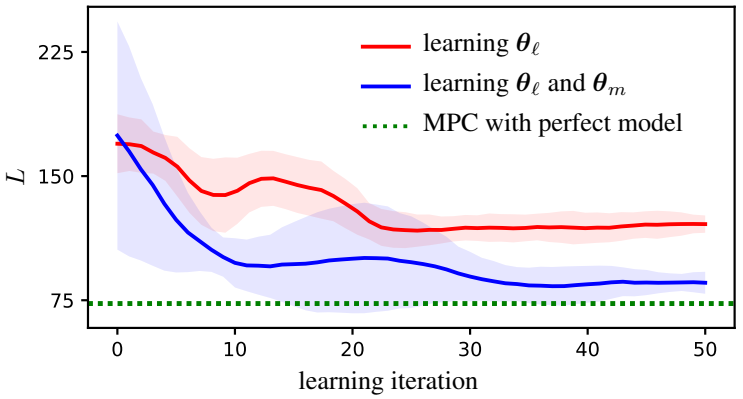


Figure 4.3: Learning MPC parameters for the perturbed model: the mean learning progress over 10 seeded random trials are presented for 50 learning iterations for two different sets of experiments described in Section 4.4.2. Stage cost, L for each trial is smoothed by calculating the moving average with a moving window size of 10 learning iterations.

task is considered as successfully completed if the pole is balanced in the upright position with maximum deviation of $\pm 5^\circ$ within 5 s. Results presented in Table 4.2 show that the MPC with learned $\boldsymbol{\theta}_\ell$ parameters could improve the success rate of the MPC with the perturbed model. But by additionally learning the $\boldsymbol{\theta}_m$ parameters, MPC achieved 100% success rate equivalent to the ideal MPC with a perfect model of the system dynamics.

4.5 Discussions

The actor-critic setting is at the core of all of the state-of-the-art policy gradient methods in RL, which brought a lot of success and research interest in RL. But this is at the expense of using additional complex function approximators for the critic and makes the RL practitioner's life difficult on different applications. Combining RL with MPC framework opens up a wide range of possibilities to simplify the RL algorithms which have been shown in the previous works combining RL and MPC [67]. The presented work is in line with the similar idea of leveraging on the MPC properties to derive better RL methods for MPC. The appeal of the proposed

approach is that the MPC delivers the respective gradients for the value function and the policy simultaneously. Therefore, a well-formulated MPC can provide a reliable estimation of the optimal value function on the system. As a result, we do not need to design and train a separate function approximator for the value function, which would be the case if we were to use the DPG algorithm directly with a DNN.

The simulation experiment with the proposed approach provided in Section 4.4 showed promising results. The result of the cartpole swing-up and balancing control problem is a practical example of the main contribution of our work, a DPG method for MPC without any difficulties of designing and using a complex function approximation for the value function. We incorporated the proposed value function approximation within compatible critic function approximation for DPG. A key feature of our approach is that the proposed value function approximation is compatible with the policy gradient. One limitation of our approach is the assumption of an initial MPC policy which is close to the optimal policy on the system. This approach could fail and result in suboptimal policies if we use a random initial policy which is often the case in RL with DNN policies. But the proposed approach is only applicable in the context of an MPC, where it is not a major limitation as we often have well-formulated MPC problems with fairly good models.

4.6 Conclusions

In this work, we presented an actor-critic DPG algorithm for RL with an MPC policy. The proposed method eliminates the need for an additional value function parameterization such as DNNs by approximating the value function using the MPC policy itself. We expect this would relieve the pain of dealing with a complex function approximation for the value function while applying DPG to RL based MPC. In the future, we will extend our approach to full critic approximation using MPC and thereby eliminating the need for a critic in DPG method for MPC.

Chapter 5

A Survey on Safe Learning for Control

This chapter is based on the following publication [8]:

Anand, A., Seel, K., Gjærum, V., Håkansson, A., Robinson, H., and Saad, A. (2021). Safe learning for control using control Lyapunov functions and control barrier functions: A review. Procedia Computer Science, 192, 3987-3997.

5.1 Introduction

The notion of safety in learning-based control for real-world systems, such as robotic systems, has two aspects. The first aspect is ensuring safety during the learning process or the exploration phase, in cases when learning is done online. Online learning is here used to describe the process of learning by simultaneously interacting with and sampling from the system. The second aspect is guaranteeing the safety of an already learned control policy. Specifically for most real-world robotic systems, learning the model or the control policy for the system should be done partially or entirely online using the physical system, depending on the availability and accuracy of a prior model of the system. Even if a model of the system is available, learning a policy in the simulation will often require fine-tuning on the physical system, to compensate for inaccuracies of the prior model. This demands guaranteeing safety during the online learning process on the physical system in order to avoid any kind of damage to the robotic system or its environment. At the same time, the resulting learned control policy should have provable safety guarantees to facilitate its deployment on the real-world robotic system. For most learning-based algorithms, incorporating these guarantees may be done either

by transforming the optimization problem or by changing the exploration process [63].

For safety-critical systems, the research community has mainly focused on safety in terms on constraint satisfaction, and mainly on state constraints. However, few of the learning algorithms used for designing learning-based controllers are capable of naturally incorporating state and input constraints. One of the predominant approaches for ensuring safe learning-based control has been model predictive control (MPC) frameworks, that naturally incorporate state and input constraint satisfaction. MPC has been combined with SL methods in order to build or improve the prediction model as in [142, 191]. There are also many examples of MPC being combined with RL, for example as a value function approximator [227, 68], where stability and constraint satisfaction is ensured by design.

Another approach for ensuring safe learning-based control, which is more agnostic with respect to the control framework, is the use of *safety filters*. Safety filters function by verifying if a learned controller ensures safety in terms of the system states remaining inside a predefined safe set for all future times. If the learned controller does not pass the verification, it can be replaced by a known, safe controller. Alternatively, the learned control input can be minimally modified using an optimization problem, such that it satisfies the safety constraints. For both types of filters, a predefined safe set is necessary. One prominent method to calculate the safe set is using reachability analysis [64]. However, this can be computationally demanding or result in potentially conservative approximations.

A different method for defining the safe sets, is employing control barrier functions (CBFs) [223]. CBFs gained popularity within the conventional control community during recent years, but have been utilized more as a safety filter for an existing nominal controller [6]. Many recent works in the learning-based control field use CBFs [207, 145]. By utilizing probabilistic data-driven methods such as Gaussian processes (GPs), complete prior system knowledge is no longer needed, and probabilistic safety guarantees can be provided [45, 107, 218, 55].

In a different but slightly more conservative approach, safety can be guaranteed using the stability guarantees of the closed-loop system. These approaches are typically based on Lyapunov stability verification, using control Lyapunov functions (CLFs)[103]. Compared to the safe sets, found either by reachability analysis or using CBFs, level sets of CLFs are both invariant and attractive. If the safe set can be designed as a subset of the region of attraction (ROA) defined by a Lyapunov function, then CLFs can be used to guarantee the safety of the closed-loop system, and exploration in closed-loop is then typically limited to this region. A combination of CLF and CBF can be used to guarantee a safe and stabilizing controller

[177], which is also utilized in learning to guarantee stability [95]. The combination of functions has been used in MPC frameworks [225], but are seldomly used in model-based RL algorithms [48].

For safety-critical systems, it can be paramount to ensure safety, either in terms of constraint satisfaction or in terms of stability, or both. Ensuring safety and stability by using CBFs and CLFs is particularly suited for learning, as both properties can be expressed using functions that can be learned. This chapter aims to review approaches using CLFs and CBFs for ensuring safety and stability in learning-based methods considering their suitability in a learning framework and because they can be applied to a broad wide range of control frameworks. Even though this review is motivated by robotic applications, these approaches are generalizable to other types of control applications e.g. process control. The rest of the chapter is organized as follows: Section 5.2 presents related work. The review of the learning algorithms is organized as follows: Section 5.3, presents a technical background of how CLFs and CBFs are used to provide safety guarantees. Section 5.4, treats RL algorithms, section 5.5, presents a review of online SL algorithms, followed by offline SL methods in Section 5.6. The mentioned categories and the corresponding relevant references treated for each of them are listed in Table 5.1. To the best of the authors' knowledge, there is currently no published research work on the combination of CLFs and CBFs for offline SL methods. Section 5.7 provides a short summary of the algorithms treated in the former section and a discussion of the suitability of the methods for different applications. Finally, in Section 5.8, the concluding remarks are presented.

Table 5.1: Overview of the relevant literature.

	CBF	CLF	CLF + CBF
RL	Sec 5.4.1 : [45, 145]	Sec 5.4.2 : [50, 49, 167, 24]	Sec 5.4.3: [48]
Online SL	Sec 5.5.1 : [107, 218, 207]	Sec 5.5.2 : [41, 152, 214, 228, 22]	Sec 5.5.3 : [95, 55]
Offline SL	Sec 5.6.1 : [201, 230, 93, 186, 176]	Sec 5.6.2 : [208]	-

5.2 Related Work

There are a few review papers in the area of learning for control, discussing different methods for ensuring safety. A survey on model learning of autonomous systems is presented in Nguyen-Tuong et al. [159], discussing safety challenges in existing methods. In [206], a comprehensive review of safety criteria and metrics, controller design along with mechanical design and actuation for domestic robotic systems is presented, where learning-based approaches were mentioned briefly. A survey on safe RL [63], discusses cases where it is important to respect safety constraints during learning and/or deployment. In this survey, the authors categorize safe RL methods, based on whether the safety criterion is incorporated

into the optimality criterion or by modifying the exploration process. In [133] and [226], the authors present a comprehensive survey of methods for safe human-robot interaction. A review work on learning-based MPC with emphasis on safe learning is presented in [79], categorizing the methods based on learning the system dynamics, learning the control policy, or using safety filters. In [112], a review of safe learning and optimization methods is presented as a continuation of [63] with an additional review of active learning and optimization methods. Kim et al. [112] review learning and optimization algorithms that ensure safety, where safety is guaranteed by adding constraints and/or ensuring that any unsafe states are avoided. But none of these surveys specifically address the CLF- and CBF-based approaches in learning-based control.

5.3 Safety Guarantees using CBFs and CLFs

Throughout this chapter, nonlinear dynamical systems in their general form (5.1) and a control affine form (5.2) is considered, where f and g are locally Lipschitz, $x \in \mathcal{X} \subseteq \mathbb{R}^n$ denotes the state and, $u \in \mathcal{U} \subseteq \mathbb{R}^m$ denotes the control input.

$$\dot{x} = f(x, u)\,, \tag{5.1}$$

$$\dot{x} = f(x) + g(x)u\,. \tag{5.2}$$

For the rest of this section, we will use the following notation. A continuous function $\alpha : [0, a) \to [0, \infty)$ is called a class $\mathcal{K}$ function, if $\alpha(0) = 0$ and it is strictly increasing. A function is called a class $\mathcal{K}_\infty$ function, if it belongs to class $\mathcal{K}$, $a = \infty$ and $\lim_{r\to\infty} \alpha(r) = \infty$. Let $L_m Q(x)$ denote the Lie derivative of a function $Q(x)$ along another function $m(x)$ i.e. $L_m Q(x) := \frac{\partial Q(x)}{\partial x} m(x)$.

5.3.1 Notion of Safety

For a dynamical system controlled by a control policy in order to perform a task, a general notion of safety is considered. For the system to be safe, it should be guaranteed that the system will never enter any unsafe region under the current policy. Safety can be enforced by ensuring the forward invariance of a safe set [6]. That is, all trajectories starting in the set of safe states will remain within the safe set for all $t \geq 0$.

Definition 1 *(Safe control) Consider a general form of dynamical systems (5.1), where the control policy $u = u(x)$ is a mapping from state to the optimal control action, $u : \mathcal{X} \to \mathcal{U}$. Consider a given set of unsafe states $\mathcal{X}_u \subseteq \mathcal{X}$, a set of initial condition $\mathcal{X}_0 \subseteq \mathcal{X}$ and a set of target/goal states $\mathcal{X}_g \subseteq \mathcal{X}$ where $\mathcal{X}_u \cap \mathcal{X}_0 = \emptyset$ and $\mathcal{X}_u \cap \mathcal{X}_g = \emptyset$. If for all the possible trajectories $x(t)$ evolving from the set of initial*

conditions to the set of goal states, such that $x(t) \notin \mathcal{X}_u$, for all time, $t \in T \subseteq \mathbb{R}^+$, the system is guaranteed to be safe under the control policy $u(x)$.

5.3.2 Control Lyapunov Functions

The notion of control Lyapunov function [60] to design asymptotically stabilizing controllers to a nonlinear system, was introduced by Artstein and generalized by Sontag [14, 200]. Extending the Lyapunov function to a control Lyapunov function (CLF) helps to find a control law that ensures the stability of a dynamical system in (5.2).

Definition 2 *A positive definite function $V : \mathbb{R}^n \to \mathbb{R}$, is a CLF to the system (5.2), if there exist a class $\mathcal{K}$ function, γ, and a control law,*

$$u_{clf} = u \in U, \quad s.t \quad L_f V(x) + L_g V(x) u + \gamma(V(x)) \leq 0\,, \tag{5.3}$$

which render the system asymptotically stable at the point $x^* = 0$ where $V(x^*) = 0$. For a system (5.2), the existence of a CLF implies that for all x in the level set $\mathcal{V}(c) = \{x \in \mathcal{X} | V(x) \leq c\}$ for $c > 0$, the control law u_{clf} asymptotically stabilizes the system. The largest level set is referred to as the region of attraction (ROA), and is forward invariant and attractive.

5.3.3 Control Barrier Functions

A set $\mathcal{C}$, defined as a super level set of a continuously differentiable function $h : \mathcal{X} \subset \mathbb{R}^n \to \mathbb{R}$, is safe if,

$$\mathcal{C} = \{h(x) \geq 0\}\,, \qquad \partial\mathcal{C} = \{h(x) = 0\}\,, \qquad \text{Int}(\mathcal{C}) = \{h(x) > 0\}\,, \tag{5.4}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ and $\partial\mathcal{C}$ represents the boundary of $\mathcal{C}$. In order to address safety while controlling dynamical systems, the control barrier function (CBF) is introduced [223]. The general definition considered here is from [6].

Definition 3 *h in (5.4) is a CBF for a dynamical system in (5.2), if there exists a class $\mathcal{K}_\infty$ function, α defined over the entire real line, $\mathbb{R}$, and a control law $u_{cbf} = u \in U$, s.t*

$$L_f h(x) + L_g h(x) u + \alpha(h(x)) \geq 0\,, \tag{5.5}$$

for all $x \in \mathcal{X}$. Given the control barrier condition (CBC) in (5.5), the safe set $\mathcal{C}$ is forward invariant for the system (5.2).

5.4 CBFs and CLFs in Reinforcement Learning

Complex dynamical systems, such as robotic systems, are often difficult to model accurately due to their highly nonlinear and uncertain dynamics. Model-based RL can be used to estimate the unknown system dynamics online, while simultaneously learning an optimal policy to perform a particular task using samples from the learned model. Whereas model-free RL algorithms can be used to learn the policy directly. Both types of algorithms come at the cost of demanding online safety certification during learning, in addition to safety certification of the learned policy. Therefore safety certification of RL algorithms for systems with unknown dynamics is two-fold, (1) safety certification during learning and (2) offline safety certification of the learned policy i.e. after convergence. Depending on the selected RL algorithm, both regards may be ensured simultaneously or separately.

5.4.1 Control Barrier Functions

There are a few different approaches to utilizing CBFs to guarantee safety in an RL setting. In [45], a framework for combining model-free RL algorithms with model-based CBFs is proposed. The approach ensures the safety and improves the exploration efficiency of the RL algorithm. The model of the dynamical system to be controlled is assumed unknown. As the RL algorithm explores the system's states, measurements are used to update a GP model used to learn the unknown system dynamics. The GP model is in turn used to derive the CBC defined in Section 5.3.3. Here, safety is determined by adding a compensating CBF-based controller to the model-free RL policy. This is formulated using a quadratic program (QP) with CBC constraints as a safety filter, which aims to modify the RL policy as little as possible, while ensuring that the state remains within the safe set. This approach resembles the general safety filter formulated in [217]. As policy iteration is done considering the altered RL policy, i.e. with the CBF addition, the learned policy is encouraged to operate in the safe part of state space. To avoid solving a QP every time-step during deployment the method is extended by approximating the compensating CBF-based controller using a neural network (NN) during the learning process.

Another approach is presented in [145], where an off-policy actor-critic method is used to learn a policy without requiring knowledge of the system dynamics, i.e a model-free RL algorithm. A safe, possibly conservative policy is used to explore while the algorithm learns, and by adding a CBF to the value function, the learned policy will stay inside the safe set determined by the CBF condition. A coefficient is used to define a trade-off between optimality, defined in terms of the original utility function, and safety, determined using the CBF. Using an off-policy algorithm, where the resulting policy is approximated using a NN, the

safety guarantees will only carry over given that the NN converges to the optimal solution.

5.4.2 Control Lyapunov Functions

For RL algorithms, a distinction is typically made between policy-gradient methods as opposed to value-based methods. For value-based methods, the Lyapunov-based approach has generated interest, because the value function can be used as a Lyapunov function. This idea is exploited in [24], where safety constraints are defined using the ROA, i.e all states inside the ROA are safe. The ROA can be approximated for a fixed policy, by formulating a Lyapunov function and taking the largest level set as the ROA, as defined in Section 5.3.2. In [24], the uncertain dynamics are learned using GPs with measurements collected online by sampling from the system. Confidence intervals, defined based on the learned system dynamics model, are used to check the Lyapunov decrease condition for the system. An optimization problem constrained by the Lyapunov decrease condition is then solved to find a policy that results in the largest possible level set (largest possible ROA). The exploration strategy is based on information maximization. By choosing state-action pairs where the dynamics are most uncertain, the confidence interval will shrink so the ROA can be expanded incrementally. This is the same exploration strategy used in [22], which is treated in Section 5.5.2.

Lyapunov functions have also been used with RL algorithms to achieve a different understanding of safety. In [49], an agent's behavior policy is defined to be safe, if the cumulative cost constraint of the constrained Markov decision problem is satisfied. The Lyapunov function is designed to be a uniform upper bound on the constraint cost, such that the corresponding algorithm guarantees feasibility and optimality under certain conditions. This approach is also extended to policy-gradient methods in [50], for which the policy function, that is a mapping from state to action, is learned directly.

In a different approach, Lyapunov functions have been used to construct safe RL agents that switch among a safe base of controllers. This was first done in [167], where Lyapunov functions are used to provide stability guarantees for each controller.

5.4.3 Control Lyapunov Functions and Control Barrier Functions

In [48], system uncertainty is estimated in the CLF and the CBFs using deep NNs. An actor-critic RL algorithm is used to minimize the effect of model uncertainty in the learned CLF and CBF using a reward function that penalizes the estimation errors in the CLF and CBF. The learned CBF and CLF constraints, compensating for model uncertainty, are exploited in a QP to find a safe and stable controller

for the uncertain system, using input-output linearization. This work assumes that CLFs and CBFs designed for the nominal model will also serve as CLFs and CBFs for the true system. This assumption holds for systems where the nominal model and the true system have the same relative degree [207].

5.5 CBFs and CLFs in Online Supervised Learning

Online SL methods are here understood as either continuous or episodic learning of system components in a closed loop. In this setting, SL methods are used for learning a model with data collected by sampling from the controlled system. Unlike RL algorithms, online SL methods are usually not used to optimize a control policy directly. However, for an uncertain system, CBFs or CLFs can be learned in order to determine a safe controller. Often a safe controller is found by solving an optimization problem, with constraints formulated using the learned models.

5.5.1 Control Barrier Functions

In [207], the authors present an approach to improve the safety of a dynamical system by estimating the model uncertainty using CBFs. Instead of estimating the uncertainty by learning the system dynamics from measurements, the uncertainty is modeled directly in the CBF. This approach is less restrictive on the types of system uncertainties, as it estimates both uncertainties due to parametric errors and unmodelled dynamics. The uncertainty is learned episodically using NNs and included as a constraint in an optimization problem modifying a nominal controller to be safe. It is assumed that if there exists a valid CBF for the nominal model of the system, then it is also a valid CBF for the uncertain model. For learning the uncertainties, an episodic learning approach is used, which alternates between collecting data using the current controller and synthesizing a new controller by solving a QP. At every iteration of the episodic learning, a heuristically weighted blend of the newly synthesized controller and the nominal controller is used to explore new data.

A different approach for learning the safe region of an unknown dynamical system is presented in [218]. This paper considers a dynamical system on the form (5.2) with an additional unknown affine disturbance term, which is modeled using a GP. A high probability confidence interval is defined over this GP model. It is assumed that an initial safe region and the corresponding barrier certificates are given. With online learning, the safe region is expanded until no more improvement is obtained with further exploration. A QP is formulated to maximize the volume of the barrier-certified safe region with CBC as the constraint. An adaptive sampling method of the discretized state space, namely an information-maximization-based exploration method, inspired by [22], is proposed. The system is driven to any se-

lected state by employing a nominal controller augmented with a safety filter (the QP with CBC as a constraint). The approach is successfully demonstrated on a quadrotor to learn maximally aggressive movements in the vertical direction with an uncertain model and limited thrust.

A more general approach in the direction of safe online learning of system dynamics is presented in [107], for a nonlinear control affine dynamical system (5.2). The unknown system dynamics are modeled as a GP and used to optimize the system behavior and to guarantee safety with high probability. A chance constraint, i.e. a constraint that needs to hold for the entire state or input trajectory with a given probability, is specified using a predefined CBF defined by the estimated dynamics. The chance-constrained version of an optimization problem for the control input is solved by providing safety guarantees for a zero-order hold (ZOH) controller over a control time step. The safe control input provided by solving the constrained QP is then used to explore in order to train the GP. Similar probabilistic safety guarantees were extended to systems with arbitrary relative degree using exponential CBFs [6].

5.5.2 Control Lyapunov Functions

Online SL methods can be used for finding safe controllers using Lyapunov analysis. In [41], GP regression is used to model the uncertainties in the system. The resulting stochastic model is used to formulate a stochastic CLF, included as a chance constraint in a second-order cone program (SOCP). A stabilizing controller for the system with probabilistic guarantees is derived by solving the SOCP.

An accurate estimate of the ROA is useful for ensuring safety as addressed in Section 5.4.2. Learning can be used to estimate the unknown parts of the dynamic model of an uncertain system and thereby expand the ROA as exploited in [22]. In [22], a GP is used to learn the uncertain parts of the dynamics and the ROA is taken as the largest level-set of the resulting CLF. The next state to be explored is chosen as the point with largest variance within the current estimate of the ROA. A fixed locally safe controller is used to drive the system to the chosen next state. The episodically updated GP model incrementally decreases its variance and thereby increases the accuracy of the ROA estimation.

A similar approach is used in [228], where the goal is to find an accurate estimate of the ROA. Here a GP is used for learning the Lyapunov function rather than dynamics as in [22]. Using the converse Lyapunov theorem, which states that for a stable system, there exists a Lyapunov function, a GP is trained with closed-loop data in order to infer the Lyapunov function, and in turn, estimate the ROA as described above. Safe samples are collected using an algorithm that aims to balance

the trade-off between exploration, in order to expand the resulting estimate of the ROA, and exploitation, i.e. reducing the uncertainty of the GP-learned Lyapunov function.

Learning a Lyapunov function can be useful for the stability analysis of MPC algorithms. In [152], a NN is used for learning a Lyapunov function for the closed-loop system, used as a terminal cost in the MPC scheme. For uncertain nonlinear systems, this often needs to be conservatively approximated, with implications on closed-loop performance. By learning the terminal cost from data, the learned Lyapunov function is used to guarantee the stability of the closed-loop system, in addition to ensuring robustness with respect to model errors in the prediction model.

Stability properties formulated using Lyapunov functions may be used to add knowledge about an existing closed-loop system, for which the system dynamics are unknown. This is investigated in [214], where the stochastic dynamics of a closed-loop system are learned based on training data and a constrained likelihood maximization problem, exploiting the fact that the closed-loop system is exponentially stable. The stability property is expressed in terms of a Lyapunov function, included as a constraint to the maximization problem.

5.5.3 Control Lyapunov Functions and Control Barrier Functions

There are two notable approaches combining CBFs and CLFs in an online SL framework to ensure safety and stability. In [55], the authors augment the approach presented in [107] by using both CLFs and CBFs. The system dynamics are learned online while satisfying safety constraints. The computationally efficient matrix variate Gaussian process regression method is used to learn the drift and input gain terms of control affine dynamical system. In addition to a CBF-based chance constraint in [107], a CLF-based chance constraint is included for specifying stability constraints. This method is extended to systems with the arbitrary relative degree to synthesize a safe control policy by solving a deterministic SOCP.

The second approach presented in [95], uses SL to learn a safe and optimal goal-reaching policy using a barrier function and a Lyapunov-like function respectively, for dynamical systems of the form (5.1) The condition for asymptotic stability is translated to goal reaching utilizing a Lyapunov-like function, considering the equilibrium point as the goal. The proposed Lyapunov-like function is less restrictive as it allows for specifying a set of goal states rather than just a fixed point. In addition, the Lie derivative is not required to always be negative definite. The policy, barrier function, and Lyapunov-like function are parameterized using deep

NNs. The loss function for learning the barrier function is designed to penalize the violation of any constraints. Similarly, another loss function is defined to penalize the violation of the Lyapunov-like function constraint. The two loss functions are then combined into a single optimization objective called the *total certificate risk*, which is positive and semi-definite. Joint training of the barrier certificate, Lyapunov-like function and the policy networks is achieved by minimizing this objective. Additionally, a verification procedure is introduced to confirm the validity of the barrier and Lyapunov-like networks. One drawback of this approach is that it does not guarantee safety during the learning process. Guarantees are only provided for the final policy obtained from the learning process upon verification of the networks.

5.6 CBFs and CLFs in Offline Supervised Learning

Similarly to online SL and RL, offline SL can be utilized to learn the unknown system dynamics, CBFs, and/or CLFs from collected data. Unlike in RL or online SL, offline SL is not an active learning method, for which an algorithm chooses the data points from the sampling space of the system using an exploration strategy. However, provided with an adequate data set, offline supervised learning can achieve the same final goals.

5.6.1 Control Barrier Functions

There are a few interesting approaches to learning CBFs for nonlinear systems with unknown dynamics in an offline supervised learning set-up. Saveriano et al. [186] focus on incremental learning of a set of linear parametric zeroing control barrier functions (ZCBFs)[6]. ZCBF is a type of CBF, which approaches infinity in the boundary of its safe set. ZCBFs are combined with a dynamical system-based motion planner such as dynamic movement primitives to ensure the constraint satisfaction for planned trajectories. The state constraint for the motion trajectory can be learned from human demonstrations and formulated as ZCBFs. A QP can then be used to find a stabilizing controller, where the states of the motion planner are constrained by the ZCBF. This enables the motion planner to generate a feasible motion trajectory satisfying the safety constraints. Another approach for estimating ZCBFs (both in offline and online settings) of control affine robotic systems from sensor data is presented in [201]. A support vector machine (SVM) approach, namely the kernel-SVM method, is used to classify the set of safe and unsafe states in the data set. An online approach is defined for the scenario where the full set of unsafe samples from the environment is not available for offline learning. Robey et al. [176] present an approach to learn CBFs for nonlinear control affine dynamical systems of the form (5.2) using expert demonstrations of safe trajectories. This approach is agnostic to the parameterization used to represent CBFs. An optimiz-

ation method is defined to synthesize valid local CBFs from the collected expert demonstrations.

An approach to synthesize NN-based controllers for nonlinear dynamical systems where safety guarantees are provided by NN-based barrier functions is proposed in [230]. The controller and the barrier functions are simultaneously trained using the same data set using a modified Stochastic Gradient Descent (SGD) optimization technique. A formal verification to guarantee the safety of the synthesized controller is provided.

In a different approach using GPs, presented in [93], the authors learn the unknown control affine nonlinear dynamics as GPs. A parametric nonlinear CBF is generated based on a counterexample-guided inductive synthesis (CEGIS) method. A control policy is synthesized with safety guarantees in three steps. In the first step a GP is learned using a data set and a confidence interval is defined using the uncertainty of the learned model. The second step involves computing a parametric CBF using CEGIS. In the third step a controller can be synthesized by solving an optimization problem with a CBC constraint.

5.6.2 Control Lyapunov Functions

For uncertain systems, offline supervised learning may be used to improve the estimate of a Lyapunov function. This approach is explored in [208], where the derivative of the Lyapunov function is learned offline. Using an episodic learning approach, the time derivative of the Lyapunov function is iteratively improved. This, in turn, is used to ensure that the nominal controller, augmented with a QP-based optimal controller, satisfies the necessary conditions of the time derivative of the Lyapunov function, and renders the closed-loop system stable. The formulation of the QP is similar to the formulation in [207], except that a CLF rather than a CBF, is used to formulate the constraints.

5.7 Discussion

This chapter presented a review of three different learning methods that use CLFs and CBFs for ensuring safe learning-based control, namely RL, online, and offline SL. RL and typically online SL are both active learning methods but differ in the way the control policy is derived. RL offers a flexible framework to learn any complex control policies based on data, while SL methods are often used in combination with optimization to find a control policy. Offline SL differs from these two approaches in its data collection strategy as it uses pre-collected data sets. Active learning methods may be more data-efficient for learning control policies for systems with complex dynamics, compared to offline SL methods [43]. Especially for high-dimensional robotic systems, the data requirements scale expo-

nentially with the state space dimension, demanding very large pre-collected data sets. Active learning methods can explore state space efficiently, using for example an information maximization exploration strategy. Low-quality data sets could adversely affect the accuracy of the estimated model and thereby the resulting control policy. On the other hand, offline SL methods are suitable for learning from expert demonstrations.

Online SL and RL methods offer an option to incrementally update/optimize the control policy and use the same policy to sample. An example is model-based policy search algorithms, which are proven to be very sample efficient for policy optimization [43]. For robotic systems, a nominal safe controller may be used for collecting data or initial exploration of state space, but this may only be valid in small local areas. For offline SL methods, this can therefore result in small and local data sets, limiting the possible control policies that can be learned. Gradually obtaining a less conservative control policy using incrementally updated estimates of the system model, as in online SL and RL methods, can therefore be very convenient.

Using NNs to model the optimal policy in RL algorithms enable approximation of arbitrary complex policies. On the downside, providing safety guarantees for the resulting policy is hard as it is only an approximation of the safe policy. SL methods will often be used in combination with optimization to derive the controller, for which it can be easier to provide stricter guarantees. Combining RL with an optimization-based safety filter could provide stricter safety guarantees, in addition to providing rich expressibility of the final policy. However, this comes at the cost of solving an additional optimization problem in real-time as discussed in Section 5.4.1. For optimization-based safety filters that alter the learned policy in order to satisfy safety constraints, a relevant question is how this modification affects the learning process. Depending on the applied RL algorithm, this can disrupt the learning process such that the learned policy becomes suboptimal. This issue is addressed for several RL algorithms in [69].

Offline SL methods can generate a model of the system dynamics, which can be used to formulate an optimal control problem for the system. Offline SL methods are predominantly used for learning CBFs rather than learning the dynamics model or CLFs, as observed in Section 5.6. For some dynamical systems, controlling the system may not be needed in order to derive the safety constraints. One example is in the case of collision avoidance, where a camera system can be used to detect possible collision objects and learn the corresponding safety constraints. In this case, offline SL may be an ideal tool for providing safe control policies. Learning CBFs using NNs could be suitable as it can represent a complex safe set accurately and incorporate constraints in real-time that are otherwise hard to model.

Robust and well-established machine learning approaches, such as clustering and classification can be utilized in learning the safe sets and thereby CBFs or CLFs. Consequently, CBFs and CLFs learned through offline SL could be used with RL or online SL methods to derive less restrictive controllers than using conservative CBFs or CLFs defined for the uncertain system.

Considering robotics applications, both RL and online SL methods are suited as it may be hard to collect data offline. For applications demanding strict safety guarantees either online SL or RL with a safety filter can be used. Wang et al. [218] provide a very good example of the practical use of a safety filter where a quadcopter's dynamics are learned safely using CBFs in an online SL setting. Offline SL methods can aid RL and online SL methods in learning the system dynamics and controller less restrictively. For robotic systems the dynamics are often control affine, an approximate prior model is usually available and a major part of the uncertain dynamics are linked to the environment rather than the robot itself. These properties make CBFs particularly suited for guaranteeing safety for a wide variety of robotic systems. CBF-based approaches can be generalized to most real-world robotic systems using exponential CBFs [6]. This includes systems such as robotic manipulators, bipedal robots, unmanned ground and aerial vehicles, autonomous underwater vehicles etc.

When learning a controller for an uncertain system, desired safety and stability properties will dictate which approach is suited for ensuring safe control. The combination of CLFs and CBFs can be used to obtain control policies that ensure stability and safety for a wide range of safety-critical systems. If only interested in constraint satisfaction, then CLF-based methods will limit the set of possible control policies. This is because a policy derived from this approach will render the system asymptotically stable in addition to guaranteeing constraint satisfaction. Therefore CBFs are particularly suited for scenarios where safety is the primary goal, as it offers a less restrictive way to ensure constraint satisfaction compared to using CLFs. In cases where stability is of major importance, CLFs can provide constraint satisfaction in addition to stability guarantees at the expense of more restrictive conditions. In case of value function-based RL algorithms, CLFs can be incorporated naturally by using the value function itself as a Lyapunov function [24].

5.8 Conclusions

This chapter presents a literature review of learning methods that incorporate CBFs and CLFs and their combination. The review summarizes the existing learning-based methods for safe control of dynamical systems with uncertainty, utilizing CBFs and CLFs. The relevant references are divided into three main categories,

decided by the learning method that CBFs and CLFs are combined with, namely RL, online, and offline SL as shown in Table 5.1. The similarities and differences between the methods used in the review references are highlighted and their suitability in different scenarios is discussed. It is observed that, despite steady progress, there still exists a large gap between the theory and practical application of the methods. Because using CLFs and CBFs with learning is a rather new approach, a major challenge ahead is demonstrating their capabilities on real-world safety-critical systems. This widens the scope for future research in the area.

Part II

Variable Impedance Learning Control for Robotic Manipulation

Chapter 6

Robotic Manipulation and Variable Impedance Control

This chapter introduces the fundamentals of compliant robotic manipulation focusing on variable impedance control. The concepts introduced in this chapter will serve as preliminary to the rest of this part.

6.1 Compliance Control for Robotic Manipulation

When a robot is physically interacting with its environment during a manipulation task, it is crucial to control this interaction. Incorporating compliance into robot behavior is the key to achieving dexterous, safe, and human-level manipulation skills, especially in interaction tasks [83]. Robotic controls for following desired trajectories while ensuring a *compliant behavior* with respect to external forces for providing safe and stable control can be developed by combining elements of motion control and force control. The research on incorporating compliance in robot control for manipulation is well-studied in the area of robotic interaction control [216]. Such compliant control approaches enable robots to perform complex manipulation tasks using interaction force feedback to provide adequate compliance.

Compliance properties can be incorporated into robots using either using *passive* mechanisms or *active* control approaches. In the passive approach, the compliant behavior is inbuilt into the robot's hardware by structural compliance on the joints, links, and end-effector or by incorporating compliance into the position servo. Soft robots inherently possess such passive compliance mechanisms [181]. Active compliance control approaches are more sophisticated as they easily enable compliance properties into mechanically stiff robots with the possibility to adapt them

freely [190]. From a control perspective, passive compliance mechanisms provide a cheaper way to achieve compliance properties in robots. But they lack flexibility as different robotic tasks might demand task-specific end-effectors or even robots be designed. Similarly, the compliance properties of such passive mechanisms are not easily adaptable during a task. Whereas in active compliance control, the compliance of the robot is modulated using a controller providing flexibility and adaptability. A reliable measurement of the interaction force is very critical in active approaches [216].

A major part of the research presented in this part falls into the category of active compliance control which can be divided into two categories, Impedance Control (IC) and Force/Motion Control (F/M Control) (also referred to as Force/Position Control in some literature) [190]. IC can be seen as indirect force control as they achieve compliance by adjusting the reference position based on force-feedback. On the contrary F/M Control rely on a combination of direct force control to regulate the contact force and motion control for trajectory tracking. This classification of the compliance control scheme is shown in as shown in Fig. 6.1. Therefore, hybrid and parallel F/M Control try to track the target force and/or position, while impedance and admittance control tries to establish a desired relationship with the force and position. The Hybrid Force-Motion Controller (HFMC) approach is suitable for systems with multiple Degree of Freedom (DoF) [174] where as parallel F/M Control can be applied to systems with a single DoF systems [47]. HFMC aims to achieve both motion and force control by dividing the task into two separate, decoupled sub-problems. By specifying which sub-spaces should be controlled by a motion- and force controller respectively, the hybrid control intends to solve the two separate control tasks simultaneously.

IC for robot control, introduced by Hogan in [83], aims to couple the manipulator dynamics with its environment instead of treating it as an isolated system while designing control strategies. IC attempts to implement a dynamic relation between manipulator variables such as end-point positions and forces rather than control these variables independently. The use of IC provides a feasible solution to overcome position uncertainties in order to avoid large impact forces since robots are controlled to modulate their motion or compliance according to force feedback [106]. IC being indirect force-control methods does not use any direct force-feedback control, they achieve force control indirectly through motion control [216], for e.g., by changing the reference position to comply with the interaction force.

There are two different types of IC (i) force-based IC and (ii) Admittance Control (AdC). Although both of these were referred to as impedance control in the original formulation by [83], these are two different ways of implementing im-

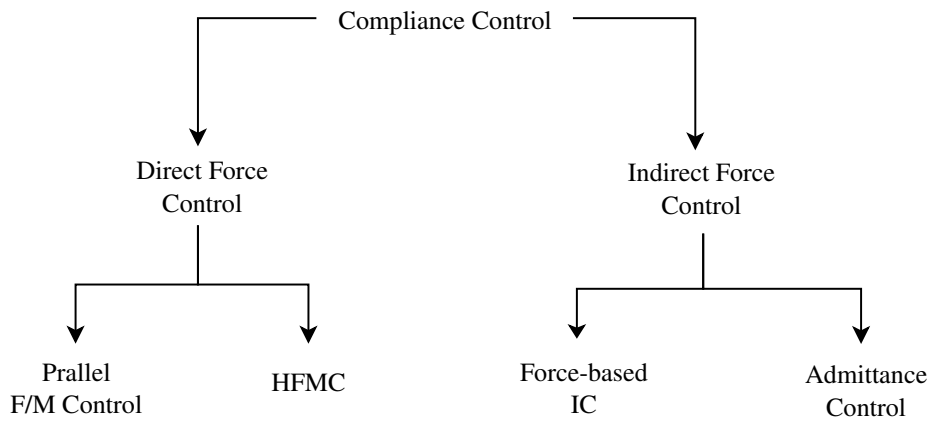


Figure 6.1: Classification of the compliance control approaches in robotic manipulation.

pedance control. A system is considered to be impedance if it receives flow as input and produces an effort as output. On the contrary, if it receives the effort as input and produces flow as output, it is an admittance. This concept helps to distinguish between the two types of IC [131]. In force-based IC, the controller is an impedance and the manipulator is admittance, whereas in AdC, the controller is an admittance and the manipulator is an impedance. Therefore in practice, a force-based IC, provides the control force/torque to a force-controlled robot manipulator and an AdC provides a target position to the position-controller robot manipulator [161]. This thesis emphasizes on force-based IC, and is interchangeably referred to as IC in rest of the this thesis. The mathematical formulation of IC is provided in the next section.

6.2 Impedance Control

This section provides the mathematical formulation for the force-based IC and describes its relationship to Variable Impedance Control (VIC). All the mathematical formulations are provided in Cartesian space which is equivalent to the task space of the robot manipulator.

6.2.1 Task-space Formulation of Robot Manipulator Dynamics

For the task at hand, it is most convenient to consider the task space formulation of the dynamical system. For a rigid n-DOF robotic arm, the task space formulation of the robot dynamics is given by

$$\mathbf{\Lambda}(\mathbf{q})\ddot{\mathbf{x}} + \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \boldsymbol{\eta}(\mathbf{q}) = \mathbf{f_c} - \mathbf{f_{ext}}\,, \tag{6.1}$$

where $\dot{\mathbf{x}}, \ddot{\mathbf{x}}$ are velocity and acceleration of the robot end-effector in task space, $\mathbf{f_c}$ is the task space control force, $\mathbf{f_{ext}}$ is the external force, $\mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{\mathbf{6\times 6}}$ is a matrix representing the centrifugal and Coriolis effects, and $\boldsymbol{\eta}(\mathbf{q}) = \mathbf{J^{-T}}\mathbf{g}(\mathbf{q}) \in \mathbb{R}^{\mathbf{6\times 1}}$ is the gravitational force, where $\mathbf{g}(\mathbf{q})$ is the joint space forces and torques. The cartesian inertia matrix, $\mathbf{\Lambda}(\mathbf{q}) \in \mathbb{R}^{\mathbf{6\times 6}}$, is calculated as

$$\mathbf{\Lambda}(\mathbf{q}) = (\mathbf{J}\mathbf{H}(\mathbf{q})^{-1}\mathbf{J}^T)^{-1}, \tag{6.2}$$

where $\mathbf{H}(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the symmetric and positive-definite joint space inertia matrix. This inertia matrix is representing the mass distribution of the manipulator and is highly state-dependent. By additionally knowing the joint space formulation of the centrifugal and Coriolis effects, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$, the corresponding wrench, $\mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}})$, is

$$\mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{J}^{-T}\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{J}^{-1} - \mathbf{\Lambda}(\mathbf{q})\dot{\mathbf{J}}\mathbf{J}^{-1}. \tag{6.3}$$

6.2.2 Task-space Formulation of Impedance Control

In the presence of a force and torque sensor measuring $\mathbf{f_{ext}}$, IC can be implemented by enabling inertia shaping [216]. Casting the control law

$$\mathbf{f_c} = \mathbf{\Lambda}(\mathbf{q})\boldsymbol{\alpha} + \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \boldsymbol{\eta}(\mathbf{q}) + \mathbf{f_{ext}}\,, \tag{6.4}$$

into the dynamic model in (6.1) results in $\ddot{\mathbf{x}} = \boldsymbol{\alpha}$, $\boldsymbol{\alpha}$ being the control input denoting acceleration with respect to the base frame.

IC in the standard form models a virtual spring–damper system between the environment and the robot end-effector [83], where the desired impedance behavior modeled as a mass-spring-damper system is as follows,

$$\mathbf{M}\delta\ddot{\mathbf{x}} + \mathbf{D}\delta\dot{\mathbf{x}} + \mathbf{K}\delta\mathbf{x} = \mathbf{f_{ext}}\,. \tag{6.5}$$

In task-space IC, the objective is to maintain this dynamics relationship (6.5) between the external force, $\mathbf{f_{ext}}$, and the error in position $\boldsymbol{\delta}\mathbf{x} = \mathbf{x^r} - \mathbf{x}$, velocity $\delta\dot{\mathbf{x}} = \dot{\mathbf{x}}^{\mathbf{r}} - \dot{\mathbf{x}}$ and acceleration $\delta\ddot{\mathbf{x}} = \ddot{\mathbf{x}}^{\mathbf{r}} - \ddot{\mathbf{x}}$. Where $\mathbf{M}$, $\mathbf{D}$ and $\mathbf{K}$ are Symmetric Positive Definite (SPD) matrices, impedance parameters, representing inertia, damping and stiffness terms, respectively. This desired dynamic behaviour (6.5) can be achieved using the following control law,

$$\boldsymbol{\alpha} = \ddot{\mathbf{x}}^{\mathbf{r}} + \mathbf{M^{-1}}(\mathbf{D}\delta\dot{\mathbf{x}} + \mathbf{K}\delta\mathbf{x} - \mathbf{f_{ext}})\,. \tag{6.6}$$

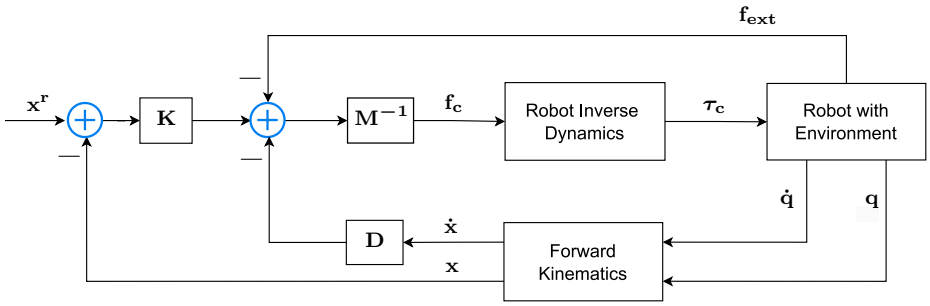


Figure 6.2: Block diagram of IC.

A schematic of the IC is shown in Fig. 6.2 for the case where $\ddot{\mathbf{x}}^{\mathbf{r}} = \dot{\mathbf{x}}^{\mathbf{r}} = 0$. IC is extended to VIC where the impedance parameters $\mathbf{M}$, $\mathbf{D}$ and $\mathbf{K}$ are adjustable parameters to regulate the impedance model (6.5). Essentially, VIC is designed to achieve force regulation by adjusting the system impedance [86], via the adaptation of the inertia, damping, and stiffness components.

Chapter 7

Survey on RL-based Variable Impedance Learning Control

This chapter provides a survey of all the existing Variable Impedance Learning Control (VILC) methods that uses Reinforcement Learning (RL).

7.1 Introduction

In order to develop compliant robotic manipulation skills, Impedance Control (IC) is naturally extended to Variable Impedance Control (VIC) as formulated in (6.6) with variable impedance gains. VILC extends this further to facilitate the use of Machine Learning (ML) approaches to identify the optimal variable impedance laws from data. As described in Chapter 1, among all ML approaches, RL promise to be an ideal tool to learn complex robotic manipulation skills and has been widely explored in recent times in various applications. RL has the potential to identify optimal controllers from data for complex control problems without necessarily having a model. This is ideal for VILC its difficulty to design optimal impedance profiles, especially for high dimensional robotic systems, for example, a seven-DoF robotic manipulator. VIC is a conventional compliant control approach providing indirect torque control with safety and stability guarantees. The combination of RL and VIC can be seen as using RL as a high-level decision maker for the robust low-level VIC. Using RL as a tool to build scalable robust control approaches can be achieved by efficiently combining conventional approaches with RL. From that perspective RL can be used to learn reliable compliant controllers with RL deciding the right impedance parameters in real-time and applying it the low level VIC. In this chapter, we explore the existing research on RL-based VILC and discuss their advantages and disadvantages. We additionally discuss the chal-

lenges RL-based VILC approaches and suggests some future directions to address them.

The rest of this chapter is organized as follows. Section 7.2 presents the survey of all existing RL-based VILC methods. Section 7.3 provides a brief summary of all the research works presented in this survey. A detailed discussion and conclusion are provided in Section 7.4 and Section 7.5 respectively.

7.2 Reinforcement Learning-based VILC

A VIC can be parametrized as a learnable policy of the form,

$$\boldsymbol{\pi}_\theta = \mathbf{K}_\theta \left(\hat{\mathbf{x}}_\theta - \mathbf{x}_t\right) + \mathbf{D}_\theta \left(\dot{\hat{\mathbf{x}}}_\theta - \dot{\mathbf{x}}\right) + \mathbf{f}_{ext} \tag{7.1}$$

where $\boldsymbol{\pi}_\theta$ is the VILC policy parameterised by θ defining impedance parameters (stiffness matrix $\mathbf{K}$ and damping matrix $\mathbf{D}$) and the desired trajectory parameters (desired pose $\hat{\mathbf{x}}$ and desired velocity $\dot{\hat{\mathbf{x}}}$). $\boldsymbol{\pi}_\theta$ can be learned using any type of RL algorithms based on the type of VIC task and the availability of data.

Model-free RL approaches are most widely used in RL applications as they can learn a policy directly from data without needing a model of the dynamics. References [109], [34], [175], [56] approach VILC with model-free RL for relatively simpler policy with fewer parameters.

Reference [109] demonstrated a relatively simpler VILC using an episodic version of the Natural Actor-Critic algorithm [169]. Here the authors proposed an algorithm for a 2-link planar manipulator with a SPD stiffness matrix which is fully represented by the magnitude, the shape, and the orientation. As the SPD stiffness matrix here only has 3 scalar values, it facilitates a faster convergence to the optimal stiffness values. But other than this simpler case, this method is not evaluated more complex real-world scenarios such a 7-DoF robotic manipulator.

In a later work proposed in Ref. [34], the impedance parameters are learned using the highly sample efficient PI^2 algorithm proposed in [210]. The PI^2 algorithm relies on a policy representation that is linear with respect to the learnable parameters. The widely used DMP framework [89] for representing robot motion trajectories offers such a policy parametrization. Authors utilized this property of DMP to represent the desired position and velocity, and a diagonal stiffness matrix $\mathbf{K}$ where the time-derivative of each diagonal term $\mathbf{K}_i$ is given by,

$$\dot{\mathbf{K}}_i = \alpha_K \left(\mathbf{g}_{t,K}^{i,T} \left(\boldsymbol{\theta}_K^i + \boldsymbol{\epsilon}_{K,t}^i\right) - \mathbf{K}_i\right) \tag{7.2}$$

where i indicate i^{th} DoF and $\boldsymbol{\epsilon}^i_{K,t}$ is a time-dependent exploration noise and $\mathbf{g}^i_{t,K}$ is the sum of all the basis functions given by,

$$[\mathbf{g}_{t,K}]_j = \frac{w_j}{\sum_{k=1}^{p} w_k} \tag{7.3}$$

The (7.2) models the time course of the position gains, coupled with a DMP based trajectory generator. $\mathbf{K}_i$ is parametrized using basis functions making it linear with respect to the learnable parameters θ.

This idea was extended further in Ref. [175] using Stable Estimator of Dynamical Systems (SEDS) based policy. The SEDS policy encodes a first-order dynamics into a Gaussian Mixture Model (GMM), providing a non-linear, time-invariant dynamical system as the policy, unlike the time-dependant policy in [34].

$$\dot{\boldsymbol{\xi}}_t = \sum_{g=1}^{G} h_g\left(\boldsymbol{\xi}_t\right) \mathbf{A}_g^{\mathcal{P}} \left(\hat{\boldsymbol{\xi}} - \boldsymbol{\xi}_t\right) \tag{7.4}$$

where $\boldsymbol{\xi}_t$ is a generic state variable and $\hat{\boldsymbol{\xi}}$ is the goal state, $0 < h_g\left(\mathbf{x}_t\right) \leq 1$ is the state-dependent mixing coeeficients and the matrices $\mathbf{A}_g^{\mathcal{P}}$ depends on the learned covariance. The GMM system (7.4) asymptotically converges to its goal $\hat{\boldsymbol{\xi}}$ if all $\mathbf{A}_g^{\mathcal{P}}$ are positive definite as given by the Lyapunov stability theory. The authors modified the PI^2 such that the stability condition is not violated during exploration. The state vector $\boldsymbol{\xi}$ is augmented to include position and stiffness and to encode a variable stiffness profile using (7.4). The corresponding diagonal stiffness matrix and reference trajectory of the VIC is derived from the learned dynamical system at each time-step.

Reference [56] utilized the Fuzzy Q-learning scheme[98] to learn the variable damping gains for an admittance control scheme with a constant inertia matrix and null stiffness. The objective of the Fuzzy Q-learning scheme here is to minimize the jerk in robot motion during human-robot co-manipulation tasks. In an experiment conducted with seven different human subjects performing a co-manipulation task with a robot, the proposed algorithm converged to a sub-optimal policy in just 30 episodes. This is highly sample-efficient compared to other model-free approaches.

All four approaches described above rely on a diagonal form for the stiffness matrix $\mathbf{K}$ to reduce the parameter space facilitating sample-efficient policy learning. But the stiffness matrix is not necessarily diagonal in real-world applications. Therefore assuming a diagonal form for the stiffness matrix neglects the interdependencies between the different DoFs. This problem is discussed in [121],

where the authors proposed to learn an acceleration command as a mixture of G proportional-derivative systems,

$$\mathbf{u}_t = \sum_{g=1}^{G} h_{g,t} \left[\mathbf{K}_g^{\mathcal{P}} \left(\hat{\mathbf{x}}_g - \mathbf{x}_t \right) - D\dot{\mathbf{x}}_t \right] \tag{7.5}$$

where $\mathbf{x}_g$ are the local target and $h_{g,t}$ are the time-varying mixing coefficients, D is a constant damping, and $\mathbf{K}_g^{\mathcal{P}}$ denotes the full stiffness matrix instead of a diagonal form. The authors refer to the full stiffness matrix as *coordination matrix* as it encodes the dependencies among different motion directions. Then proposed approach is demonstrated on a highly-dynamic manipulation task of flipping pancakes.

All five approaches mentioned above rely on highly structured and simpler policies to encode variable impedance profiles for manipulation tasks. This limits their usability in complex manipulation tasks, and it can be difficult to design the right policy structure even if these methods are generalizable to complex tasks. Deep-RL has gained popularity by solving complex control and decision-making problems by incorporating DNN to learn a policy [156]. The deep-RL approach is not sample-efficient but it is easily generalizable to most complex manipulation tasks. References [144, 20, 21, 30, 215] are some examples of using deep RL for VILC applied to different robotic manipulation tasks.

Ref. [144] presented a comparison between well-known robot motion controllers when a deep-RL policy is used to learn the controller parameters. The authors compared five popular controllers, (i) joint position, (ii) velocity, (iii) torque, (iv) Cartesian pose, and (v) Cartesian variable impedance controller. The comparison was conducted over three tasks of path following, door opening, and surface wiping. The controllers were evaluated for five different criteria, (i) sample efficiency and task completion, (ii) energy efficiency, (iii) physical effort (wrenches applied by the robot), (iv) transferability to different robots, and (v) sim-to-real transfer. Proximal Policy Optimization (PPO) algorithm is used to learn a DNN policy for all the five robot controllers. Their findings show that the Cartesian variable impedance control performs well for all the evaluation criteria. One of the most interesting findings of this work is that a VIC policy is easier to transfer between different robots and facilitates an almost seamless transfer of policy learned in simulation to a real robot (sim-to-real transfer). This is highly valuable in many robotic manipulation tasks and helps to tackle the issue of sample-efficiency of RL-based VILC as the simulator offers a cheaper way to generate data.

A similar study in reference [30] explored the effect of different action spaces for

deep RL-based robotic manipulation for contact-sensitive tasks. The authors compared a variable impedance control policy with position control and torque control policies, all in joint space, under different contact uncertainties. For learning the DNN control policies the authors used the popular off-policy RL algorithm, Deep Deterministic Policy Gradient (DDPG). The off-policy RL algorithm was chosen to reduce the issue of local minima which could arise in learning to control in joint space with discontinuities in the dynamics arising from contact interaction, and additionally the complex and multi-part reward functions. An additional reward term is designed to regularize these variable impedance control policies. This additional reward term helps to force the policy to generate desired positions that can be effectively tracked, which helps to learn interpretable policies and allows easier transfer to real systems from simulations. The controllers were evaluated in both simulations and real-world experiments with floating and fixed-base systems in tasks involving contact uncertainties. The evaluation demonstrated VIC provided better task performance and reliability compared to joint position and joint torque controllers.

Further, a similar line of work in [215] evaluates how the choice of action space for dynamic manipulation tasks affects the sample complexity as well as the quality of learned policies. The authors compared four controllers, (i) torque, (ii) joint space PD, (iii) inverse dynamics, and (iv) task-space impedance control The authors compared the learning efficiency for these controllers on three tasks (i) peg insertion, (ii) hammering, and (iii) pushing). Two different deep-RL algorithms (i) PPO and (ii) Soft Actor Critic (SAC)) were used to learn DNN-based control policies. The results of the evaluation supported the original hypothesis of a learning-based task-space impedance controller could significantly reduce the number of samples required to achieve good performance across all tasks and algorithms evaluated. The policies for torque control and PD control were learned significantly slower than impedance control or inverse dynamics control in all the experiments.

In all these three references [144, 30, 215], compared the different learning-based robot motion controllers in the context of deep-RL. The results of all this research support the superiority of VILC for robotic manipulation and contact-rich tasks. RL methods have great potential and are effective in discovering sophisticated control policies. One major drawback of model-free RL approaches is their low sample efficiency which can hinder their application in real-world scenarios. One possibility to alleviate this issue is to use a "good" initial policy and locally refine it done in [109]. [110] combines human demonstrations with RL, providing improved sample efficiency for learning stiffness control policies. But it is not suitable for VILC, as, unlike stiffness values, the impedance parameters can not be

estimated directly from kinesthetic demonstrations used in [110].

But alternatively, MBRL offers a framework to use the knowledge of the system dynamics to improve sample efficiency of RL [170, 220]. While model-based policy search is computationally more expensive than model-free methods, it requires less data to solve a task. A major challenge for MBRL in case contact-rich manipulation is learning/obtaining an accurate model of contact dynamics. Few MBRL approaches have been applied for VILC. The references [137, 136, 179, 10, 9] used different MBRL for VILC.

In the context of VILC for position-controlled industrial robots, references [137, 136] used GP to learn the contact dynamics. The long-term predictions from the learned model are used with the PILCO algorithm [53] to optimize the policy using gradient-based optimization. The effectiveness and sample efficiency of the system were evaluated both in simulations and experiments using the six-DoF Reinovo industrial robot. The results showed the proposed method could outperform model-free VILC methods by at least one order of magnitude.

In Ref. [10], the authors followed a similar MBRL approach to compare the HFMC and VIC for contact-rich manipulation task with GP based dynamic model and PILCO [53] based policy optimization. Evaluating both controllers on a contact force-tracking task in simulation and real-world experiments showed significant improvement in force-tracking ability using MBRL. While introducing learning led to faster convergence to the desired contact force in VIC, it led to a significant improvement in the force tracking error in HFMC. The results showed that having highly accurate contact dynamics models are key to having accurate force tracking.

GP based are extremely sample-efficient and provide smooth function approximations. However, they do not scale with large datasets and tend to smooth out discontinuities that are typical in interaction tasks. Additionally, the PILCO based optimization approach used in [137, 136, 10] restricts the use of complex policies and reward functions. Alternatively in [179], authors used the MBRL algorithm PETS [51] for learning a position-based VIC strategy for HRC tasks. This approach uses an ensemble of Probabilistic Neural Networks (PNN) termed as PENN to learn human–robot interaction dynamics and a CEM based optimizer for varying impedance gains online. The PENN dynamics model is updated offline after every learning episode captures both the epistemic and aleatoric uncertainties in the dynamics. The CEM based optimizer uses the multi-step predictions from the learned PENN model to optimize the variable impedance gains in an MPC fashion. This approach is proved to be highly sample efficient while able to optimize variable impedance gains for complex manipulation tasks.

Reference [9] extended the PETS framework [51] for VILC for general force-based robotic manipulation tasks more comprehensively by learning a generalized PENN model of the robot dynamics. The authors termed the VILC framework as Deep-MPVIC as it combines a force-based VIC with DNN based MPC. But this can be seen as an MBRL framework with an MPC based policy. The proposed framework was evaluated for performance and easiness of the policy to transfer between different tasks. The authors introduced uncertainty targeted exploration to learn a generalized Cartesian impedance model of the robot. Learning a high-quality generalized Cartesian impedance model helped to achieve high performance and facilitated easy transfer between tasks without having to relearn the model. The approach was evaluated on both simulations and real-world experiments on a variety of tasks demanding impedance adaptation such as pushing an object, catching a falling object, door opening, etc. The approach demonstrated high sample efficiency and better performance compared to model-free RL approaches.

HFMC offers an alternative to VIC in contact-sensitive tasks, especially in cases where force and motion tracking direction are decoupled. Similar RL approaches are applied to learn the variable impedance gains for the HFMC in references [141, 21, 20]. Ref. [141] used RL to learn a task-space HFMC for performing high-precision assembly tasks. Ref. [20] used the off-policy, model-free RL method SAC for learning a position-based VIC for peg-in-hole assembly tasks. The major focus of this work is to solve the peg-in-hole tasks with hole-position uncertainty. The VILC policy is represented using time convolutional neural networks (TCNs), providing robustness properties. The control policy consists of three networks, (i) first, proprioception information is processed through a 2-layer NN, (ii) second, force/torque information is processed with a temporal convolutional network, (iii) third, extracted features from the first two networks are concatenated and processed on a 2-layer NN to predict actions. The compliance controller utilized in this work is parallel position-force control instead of standard VIC. Domain randomization technique is used to close the mismatch between the physics simulator and real-world dynamics. One key limitation of this approach is on having a predefined narrow range of parameter values. Although the narrow range of parameter values helps to make it easier and faster to learn a task, it makes it difficult to generalize well across different environments. This approach was adapted for position-controlled robots in Ref. [21] with parallel position/force control, and an admittance control scheme.

7.3 Summary

References [144, 20, 21, 30, 215] are some examples of the use of deep RL for VILC applied to different robotic manipulation tasks. [144] compares different action spaces in deep-RL for robotic manipulation. Compared to the action spaces, joint position, joint torque, joint velocity, and variable impedance control in joint space, the variable impedance control in end-effector space demonstrated superior sample efficiency, energy efficiency, safety, and sim-to-real transferability in learning for multiple robotic manipulation tasks. In [20] an RL framework for learning contact-rich manipulation tasks is proposed where an off-policy model-free RL algorithm (SAC) is used to learn the stiffness and position parameters of a parallel position-force controller. This approach could learn policies achieving high success rates in insertion tasks even under uncertainties. Ref. [21] extended it to real-world robotic manipulators for safely learning contact-rich manipulation tasks. A parallel position/force control and admittance control were evaluated in this framework on position-controlled robots. Similar to other approaches, in [30] the right choice of action space for learning contact-rich tasks in presence of uncertainties was investigated. A model-free RL approach is used to learn policies for direct torque control, fixed gain PD control and variable gain PD control. On comparing the three controllers, the variable gain PD controlled demonstrated superior performance and reliability. Similarly in [215], on comparing direct torque control, joint PD, inverse dynamics, and task-space impedance control, the impedance control showed superior performance compared to the other three. But all of these approaches have the drawback of model-free RL in terms of low sample efficiency and lack of task transferability.

All these approaches could learn complex VIC policies for specific tasks, however, at the expense of sample efficiency. Ref. [34] demonstrated model-free RL-based VILC using DMP policy and PI^2, which is sample efficient but it fails to scale to complex policies. In [99], PI^2 approach was used to learn torque control profiles for robot manipulators for compliant manipulation using desired position trajectories from kinesthetic demonstrations. But it is not suitable for force-based VIC, as unlike stiffness values the impedance parameters can not be estimated directly from kinesthetic demonstrations used in [110]. [110] demonstrated that augmenting position demonstrations with stiffness estimates and using it to learn a stiffness controller could provide superior manipulation performance compared to a position controller. Alternatively, MBRL approaches offer a sample efficient and scalable framework leveraging on a learned dynamical model. In [137] MBRL is used to learn position-based VIC on industrial robots using GP models. Ref. [10] used a similar approach for force-based VIC and hybrid force-motion control for contact-sensitive tasks. In [179], PETS approach is used to learn a position-based

VIC strategy for HRC tasks. Although this MBRL methods are sample efficient compared to model-free RL, it still demands a lot of interactions, making it difficult to apply to robotic manipulation.

7.4 Discussion

The presented survey underlines the increasing interest and the suitability of RL for VILC. This is still an early stage for this area of research and has many open questions to be addressed. The results from the existing research show that RL-based VILC methods can be very effective in learning complex compliant manipulation skills with minimal modeling effort. These results show that RL with VIC can be a tool to achieve human-like compliant manipulation skills in robotic manipulators, but this demands further research. Even though VIC provides a robust low-level control for a RL agent to identify the optimal parameters based on state-feedback, many challenges in RL are carried into RL-based VILC. Generally, RL learns task specific VILC skills and is difficult to transfer to another task or even to a different specification of the same task. This can be very important in real-world robotic tasks, especially in unstructured environments. It is important for robots to efficiently use the existing knowledge in new scenarios and be able to improve or generalize the skills to new scenarios rather than having to relearn for every new scenario. More research in transfer learning and generalization is key to further advancing RL-based VILC methods. Chapter 11 presents some new results in this direction. Most of the existing RL-based VILC approaches especially the model-free approaches are not sample efficient to use in a real system whereas the model-based approaches are not scalable because of the limitation of the model structure. GP are popularly used to model the dynamics in [137, 136, 10] in order to achieve sample efficient VILC. But GP has limited ability in modeling complex dynamics and is not well suited for highly noisy data. Additionally, GP models are not good at representing non-smooth dynamics such as contact dynamics or human-robot-interaction dynamics and pose challenges in computational effort when the dataset is large. While MBRL approaches could improve sample efficiency and can be useful in providing safety and stability guarantees, there is a need for further research in this direction facilitating complex model structure such as DNN to build scalable and sample efficient VILC approaches with theoretical guarantees.

If the RL agent could identify the optimal impedance parameters, we could achieve complex compliant manipulation skills with safety and stability guarantees. But this is not obvious, especially with model-free RL, and remains an open question with the area of RL. Guaranteeing safety, stability, and robustness for controllers in a complex robotic manipulator operating in uncertain environments is challenging. In the case of VIC, often passivity theory is used to provide theoretical

guarantees under relatively general working assumptions. However, this approach is model-based and not suitable in the case of model-free RL and the passivity property is lost if arbitrary variations of the impedance parameters are allowed. Passivity-based approaches are often concerned with the analysis of variable impedance profiles that already exist prior to task execution [128, 150, 149]. This is not suitable for guaranteeing the stability of state-dependent real-time impedance variations. In another recent approach, a modified impedance control strategy allows the reproduction of a variable stiffness while preserving the passivity, and therefore a stable behavior both in free motion and in interaction with partially known environments [59]. In Ref. [59], the goal is to modify the impedance control in order to allow stiffness variations while preserving passivity and, consequently, stable interactive behavior and asymptotic tracking in free motion. This tank-based strategy has been shown very well suited for VIC, in spite of some difficulties to tune its parameters. Nevertheless, it is dependent on the states of the system, measured during task execution and so can only be applied online. [97] propose an approach using a designed Lyapunov candidate function to stabilize the learned impedance system with an optimal input law in analytical form. But this requires solving an additional convex optimization problem which could be computationally very expensive so not feasible practically. Ref. [18] proposed an approach based on the combination of passivity conditions with an adaptation law on the impedance profile. This method allows for verifying whether a given profile is passive and if it is not, it provides a method to modify it in a way to guarantee passivity.

The safe RL approaches described in Chapter 5 are interesting to explore for RL-based VILC. A feasible approach in this direction could be to provide probabilistic safety guarantees using safety filters such as CBF and solving constrained optimization problems over the GP model [108]. Guaranteeing stability properties to the resulting VILC is challenging as guarantees have to be provided in real-time in an online fashion as the stiffness values predicted by the policy are state-dependent. The approach proposed in [97] by designing a quadratic Lyapunov candidate function could be coupled with GP models to provide probabilistic stability guarantees similar to safety guarantees in [108].

7.5 Conclusions

This chapter presented a survey of existing VILC methods that utilize RL. This survey serves as a background for the rest of the chapters in this part of the thesis. This chapter also presented a discussion on the major challenges in using RL for VILC and some future directions to address them. Some of these challenges are addressed in Chapter 10 and Chapter 11 in this thesis.

Chapter 8

Real-time Dynamic Movement Primitives for Moving Targets

This chapter is based on the following publication [11]:

Anand, A. S., Østvik, A., Grøtli, E. I., Vagia, M., and Gravdahl, J. T. (2021, December). Real-time temporal adaptation of dynamic movement primitives for moving targets. In 2021 20th International Conference on Advanced Robotics (ICAR) (pp. 261-268). IEEE.

8.1 Introduction

The problem of targeting and manipulating a moving object with a robotic arm is of great importance in numerous industrial applications. Up until now, researchers mainly focused on the problem of static manipulation. However, in order to achieve higher levels of dexterity in robotic manipulators, moving target scenarios need to be addressed. For example, grasping moving objects is challenging as opposed to picking up a stationary object, such as the adaption of the motion plan, efficient trajectory tracking, modeling and estimating the motion of the object, and so on. Learning from demonstration (LfD) has been widely deployed in many robotic manipulation tasks, but similarly, it mainly involves static targets. Among the various LfD methods that exist, the Dynamic Movement Primitive (DMP) framework [188, 90, 189, 88, 89] is predominantly used for motion planning in manipulation tasks in order to learn from human demonstrations [125, 194].

The DMP framework offers a simple way to learn a complex motion trajectory from a single human demonstration without the need for any complex modeling. DMP framework encodes an arbitrary motion pattern using a second-order non-

linear system consisting of a linear point attractor modulated by a learned non-linear forcing function. Additionally, the DMP system is capable of generalizing to different goal positions and task execution speeds using spatial and temporal scaling properties respectively. The DMP framework can be used for both point-to-point as well as rhythmic movements. The robustness to perturbations and collision avoidance capabilities can be incorporated into DMP [163, 172, 46], making it a highly useful framework for learning robotic manipulations skills. The DMP framework is further improved with the possibility of learning from multiple demonstrations [146, 147, 65]. To facilitate a singularity-free representation of orientation in Cartesian coordinates, DMP is represented using unit quaternions [213, 123].

There has been limited work conducted regarding DMP systems naturally adapting to a moving target. A bio-inspired formulation was developed for human-robot interaction by including a velocity feedback term into the DMP system in [171]. In [233], an interactive movement primitive is formulated to reach a moving target in a leader-follower configuration. Other approaches involve reaching a moving object by predicting the target trajectory in advance, as presented in [164, 115], based on a dynamic model of the moving target. However, in many robotic tasks, including human-robot interaction, it is difficult to model the movement of the target object and thereby generate an accurate predefined DMP. Such tasks, demand real-time adaptation of the DMP to continuously change the target/goal position and desired execution time. Consider the example of a human-to-robot object handover task, where the object's velocity depends on the human individual's movement. In such a scenario, the DMP system needs to slow down or speed up based on human behavior, while adapting to a moving target. Another example is a robotic grasping task where the target object is moving continuously, but the motion pattern of the object is unknown. In [111], an approach to achieve real-time control over the execution time by forward simulating the entire DMP execution at each time step was proposed. This approach is computationally expensive and thus less desirable for higher control frequencies.

In this chapter, we propose an extension of the standard DMP framework for adaptation to real-time changes in task execution time. We define *real-time control of the execution time* as, how the DMP system is adapting to real-time changes in its desired execution time during the task. In order to achieve this, we only manipulate the temporal scaling of DMP system while preserving its spatial properties. We formulate two methods to achieve efficient real-time temporal scaling, (i) a control law to vary the temporal scaling term of the standard exponential canonical system and (ii) an alternate polynomial-based canonical system with a suitable control law for temporal scaling. This is useful in a manipulation task with an un-

derlying DMP planner, where the task execution time needs to be changed during the execution phase. Additionally, in the case of moving targets, velocity feedback of the target and a simple estimate of the goal position based on the current target position and velocity are included in the DMP system. The proposed approach is tested on simulation and experimental setups with a moving object, with the use of a UR5 robotic manipulator.

The rest of the chapter is organized as follows. Section 8.2 describes the necessary background on DMP. Section 8.3 describes the proposed extention to the standard DMP framework briefly. Section 8.4 presents the evaluation of our approach on simulation and experimental setups using the UR5 robotic manipulator. Conclusions are presented in Section 8.5.

8.2 Dynamic Movement Primitives

DMP framework provides an elegant way to encode any arbitrary spatial trajectory as a stable second-order nonlinear system, which is well suited and widely utilized controlling for robotic systems [184]. The standard DMP system consists of a point attractor formulated as a second-order ordinary differential equation (ODE) with a nonlinear forcing term. In the DMP framework this is called the transformation system, each degree of freedom (DOF) in the operation space will be denoted by a separate transformation system,

$$\begin{aligned} \tau \dot{x} &= v \,, \\ \tau \dot{v} &= K(x_g - x) - Dv + (x_g - x_0)\, f(s) \,. \end{aligned} \tag{8.1}$$

Here, $x, v \in \mathbb{R}$ are the position and velocity of the system at time t respectively. The initial position and the goal/target are given by $x_0, x_g \in \mathbb{R}$ respectively. $K, D \in \mathbb{R}^+$ represents the stiffness and damping coefficient terms of the second-order system. f denotes the nonlinear forcing term, which is a function of a phase variable s. If $f = 0$, these equations represent a globally stable second-order linear system with $(x, v) = (x_g, 0)$ being a point attractor. The forcing term is modelled to match the system output with any arbitrary trajectory demonstrated, which is then generalised to different goals and initial conditions. $\tau \in \mathbb{R}^+$ is a temporal scaling term, which is normally set equal to the task execution time or the motion duration while modelling/learning the forcing term. The temporal evolution of the DMP system can be modulated by varying τ. For generalization in time, the explicit time dependency of f is avoided by parameterising time by a phase variable s guided by a first-order linear dynamical system, termed as the canonical system, thus making the system temporally scalable by varying τ,

$$\tau \dot{s} = -\alpha s \,. \tag{8.2}$$

The initial state of the canonical system is $s_0 = 1$ and $\alpha \in \mathbb{R}^+$ is the decay constant, where s decays exponentially from 1 to 0. The forcing term can be expressed as a function of the phase variable s using Gaussian kernel functions ψ_i with corresponding weights ω_i,

$$f(s) = \frac{\sum_{i=1}^{N} \psi_i(s)\omega_i}{\sum_{i=1}^{N} \psi_i(s)} s\,, \tag{8.3}$$

where,

$$\psi_i(s) = \exp\left(-h_i\left(s - c_i\right)^2\right)\,. \tag{8.4}$$

N denotes the number of Gaussian kernels used, which is a hyper-parameter decided based on the geometric complexity of the demonstration trajectory. The centres and widths of Gaussian kernels are given by c_i and h_i respectively for a demonstration trajectory of time duration, T,

$$c_i = \exp\left(-\alpha i \frac{T}{N}\right), \quad i = 1, 2, \ldots, N\,, \tag{8.5}$$

$$\begin{aligned} &h_i = \tfrac{1}{(c_{i+1} - c_i)^2}, \quad i = 1, 2, \ldots, N-1\,, \\ &h_N = h_{N-1}\,. \end{aligned} \tag{8.6}$$

The DMP system is easily translated to the multidimensional problem as a separate transformation system is learned along each dimension while having a single canonical system. The conventional transformation system representation in (8.1) has few drawbacks, such as the case when the goal position coincides or is too close to the initial position, i.e ($x_g = x_0$) or when ($x_g - x_0$) flips sign from the demonstrated trajectory. These problems are addressed in [163, 171, 164, 82] to formulate an improved version,

$$\begin{aligned} &\tau \dot{x} = v\,, \\ &\tau \dot{v} = K(x_g - x) - Dv - K\left(x_g - x_0\right) s + Kf(s)\,. \end{aligned} \tag{8.7}$$

The canonical system remains the same as (8.2). Unlike (8.1), generalization to different goal positions is possible in (8.7) as f is independent of the spatial scaling. This generalization is obtained by utilizing the invariance property of the transformation system in (8.7) and using the roto-dilation based transformation to find the transformation matrix [65]. Given a transformation matrix $\boldsymbol{S} \in \mathbb{R}^{n \times n}$, where n is number of DMPs or the dimension of trajectory, $X \in \mathbb{R}^n, V \in \mathbb{R}^n, X_0 \in \mathbb{R}^n, X_g \in \mathbb{R}^n, F \in \mathbb{R}^n$ respectively are the vector representation of

x, v, x_0, x_g, f, and $\boldsymbol{K} \in \mathbb{R}^{n\times n}, \boldsymbol{D} \in \mathbb{R}^{n\times n}$ are the matrix representation of K, D. The transformed states and variables are:

$$\begin{aligned} X' &= \boldsymbol{S}X, \quad V' = \boldsymbol{S}V, \qquad X'_0 = \boldsymbol{S}X_0, \qquad X'_g = \boldsymbol{S}X_g, \\ F' &= \boldsymbol{S}F, \quad \boldsymbol{K}' = \boldsymbol{S}K\boldsymbol{S}^{-1}, \quad \boldsymbol{D}' = \boldsymbol{S}D\boldsymbol{S}^{-1} \end{aligned} \tag{8.8}$$

The DMP formulation has some drawbacks for real-time control during execution. This becomes very pronounced if the target is non-stationary. In the standard DMP formulation, we can vary the execution time in real-time by modulating τ according to an adaptive law based on a complete forward simulation of the entire DMP, given by,

$$\begin{aligned} &\tau' = \lambda\tau\,, \\ &\lambda^{t_{i+1}} = \lambda^{t_i} + k_p\left(\hat{T}^{t_i} - T\right) - k_d\left(\hat{T}^{t_i} - \hat{T}^{t_{i-1}}\right)\,. \end{aligned} \tag{8.9}$$

where t_i is the time at the i^{th} control step, $\lambda^{t_{i+1}}$, is the temporal scaling factor at time t_i, $t_0 = 0$; $\lambda^{t_0} = 1$; k_p and k_d are the specified proportional and derivative gains. $\hat{T}^{t_i}$ is the total execution time (from start) as computed at time t_i. $\hat{T}^{t_i}$ is estimated by doing an entire forward simulation of the DMP at time t_i which is computationally inefficient for a real-time task. Also, sudden changes in τ could create unexpected behaviors in the DMP system and make it less robust in a moving target scenario.

8.3 Extension of the DMP Model

8.3.1 Real-time Control of the DMP Execution Time

We propose an extension of the standard DMP framework to provide real-time control over the execution time T, while preserving the shape properties of the DMP system for tasks involving moving targets. Although we consider the moving target scenarios here, our approach is equally applicable to a stationary target scenario. In the standard DMP formulation, the execution time can be varied in real-time by varying τ according to a suitable adaptive law. However, such an adaptive law can not guarantee an exact final execution time. This error in the final execution could be higher when T is varied during the later stages of DMP execution. An adaptive law is formulated for τ based on the value of the phase variable, s from the canonical system (8.2) and the fraction of DMP executed at s in each DOF in comparison to the demonstration trajectory. The dynamics of the temporal scaling term τ of the canonical system is represented using a first-order linear system modulated by the input u,

$$\dot{\tau} = -k_\tau(\tau - u)\,. \tag{8.10}$$

Here k_τ is a positive constant. An adaptive law is defined for control input u, such that the decay rate of the canonical system in (8.2) is slowed down if the desired

execution time is increased or sped up if the desired execution time is reduced. The value of u is derived at each time step in order to have continuous control over the execution time. As the nonlinear forcing term f is a function of the phase variable s, the shape of the trajectory can be preserved while the execution time of the task is changed in real-time. The rate of change of s is always negative as it is a monotonically decreasing function. The control law for u is given by

$$u = \begin{cases} \frac{\hat{s}_t}{s_t} \frac{\|\overline{x}\|}{\|\overline{x}_d\|} \hat{\tau}_t & \text{if } s_t > \delta \\ \hat{\tau}_t & \text{otherwise .} \end{cases} \tag{8.11}$$

δ is chosen to be a very small value close to 0, to guarantee finite value for $u \,\forall\, t$. $\|\overline{x}_d\|$ denotes the norm of the fraction of DMP executed in the demonstration trajectory for a specific value of phase variable at time t denoted by s_t. Similarly $\|\overline{x}\|$ denotes the norm of the fraction of the current DMP corresponding to s_t. $\hat{s}_t$ and $\hat{\tau}_t$ are the desired value of s and τ at time t, which is described later. The mathematical expressions are given by,

$$\overline{x}_d(s) = \frac{g_d - x_d(s)}{g_d - x_d^0}\,, \quad \overline{x}(s) = \frac{\hat{x}_g - x(s)}{\hat{x}_g - x^0}\,. \tag{8.12}$$

$\overline{x}_d,\ \overline{x} \in [0, 1]$, where $\overline{x}_d$ is derived from the transformed demonstration trajectory used for learning the DMP, whereas $\overline{x}(s)$ is calculated in real-time based on the value of s_t. In (8.12), it is assumed that the initial pose of the robot is different from the goal trajectory and needs to reach the goal point only once during a trajectory execution to avoid the division-by-zero condition. This is valid for general robotic manipulation tasks where the initial pose of the robot is different from the object's trajectory in the workspace and the robot needs to approach the object. In scenarios where the starting and goal positions coincide, rhythmic DMPs should be used instead of discrete DMP, which is not considered in this chapter.

Furthermore, $\hat{s}_t$ is the desired value of the phase variable from the canonical system given the value τ, scaled linearly in the desired execution time $\hat{T}_t$ at time t,

$$\hat{\tau}_t \dot{\hat{s}}_t = -\alpha \hat{s}_t\,, \tag{8.13}$$

where,

$$\hat{\tau}_t = \tau_0 \frac{\hat{T}_t}{T_0}\,. \tag{8.14}$$

Here τ_0 and T_0 are the initial value of τ and the initial value of the task execution time respectively.

A similar approach of defining first-order dynamics on τ is provided in recent work [124]. Here, τ is adapted solely to maintain the demonstrated velocity levels

in case of DMP execution for a task involving a moving goal. However, our work considers the aspect of real-time control of task execution time, which is more applicable in practical scenarios with strict timing requirements.

8.3.2 Polynomial Canonical System

The standard canonical system (8.2) in the DMP formulation can not guarantee complete control over the execution time with an adaptive law (8.10) because of its exponentially decaying nature. As the canonical system decays exponentially in time, the value of s approaches zero rapidly, with only very small changes in s in the later phases. This behavior of the canonical system makes it impossible to have control over the time scaling by varying τ. For larger values of s during the early phase of DMP execution, a better level of real-time control over the task execution time is feasible. In order to improve this drawback of the DMP framework, we propose to use an alternative canonical system that could provide a higher degree of control over the execution time in later phases of the DMP. A polynomial function that decays slowly at the beginning and then rapidly converges to 0 at $t = T$, or any similar functions could offer such property. Considering this benefit, we choose a polynomial function that can be represented using an inverse gradient formulation of the standard canonical system in (8.2),

$$\tau \dot{s} = \begin{cases} -\alpha^{-1} s^{-1} & \text{if } s > \delta \\ -\alpha' s & \text{otherwise} \end{cases} . \tag{8.15}$$

To have s monotonically decreasing in the interval $[1, 0]$ and asymptotically converging to 0, the final stage of DMP where $s \leq \delta$ is represented using the standard canonical system (8.2). δ is a small positive constant close to 0. The value of τ for $s > \delta$ at any time instant t can be derived from the solution of the system in (8.15) based on the condition, $t = \hat{T} \implies s = 0$ assuming the polynomial function for s,

$$\tau = \frac{2\alpha(\hat{T} - t)}{s_t^2} . \tag{8.16}$$

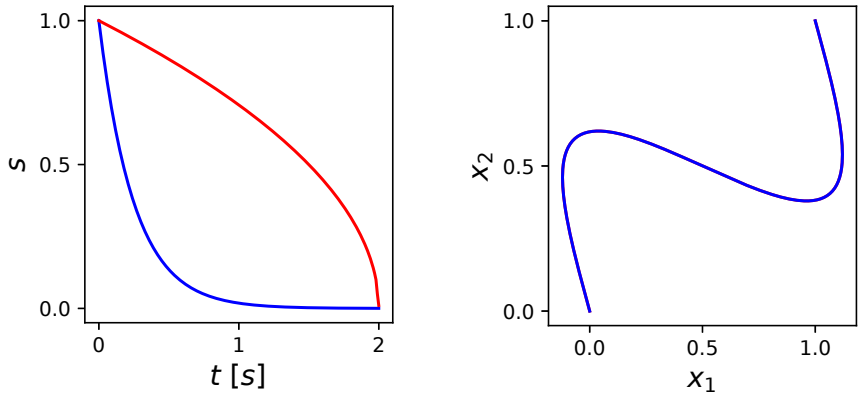
Here the value of τ needs to be updated only when there is a change in $\hat{T}$. From (8.15) and (8.16), $\tau > 0 \ \forall \ t$ as $t < \hat{T}$ when $s > \delta$. Additionally, the polynomial canonical system in (8.15) is continuous, which can be made smooth by choosing $\alpha' = 1/(\delta^2 \alpha)$. An analytical solution for the polynomial canonical system when $s > \delta$ is derived from (8.15), (8.16) as,

$$s = \left(\frac{T - t}{T} \right)^{\frac{1}{2\alpha^2}} . \tag{8.17}$$

The value of α decides the nature of this polynomial canonical system, where $\alpha = 1$ defines a parabola. α is chosen as $\alpha \geq 1$, to make better use of the higher-order polynomial behavior motivating this approach. For $s \leq \delta$ the value of τ could be found using (8.10). The difference between systems (8.2) and (8.15) is shown in figure 8.1.a. The larger variation in s during the later phases of execution provides the option to have better control over the execution time. The intersection time t_{int} of the polynomial canonical system in (8.17) and the exponential canonical system from (8.2) (assuming $\tau/\alpha' = T/4$ for (8.2) for the sake of comparison) can be found to be,

$$t_{int} = T + \frac{TW\left(-8e^{-8\alpha^2}\alpha^2\right)}{8\alpha^2}, \tag{8.18}$$

where t_{int} is the time at which both the systems in figure 8.1.a intersects and W is the Lambert W-function [113]. As $\alpha \geq 1 \implies t_{int} \longrightarrow \hat{T}$ it follows that the polynomial canonical system maintains a higher value of s until s is close to 0.



(a) (b)

Figure 8.1: (a) Evolution of the canonical systems, the exponential canonical system is shown in the blue and polynomial canonical system in red. (b) The behavior of learned DMP systems corresponding to both canonical systems, overlap as the same demonstration trajectory is used to learn both systems.

8.3.3 Stability Guarantee

For the adaptive law discussed in section 8.3.1, from (8.10) and (8.11) it can be observed that $\tau > 0 \ \forall \ t$, which guarantees the asymptotic convergence of the exponential canonical system (8.2) to 0. From (8.17), the polynomial canonical system monotonically decreases to 0 as $t \to \hat{T}$ with $t = \hat{T} \implies s = 0$. Therefore, $\exists\, \Delta > 0$, such that, $(\hat{T} - t) \to \Delta \implies s \to \delta > 0$. The existence of δ guarantees

the switching of the polynomial canonical system to the exponential canonical system (8.15), which in turn guarantees asymptotic convergence of s to 0. The existence of $\delta > 0$ also guarantees a finite value of τ at switching (8.16) and τ is bounded $\forall\, t$.

In order to analyze the stability of the transformation system, the contraction analysis method is used [140, 222]. The stability analysis presented here is very similar to the one presented in [124]. Consider the transformation system (8.7) with $e = x - x_g$,

$$\begin{bmatrix} \dot{e} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau} v \\ \frac{1}{\tau}(-Ke - Dv - K(x_g - x_0)s + Kf(s)) \end{bmatrix}. \quad (8.19)$$

As s asymptotically converges to 0, for the fixed point of $s = 0$ with a finite value of $\tau = \tau_s$ and $f(s) = 0$, (8.19) can be written as,

$$\begin{bmatrix} \dot{e} \\ \dot{v} \end{bmatrix} = \frac{1}{\tau_s} \begin{bmatrix} 0 & 1 \\ -K & -D \end{bmatrix} \begin{bmatrix} e \\ v \end{bmatrix}. \quad (8.20)$$

This is a linear time-varying system with a bounded time-dependent parameter, τ_s.

Theorem 4.6 from [143]: Consider a linear time varying system of the form $\dot{x} = \boldsymbol{A}(t)x$ where $x \in \mathbb{R}_n$ and $\boldsymbol{A}(t) \in \mathbb{R}_{n\times n}$. The equilibrium state $x = 0$ is exponentially stable if and only if for any given symmetric, positive definite, continuous, and bounded matrix $\boldsymbol{Q}(t)$, there exists a symmetric, positive definite, continuously differentiable and bounded matrix $\boldsymbol{P}(t)$ such that $-\boldsymbol{Q}(t) = \boldsymbol{P}(t)\boldsymbol{A}(t) + \boldsymbol{A}^T(t)\boldsymbol{P}(t) + \dot{\boldsymbol{P}}(t)$.

A similar approach is provided in Example 4.21 of [102] using Theorem 4.10 [102]. Choosing the matrix,

$$\boldsymbol{P} = \frac{1}{2} \begin{bmatrix} \frac{K}{D} + \frac{1}{D} + \frac{D}{K} & \frac{1}{K} \\ \frac{1}{K} & \frac{1}{D} + \frac{1}{KD} \end{bmatrix},$$

which is constant, bounded and positive definite, gives $\boldsymbol{Q}(t) = \frac{1}{\tau_s}\boldsymbol{I}$. Since τ_s is bounded, $\boldsymbol{Q}$ is also bounded $\forall\, t$. Based on Theorem 1 in [222], the entire DMP system is globally contracting. Therefore the DMP system with the proposed adaptive laws and polynomial canonical system asymptotically converges to a unique equilibrium point with $s = 0$, $e = 0$, $v = 0$.

8.3.4 Moving Target DMP with Velocity Feedback

In order to better preserve the shape properties of DMP while following a moving target, a velocity feedback of the moving target is incorporated into the DMP formulation similar to [171].

$$\begin{aligned} \tau\dot{x} &= v\,, \\ \tau\dot{v} &= K(\hat{x}_g - x) - D(v - \dot{x}_g) - K\,(\hat{x}_g - x_0)\,s + f(s)\,. \end{aligned} \tag{8.21}$$

Here $\hat{x}_g$ is an estimate of the final goal position. A fair assumption is that the goal is moving slowly enough such that $v \geq \dot{x}_g$ the convergence properties holds for the system in (8.21). The estimate of the final goal position at time t is updated with a simple weighted average of the current goal position and the position estimated using the goal velocity at time t,

$$\hat{x}_g(t) = x_g(t) + \frac{\dot{x}_g(t)(\hat{T}_t - t)t}{\hat{T}_t}\,. \tag{8.22}$$

At any time instant t, $x_g(t)$ is the measured current goal position, $\dot{x}_g(t)$ is the velocity of the goal trajectory, and $\hat{T}_t$ is the desired time for the entire execution of the DMP.

8.4 Evaluation

8.4.1 Simulations

Two separate sets of simulations were conducted to evaluate the performance of our approach. In the first simulation setup, a 2 DoF DMP system was set up with a moving target. The control law for τ, for both the approaches, is evaluated based on how accurately we can control the execution time of the DMP system. For both approaches, a single demonstration trajectory of a rotated sinusoidal pattern with $x_0 = (0, 0)$ and $x_g = (1, 1)$ is used. The execution time of the demonstrated trajectory is set as 2s. The sampling time for all the simulations is $dt = 0.01$s, the position tolerance for DMP convergence to the goal/target point is $\mu = 0.01$, i.e $\|x_g - x\| < \mu\,\|x_g - x_0\|$. The other system parameters are $K = 1000, \alpha = 4, N = 100, k_\tau = 1$, $D = 2\sqrt{K}$ and $\delta = 0.01$. For the purpose of analysis, the initial pose is always kept constant at the origin (0,0).

The system is analyzed for the following four scenarios in simulation, regarding its adaptation to real-time changes in the desired execution time $\hat{T}$ as shown in figure 8.2 and summarised in the following,

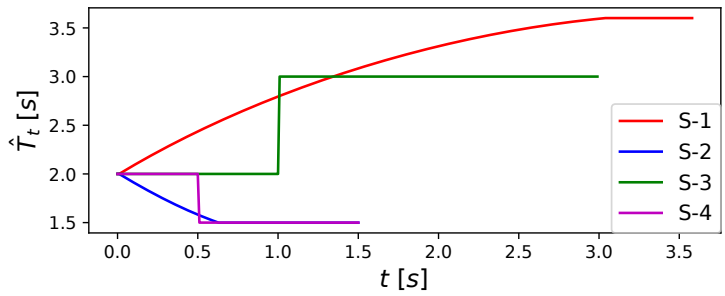


Figure 8.2: The change in desired execution time $\hat{T}_t$ for (S-1 to S-4).

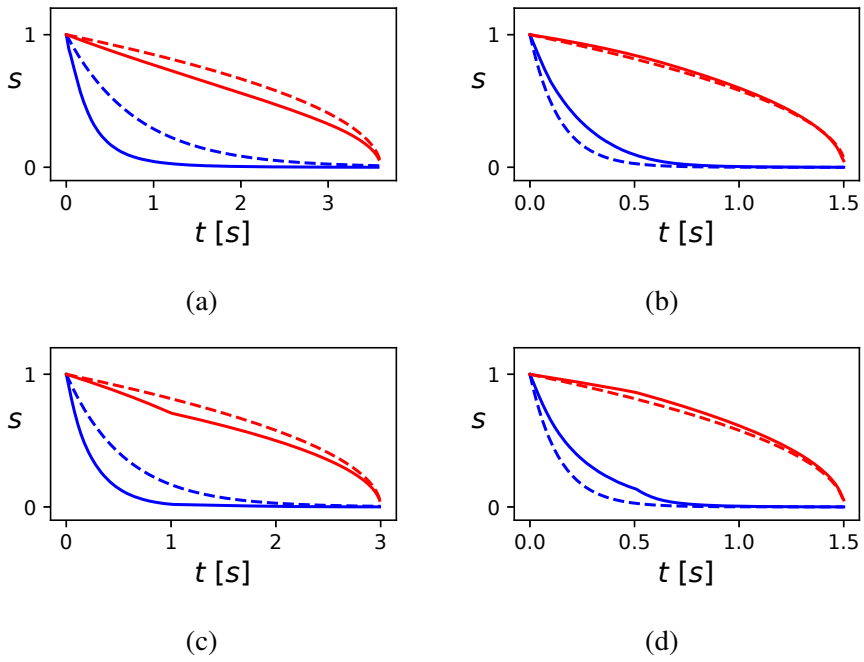


Figure 8.3: (a) - (d) represent the behavior of the canonical systems for S-1 to S-4 respectively. The dashed lines represent the ideal DMP profile when the final desired execution time T is known at $t = 0$. The solid lines represent the real-time behavior of the canonical system based on the control law u given only the current $\hat{T}$. The red and blue lines correspond to the standard exponential and the proposed polynomial canonical systems respectively.

- S-1: $\hat{T}_t$ gradually increases from 2s to 3.6s.

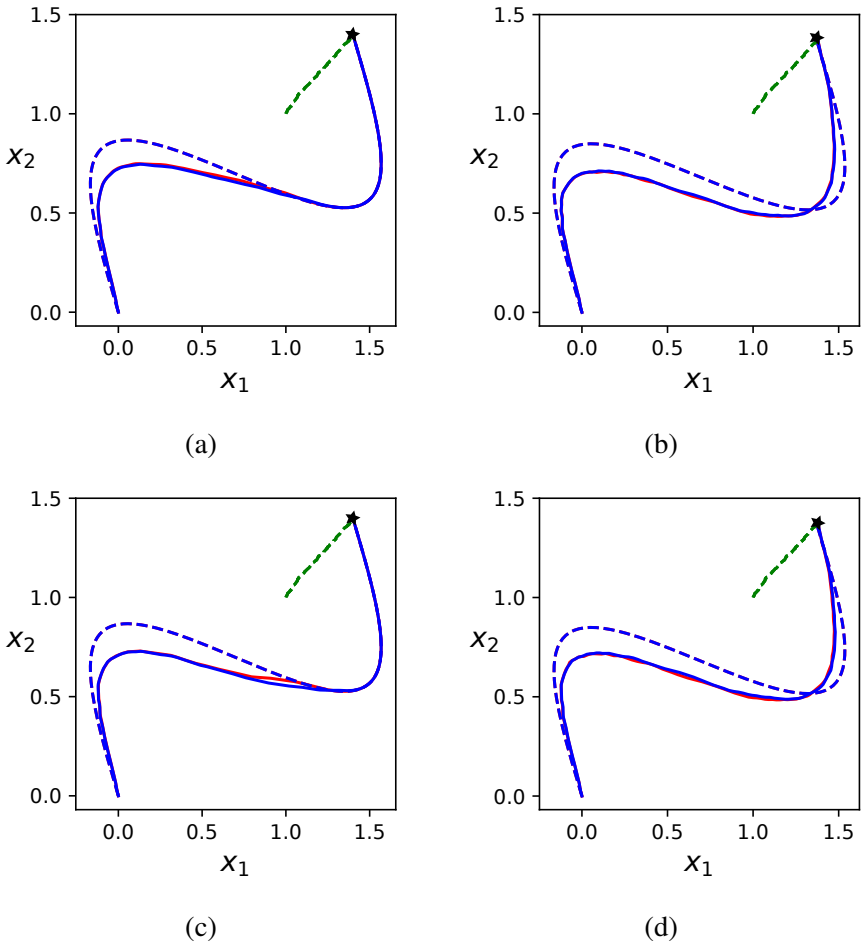


Figure 8.4: (a) - (d) represent the trajectory generated by their corresponding transformation systems for S-1 to S-4 respectively. The dashed lines represent the ideal DMP profile when the final desired execution time T and the exact end position of goal $x_g(T)$ are known at $t = 0$. The solid lines represent the real-time behavior of DMP based on the adapted canonical system and the green dotted lines represent the trajectory of the moving target with a star shape denoting its final position. The red and blue lines correspond to the standard exponential and the proposed polynomial canonical systems respectively.

- S-2: $\hat{T}_t$ gradually decreases from 2s to 1.5s.
- S-3: at $t = 1$s $\hat{T}_t$ switches from 2s to 3s.
- S-4: at $t = 1$s $\hat{T}_t$ switches from 2s to 1.5s.

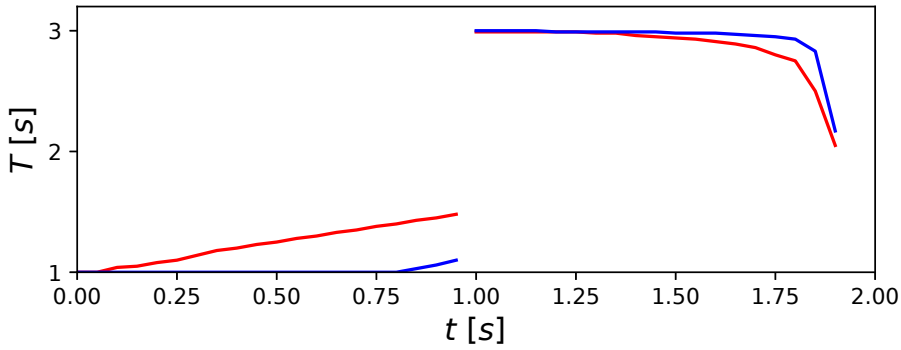


Figure 8.5: Performance comparison of DMP system with the two canonical systems, the standard exponential system (in red) and the proposed polynomial canonical system (in blue). The lines denote the final execution time T for the DMP system when perturbed with a step change $\hat{T}$ from 2s to 1s and from 2s to 3s evaluated every 0.05 seconds for $t \in [0, 1)$s and $t \in [1, 2)$s respectively.

All four scenarios are simulated with a moving target with a random positive velocity $\dot{x}_g \in [0, 0.5]$ at every time instant along both the DOFs. The goal position is updated in real-time based on a simple estimate from equation (8.22). The DMP system is simulated with velocity feedback from the moving goal based on the transformation system in (8.21). For all the scenarios, the real-time behavior of DMP systems with both exponential and polynomial canonical systems described by (8.2) and (8.15) respectively are shown in figure 8.4. The behavior of their corresponding adapted canonical systems is shown in figure 8.3.

In all four scenarios, the DMP trajectories converge smoothly to a moving goal as shown in figure 8.4. The trajectories generated by both DMP systems with standard exponential and the proposed polynomial canonical systems in figure 8.4 are very similar in their shape characteristic. The adaptation to real-time changes in $\hat{T}$ can be seen in figure 8.3 from the evolution of canonical systems. For gradual changes in $\hat{T}$ as shown in figures 8.3.a and 8.3.b, both the exponential and polynomial canonical systems adapt smoothly with the desired values of τ and converge to the moving goal exactly at the final desired execution time $T = 3.6$s and $T = 1.5$s for S-1 and S-2 respectively. In response to a negative step change in $\hat{T}$ in S-3 as shown in figure 8.3.c , the exponential canonical system converges to the moving goal $t = 2.98$s with an error of 0.02s, whereas the proposed polynomial based canonical system converges exactly at the final desired execution time $T = 3$s. But in S-4 in response to a negative step change in $\hat{T}$ as shown in figure 8.3.d, both

canonical systems adapts with the desired values of τ and converges to the moving goal exactly at final desired execution time, $T = 1.5$s.

However, these simulations do not reflect the effect of larger changes in the execution time during the later phases of execution. As the difference is expected to become more pronounced when changes are made on $\hat{T}$ towards the later stages of trajectory execution, a second set of simulations is conducted to evaluate the real-time performance of the approach during the DMP execution. Two scenarios of step changes in $\hat{T}$ are considered in the simulation with a stationary target, $x_g = (1, 1)$, all other system parameters remain unchanged.

Negative step change in T from 2s to 1s: This step change in T is simulated with a time step of 0.05s for $t \in [0, 1)$. The resulting performance and the corresponding error are shown in figure 8.5 for $t \in [0, 1]$. The proposed polynomial canonical system outperforms the standard exponential canonical system, as seen by the increased difference towards the later phases of DMP execution as shown in figure 8.5. Take $t = 0.75$s, on changing T from 2s to 1s the adaptive version of the standard exponential canonical system takes 1.38s to finish execution which is a 0.38s delay over the desired time. But with the proposed polynomial canonical system the DMP system converges to the goal at exactly 1s.

Positive step change in T from 3s to 2s: This step change in T is simulated at every 0.05s for $t \in [1, 2)$: The resulting performance and the corresponding error are shown in figure 8.5 for $t \in [1, 2]$. Here the proposed polynomial canonical system outperforms the standard exponential canonical system. Again, this difference increases towards the later phases of DMP execution as shown in 8.5. Take $t = 1.45$s, on changing T from 2s to 3s the adaptive version of the standard exponential canonical system takes 2.95s to finish execution which is 0.05s earlier that the desired time. But with the proposed polynomial canonical system the DMP system converges to the goal at 2.99s. At $t = 1.75$s, this is 2.8s and 2.95s respectively.

8.4.2 Experiments

To validate the performance of the proposed canonical system, experiments were conducted on a UR5 robot manipulator for the task of approaching a moving object. The 3D-cartesian position of the object is tracked in real-time using a Polaris Vicra optical tracking system. A PID trajectory tracking controller is utilized to track the trajectories generated by the DMP. In the experiments, the proposed polynomial canonical system is evaluated for its performance on reaching a moving object within a specified time which is varied during the DMP execution similar to the simulations conducted. The experiment consists of an object moved around by a human operator and the UR5 robot manipulator reaching it from a

fixed starting pose. The object is moved around slowly in order to keep the robot's velocity within safe limits. An open-source robot control framework developed for UR robots is used for controlling and communicating with the UR5 robot [160]. A demonstrated robot trajectory is collected using the optical tracking system with a static goal as shown in figure 8.7. The time duration of the demonstration trajectory is $T_d = 10$s. The sampling time for all the experiments is 60Hz, the position tolerance for DMP convergence in Cartesian space is defined to be $\mu = 0.01$, other system parameters are $K = 800, \alpha = 1/2T_d, N = 100, k_\tau = 1$, $D = 2\sqrt{K}$ and $\delta = 0.01$. Four sets of experiments (E-1 to E-4) were conducted similar to the simulations (figure 8.8),

- E-1: $\hat{T}_t$ gradually increases from 35s to 50s.
- E-2: $\hat{T}_t$ gradually decreases from 50s to 35s.
- E-3: at $t = 15$s, $\hat{T}_t$ switches from 25s to 40s.
- E-4: at $t = 20$s, $\hat{T}_t$ switches from 50s to 40s.

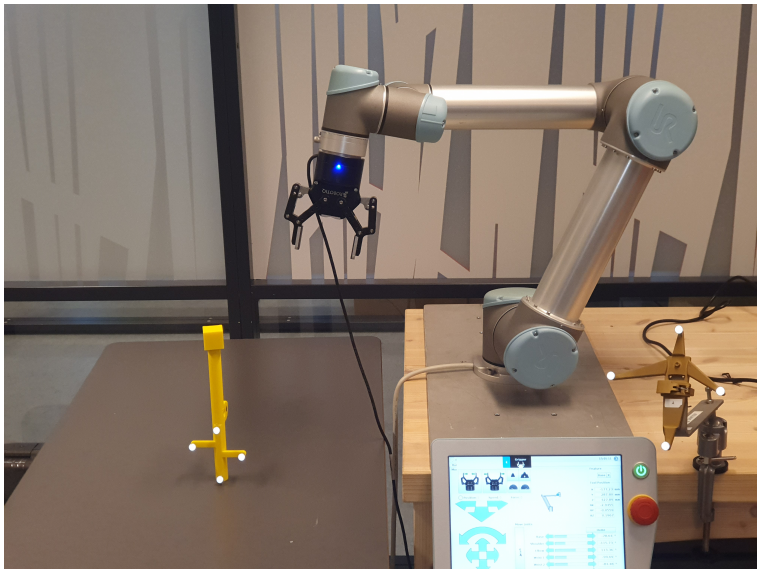
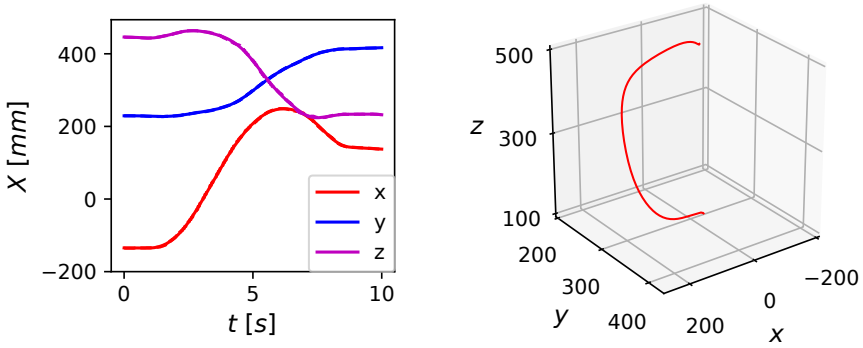


Figure 8.6: The experimental setup with UR5 robot and an object fitted with motion capture markers, which is moved around manually by hand.

The moving object scenario is created by moving the target by hand approximately within a Cartesian space of 1.0m, 0.3m, and 0.3m along x, y, and z directions



(a) (b)

Figure 8.7: Demonstration trajectory used for learning the DMP. In (a) time evolution of the Cartesian position (x, y, z) is shown, and in (b) the corresponding 3D trajectory in the Cartesian space is shown.

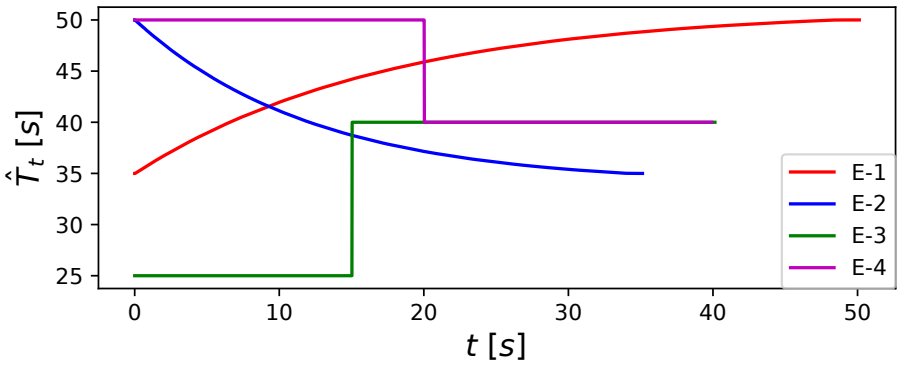


Figure 8.8: The change in desired execution time $\hat{T}_t$ for E-1 to E-4

respectively. The movement of the object is tracked at approximately 60 Hz using the optical tracking system. The corresponding velocity and estimate of the goal position are fed back to the DMP system. Figure 8.9 and 8.10 show the generated DMP trajectories and the behavior of the polynomial canonical system respectively for E-1 to E-4. We found the experimental results to be very consistent with the simulation results. The experimental results validate the usefulness of the proposed polynomial canonical system in adapting the DMP to the changes in desired task

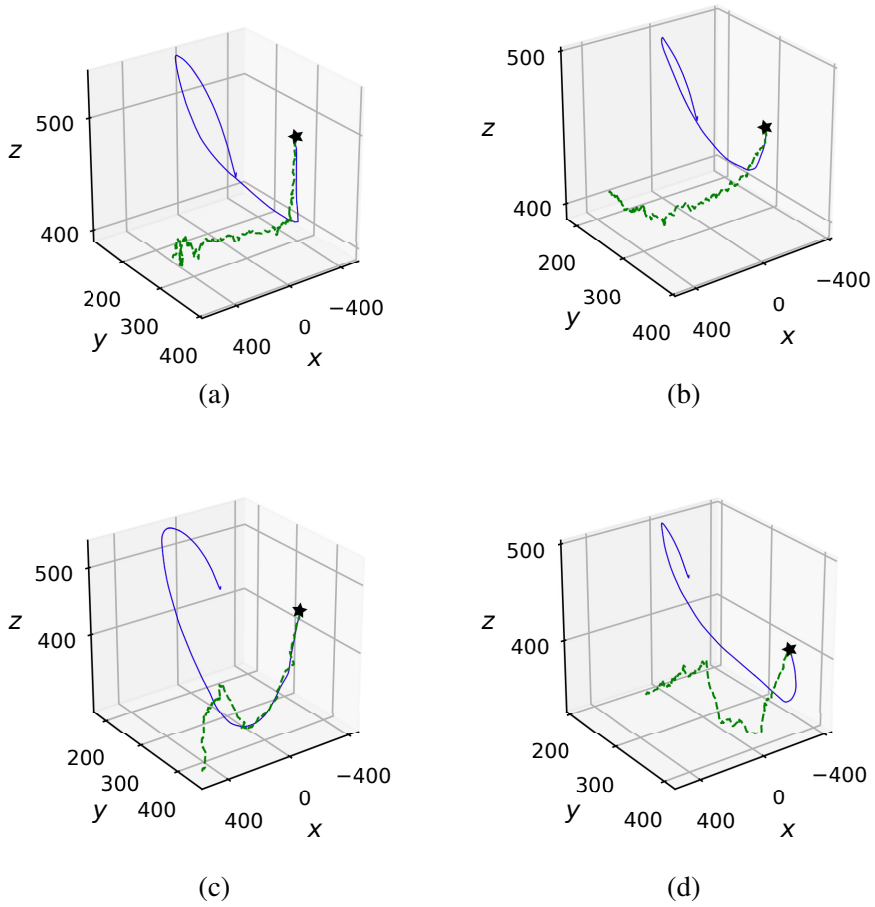


Figure 8.9: (a) - (d) represents the 3D Cartesian trajectory generated by the DMP in red and trajectory followed by the UR5 robot in blue for E-1 to E-4 (the blue and red trajectories approximately overlap). The trajectory of the target object is shown in green with a star shape denoting its final position. The target trajectory is noisy as the target object is moved around by hand.

execution time. In all four trials, the difference between the desired execution time and final execution time was within ± 0.1s which is acceptable compared to the total execution time of the task.

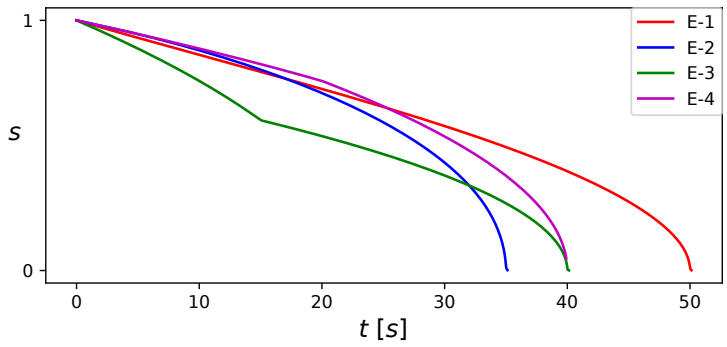


Figure 8.10: Behaviour of the polynomial canonical system for E-1 to E-4.

8.5 Conclusions

In this study, we extended the DMP framework with real-time control over the execution time while preserving the shape characteristics. We used a DMP formulation focused on a moving target scenario, incorporating the target's velocity feedback and a simple estimation of the target's final position based on its current position and velocity. We formulated an adaptive law for the standard exponential canonical system and an alternative polynomial canonical system to have better real-time control of the task execution time even during the later phases of DMP execution. We compared the proposed polynomial canonical system with the standard exponential canonical system for their relative performance. On comparing both canonical systems, the proposed polynomial canonical system was found to perform better in all the simulations conducted. The proposed polynomial canonical system was evaluated in an experiment using a UR5 robotic manipulator with a moving target. The extended DMP framework is useful in various robotic tasks with strict task execution time requirements when the desired task execution time is unknown prior to performing the task itself. This requirement is very relevant to robotic tasks involving moving targets such as industrial handling of moving parts, and human-robot collaborative tasks.

Chapter 9

Evaluation of Compliant Controllers for Learning Force Tracking Skills

This chapter is based on the following publication [10]:

Anand, A. S., Myrestrand, M. H., and Gravdahl, J. T. (2022, January). Evaluation of Variable Impedance-and Hybrid Force/MotionControllers for Learning Force Tracking Skills. In 2022 IEEE/SICE International Symposium on System Integration (SII) (pp. 83-89). IEEE.

9.1 Introduction

Compliant behavior for robots can be achieved by means of passive mechanical compliance built into the manipulator, or by active compliance control implemented in the servo control loop, for example, admittance control [190]. Compliant behavior can also be achieved using direct force control, where Variable Impedance Control (VIC) [91] and Hybrid Force-Motion Controller (HFMC) [174] are two prominent approaches. While VIC and HFMC are robust interaction control strategies, their performance depends heavily on setting the right parameters according to the environment's compliance properties [216]. While MBRL offers an ideal framework to learn the parameters of these compliant control, it is necessary to evaluate and compare them in a state-of-the-art MBRL framework. Among the MBRL approaches, the Probabilistic Inference for Learning Control (PILCO) algorithm is considered to be state-of-the-art for sample efficiency [53]. Even though PILCO is not directly scalable to high dimensional problems and complex

dynamical models, it offers a framework to develop and evaluate learning-based controllers with minimal robot-environment interactions. PILCO style approach is adapted to cases where a simplified prior model is available [187].

The majority of the learning-based methods in robotic interaction control have focused on using VIC to solve specific tasks, with a major focus on robotic assembly and human-robot interaction tasks. This chapter focuses on the implementation and evaluation of two prominent force control approaches (VIC and HFMC) in a model-based learning framework for an interaction task demanding force and motion tracking. The PILCO algorithm is chosen for evaluating the controller considering its high sample efficiency which facilitates learning directly in the experimental set-up in a handful of trails. The force controllers are implemented as Open-AI gym environments to seamlessly integrate with various learning frameworks.

The rest of this chapter is organized as: Section 9.2 and 9.3 present the necessary background on force tracking VIC and HFMC respectively. Section 9.4 presents force control as a learning problem. Section 9.5 presents the evaluation of the controllers in the learning framework. Section 9.6 presents a discussion of the findings and Section 9.7 concludes the work.

9.2 Force Tracking Variable Impedance Control

Consider a VIC with position and force errors defined by $\mathbf{E}_1 = \mathbf{x} - \mathbf{x}_d$ and $\mathbf{E}_f = \mathbf{f_{ext}} - \mathbf{f}_d$ respectively, where $\mathbf{f_{ext}}$ is the sensed external force and $\mathbf{f}_d$ is the desired force. The $\mathbf{M}$, $\mathbf{K}$ and $\mathbf{D}$ are adjusted using the adaptive law proposed in [86],

$$\dot{\boldsymbol{\beta}} = -(\mathbf{E}_f^T \mathbf{P} \mathbf{K}_v^{-1} \boldsymbol{\xi} \boldsymbol{\Gamma}^{-1})^T = -\boldsymbol{\Gamma}^{-1} \boldsymbol{\xi}^T \mathbf{K}_v^{-1} \mathbf{P} \mathbf{E}_f \,. \tag{9.1}$$

Where $\mathbf{K}_v \in \mathbb{R}^{n \times n}$ is the gain matrix,

$$\boldsymbol{\xi} = \boldsymbol{\xi}(\mathbf{E}_1, \dot{\mathbf{E}}_1, \ddot{\mathbf{E}}_1) \,, \tag{9.2}$$

is a $n \times 3n$ matrix and

$$\boldsymbol{\beta} = \boldsymbol{\beta}(\Delta\mathbf{M}, \Delta\mathbf{D}, \Delta\mathbf{K}) \tag{9.3}$$

is a $3n \times 1$ vector. The corresponding updates are,

$$\begin{aligned} \mathbf{M} &\rightarrow \mathbf{M} + \Delta\mathbf{M} \,, \\ \mathbf{D} &\rightarrow \mathbf{D} + \Delta\mathbf{D} \,, \\ \mathbf{K} &\rightarrow \mathbf{K} + \Delta\mathbf{K} \,. \end{aligned} \tag{9.4}$$

If the desired contact force $\mathbf{f}_d$ is large and the position error $\mathbf{E}_1$ is small, the adaptive law will adjust $\mathbf{M}$, $\mathbf{D}$ and $\mathbf{K}$ until $\mathbf{f}^* \rightarrow \mathbf{f}_d$, potentially causing instability issues. Hence, upper bounds should be set for $\mathbf{M}$, $\mathbf{D}$ and $\mathbf{K}$, avoiding instability at the expense of force tracking ability [86].

9.3 Hybrid Force-Motion Control

First proposed in [174], HFMC aims to achieve both motion and force control by dividing the task into two separate, decoupled sub-problems [216]. By specifying which sub-spaces should be controlled by a motion- and force controller respectively, the hybrid control intends to simultaneously solve the two separate control tasks. The selection matrices $\mathbf{S}_v$ and $\mathbf{S}_f$ is used to specify these subspaces. In the case of performing force control along the z-axis, and motion control in the remaining five dimensions, given by,

$$\begin{aligned} \mathbf{S}_v &= diag\begin{pmatrix}1 & 1 & 0 & 1 & 1 & 1\end{pmatrix}, \\ \mathbf{S}_f &= \begin{pmatrix}0 & 0 & 1 & 0 & 0 & 0\end{pmatrix}^T. \end{aligned} \tag{9.5}$$

When dealing with a compliant environment, the end-effector twist caused by environmental deformation in the presence of a wrench is given by [216],

$$\mathbf{v}_e = \mathbf{S}_v\mathbf{v} + (\mathbf{I} - \mathbf{P}_v)\mathbf{C}\mathbf{S}_f\dot{\boldsymbol{\lambda}}\,, \tag{9.6}$$

where $\mathbf{C} = \mathbf{K}^{-1} \in \mathbb{R}^{6\times6}$ represents the compliance matrix between the end-effector and the environment and $\boldsymbol{\lambda}$ is the force multiplier. $\mathbf{P}_v$ is a projection matrix that filters out all the end-effector twists that are not in the range space of $\mathbf{S}_v$. $\mathbf{I} - \mathbf{P}_v$ has the opposite effect of filtering out the twists that are in the range space of $\mathbf{S}_v$. $\mathbf{P}_v$ is calculated as $\mathbf{P}_v = \mathbf{S}_v\mathbf{S}_v^\dagger$, where $\mathbf{S}_v^\dagger$ is a suitable weighted pseudoinverse of $\mathbf{S}_v$,

$$\mathbf{S}_v^\dagger = (\mathbf{S}_v^T\mathbf{W}\mathbf{S}_v)^{-1}\mathbf{S}_v^T\mathbf{W}\,. \tag{9.7}$$

Setting $\mathbf{W}$ equal to the inertia matrix $\mathbf{H} \in \mathbb{R}^{6\times6}$ corresponds to defining a norm in the space of twists based on the kinetic energy [216]. Assuming, $\mathbf{S}_v$ and compliance,

$$\mathbf{C}' = (\mathbf{I} - \mathbf{P}_v)\mathbf{C} \tag{9.8}$$

to be a constant, (9.6) leads to the following decomposition of acceleration

$$\dot{\mathbf{v}}_e = \mathbf{S}_v\dot{\mathbf{v}} + \mathbf{C}'\mathbf{S}_f\ddot{\boldsymbol{\lambda}}\,. \tag{9.9}$$

Casting the control law (6.4) into the dynamic model (6.1) results in, $\dot{\mathbf{v}}_e = \boldsymbol{\alpha}$, $\boldsymbol{\alpha}$ is the control input denoting the acceleration with respect to Σ. By choosing,

$$\boldsymbol{\alpha} = \mathbf{S}_v\boldsymbol{\alpha}_v + \mathbf{C}'\mathbf{S}_f\mathbf{f}_\lambda\,, \tag{9.10}$$

allows decoupling of the respective controllers, $\boldsymbol{\alpha}_v$ relating to motion control and $\mathbf{f}_\lambda$ to force control. By choosing

$$\boldsymbol{\alpha}_v = \ddot{\mathbf{r}}_d(t) + \mathbf{D}_r[\dot{\mathbf{r}}_d(t) - \mathbf{v}(t)] + \mathbf{K}_r[\mathbf{r}_d - \mathbf{r}(t)]\,, \tag{9.11}$$

guarantees asymptotic tracking of desired velocity $\mathbf{v}_d$ and acceleration $\dot{\mathbf{v}}_d$ with exponential convergence [216]. Choosing,

$$\mathbf{f}_\lambda = \ddot{\boldsymbol{\lambda}}_d(t) + \mathbf{D}_\lambda[\dot{\boldsymbol{\lambda}}_d - \dot{\boldsymbol{\lambda}}(t)] + \mathbf{K}_\lambda[\boldsymbol{\lambda}_d(t) - \boldsymbol{\lambda}(t)]\,, \tag{9.12}$$

guarantees asymptotic tracking of a desired force trajectory $(\ddot{\boldsymbol{\lambda}}_d(t), \dot{\boldsymbol{\lambda}}_d(t), \boldsymbol{\lambda}_d(t))$, with exponential convergence [216]. $\mathbf{D}_\lambda$ and $\mathbf{K}_\lambda$ are positive-definite matrices. $\dot{\boldsymbol{\lambda}}$ (9.12) can be computed from the force measurements $\mathbf{f_{ext}}$ as,

$$\dot{\boldsymbol{\lambda}} = \mathbf{S}_f^\dagger \dot{\mathbf{f}}_{\mathbf{ext}}\,. \tag{9.13}$$

where $\mathbf{S}_f^\dagger$ is the pseudoinverse of $\mathbf{S}_f$, computed using, $\mathbf{W} = \mathbf{C}$ in (9.7). Due to the noisy force measurements, the estimate

$$\dot{\boldsymbol{\lambda}} = \mathbf{S}_f^\dagger \mathbf{K}' \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}\,, \tag{9.14}$$

is often preferred, where $\mathbf{K}' = \mathbf{P}_f\mathbf{K}$ and $\mathbf{P}_f = \mathbf{S}_f\mathbf{S}_f^\dagger$.

9.4 Learning Force Tracking

9.4.1 PILCO

PILCO [53] is a data-efficient model-based Policy Search method considered as state-of-the-art in terms of sample efficiency in model-based RL. It is formulated to reduce model bias, one of the key problems of model-based reinforcement learning. This is achieved by learning a probabilistic dynamics model and explicitly incorporating model uncertainty into long-term planning. This way PILCO can cope with a small amount of data, facilitating learning in a handful of trials. Policy evaluation is performed using approximate inference, and policy improvement by computing policy gradients analytically. PILCO considers a dynamic system on the form, $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t)$ with unknown transition dynamics f, and continuous-valued states $\mathbf{x} \in \mathbb{R}^D$ and control-input $\mathbf{u} \in \mathbb{R}^F$, where D and F are the dimensions of the state and input space respectively. The objective is to find a policy π that minimizes the expected return

$$J^\pi(\theta) = \sum_{t=0}^{T} \mathrm{E}_{\mathrm{x}_t}\left[c\left(\mathbf{x}_t\right)\right], \quad \mathrm{x}_0 \sim \mathcal{N}\left(\mu_0, \boldsymbol{\Sigma}_0\right), \tag{9.15}$$

over the next T time steps, where $c(\mathbf{x}_t)$ is the cost associated with the state $\mathbf{x}$ at time t.

9.4.2 Learning Framework

A robot learning framework that can be easily integrated with different RL algorithms is developed in Python. The framework has three components, (i) a set of force controllers, (ii) RL algorithms and (iii) robot simulator with the three components interacting with each other. The force controllers (VIC and HFMC) are implemented as OpenAI Gym environments [32] for easy integration with various RL algorithms. A Gazebo-based [116] simulator is set up with a Franka Emika Panda robotic manipulator using the franka simulator framework [196]. For performing the learning trials in simulations and experiments, the framework is integrated with the robotic manipulator using the Franka ROS Interface framework [196]. In both simulation and experimental setups, a python interface is used to fetch the system states and command the control actions.

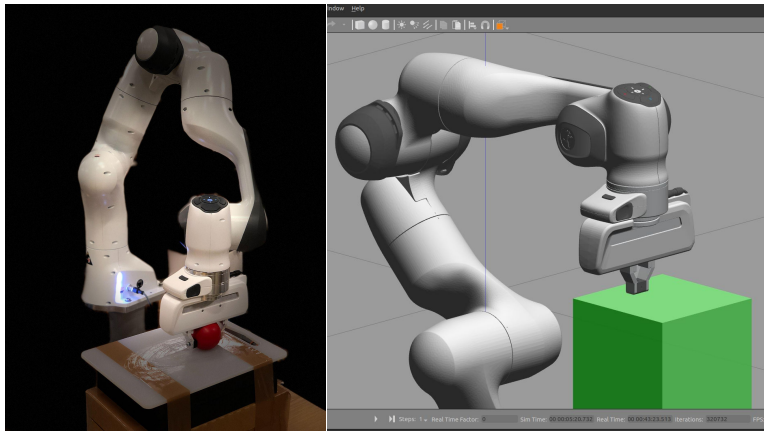


Figure 9.1: The setup for testing the force controllers. The experimental set-up is shown on the left and the Gazebo simulator set-up on the right

9.4.3 Force Controller Implementation

Accurate modeling of contact interaction behavior is important in achieving precise force tracking using model-based learning approaches. For modelling the contact transition dynamics, the state vector $\mathbf{x}$ was chosen as $\begin{pmatrix} F & p_z & v_z \end{pmatrix}$, where F is the contact force, p_z is the z-position and v_z is the z-velocity all defined in Σ. For rest of this chapter we use F_d for the desired contact force. The action spaces for learning are decided based on the controller, but both the controllers have one action each related to stiffness and damping, in the force tracking direction. Consider an action space, $\mathbf{u} \in \mathbb{R}^2$, the training inputs of the model are $(\mathbf{x}_t, \mathbf{u}_t) \in \mathbb{R}^5$ and the output targets are given by, $\boldsymbol{\Delta}_t = \mathbf{x}_{t+1} - \mathbf{x} + \boldsymbol{\epsilon} \in \mathbb{R}^3$. Where, $\varepsilon \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\varepsilon)$, $\boldsymbol{\Sigma}_\varepsilon = \text{diag}([\sigma_{\varepsilon_1}, \ldots, \sigma_{\varepsilon_D}])$. Radial basis functions (RBF)

are used to approximate the policy in the simulations and a linear policy is used in the experiments. An exponential cost function is specified based on the desired behavior of the system with a constant, σ_c deciding the shape of the cost function.

$$c_t = 1 - \exp\left(-\left\|F_t - F_d\right\|^2 / \sigma_c^2\right) \in [0, 1]\,. \tag{9.16}$$

The learning algorithm according to [53] is presented in Algorithm 4. An additional option for using two different GP contact models is implemented to separately model the transition dynamics for the contact establishment and motion phase.

Algorithm 4 PILCO

init: Sample control parameters $\theta \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\pi(\boldsymbol{\theta})$ and dataset $\mathcal{D}$.
Apply random control signals and record data into $\mathcal{D}$.
repeat
  Learn probabilistic (GP) dynamics model, using $\mathcal{D}$.
  Model-based policy search
  repeat
    Approximate inference for policy evaluation,
    obtain $J^\pi(\theta)$.
    Gradient-based policy improvement, obtain
    $\mathrm{d}J^\pi(\theta)/\mathrm{d}\theta$.
    Update parameters θ .
  until *convergence;* ***return*** θ^*;
  Set $\pi^* \leftarrow \pi(\theta^*)$.
  Apply π^* to system and record data into $\mathcal{D}$.
until *task learned*;

VIC Implementation

The VIC controller is implemented by adapting the impedance control law in (6.4) with a modified version of the adaptive impedance law in (9.1). For force control in z direction, the adaptive laws in (9.1) only applies to the z-dimensional properties of $(\mathbf{M}, \mathbf{D}, \mathbf{K}) \in \mathbb{R}^{6\times6}$. However, since an adaptive gain matrix, $\mathbf{M}$ easily can lead to instability, $\mathbf{M}$ is chosen to be static. This results in a system with adaptive damping and stiffness in z. $\mathbf{K}_v$ and $\mathbf{P}$ are chosen to be $\mathbf{I} \in \mathbb{R}^{6\times6}$ in the adaptive law (9.1), the law is reduced to,

$$\begin{aligned}\dot{\boldsymbol{\beta}}(\dot{\Delta\mathbf{D}}, \dot{\Delta\mathbf{K}}) = \\ \begin{pmatrix}0... & ... & \gamma_D^{-1}\dot{E}_z E_{fz} & 0... & \gamma_P^{-1} E_z E_{fz} & 0...\end{pmatrix}^T,\end{aligned} \tag{9.17}$$

where E_z is the error in z-position, E_{fz} is the error in z-force, and γ_D^{-1} and γ_P^{-1} are the rates of adaptability for damping and stiffness in z direction respectively. The change in damping and stiffness matrices due to the adaptive law is thus given by,

$$\begin{aligned} \Delta\dot{\mathbf{D}} &= diag\begin{pmatrix}0 & 0 & \gamma_D^{-1}\dot{E}_z E_{fz} & 0 & 0 & 0\end{pmatrix}, \\ \Delta\dot{\mathbf{K}} &= diag\begin{pmatrix}0 & 0 & \gamma_P^{-1} E_z E_{fz} & 0 & 0 & 0\end{pmatrix}. \end{aligned} \tag{9.18}$$

The parameter space for learning are chosen as γ_P and γ_D.

HFMC Implementation

In-order to perform force control in z direction and motion control in the remaining five dimensions the selection matrices, $\mathbf{S}_f$ and $\mathbf{S}_v$ are chosen as, $\mathbf{S}_v$ = diag(1 1 0 1 1 1) and $\mathbf{S}_f$ = [0 0 1 0 0 0]^T. All parameters of the control law (6.4) are purely state-dependent except the compliance matrix $\mathbf{C} = \mathbf{K}^{-1}$, the gains of the motion controller ($\mathbf{D}_r, \mathbf{K}_r \in \mathbb{R}^{5\times5}$), and the gains of the force controller ($K_{D\lambda}, K_{P\lambda} \in \mathbb{R}$). The motion controller gains, $\mathbf{D}_r, \mathbf{K}_r$ are kept unchanged as we are interested in improving the force tracking behavior. Considering the task with a constant desired tracking force, the compliance matrix, $\mathbf{C}$ is chosen to be constant. Therefore, D_λ and K_λ are chosen for tuning. All matrices are chosen to be diagonal, with values derived from testing in simulation and in experiments.

9.5 Evaluation

The VIC and HFMC were tested and evaluated in both simulation and experimental set-ups. The controllers were compared based on their performance with and without using the learning framework in Section 9.4. A common force-motion tracking task is set up in simulations and experiments. The task is a sweeping task over a flat surface demanding force tracking in the vertical (z) direction and motion tracking in the remaining 5 DOFs (x, y direction and orientations along x, y and z). The controllers are implemented such that only the force-tracking parameters are learned by the PILCO algorithm and the motion-tracking parameters are kept unchanged. The desired tracking force is 3N while performing a sweeping movement of 5cm over the surface in x direction. The task can be divided into two phases, where in phase 1 the robot establishes stable contact with the object by achieving the desired contact force. Phase 2 is the motion phase, where the robot performs a sweeping motion across the surface, tracking desired force and motion trajectories.

The controller is considered to be at a steady state when the actual contact force has reached a steady state value range around the desired contact force. The force tracking performance is compared by calculating the Mean Squared Error (MSE)

between the target and actual contact forces, denoted by Δ_F. The motion tracking behavior is not considered for evaluation since they are very similar in all the tests as the motion tracking parameters remained unchanged. The simulation and experimental set-ups are illustrated in Fig. 9.1. In all figures, F_d, denotes the desired force, and F, F^{PILCO}, denotes the actual contact force when not-, and *when* using PILCO to learn the parameters respectively. x, y and x_d, y_d, represents the actual and desired positions in x and y directions respectively. $\Delta q_x, \Delta q_y, \Delta q_z$ represents the differences in orientations along x, y, z directions in quaternions. K_{Pz}, K_{Dz} and ${K_{Pz}}^{PILCO}$, ${K_{Dz}}^{PILCO}$ represents the stiffness and damping in the force tracking direction z, identified without and with learning framework respectively.

9.5.1 Simulations

The simulations are performed for force and motion tracking on a flat compliant surface modeled in the Gazebo simulator with a stiffness coefficient of $k = 5$, and a damping coefficient of $d = 3$. In order to test the robustness of the controllers in simulation, a Gaussian noise $\mathcal{N}\left(\mu = 0, \sigma_\varepsilon^2 = 0.015 h_e\right)$, is added to force estimate based on comparing with the real system used in experiments. Fig. 9.2 and Fig. 9.3 represent both the contact establishment phase and sweeping movement over the surface as shown in the motion trajectories in Fig. 9.2.c and 9.3.c.

VIC

Fig. 9.2 illustrates the force, motion tracking and the varying damping and stiffness in z. The force tracking error, Δ_F in Fig. 9.2.a is decreased from 0.26 to 0.18 by introducing learning. Steady-state is achieved in 0.35s for the learning-based controller compared to 1.5s for the adaptive controller. The mean value of both F and F^{PILCO} is 2.94N and the variances are 0.009 and 0.007 respectively. Fig. 9.2.b shows how the stiffness and damping in z direction are adapted for achieving the desired force tracking behavior. The motion tracking behavior is shown in Fig. 9.2.c and 9.2.d.

HFMC

The performance of the HFMC in the simulation is shown in Fig. 9.3, illustrating the force and motion tracking results. Introducing learning has reduced Δ_F from 0.28 to 0.14, but the force tracking behavior of the learned controller is not smooth as desired. Steady-state is achieved in 0.32s for a learning-based controller with a mean value of F^{PILCO} is 3.03N and variance of 0.01 compared to 2.77N and 0.03 for F. The motion-tracking behavior is hand-tuned and is comparable with VIC.

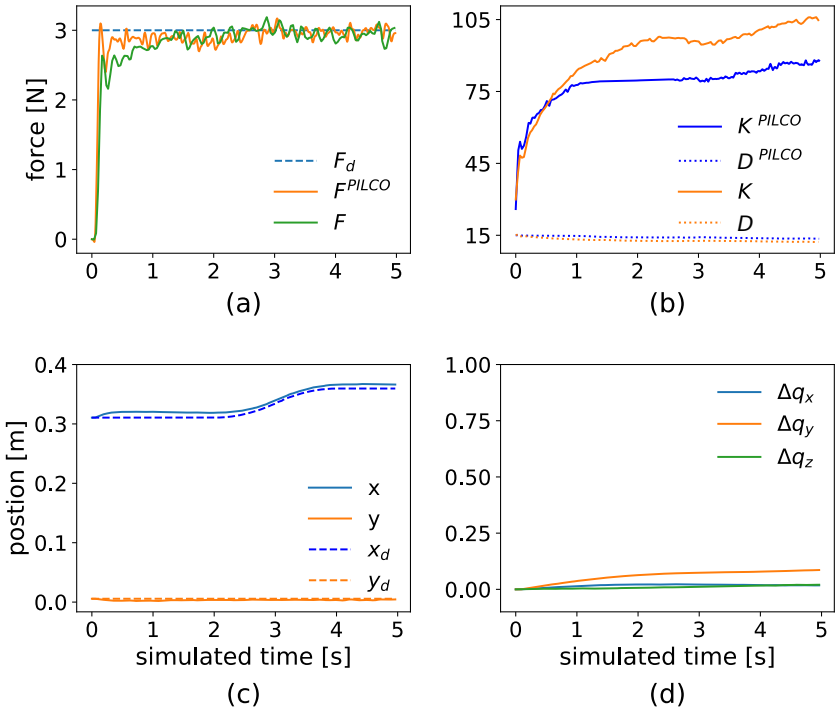


Figure 9.2: Simulation results for VIC using PILCO for the set-up in Fig. 9.1. (a) Force tracking behavior, (b) stiffness and damping behavior in z direction, (c) and (d) motion tracking behavior.

9.5.2 Experiments

The simulation task is replicated in the experimental setup to perform force-motion tracking on a flat surface. In order to have a compliant contact between the robot and the rigid surface, the manipulator's end-effector is equipped with a soft ball as shown in Fig. 9.1. The robot should perform a contact establishment to reach a desired force of 3N and then perform a sweeping motion over the surface for 5cm tracking a desired force. The results of the experiment for both VIC and HFMC in the learning framework is shown in Fig. 9.4. The introduction of learning in VIC has reduced Δ_F from 0.13 to 0.04 as shown in Fig. 9.4.a. During the contact establishment phase, the leaning-based VIC converged to a steady state with an overshoot of 0.51N in 1.2s whereas VIC with fixed adaptive law (9.1) converged to steady state in 3.5s. The learning-based VIC has a steady state variance of 0.002 compared to 0.028 for the VIC with fixed adaptive law. While experimenting with learning-based HFMC with a single GP model for contact establishment and mo-

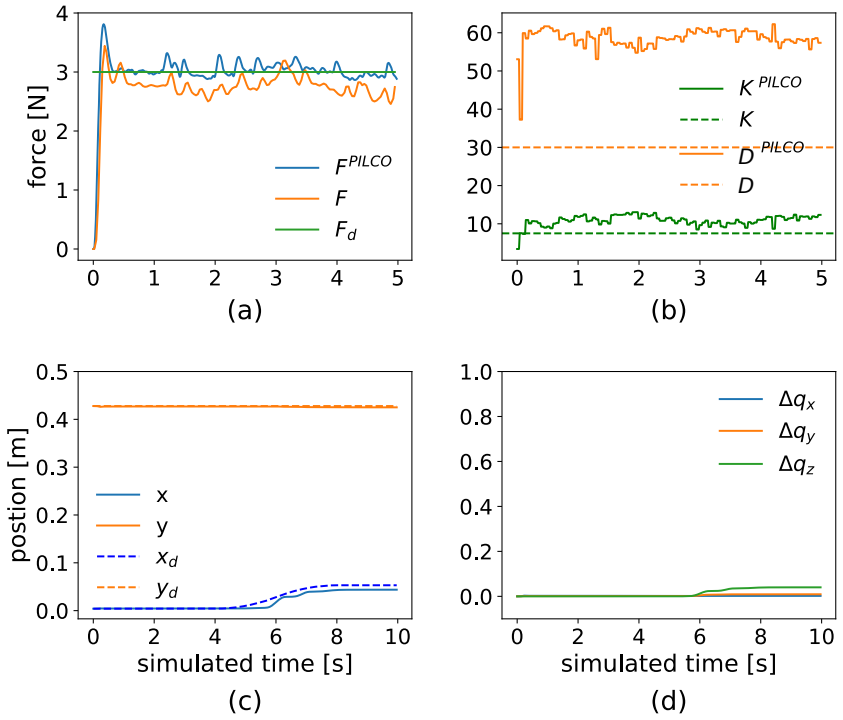


Figure 9.3: Simulation results for HFMC using PILCO for the set-up in Fig. 9.1. (a) Force tracking behavior, (b) stiffness and damping behavior in z direction, (c) and (d) motion tracking behavior.

tion phase, the Δ_F was increased to 1.20 compared to 1.17 for the manually tuned HFMC. But this drawback was eliminated by introducing separate contact models for the two phases and thereby significantly reducing the Δ_F to 0.29. By introducing dual contact models for the learning-based HFMC, the Δ_F of the motion phase (phase 2) was decreased from 1.8 to 0.35.

Fig. 9.5.a represents how the variance of the learned force GP model changed over the learning iterations in two different experimental trials using learning-based VIC. In Each trial, the experimental task was executed for 14 learning iterations. The trial 1 failed in force tracking with reaching unsafe forces above 5N. Whereas, the successful trial 2 corresponds to the results in Fig. 9.4.a. Fig. 9.5.b represents the corresponding change in the cost for the successful trial (trial 2). The decrease in variance after the initial step is minimal during all the successful learning trials conducted with the noisy force data.

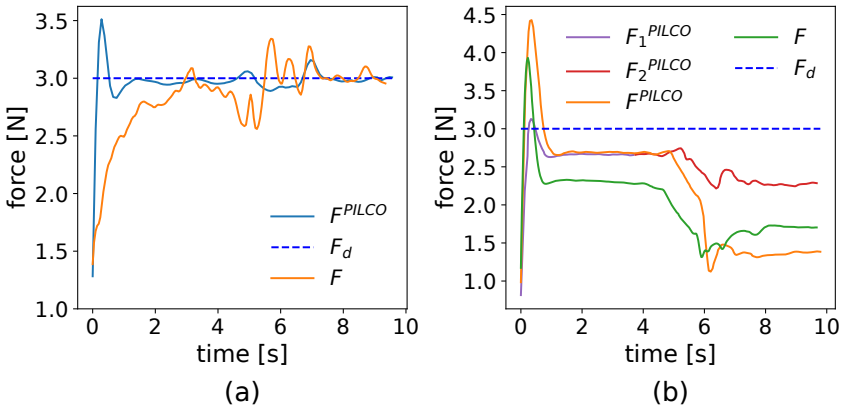


Figure 9.4: Experimental results for VIC and HFMC using PILCO for the set-up in Fig. 9.1. (a) Force tracking behavior of VIC, (b) force tracking behavior of HFMC, where ${F_1}^{PILCO}$, ${F_2}^{PILCO}$ represents the contact establishment and motion phase while using separate contact models for both the phases. While F^{PILCO} represents the force behavior when using a single model for both the phases as in the case of VIC.

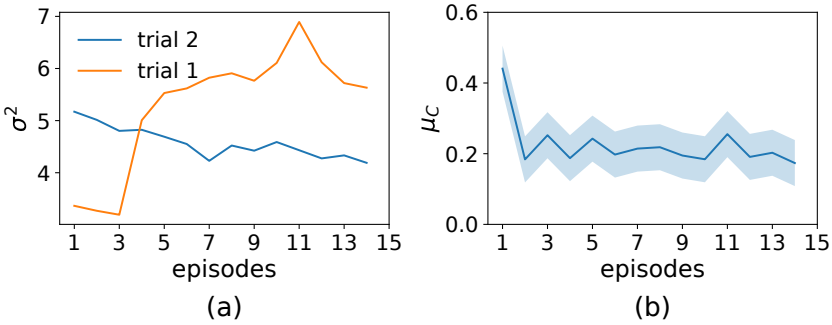


Figure 9.5: (a) Variance of the force GP model for two different training trials, (b) cost for the successful trial (trial 2) across the learning iterations.

9.6 Discussion

The simulation and experimental evaluation demonstrated the advantage of introducing model-based learning to achieve better force tracking in both VIC and HFMC for the robotic interaction task demanding both force and motion tracking. The simulated and experimental results for VIC, are shown in Fig. 9.2 and 9.4.a suggests significant improvement in force tracking by introducing model-based learning to adapt the stiffness and damping parameters. The robust adaptive law (9.1) ensures better force tracking capabilities in VIC even without introducing

learning in simulations. Faster convergence to the steady state is a major impact of learning-based VIC which is consistent in both simulation and experiment. The fast convergence to the desired contact force and less noisy force tracking behavior is a result of a better impedance strategy optimized by PILCO. Furthermore, the flexibility of the controller allowed it to start off with high compliance in z direction, avoiding a high initial force overshoot.

The effect of learning in the stiffness and damping parameters are well demonstrated in Fig. 9.2.b. Similar to VIC, introducing learning has significantly improved the force-tracking behavior of HFMC. The learned stiffness in Fig. 9.3.b has a noticeable decrease in-order to reduce high impact forces during the contact establishment phase. Introducing learning produced significant improvement in the force tracking capabilities as shown in Fig. 9.3.a except for a higher overshoot in the initial phase. The learned impedance strategy could exploit the force-tracking capabilities of HFMC to produce better steady-state behavior both in simulations and experiments. Introducing learning in HFMC had significant improvement in the force tracking error and slightly faster convergence to the desired contact force. The learning-based VIC executed a smooth force-tracking behavior while performing the sweeping motion. Even though the motion tracking ability is not evaluated thoroughly, it was difficult to hand-tune the VIC to achieve accurate motion tracking.

In simulations, a single GP model was used to model both the contact establishment phase and motion phase. However, this could not be translated into the experiments because of the frictional effects. The HFMC has no integral action on the force and relies on accurate models to avoid steady-state errors. The introduction of separate models for the two phases was found to yield better force tracking in the experiments. As in all model-based RL algorithms, the performance of the PILCO algorithm is highly dependent on the accuracy of the dynamics model. The GP model has a limited ability in modeling complex dynamics and is not well suited for highly noisy data. Fig. 9.5.a represents the variance for different GP models across different trials, where difficulty in learning the model is translated into the performance as shown using the variances and the cost. The unsuccessful trials failed at reducing the model variances, thereby learning largely uncertain models. This could be improved using more complex models such as ensembles of Neural Networks or Bayesian Neural Networks. There are few promising model-based RL frameworks using such approaches [220], but these methods have a lower sample efficiency compared to PILCO. However, further improvements in model-based RL and the scope of sim2real transfer of task space controllers [144] could realize learning-based force controllers for robotic interaction tasks.

Despite the high data-efficiency offered by the PILCO framework, it imposes con-

straints on the reward functions and policy structure. This prevents the use of any arbitrary reward function and policy, therefore it is not a flexible RL framework. Additionally, analytical approaches like PILCO can not be efficiently parallelized in a multi-core computer. These limits the application of such an approach to VILC for complex manipulation tasks.

9.7 Conclusions

This chapter presented the implementation and evaluation of two fundamental approaches in robotic force control, HFMC and a Force-based VIC in the PILCO framework. It was shown that combining a learning-based approach with force controllers has the ability to improve robotic interaction control. For the HFMC, the framework was used to learn direct strategies for its damping- and stiffness parameters. In the VIC, strategies were learned for the parameters of an adaptation law. Both controllers showed significant improvement in force tracking ability by introducing model-based learning. While introducing learning led to faster convergence to the desired force in VIC, it led to a significant improvement in the force tracking error in HFMC. The results showed that having highly accurate contact dynamics models are key to having accurate force tracking. GP model does not offer the flexibility required to model the complex robot-object contact dynamics. Hence, in Chapter 11, we address this issue by modeling the interaction dynamics using an Ensemble of probabilistic neural networks.

Chapter 10

A Data-Efficient Variable Impedance Learning Control Framework

This chapter is based on the following paper under review: *Anand, A. S., Abu-Dakka, F. J., and Gravdahl, J. T. (2023). Data-Efficient Variable Impedance Control, IEEE Access.*

10.1 Introduction

Based on the survey presented in Chapter 7, it is crucial to improve the sample efficiency of Variable Impedance Learning Control (VILC) approaches to make them useful for real-world robotic manipulation tasks. Although the existing Model-based Reinforcement Learning (MBRL) methods are sample efficient compared to model-free methods, it still demands a lot of interactions, making it difficult to apply to robotic manipulation. In this work, we tackle this issue by using a micro-data-based policy optimization method combining Gaussian Processes (GP) models and Covariance Matrix Adaptation (CMA-ES) based policy optimization [42].

Probabilistic Inference for Learning Control (PILCO) [53] is a highly data-efficient MBRL approach, has been used for VILC in Chapter 9 [10]. But it imposes constraints on the reward functions and policies structure that prevent the use of any arbitrary reward unlike in a general Reinforcement Learning (RL) setting. Additionally, analytical approaches like PILCO can not be efficiently parallelized in a multi-core computer. These limits the application of such an approach to VILC

for complex manipulation tasks. These deficiencies are addressed in the alternate MBRL approach using GP based dynamical models and CMA-ES based policy optimization in [42]. This approach does not impose any constraints on reward functions or policies and can be easily parallelized. A combination of GP based dynamics modeling and CMA-ES based policy optimization provides a highly data-efficient MBRL framework which takes into the uncertainty of the model and performs a global policy search. In this chapter, we extend this approach to develop a data-efficient VILC framework for robotic manipulation. All the relevant RL-based VILC approaches are presented in Chapter 7 Section 7.3.

This chapter introduces Data-Efficient Variable Impedance Learning Controller (DEVILC), a MBRL framework for VILC focusing on finding an optimal impedance adaptation strategy for a VIC from a few data samples in the context of robotic manipulation. In summary, the main contributions of this chapter are;

- a Model-based VILC framework using GP models and using the evolution strategy, CMA-ES to optimize a NN policy.
- demonstrates a highly data-efficient approach for learning impedance adaptation strategy for robotic manipulation.

The rest of this chapter is organized as follows. In Section 10.2 presents the details of the DEVILC framework proposed. Section 10.3 presents the evaluation of the proposed VILC framework on simulation and experimental setups using Franka Panda robotic manipulator. Detailed discussion on the results and conclusion are presented in Section 10.4 and Section 10.5 respectively.

10.2 Data-Efficient Variable Impedance Learning Framework

The DEVILC framework utilizes GP models to learn the Cartesian impedance model of the system. The learned GP model is then used to optimize a NN-based impedance adaption policy using CMA-ES. The cartesian impedance model represents the environment-robot dynamic relationship in (6.5). We learn the following Cartesian impedance model of the robot manipulator [224] using GP:

$$\mathbf{s}_{t+1} = \mathbf{s}_t + f(\mathbf{s}_t, \mathbf{u}_t) + \boldsymbol{\omega}\,. \tag{10.1}$$

Where $\mathbf{s}_t$ is the state of the robot end-effector at time step t, $\mathbf{u}_t$ is the applied action, $\mathbf{s}_{t+1}$ is the next state, and $\boldsymbol{\omega}$ is the i.i.d Gaussian noise. The GP model with these inputs predicts the mean $\boldsymbol{\mu}(\mathbf{s}_{t+1})$ and variance $\boldsymbol{\sigma}^2(\mathbf{s}_{t+1})$ based on the current state and the action. We define the state $\mathbf{s}_t$ as $[\mathbf{x}_t,\, \dot{\mathbf{x}}_t]$, and action $\mathbf{u}_t$ as $[\mathbf{f}^t{}_{\mathbf{ext}},\, \mathbf{K}_t]$. $\mathbf{K}_t$ is sampled from a parameterized impedance adaptation policy

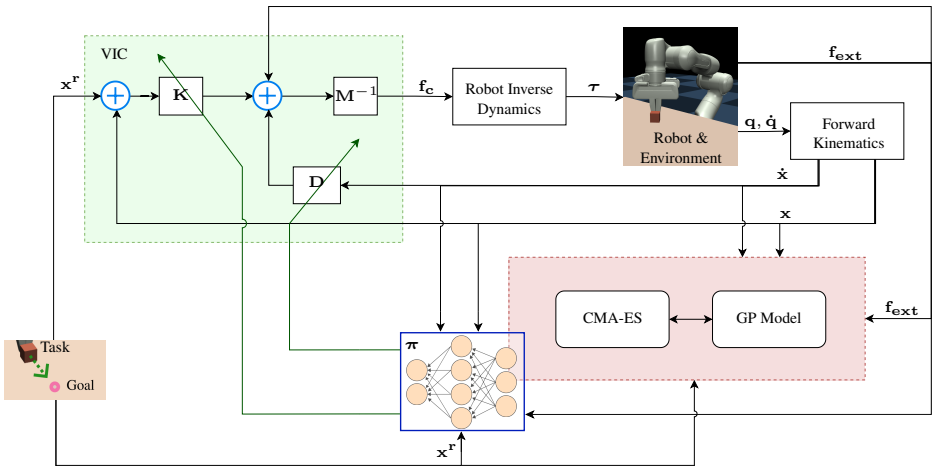


Figure 10.1: DEVILC framework with GP dynamics model, NN based impedance adaptation policy and CMA-ES optimizer.

with $\boldsymbol{\pi}$ s.t $\mathbf{K}_t = \boldsymbol{\pi}\left(\mathbf{s}_t, \mathbf{f}^t{}_{\mathbf{ext}} \mid \boldsymbol{\theta}\right)$ which can be a NN. $\mathbf{f}^t{}_{\mathbf{ext}}$ is the sensed external force acting on the robot at time instant t, this is an uncertain external factor the VILC needs to compensate for. The damping parameters are chosen according to the critical damping condition, $\mathbf{D} = 2\sqrt{\mathbf{K}}$. For an N DoF Cartesian impedance dynamics considered, f contains N independent GP model with each GP model approximating the dynamics along one DoF.

We aim to optimize the compliant behavior of the robot end-effector can be optimized by designing a suitable variable impedance control strategy. Within the proposed DEVILC framework, given a manipulation task objective, this impedance adaptation strategy for the underlying VIC is optimized using CMA-ES and the GP-based Cartesian impedance model (10.1). We learn a NN based impedance adaptation policy $\boldsymbol{\pi}$ in an episodic MBRL setting, where after each episode of interaction with the real system, the GP model is updated and a policy is optimized using CMA-ES for the entire task horizon. The objective of compliant robot manipulation is defined to achieve manipulation task requirements/goals while executing a high level of compliance. In this work, we consider the scenario where the manipulation task requirement can be represented as tracking a desired robot state, but it is generalizable to any objective that can be measured. The compliance objective is to minimize the stiffness of the underlying VIC controller. A cost function describing the task objective and the compliance objective is designed for the real system,

$$C\left(\mathbf{s}_t, \mathbf{K}_t\right) = \delta\mathbf{s}_t^T\mathbf{Q}_t\delta\mathbf{s}_t + \lambda(\mathbf{K}_t)^T\mathbf{R}_t\lambda(\mathbf{K}_t)\,, \tag{10.2}$$

where $\mathbf{s}_t^r$ denotes the reference or goal states. $\lambda(\mathbf{K}_t)$ is the Eigenvalues of the stiffness matrix represented in a vector form, $\delta\mathbf{s}_t = \mathbf{s}_t^r - \mathbf{s}_t$ and $\mathbf{Q}_t$ and $\mathbf{R}_t$ are diagonal gain matrices for task and compliance components respectively. These gain matrices can be either constant or can be a function of the robot's states. For example, in the case of a reference tracking task $\mathbf{Q}_t$ can be chosen as a linear function of $\|\delta\mathbf{s}_t\|$ in order to have higher penalties for larger deviations from the target. While this cost is defined over the real system, it can be used as a target to train the reward function $r = -C$, i.e by taking the negative of this cost at every time instant.

Given the GP model f (10.1) and the cost function (10.2), the goal is to find the optimal policy parameters $\boldsymbol{\theta}$ that maximizes the reward over the entire task horizon given by:

$$J(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{t=1}^{T} r\left(\mathbf{s}_t, \mathbf{K}_t\right) \mid \boldsymbol{\theta}\right] . \tag{10.3}$$

This is achieved by predicting the state evolution over the GP dynamics model. State-to-next-state propagation is carried out as in Monte Carlo estimation by sampling according to the GP model. But additionally, each of these rollouts is considered as a measurement of a function $G(\boldsymbol{\theta})$ that is the actual function $J(\boldsymbol{\theta})$ perturbed by a noise $N(\boldsymbol{\theta})$: [42]

$$\begin{aligned} G(\boldsymbol{\theta}) &= J(\boldsymbol{\theta}) + N(\boldsymbol{\theta}) \\ &= \sum_{t=1}^{T} r\left(f\left(\mathbf{s}_{t-1}, \mathbf{u}_{t-1}\right)\right) . \end{aligned} \tag{10.4}$$

Where $\mathbf{K}_{t-1} = \boldsymbol{\pi}\left(\mathbf{s}_{t-1}, \mathbf{f}_{\mathbf{ext}}^t \mid \boldsymbol{\theta}\right)$ and $\mathbf{f}_{\mathbf{ext}}^t$ is chosen from a random episode of interaction data. We would like to maximize its expectation:

$$\begin{aligned} \mathbb{E}[G(\boldsymbol{\theta})] &= \mathbb{E}[J(\boldsymbol{\theta}) + N(\boldsymbol{\theta})] \\ &= \mathbb{E}[J(\boldsymbol{\theta})] + \mathbb{E}[N(\boldsymbol{\theta})] \\ &= J(\boldsymbol{\theta}) + \mathbb{E}[N(\boldsymbol{\theta})] . \end{aligned} \tag{10.5}$$

With the assumption that $\mathbb{E}[N(\boldsymbol{\theta})] = 0$ thereby maximizing $\mathbb{E}[G(\boldsymbol{\theta})]$ is equivalent to maximizing $J(\boldsymbol{\theta})$. In order to search for the optimal policy $\boldsymbol{\pi}^\star$ with parameters $\boldsymbol{\theta}^\star$, we utilize the evolutionary optimization strategy, CMA-ES.

The CMA-ES is an evolutionary strategy designed to solve non-convex and non-linear black-box optimization problems in continuous domain [75]. It is one of the state-of-the-art methods in evolutionary computation, specifically for continuous optimization. CMA-ES uses a multivariate Gaussian distribution $\mathcal{N}\left(\boldsymbol{\mu}^{\mathbf{d}}, \boldsymbol{\Sigma}^{\mathbf{d}}\right)$

where $\boldsymbol{\mu}^{\mathbf{d}} \in \mathbb{R}^d, \boldsymbol{\Sigma}^{\mathbf{d}} \in \mathbb{R}^{d \times d}$ is a positive definite symmetric matrix and d is the dimension of a solution vector. CMA-ES solves the maximization of an objective function $J(\boldsymbol{\theta})$ as the optimization of the noisy function $G(\boldsymbol{\theta}) = J(\boldsymbol{\theta}) + N(\boldsymbol{\theta})$ where $N(\boldsymbol{\theta})$ is the noise [75, 42]. This facilitates maximizing the objective without computing or estimating it explicitly. CMA-ES performs four steps at each generation i:

1. sample β new candidates according to a multivariate Gaussian distribution of mean ${\boldsymbol{\mu}^d}_i$ and covariance ${\boldsymbol{\Sigma}^d}_i$.
2. rank the β sampled candidates based on their noisy performance over the objective $G(\boldsymbol{\theta}_i)$.
3. compute ${\boldsymbol{\mu}^d}_{i+1}$ by averaging n best candidates: ${\boldsymbol{\mu}^d}_{i+1} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{\theta}_i$.
4. updates the covariance matrix to match the distribution of the $\boldsymbol{\mu}^d$ best candidates identified.

One key advantage of this ranking-based approach is it reduces the impact of noise on the performance function. This is because the solution is looking for the β best candidates and errors can only happen at the boundaries between the low-performing and high-performing solutions. Even if a candidate is misclassified because of the noise, this error is smoothed out by taking the average in step 3 to calculate ${\boldsymbol{\mu}^d}_{i+1}$ [96].

By combining the GP-based model learning, policy evaluation based on the noisy function (10.4) (10.5), and CMA-ES based policy search forms a MBRL based VILC framework DEVILC as shown in Fig. 10.1. The DEVILC method alternates episodically between the robot interacting with the real system and learning the impedance adaptation policy (which also includes updating the Cartesian impedance model). The DEVILC Algorithm is shown in Algorithm 5.

10.3 Evaluation

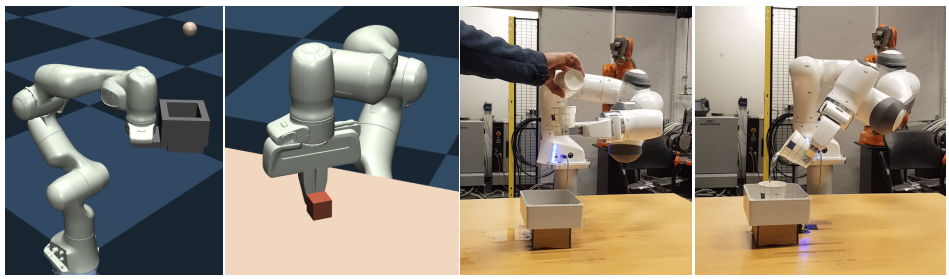
We evaluated the proposed DEVILC framework using a couple of simulation setups in addition to a real experiment. The focus of our evaluation is on demonstrating the effectiveness of the proposed approach to learning suitable VIC policy for robotic manipulation tasks. For evaluation, depending on the task, we consider the adaptation of the stiffness along the specific DoF of the robot manipulator while keeping the stiffness values along the other DoFs constant. GP models are used to learn the system dynamics, while a NN is used as the policy. The input state space for the GP model contains the Cartesian position and velocity. The input action

Algorithm 5 DEVILC

Given a cost function C, initialise an NN policy $\boldsymbol{\pi}$.
Populate dataset $\mathcal{D}$ using a VIC with random $\mathbf{K}$ values for N_i initial trials .
while *task* $\neq$ *solved* **do**
  Learn the GP dynamics model f *on* $\mathcal{D}$.
  $\boldsymbol{\theta}^* = \text{argmax}_{\boldsymbol{\theta}} \mathbb{E}[G(\boldsymbol{\theta})]$ *(Section 10.2)* .
  for $i \leftarrow 1$ **to** *TaskHorizon* **do**
    $\mathbf{K}_i = \boldsymbol{\pi}\left(\mathbf{s}_i, \mathbf{f}^i{}_{\mathbf{ext}} \mid \boldsymbol{\theta}^\star\right)$.
    $\mathbf{s}_{i+1}, \mathbf{f}^{i+1}{}_{\mathbf{ext}}$ = *execute VIC with* $\mathbf{K}_i$ *(*(6.6)*)* .
    $\mathcal{D} = \mathcal{D} \cup \left\{\mathbf{s_{i+1}}, \mathbf{f}^{i+1}{}_{\mathbf{ext}}, \mathbf{K_i}\right\}$
  end
end

space for the GP models contains the external forces acting on the end-effector ($\mathbf{f_{ext}}$) and the stiffness values ($\mathbf{K}$). Given these inputs GP model predicts the next Cartesian pose ($\mathbf{x}$) and velocity ($\dot{\mathbf{x}}$) of the robot end-effector. The input state space of the NN policy contains the $\mathbf{x}$, $\dot{\mathbf{x}}$, and $\mathbf{f_{ext}}$. The output of this NN policy is the predicted stiffness values. Dimensions of all these state and action spaces for the GP model and NN policy are dependent on the number of DoFs considered for the task. We use a GP structure with an exponential kernel with automatic relevance determination [224]. The NN policy in all the experiments contained one hidden layer with 32 neurons. For the CMA-ES optimizer we utilize BIPOP-CMA-ES with restarts [73] in combination with UH-CMA-ES for noisy functions [74] as proposed in [42].

The values for the damping component are chosen as $\mathbf{D} = 2\sqrt{\mathbf{K}}$. he mass matrix $\mathbf{M}$ is kept constant to avoid stability issues during the experiment. The sampling frequency which is equivalent to the VIC frequency is set as $10\,\text{Hz}$ for all the tasks. For all the experiments 10 learning trials/episodes were conducted alternating between model learning and policy optimization. For the real-world experiments, the interaction data was downsampled to 20 data points per episode due to the low computational speed of the GP inference. For all chosen tasks, the requirements can be defined as achieving a desired goal pose for the robot end-effector. However, the robot is also required to be highly compliant whenever it is possible or be stiff only when it is necessary. This is achieved by using a weighted reward in (10.2) for the task requirement (first term) and maximizing compliance (second term). In the considered tasks, we are, essentially, trying for a trade-off between position control and compliance. The cost function for each task has different values of the gain matrices $\mathbf{Q}_t$ and $\mathbf{R}_t$ in (10.2), defining the desired trade-off between position accuracy and compliant behavior. The values of these gain matrices are



(a) (b) (c) (d)

Figure 10.2: Simulation tasks, (a) Reacting to falling object: The robot manipulator with cup end-effector should hold a Cartesian position while smoothly catching a ball of weight $0.5\,\mathrm{g}$ falling into the cup. (b) Pushing task: A robot manipulator with a gripper end-effector should push an object over a rigid surface with friction to a target position. The experimental tasks, (c) water filling: robot end-effector is fitted with an empty cup to which water is filled by a person, (d) water pouring: robot end-effector is fitted with a cup filled with water and the robot transfers this water to another cup.

hand-tuned for each task and kept constant during the learning process. In all the results force is represented in Newtons and distance in cm.

10.3.1 Simulations

Two simulation experiments were conducted to evaluate the effectiveness of the proposed approach on learning VILC. We chose two manipulation tasks with different dynamics, (i) catching falling objects and (ii) pushing an object along a surface. In the first task, the robot has to adapt its impedance to optimally react to the impact of the falling object and also to carry the additional weight added by the object. Whereas in the pushing task the robot has to adapt the impedance necessary to overcome the inertia of the object and the frictional forces and be more compliant towards the end of the task.

Reacting to falling object: In the task (Fig. 10.2 (a)), the robot manipulator is fitted with a tray as the end-effector and an object is dropped to the tray first and then removed from the tray after one second. The robot is initialized to be highly compliant at-rest position and is expected to hold the pose when the object is dropped to and removed from the tray. The robot is additionally expected to be as compliant as possible and be stiff only when necessary as in (10.2). Multiple trials were performed with the object being dropped from different heights to the cup, resulting in robot behavior as shown in Fig. 10.3. The policy is optimized such that the deviation of the robot from its initial position is minimal while being as compliant as possible in reacting to the falling object. We only consider stiffness adaptation along the z direction for this task and the stiffness values along all other

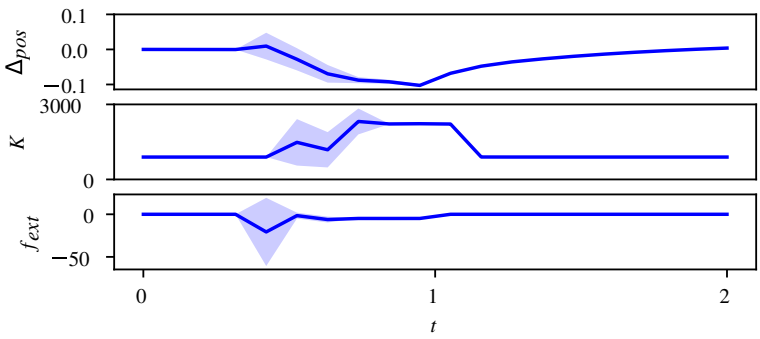


Figure 10.3: Reacting to falling object: the robot is expected to have minimal deviation Δ_{pos} from the initial pose in z direction. The results shown here are the mean values over 10 trials where the objects are dropped from randomly chosen heights between $(0.5 - 1.0)$ m.

DoFs are kept unchanged during the learning. The result shows that the robot is at rest with low stiffness and the stiffness K_z is increased instantaneously in response to the impact force $\mathbf{f_{ext}}$ and further increases in response to deviation from the initial position. Upon removing the object, the stiffness is decreased enough to drive the robot back to the initial position.

Pushing task: In this task (Fig. 10.2 (b)), the robot should push an object placed on a table with friction to a target position. The policy is learned to adapt the stiffness in the pushing directions (x and y) to push the object to the target while stiff only when necessary and being compliant otherwise. The stiffness along other DoFs are kept constant. The robot is initialized with low stiffness values along the pushing directions. The results in Fig. 10.4 show that the stiffness is initially increased to larger values as expected along the pushing directions to overcome the inertia of the object. The stiffness is decreased when the object is close to the target position, executing high compliance. The robot learned to adapt the stiffness profiles in a suitable way to push the object to the target with high accuracy while being stiff only when necessary.

10.3.2 Real-world Experiments

Two experiments were conducted to evaluate the effectiveness of the proposed approach in real-world robotic tasks demanding impedance adaptation. We chose the water pouring and filling task with the Franka-Emika Panda Robot manipulator. This is inspired by human manipulation behavior, we adapt our arm stiffness continuously in both pouring and fillings tasks to be efficient. For both filling and poring tasks, the cost function is described as reaching a target pose while being as compliant as in(10.2). For both tasks, we consider only stiffness adaptation along

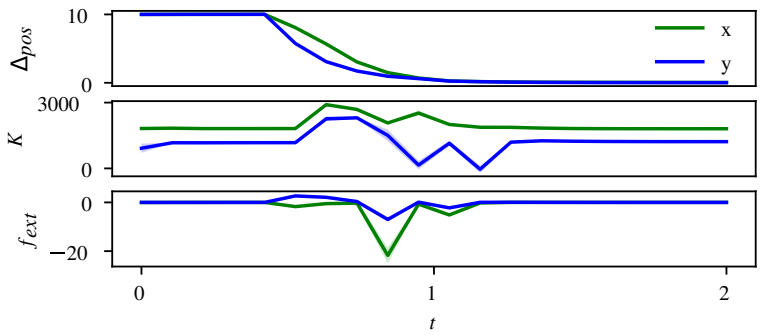


Figure 10.4: Pushing task: the robot is tasked to push an object on the table to $10\ \mathrm{cm}$ in x and y directions at $t = 0.5\ \mathrm{s}$. The results shown here are over 10 trials with objects of random weights between $(0.5 - 1.0)$.

the z axis while maintaining a constant stiffness along all other DoFs.

Water Filling: The experimental setup is shown in Fig. 10.2 (c) where the robot end-effector is fitted with an empty cup and water is poured into the cup. The robot is initialized with a low stiffness value to be highly compliant at the initial position and is expected to react optimally while the water is poured into the cup. The optimization objective of the form (10.2) is defined to hold the initial pose by continuously increasing the stiffness enough to hold the extra weights of the water being added to the cup. Results in Fig. 10.5 show that the learned VILC continuously increases the robot stiffness increases while water is added to the cup allowing only a small deviation of only $3\,\mathrm{mm}$ from the desired pose. The stiffness profile generated by the VILC during the tasks appears to be very well correlated with the sensed external force on the end-effector imparted by the water.

Water Pouring: The experimental setup for the task is shown in Fig. 10.2 (d) where the robot end-effector is fitted with a cup filled with water. The VIC is initialized with the right amount of compliance such that the robot hold the cup filled with water at the initial pose. The pouring task is defined as the robot pouring the water into a second cup placed on the table. The robot movement for the pouring task is defined as reaching a target end-effector pose such that the water is entirely transferred to the second cup. The optimization objective is to reach the target pose while being as compliant as possible. The robot is expected to learn to be less stiff as the water is transferred to the second cup. Results in Fig. 10.6 show that the learned VILC increases the robot stiffness in relation to the increased sensed forces during the first phase of the task where the cup is tilted to start transferring the water to the second cup. In the second phase of the task, (i.e once the water starts to flow into the second cup), the stiffness is decreased in

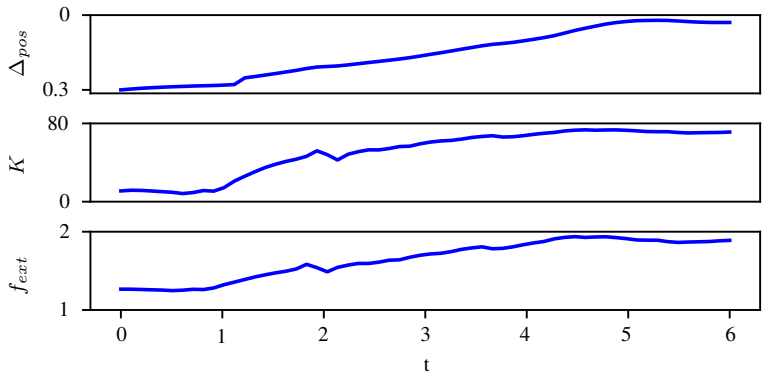


Figure 10.5: Water filing: water is filled into a cup fitted to the robot end-effector and the robot is expected to have a minimal deviation from the initial pose while executing compliance. The stiffness is varied only z direction and is kept constant in along all other DoF, all the values shown here are along z direction.

relation to the decreased weight of the water robot has to hold. Overall the learned VILC policy is able to reach the target pose while being compliant.

10.4 Discussion

The VILC approach presented in the work is evaluated on different tasks in Section 10.3 to learn impedance adaptation strategies. The optimization objective in all experiments has been to maintain a high level of compliance in general while being stiff only when demanded by the task. In all the tasks, the task requirement is defined by achieving a desired goal pose for the robot end-effector. The performance of the impedance adaptation strategy is evaluated based on how well it is able to achieve this requirement while being maximally compliant. In the case of tasks demanding positional accuracy, this means a suitable trade-off between accuracy and compliance. But IC under stable behavior allows the robot to asymptotically converge to the target pose. This property allows the learning methods to suitably vary the impedance to be maximally compliant without necessarily sacrificing the positional accuracy, especially in tasks that don't demand strict real-time trajectory tracking. Whereas optimizing impedance profiles to be maximally compliant allows robots to be more dexterous, safe, and energy efficient. The NN policy obtained using CMA-ES based optimization could adapt the stiffness profile in response to external forces and deviations from the target pose while maintaining a high level of compliance whenever possible.

Compared to the existing approach, the main advantage of our VILC approach is

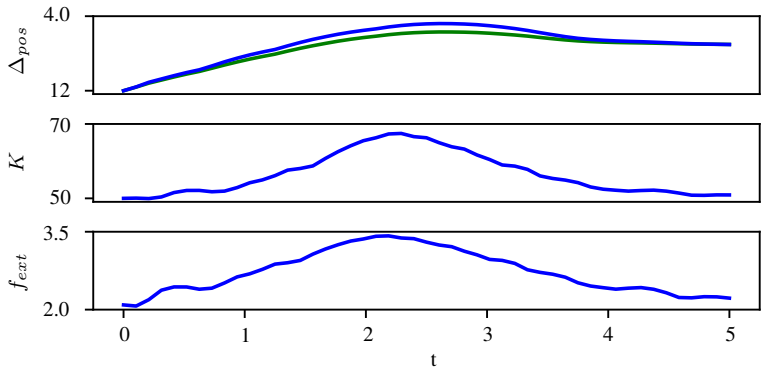


Figure 10.6: Water pouring: the robot is tasked to pour the water into a cup placed on the table. The pouring task is defined by commanding the robot to move to a pre-defined goal pose. The stiffness is varied only z direction and is kept constant in along all other DoF, all the values shown here are along z direction.

data efficiency as the stiffness adaptation policy is learned from a handful of trials without any constraint on the optimization objective and policy structure. The existing VILC methods with comparable data-efficiency are only PILCO based approaches and PI^2 [34]. While the PILCO-based approaches in [10, 137] offer a highly data-efficient approach is limited by the type of cost functions and a differentiable policy and higher computational effort on optimizing the policy. Whereas, the proposed approach is generalizable to any policy and cost structure. Whereas PI^2 approach in [34] is not directly applicable to the force-based VIC considered in this work. CMA-ES based policy optimization is shown to achieve similar performance to PILCO and PI^2 in robotic manipulation tasks while being more data-efficient [42]. Because of these reasons, we have not provided any comparison with these approaches in this work. Even though there are other RL approaches using complex dynamical models such as NN as discussed in Chapter 7, they pose challenges to real-world applications due to low data efficiency. The learning progress of the proposed DEVILC approach in Fig. 10.7 shows that the learning saturates within 10 iterations providing a local optimal solution of the impedance adaptations strategy.

We have not discussed the aspects of safety/stability in this work. We assumed a stiffness parameter range learned is with the stable region for the underlying VIC. Guaranteeing stability properties to the resulting VILC is challenging as guarantees have to be provided in real-time in an online fashion as the stiffness values predicted by the policy are state-dependent. The approach proposed in [97] by designing a quadratic Lyapunov candidate function could be coupled with GP

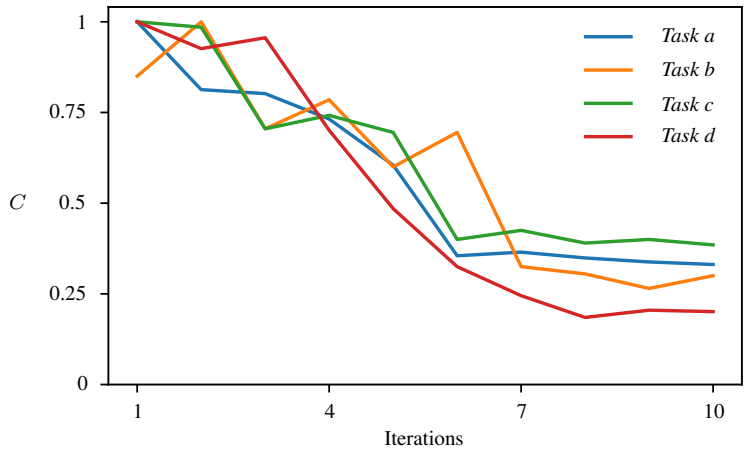


Figure 10.7: Mean cumulative cost after each learning iteration for all tasks (a) - (d) in Fig. 10.2. The values are normalized between [0,1] for each task.

models to provide probabilistic stability guarantees similar to safety guarantees in [108]. GP models allow for providing such guarantees on safety and stability by using additional optimization constraints [8]. But this needs further research in the case of VILC for providing closed-loop safety and stability guarantees during learning and for the final policy. The safe learning approaches described in [8] are interesting to explore for model-based VILC. One feasible approach in this direction could be to provide probabilistic safety guarantees using CLF for stability and CBF as a safety filter to solve constrained optimization problems over the GP model [108].

The strength of the proposed approach relies on a trade-off between high data efficiency and scalability to complex problems demanding richer model representations. GP models have high data efficiency [53], providing a reliable estimate of model uncertainties which is very suitable for model-based policy optimization [53] and MBRL in general. GP have limited ability in modeling complex dynamics and is not well suited for highly noisy data. Additionally, GP models are not good at representing non-smooth dynamics such as contact dynamics or human-robot-interaction dynamics. This effect was observed in task (b) where the robot is pushing an object to a goal position. Here the robot end effector is prone to lose contact with the object during the task and making it difficult for the GP model to learn the dynamics. This results in slightly noisy impedance profiles in Fig. 10.4. GP models also poses challenges in computational effort when the dataset is large. Similarly, CMA-ES limits the size of the policy parameter size for computational speed, which is a common drawback of most evolutionary algorithms. More scal-

able model-based VILC approaches can be developed using DNN models and RL. Still, they have much higher sample complexity and low scope of providing safety guarantees.

10.5 Conclusions

In this chapter, we presented DEVILC, a data-efficient model-based VILC approach to learning compliant robotic manipulations skills. The Cartesian impedance dynamics of the robot controlled using a VIC is learned using GP models. The learned dynamics model was coupled with CMA-ES optimization strategy to find a suitable impedance adaptation policy for a task. The optimization objective was designed such that the robot should be compliant unless it is necessary to be stiff, which is fundamental to how humans manipulate objects. We evaluated our approach to simplified robotic manipulation tasks in simulations and experiments. The impedance adaptation policy optimized exhibited the desired compliance behavior by being highly compliant unless acted upon by an external force or the robot pose deviated from the desired pose. In future work, we aim to extend this approach to incorporate safety and stability constraints.

Chapter 11

Deep Model Predictive Variable Impedance Control

This chapter is based on the following paper under review (publicly available on arxiv currently [9]):

Anand, A. S., Abu-Dakka, F. J., and Gravdahl, J. T. (2023). Deep Model Predictive Variable Impedance Control, Journal of Robotics and Autonomous Systems.

11.1 Introduction

In general, the control policies obtained using Reinforcement Learning (RL) are optimal for particular a task or a task scenario. This depends on the state-action pairs it explored during the learning phase. Such task-specific control policies are not necessarily transferable to different tasks or task scenarios. Learning control policies that can be easily transferred to different tasks could be very useful in robotics as it could enable robots to perform multiple tasks without having to learn policies specific to each task. Although there are approaches available to tackle this issue of transferability in RL such as in a hierarchical RL setup, this remains a major drawback in applying RL in robots. This is a major drawback for all the existing RL-based Variable Impedance Learning Control (VILC) approaches provided in Chapter 7 and the VILC approaches presented in the previous two chapters. Another important insight from the previous two chapters is on the limitations of Gaussian Processes (GP) to learn the Cartesian impedance model of the robot as they do not scale well with high-dimensional data and complex dynamics. Alternatively, such complex dynamics can be estimated from data using a Neural Network (NN), but they suffer from over-fitting and do not quantify uncertainties

Table 11.1: Comparison among state-of-the-art of VILC approaches

	Data-efficiency	Task transferability	Model-based/ Model-free	Computation time	force-/position-based VIC
[144, 20, 30, 215]	low	-	model-free	low	force
[21, 110]	low	-	model-free	low	position
[34]	high	-	model-free	low	force
[137, 179]	high	-	model-based	high	position
[10]	high	-	model-based	high	force
Our MPVIC	high	✓	model-based	high	force

unlike GP-based models. Probabilistic Ensemble NN (PENN) models introduced in [51] overcome these limitations of NNs, offering a way to quantify both aleatoric and epistemic uncertainties.

In this chapter, we propose a deep Model Predictive Variable Impedance Control (MPVIC) framework, where a NN based Cartesian impedance model of the robotic manipulator is used in a Cross Entropy Method (CEM) based MPC for online adaptation of the impedance parameters of a VIC. This deep MPVIC framework is utilized to learn impedance adaptation strategy for various robotic manipulation tasks by specifying a suitable cost function. The main contributions of this chapter are:

- a novel VIC framework, we call it deep MPVIC. It combines a CEM-based MPC with PENN dynamical model for compliant robotic manipulation offering the following properties.
  - transferability: the key property of the deep MPVIC framework is that it facilitates the transferability of the impedance adaptation strategy between different manipulation tasks without any need for relearning or fine-tuning.
  - data efficiency and scalability: the proposed framework can learn VIC from fewer data samples while it is scalable to complex manipulation tasks.
- an uncertainty-based exploration scheme is integrated into the proposed framework to facilitate learning a generalized Cartesian impedance model of the robot in a data-efficient manner.
- an extensive evaluation in simulation and real setups, in addition to a comparison between our approach and the state-of-the-art model-free and model-based RL approaches on transferability and performance.

A comparison between existing RL-based VILC approaches is summarised in Table 11.1.

The rest of the chapter is organized as follows. Section 11.2 describes the existing references relevant to our work. Section 11.3 presents the details of the deep MPVIC framework proposed. Section 11.4 presents the evaluation of our approach on simulation and experimental setups using Franka Panda robotic manipulator. Detailed discussion on the results and the limitations of our approach is presented in Section 11.5 and conclusion in Section 11.6.

11.2 Related Work

All the relevant RL-based VILC approaches are presented in Section 7.3. In this section, we discuss three relevant works using MPC to optimize the robot impedance for manipulation tasks. In literature, MPC is used in robotic interaction control for manipulations tasks [151, 66], where MPC optimizes the robot control input but not the stiffness itself, while in our approach the MPC adapt the stiffness values directly. It is possible to couple our deep MPVIC with the approach in [151] where it can be used as a low-level optimizer to solve additional constraints. Haninger *et al.* [71] used an MPC scheme with GP models for human-robot interaction tasks. The MPC scheme used could optimize the impedance parameters for an admittance controller, but it is task-specific as the human force model is estimated from demonstrations as a function of robot states. Using GP models limits the complexity and generalizability of the model as pointed out by the authors in [71]. Unlike [71], we optimize the impedance parameters for a force-based VIC in our deep MPVIC framework using PENN to model the Cartesian impedance behavior of the robot manipulator.

11.3 Deep MPVIC Framework

The deep MPVIC framework is formulated to optimize a VIC utilizing a learned PENN based Cartesian impedance model of the robot manipulator within a CEM based MPC.

11.3.1 Learning Cartesian Impedance Model

A Cartesian impedance model of the robot manipulator system controlled using a VIC is learned as a PENN model in an MBRL setting alternating between model learning and CEM based exploration strategy. The Cartesian impedance model represents the environment-robot dynamic relationship in (6.5). We define the state s_t as $[\mathbf{x}_t, \dot{\mathbf{x}}_t]$, and action u_t as $[\mathbf{f}^t{}_{\mathbf{ext}}, \mathbf{K}_t]$. $\mathbf{K}_t$ is given by the CEM-based MPC scheme. $\mathbf{f}^t{}_{\mathbf{ext}}$ is the sensed external force acting on the robot at time instant t, this is an uncertain external factor the VIC needs to compensate for. The damping

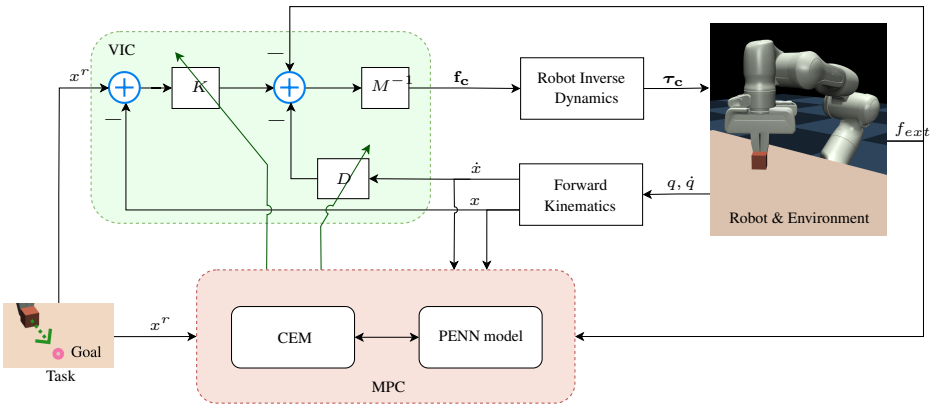


Figure 11.1: Block diagram of the deep MPVIC with PENN Cartesian impedance model and the proposed CEM-based MPC scheme for impedance adaptation. This impedance adaptation scheme along with the VIC forms the deep MPVIC framework. The task objective is represented by (11.2).

parameters are chosen according to the critical damping condition, $\mathbf{D} = 2\sqrt{\mathbf{K}}$.

To learn a generalized model, an exploration strategy is designed to minimize the epistemic uncertainty of the model across the entire state space. The exploration strategy chooses the actions which maximize the epistemic uncertainty estimate from PENN. Given a PENN model $\tilde{f}$ of B bootstrap models $\tilde{f}_b$, the uncertainty of the model prediction at the current state can be estimated by calculating the model variance [193], $\rho = \sigma^2$, given by

$$\rho(s,u) = \frac{1}{B-1}\sum_{b=1}^{B}\left(\tilde{f}_b(s,u) - \overline{\tilde{f}(s,u)}\right)^2 . \tag{11.1}$$

The designed exploration scheme will excite the system in areas in its state space where the model is more uncertain, thereby maximizing the information gained during exploration. The exploration scheme relies on a control strategy that chooses the actions that provide the highest uncertainty estimate from any given state according to (11.1). We employ a CEM-based MPC strategy to optimize for the actions that will excite the system to the most uncertain areas. In order to achieve this we define the MPC cost to maximize the variance of the outputs from all the individual NN models in the PENN, $C_\rho = \rho(s_t, u_t)$. At any given state s_t, CEM-based MPC scheme works by (i) sampling a set of actions from the defined time-evolving distribution, (ii) sorting the actions according to the uncertainty estimate in (11.1), (iii) apply the action u_t^* with the highest value of ρ, and (iv) update the Gaussian distribution. This exploration strategy enables learning a generalized

Algorithm 6 Learning a generalized Cartesian impedance model

Initialize dynamics model $\tilde{f}$.
Populate dataset $\mathcal{D}$ using random controller for n initial trials.
for $k \leftarrow 1$ **to** K *Trials* **do**
  Train dynamics model $\tilde{f}$ on $\mathcal{D}$.
  for $t \leftarrow 1$ **to** *TaskHorizon* **do**
    for *Actions* $u_{t:t+T} \sim$*CEM*$(\cdot)$, *1* **to** *CEM Iterations* **do**
      Evaluate and sort the actions by based on the uncertainty estimate in (11.1).
    end
    Execute first action u_t^* from optimal action sequence $u_{t:t+T}^*$.
    Record outcome: $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, u_t, s_{t+1})$.
  end
end

model in a sample-efficient way. The model learning approach is summarized in Algorithm 6. Learning a model with low uncertainty over the entire state-space facilitates reusing the model for different tasks.

A free-space unconstrained manipulation task where the robot has to interact with its external environment can be described by a scenario where a robot in its current state s_t under the influence of an external force or sensed force f_t provided with a goal state s_t^r and a control input u_t transitions to the next state s_{t+1}. The dynamics model shown in Fig. 11.1 represents a generalized Cartesian behavior of an unconstrained end-effector of a robot manipulator controlled by a VIC.

11.3.2 Impedance Adaptation

The compliant behavior of the robot end-effector can be optimized by designing a suitable impedance adaptation strategy. The Cartesian impedance model of the robotic system $\tilde{f}$ can be utilized in a MPC framework to adapt the impedance parameters of the VIC by designing a suitable optimization objective as shown in Fig. 11.1. The MPC scheme uses the prediction of the PENN model $\tilde{f}$ to plan action trajectories yielding the highest reward. At every time-step, an MPC with a horizon length of n, samples the current state and optimizes a control trajectory $u_{t:t+n}$ for n future time-steps and applies the first control input, u_t, to the system. The action optimal action sequence is chosen by: $\arg\min_{u_{t:t+n}} \sum_{i=t}^{t+n} \mathbb{E}_{\tilde{f}}\left[C\left(s_i, u_i\right)\right]$, where C is the cost function. A gradient-free optimization method, CEM is used in an MPC setting to optimize the controller over the PENN model. CEM samples actions from a distribution closer to previous action samples achieved the minimum cost.

Algorithm 7 deep MPVIC

Given a cost function C and a PENN dynamics model $\tilde{f}$.
MPC based optimization
for $t \leftarrow 1$ **to** *TaskHorizon* **do**
  CEM-based optimization
  for $i \leftarrow 1$ **to** *CEM Iterations* **do**
    Generate N samples.
    Sample N stiffness profiles $K_{t:t+T} \sim \text{CEM}(\cdot)$.
    Evaluate samples.
    Calculate C (11.2) for all $K_{t:t+T}$ on $\tilde{f}$ with actions $[K_{t:t+T}, f_t, s_t^r]$ using trajectory sampling (Section 11.3.2).
    Sort stiffness profiles K based on C.
    Update CEM$(\cdot)$ distribution.
    Choose optimal K^* where C is minimum.
  end
  Adapt the impedance parameters of VIC.
  Execute first action K_t^* from optimal action sequence $K_{t:t+T}^*$.
end

In order to calculate the cumulative cost of the action trajectories, we use particle-based propagation as they are specifically suited for PENN dynamics models, [51]. P particles are created from the current state, $s_{t=0}^p = s_0 \forall p$ in order to predict the state trajectories using particle-based propagation. Each of these particles are propagated along the PENN model as, $s_{t+1}^p \sim \widetilde{f}_{b(p,t)}\left(s_t^p, u_t\right)$ based on a bootstrap $b(p,t)$ in $\{1, \ldots, B\}$. We keep the particle bootstrap index constant during a trial as it allows us to separate between aleatoric and epistemic uncertainties [54]. The aleatoric uncertainty can be quantified using the average variance of particles of the same bootstrap whereas epistemic uncertainty can be quantified using the variance of the average of particles of the same bootstrap indexes.

The proposed deep MPVIC approach utilizing PENN models is described in Algorithm 7. The objective of the impedance adaptation strategy is to achieve the manipulation task requirement while executing a desired level of compliance. A cost function describing the task objective and the compliance objective is designed for the CEM-based MPC as,

$$C\left(s_t, u_t\right) = \delta s_t^T \mathbf{Q}_t \delta s_t + \lambda(K_t)^T \mathbf{R}_t \lambda(K_t)\,, \tag{11.2}$$

where $\lambda(K_t)$ are the eigenvalues of the stiffness matrix represented in a vector form, $\delta s_t = s_t^r - s_t$ and $\mathbf{Q}_t$ and $\mathbf{R}_t$ are diagonal gain matrices for task and com-

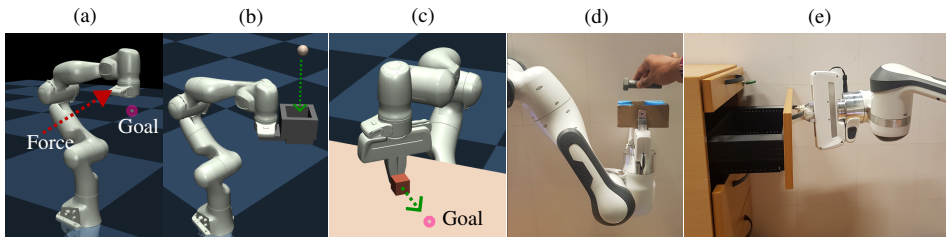


Figure 11.2: Three simulation tasks, (a) Cartesian compliance task: the robot manipulator end-effector should hold its pose in the Cartesian space compliantly while reacting to the external forces acting on it. (b) Reacting to falling object: The robot manipulator with cup end-effector should hold a Cartesian position while smoothly catching a ball of weight 0.5g falling into the cup. (c) Pushing task: A robot manipulator with a gripper end-effector should push an object over a rigid surface with friction to a target position. Two experimental tasks, (d) Reacting to falling objects: robot end-effector is fitted with a tray, where objects of different weights are dropped into the tray at regular intervals. (e) Drawer opening task: Robot manipulator opening a table drawer.

pliance components respectively. These gain matrices can be either constant or can be a function of the robot's states. The MPC output behavior will be tightly coupled with the gain matrices. In case of reference tracking tasks, we chose $\mathbf{Q}_t$ to be a linear function of $\|\delta s_t\|$ so that MPC will penalize larger deviations from target more than small deviations.

11.4 Evaluation

For evaluation, we consider only the stiffness adaptation along the x, y, z directions of the robot manipulator while keeping the stiffness values along orientations constant. However, before evaluation, we first need to learn the Cartesian impedance model of the robot manipulator. To do so, a free-space goal-reaching task with random external force is used to train the PENN model with ensembles of 5 NN with 3 hidden layers, each with 256 neurons. The network structure is chosen based on one-step prediction accuracy empirically over a pre-collected dataset. Its state space is chosen as $s = [x, y, z, \dot{x}, \dot{y}, \dot{z}]$, while the sensed external forces are denoted as $f = [f^x_{ext}, f^y_{ext}, f^z_{ext}]$. $s^r = [x^r, y^r, z^r]$ represents the target positions in x, y and z directions, $\mathbf{K}$ denotes the Cartesian stiffness matrix. The damping matrix is chosen as $\mathbf{D} = 2\sqrt{\mathbf{K}}$. The mass matrix $\mathbf{M}$ is kept constant to avoid stability issues during the experiment. CEM is used to optimize the exploration strategy based on uncertainty maximization. The control frequency for low-level VIC is set at 100Hz.

For learning the model, the robot manipulator is excited at every time-step with $f_{ext} \sim \mathrm{U}(-20, 20)$ N and s^r_t, where $x^r_t, y^r_t, z^r_t \sim \mathrm{U}(-10, 10)$ cm. The gain

matrices $\mathbf{Q}$ and $\mathbf{R}$ are kept constants for a specific task. However, while transferring to a new task, they can be scaled using scalar values α_Q and α_R as $\mathbf{Q_{new}} = \mathbf{Q} * \alpha_Q$ and $\mathbf{R_{new}} = \mathbf{R} * \alpha_R$ respectively to trade-off between compliance and accuracy depending on the task requirement. The model was trained for $100\,000$ time-steps with a control frequency of 10Hz which is equivalent to $2.77\ \text{h}$ of real-world training. For experiments, a prior model trained in simulations over $50\,000$ time-steps is fine-tuned in the experimental scenario instead of training from scratch. The model was fine-tuned for $10\,000$ time-steps which is equivalent to $33.33\ \text{min}$ of real-world training. Similar to in simulations random external forces were manually applied to the robot end-effector using ropes attached to the gripper.

After learning the Cartesian impedance model of the manipulator, to evaluate the effectiveness of the proposed deep MPVIC, three different simulation tasks and two experimental tasks using a Franka Emika Panda manipulator are designed. Tasks requiring real-time stiffness adaptation are suitable for evaluating the stiffness profile generated by the deep MPVIC controller. For all chosen tasks, the requirements can be defined as achieving a desired goal pose for the robot end-effector. However, the robot is also required to be highly compliant whenever it is possible or be stiff only when it is necessary. This is achieved by using a weighted reward in (11.2) for the task requirement (first term) and maximizing compliance (second term). In the considered tasks, we are, essentially, trying for a trade-off between position control and compliance.

The three different simulation tasks are modeled in the MuJoCo physics simulation framework [212], see Fig. 11.2 (a), (b), and (c). The two real experimental scenarios are shown in Fig. 11.2 (d) and (e). The aleatoric uncertainties in these robotic tasks is majorly due to measurement noise, whereas the epistemic uncertainty we target during exploration arises from not having enough data to model the Cartesian impedance dynamics in the system state-space we are interested in. *In simulations*, the population size for CEM is chosen as 200 and elite size of 40 and learning rate of 0.1 and number of CEM iterations as 10. The MPC planning horizon is set to 5. *While for the real experiments*, the control frequency is set as 5Hz. The CEM is chosen as 64 and elite size of 32 and learning rate of 0.5, number of CEM iterations as 5 and MPC planning horizon is set as 5. In all the simulations and experiments the model described here is used without any further fine-tuning. Here we consider only fixed goal states, therefore s_t^r is a constant value, s^r for all timesteps.

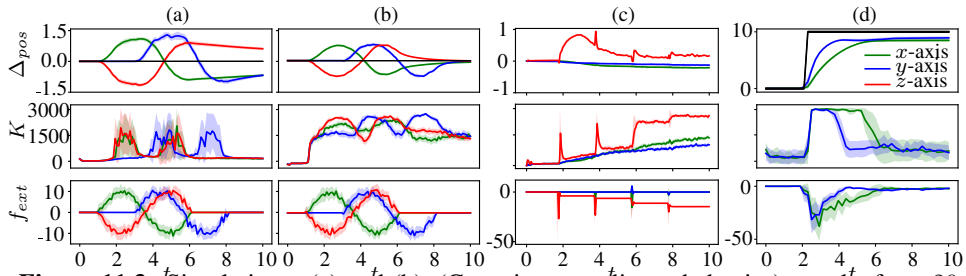


Figure 11.3: Simulations: (a) and (b), (Cartesian compliance behavior), results from 20 trials where a sinusoidal force profile with amplitude of 10 N with random noise of (± 5) N is applied to the robot end-effector. (a) High compliant behavior optimized using a cost function with larger compliance factor $\alpha_R = 0.1$, (b) Low compliant behavior optimized using a cost function with $\alpha_R = 0.01$. (c), (Reacting to falling objects) The robot is initialized at a rest position being very compliant with $K \to 0$. Objects of different weights are dropped at regular intervals of 2s, from random heights between $(0.5-1.0)$ m. Results shown are here are over 10 such random trials with $\alpha_R = 0.1$. (d), (Pushing task) Robot with a gripper end-effector is at rest with $K \to 0$. At $t = 1$ s, it is commanded to push an object to a target position given by Δ_{pos} of 10 cm in x and y directions (shown in solid black line) on a surface. The results shown here are over 10 trials with objects of random weights between $(0.5 - 3.0)$ kg and $\alpha_R = 0.1$.

11.4.1 Simulations

Cartesian compliant behavior: In this task (Fig. 11.2 (a)), the robot is expected to behave highly compliant to hold its pose allowing only small deviations. Upon applying an external force to the robot's end-effector, it is expected to counter the force by adapting its stiffness such that it achieves a new rest position close to the initial position. This task is ideal to test the impedance adaptation strategy as it needs to increase the stiffness in case of large external forces and larger deviation from its initial position. Two scenarios with different compliance behavior are evaluated here by changing the compliance maximization component in the cost function. The results in Fig. 11.3 (a) and (b) show that the robot which is highly compliant at rest adapts the stiffness in response to the external forces and deviation from the rest position. Having a higher value of compliance factor α_R allows for larger deviations from the initial position when applied with an external force while having a lower α_R limits this deviation. It is also noted that higher α_R results in noisy stiffness adaption behavior as larger Δ_{pos} (the deviation from the desired pose) creates larger gradients in the cost function.

Reacting to falling object: In this task (Fig. 11.2 (b)), a robot with a cup end-effector that is highly compliant at rest position is expected to react optimally to objects falling into the cup end-effector. Four different objects are dropped from different heights to the cup in different trials resulting in large variations in the

impact force. The desired behavior of the robot is not to deviate largely from the rest position while reacting to falling objects. The robot is additionally expected to be as compliant as possible and be stiff only when necessary as in (11.2). The resulting robot behavior is shown in Fig. 11.3 (c), which shows a sudden increase in K_z upon a spike in f_{ext} in z direction induced by the impact of the falling object. The robot increases its stiffness every time a new object is falling into the cup and maintains a higher level of stiffness during the later phases to hold the robot back to a new rest position.

Pushing task: In this task (Fig. 11.2 (c)), the robot is expected to push a cube-shaped object to a target position on a surface with friction. Here, K_z is set constant as 1000 as the robot is not expected to move in z direction. Stiffness in x and y directions are optimized to push the object to the target while being compliant and stiff only when necessary. The results in Fig. 11.3 (d) show that the stiffness is increased to its upper limit in the pushing directions initially to overcome the static friction. Upon reaching close to the target position the stiffness is decreased to be more compliant.

11.4.2 Comparison with Model-free/based RL:

The deep MPVIC is compared with RL based VILC approaches for their transferability between tasks which is the main contribution of this work while also comparing their performance. Specifically, in these comparisons, we utilize the PENN model trained with curiosity driven exploration with our deep MPVIC for different tasks without retraining or fine-tuning the model. This enables the deep MPVIC to generalize over multiple tasks where the RL approaches are task-specific.

Model-free RL approaches have been successfully used in VILC for robotic manipulation tasks in multiple previous works [144, 30, 215]. Out of which we have chosen the off-policy RL algorithm SAC because of its high sample efficiency. All the three simulation tasks shown in Fig. 11.2 are trained using SAC implementation from *stable-baselines* [81] for $500\,000$ time-steps.

In addition, we compare our approach with the MBRL approach PETS [51]. In the case of PETS, the simulation tasks are trained for $100\,000$ time-steps. The PETS policies were trained with the same CEM parameters and cost functions used for the corresponding tasks in our deep MPVIC. The performance and the transferability of the learned policies in both of these approaches were compared with our MPVIC approach in Fig. 11.6.

Performance: The resulting robot behavior on applying the learned model-free RL and PETS policies on the three simulation tasks are shown in Fig. 11.4 and Fig. 11.5 respectively. We compare the performance in terms of the reward obtained

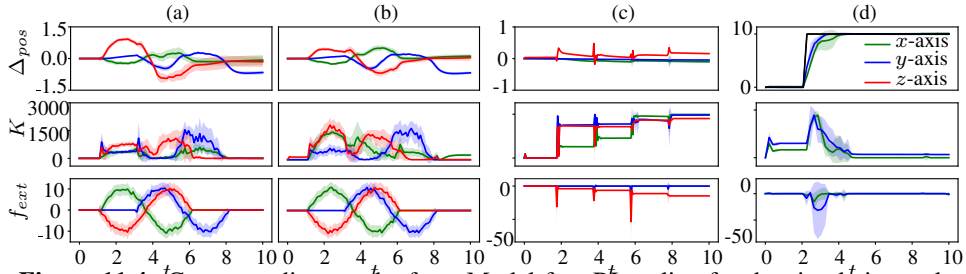


Figure 11.4: Corresponding results from Model-free RL policy for the simulation tasks shown in Fig. 11.3

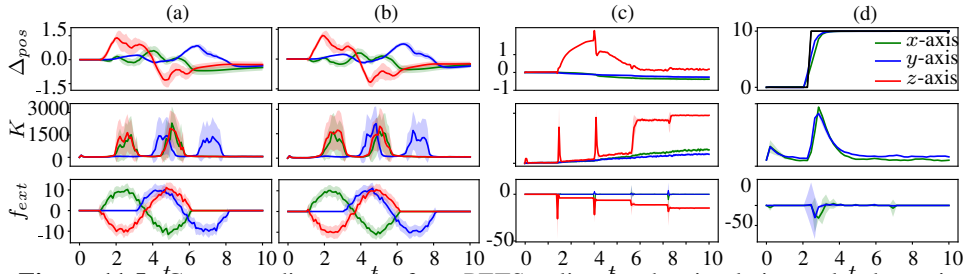


Figure 11.5: Corresponding results from PETS policy for the simulation tasks shown in Fig. 11.3

by the final policies from each of the approaches on the three simulation tasks. We don't compare the reward during the learning process as the MPC scheme does not have any policy learning process. The rewards obtained while applying the learned policies are shown in Fig. 11.6. Deep MPVIC performed better on *task a*, the performance was similar on *task b* and model-free RL and PETS policies performed better on the *task c* by minimizing the stiffness more effectively. The performance of MPVIC is lower in *task c* as the model is learned on task a which has different dynamics. The key difference between the dynamics of *task a* and *task c* is the robot movement is unconstrained in *task a*, whereas in *task c* it is constrained by the object. That means the model can not accurately predict the dynamics as in other unconstrained tasks.

Task transferability: In order to evaluate how efficiently the policy learned on a task can be transferred to another task, the model-free RL and PETS policies learned on the simulation *task a* was tested on *task b* and *task c* without retraining the policy/model. The performance of the transferred model-free RL and PETS policies on *task b* and *c* were compared with the corresponding performance of deep MPVIC using the PENN model trained on *task a*. Figure 11.6-*right* illustrates the transferability of our deep MPVIC in comparison with RL-based approaches, where deep MPVIC demonstrates the major advantage (green bars). Further, the

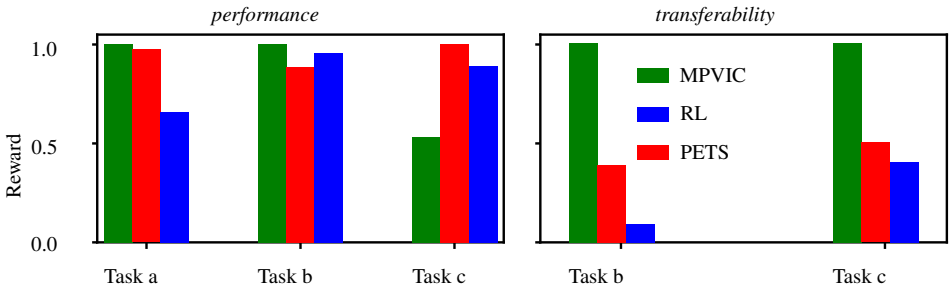


Figure 11.6: (left) Comparing the normalized value of the reward (mean value over 20 trials) obtained using Model-free RL, PETS, and our MPVIC framework on all the three simulation tasks. (right) Comparing the transferability of the Model-free RL and PETS based policy with our MPVIC framework based on the normalized value of the mean reward over 20 trials.

model-free RL and PETS policies have been retrained to achieve similar performance as our deep MPVIC. A comparison of the additional *data samples/time steps* required for retraining the models/policies for the tasks are shown in Table 11.2. For example, the model-free RL policy trained on *task a* needed additional training on *task b* with 38.6×10^5 data samples to learn the task b. Whereas MPVIC did not need any training at all (shown as 0 training samples in the table). The number of additional training samples required is correlated with the computational time. While RL approaches demand additional computational/training time to perform a new task, the proposed deep MPVIC can be deployed without any additional computational effort.

Table 11.2: Comparison on transferability between tasks

	Training samples ($\times 10^5$)		
	Transferability to		
	Task a	*Task b*	*Task c*
Model-free RL	50	38.6	27.95
PETS	10	3.2	3.9
Our MPVIC	10	0	0

11.4.3 Real-world Experiments

Reacting to falling objects: The experimental setup is shown in Fig. 11.2 (d) where the robot end-effector is fitted with a tray and four objects of different weights are added to the tray at regular intervals. The optimization objective here is similar to the simulation task (c), the robot is expected to hold a pose while being highly compliant and becoming stiffer with extra weights being introduced

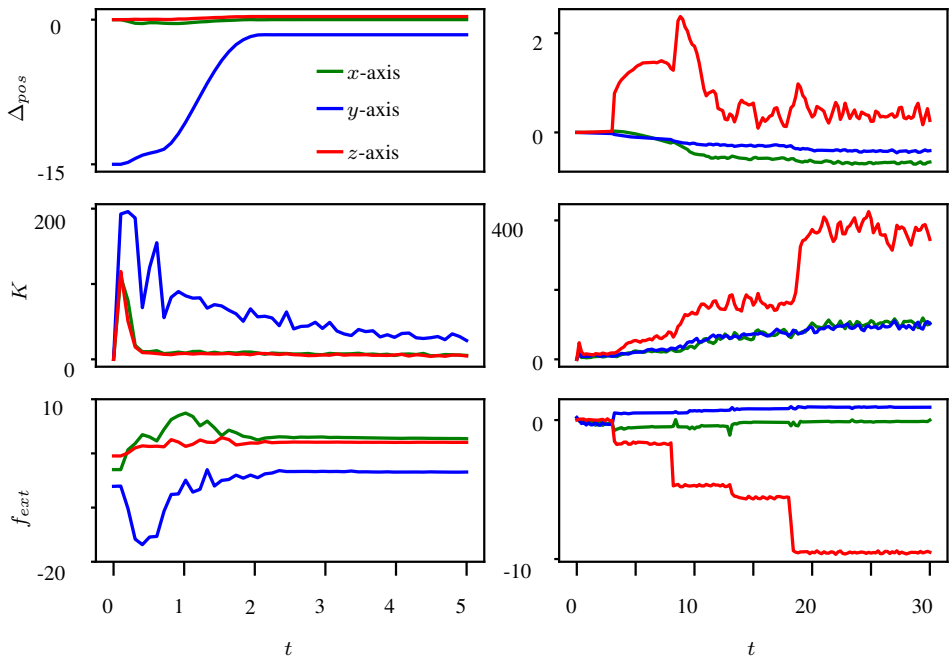


Figure 11.7: Experiments: (left) task (e), Robot manipulator opening a table drawer. (right) task (d) The robot manipulator with a tray holding its pose while objects are dropped to the tray.

to the tray. In Fig. 11.7 (right-column) the robot with a very low initial stiffness increases the stiffness every instant a new object is introduced to the tray in order to maintain it at the desired pose.

Opening a drawer: The pulling task is similar but in the opposite direction of the pushing task. The experimental setup is shown in Fig. 11.2 (e) where the robot is opening a table drawer to a desired position ($15\,\mathrm{cm}$ in x direction) in the Cartesian space. The results shown in Fig. 11.7 (left-column), show the impedance adaptation behavior similar to the pushing task in the simulation where the robot increases its stiffness initially to overcome the inertia of the drawer and then decreased once the drawer starts to move closer to the desired position.

11.5 Discussion

11.5.1 Variable Impedance Learning Control

The deep MPVIC-based approach presented in the work is evaluated over different tasks in Section 11.4 for optimizing impedance adaptation strategies. The object-

ive in all experiments has been consistent in having high stiffness values for the VIC only when the task objective demands that. This objective is motivated by human manipulation behavior and can increase the dexterity of the robot while encouraging energy-efficient and safe behaviors. We considered three simulations and two experimental tasks for evaluating the proposed method. In all the tasks, the task requirement is defined by achieving a desired goal pose for the robot end-effector. The performance of the impedance adaptation strategy is evaluated based on how well it is able to achieve this requirement while being maximally compliant. In all the evaluation scenarios, both in simulation and experiments, the stiffness adaptation guarantees a high level of compliance unless there is a large deviation from the target position or an external force is applied to it. The deep MPVIC scheme is able to adapt the impedance profiles to counteract the external forces and also to trade-off effectively between position accuracy and compliance during the task.

The modeling approach using PENN combined with uncertainty-targeted exploration has been found to be very useful in learning a generalized unconstrained Cartesian impedance model of the robot. In addition, combining it with MPC based optimization has enabled to solve different manipulation tasks demanding stiffness adaptation. The proposed deep MPVIC approach succeeds in generalizing a single model to solve multiple manipulation tasks. The versatility of the impedance adaptation strategy is evident in the scenarios of impact force from falling objects, overcoming the inertia of the objects in the pushing and drawer opening tasks respectively. While a majority of robot manipulation tasks rely on trajectory planning and tracking, our approach is not straightforward in solving complex manipulation problems. Nevertheless, it can be combined with a high-level planning approach where the low-level VIC will modify the given trajectory to ensure compliant behavior. Incorporating such compliant behaviors could improve manipulation skills, especially in tasks involving contacts.

The deep MPVIC framework was compared with model-free and model-based RL approaches utilized successfully in various previous works [144, 30, 215, 179] to solve complex manipulation tasks. The results show that the deep MPVIC framework is able to achieve similar performance to model-free and model-based RL approaches while being highly sample efficient and able to seamlessly transfer the controller between different tasks without any further training of the model. Whereas in model-free and model-based RL, transferring policy between different tasks demand relearning the policy on the new task or extensive fine-tuning of the existing policy. PETS shows better task transferability compared to model-free RL, this can be justified by the use of a model in PETS for impedance optimization even though it is not a generalized model as in deep MPVIC. It is important

to note that the transferability of deep MPVIC is dependent on the quality of the model. This is evident in its lower performance in the *task c* where the model can not accurately predict the dynamics of a constrained task. RL has the potential to solve very complex tasks at the expense of high sample complexity. It would be ideal to combine this aspect of RL with sample efficiency and easy transferability of the learned controller between tasks as in our deep MPVIC framework. Further extending the model-based RL approaches for VILC could be a promising approach in this direction.

11.5.2 Stability Analysis

MBRL approaches could improve sample efficiency and can be useful in providing stability and safety guarantees, but there is a need for further research in this direction facilitating complex model structures such as DNN to build scalable and sample efficient VILC approaches with theoretical guarantees. In general, the stability of a dynamic system is not necessarily guaranteed when it is coupled to a stable dynamic environment. However, [85] showed stability of the manipulator is preserved when it is coupled to a large class of stable environments if the manipulator has the behavior of a simple impedance. An impedance controller with constant gains makes the closed-loop robot-environment system passive and hence stable in interaction with passive environments [85]. However, this passivity property is lost if the impedance parameters are varied. If the learning-based controller could identify the optimal impedance parameters, one could achieve complex compliant manipulation skills with safety and stability guarantees. But this is not obvious while using RL or CEM-based MPC. One alternative in the case of RL is to use structured policies as done by [101] where the authors use Integrated MOtion Generator and Impedance Controller (iMOGIC) framework to guarantee stable VILC with model-free RL. Even though safety can be achieved using constrained-CEM method [221] in the proposed MPVIC framework, stability guarantees are difficult due to the PENN dynamical models.

Guaranteeing stability and robustness for controllers in a complex robotic manipulator operating in uncertain environments is challenging. In the case of VIC, often passivity theory is used to provide theoretical guarantees under relatively general working assumptions. However, this approach is model-based and the passivity property is lost if arbitrary variations of the impedance parameters are allowed. Passivity-based approaches are often concerned with the analysis of variable impedance profiles that already exist prior to task execution [128]. This is not suitable for guaranteeing the stability of state-dependent real-time impedance variations. In another recent approach, a modified impedance control strategy allows the reproduction of a variable stiffness while preserving the passivity, and therefore a stable behavior both in free motion and in interaction with partially

known environments, of the robot [59]. In [59], the goal is to modify the impedance control in order to allow stiffness variations while preserving passivity and, consequently, stable interactive behavior and asymptotic tracking in free motion. This tank-based strategy has been shown very well suited for VIC, in spite of some difficulties to tune its parameters. Nevertheless, it is dependent on the states of the system, measured during task execution and so can only be applied online. An approach based on the combination of passivity conditions with an adaptation law on the impedance profile was proposed in [18]. This method allows for verifying whether a given profile is passive and if it is not, it provides a method to modify it in a way to guarantee passivity. But none of these approaches are directly extendable to the proposed MPVIC framework to guarantee stability. Ref. [97] proposed an approach using a designed Lyapunov candidate function to stabilize the learned impedance system with an optimal input law in analytical form. But in the case of our MPVIC, this requires solving an additional convex optimization problem at every MPC solution, which could be computationally very expensive and not feasible practically.

The safe learning approaches described in [8] are interesting to explore for model-based VILC. A feasible approach in this direction could be to provide probabilistic safety and stability guarantees using CBF and CLF and solving constrained optimization problems over the GP model [107]. Guaranteeing stability properties to the resulting VILC is challenging as guarantees have to be provided in real-time in an online fashion as the stiffness values predicted by the policy are state-dependent. The approach proposed in [97] by designing a quadratic Lyapunov candidate function could be coupled with GP models to provide probabilistic stability guarantees similar to safety guarantees in [107]. But in the proposed MPVIC framework with PENN dynamics model, such methods are not straightforward to apply. This problem in CEM-based MPC is partly addressed in [232] using available prior models and an auxiliary controller based on CLF and CBF to provide guarantees.

11.5.3 Limitations

While guaranteeing stability is one major challenge, there are other identified limitations to the proposed approach. Applying our approach to tasks with non-continuous contacts is not possible as the model is not aware of the contact dynamics, which could lead to unstable behavior. Detecting contact discontinuities and switching to a different contact re-establish policy could be a solution to this issue. Whereas a more general approach could be to learn a model aware of contact constraints, incorporating such constraints into the model state-space is challenging. In future work, we will explore ways to sufficiently incorporate contact constraints to the model to aid faster fine-tuning of the VILC for different manipulation tasks. In addition, there are limitations inherited from applying CEM to

a real robotic system because of the high computation time, where the trade-off is between optimization performance and the control frequency. Eventhough VIC can be operated generally at lower control frequencies, in tasks with complex contact dynamics this might not be sufficient. The level of impedance adaptation or the compliance behavior can be adjusted by tuning the Q and R parameters in the cost function (11.2). However, it is not obvious how to find optimal values for these parameters.

11.6 Conclusions

In this chapter, we presented a deep MPVIC approach for compliant manipulation skills for a robotic manipulator by optimizing the impedance parameters. By utilizing PENN, a Cartesian impedance model of the robot is learned using an exploration strategy maximizing the information gain. The PENN dynamic model is coupled with a CEM-based MPC to optimize impedance parameters of a low-level VIC. We identified an impedance optimization objective-based human manipulation skill and replicated it on a robot manipulator for simplified scenarios in simulations and experiments. The deep MPVIC was compared with model-free and model-based RL approaches in VILC. The approach proved experimentally to be beneficial for solving multiple tasks without any need to relearn the model or policy as opposed to other VILC approaches. In the future work, we aim to extend this approach to scenarios with constraints, such as in-contact interaction tasks.

Chapter 12

Conclusions and Future Work

12.1 Conclusions

This thesis is a presented collection of new research works in the area of RL and robotic manipulation. This core research topic addressed in this thesis is *exploring the prospects of machine learning, specifically RL for developing compliant-control methods control for robotic manipulation*. This research is motivated by the possible positive benefits of safe and reliable robotic solutions in the area of healthcare, rehabilitation, elderly care, and in industrial applications. The results of this research over the last three and half years were presented in two parts in this thesis. Part I presented the research works on RL for robotic control with a special focus on MBRL. The main contribution of Part I was on developing RL methods that can be applied to real-world systems and discussing the safety and stability guarantees of learning-based methods. Part II presented the research works on robotic manipulation with a special focus on compliant robotic manipulation using VILC approaches. The contribution of the second part is on developing and evaluating VILC methods for robotic manipulation while additionally exploring the planning component of robotic manipulation.

The first part of the thesis focused on developing MBRL method to advance the research on RL for real-world applications. Chapter 3 presented an approach to improve the sample efficiency and performance of the MBRL using uncertainty-targeted exploration and incorporating a critic-value estimate from the data. We showed that an MBRL algorithm that implements this approach could optimize a policy with better asymptotic performance compared to existing methods. A sampling-based MPC was adopted as the policy in the proposed MBRL framework. Further, combining a conventional combining MPC with RL was identified

as a fruitful MBRL approach. In order to facilitate the easy application of RLMPC methods a novel actor-critic DPG algorithm for RLMPC is presented in Chapter 4. This algorithm facilitates easier use of DPG in RLMPC by eliminating the need for an additional value function parameterization such as DNNs by approximating the value function using the MPC policy itself.

The second part of the thesis emphasizes on VILC for enabling compliant control in robotic manipulation. RL has been explored for its suitability to develop VILC for complex manipulation tasks. The importance of RL in VIC is well evident from the increasing research interest in the area of RL-based VILC. A thorough review of the existing RL-based VILC approaches are presented in Chapter 7. Focusing on real-world manipulation tasks the compliant-control methods presented in this thesis relies on MBRL. Chapter 9 presented the implementation and evaluation of two fundamental compliant-control approaches for robotics, (i) HFMC and (ii) force-based VIC in a MBRL framework using GP dynamic models. It was shown that combining a learning-based approach with these compliant controllers has the ability to improve robotic interaction control. Chapter 10 presented a highly data-efficient MBRL approach for VILC using GP dynamic models and CMA-ES based policy optimization. This approach was evaluated on different simple real-world tasks demanding real-time stiffness adaptation. Even though GP provides an efficient way to approximate system dynamics, they pose limitations on scalability and computational effort. Adopting complex models and policy are key in developing model-based VILC for complex manipulation tasks. Chapter 11 presented an approach in this direction termed as deep MPVIC. The deep-MPVIC framework used the PENN for learning the system dynamics and a CEM-based MPC to optimize impedance parameters of a low-level VIC. We identified an impedance optimization objective-based human manipulation skill and replicated it on a robot manipulator for simplified scenarios in simulations and experiments. One key contribution of this approach is its transferability over multiple tasks without any need to relearn the model or policy as opposed to other VILC approaches.

Additionally, this thesis explored the aspect of safety guarantees in learning-based control methods and the aspect of planning in robotic manipulation. Safety guarantees are critical to learning-based control methods for real-world applications. A detailed review of existing learning-based methods for safe control of dynamical systems with uncertainty, using tools from control theory such as CBF and CLF are presented in Chapter 5. This review could serve as a useful document to choose a suitable method for specific applications. A combination of trajectory planning and control is central to achieving dexterous robotic manipulation. Even though this thesis was majorly focused on the control aspect, a learning-based trajectory planning approach for robotic manipulation for uncertain environments is presen-

ted in Chapter 8. This work extended the DMP framework with real-time control over the execution time while preserving the shape characteristics.

12.2 Future Works

The future directions of this research are multifaceted along RL and compliant robotic control. There is a wider scope of future research in both parts of this thesis, but one obvious future direction is on applying the research presented in Part I to VILC approaches discussed in Part II. The MBRL framework presented in Chapter 3 can be extended to targeted exploration into VILC for more complex manipulation tasks including contact-rich tasks. RL for real-world robotic demands a greater extent of future research. We have identified a few interesting directions in MBRL which was a key component of this thesis. One main drawback for MBRL we addressed in Chapter 3 is the model-bias, as the learned policy is biased to an incorrect model. As it is highly challenging to model complex dynamics accurately one interesting direction would be to develop methods that can find an optimal policy using an imperfect model. We aim to incorporate the RLMPC approaches to MBRL with DNN based policies by shaping the model and reward functions such that it can provide optimality even with imperfect models. RLMPC as an MBRL framework has great scope of real-world problems that can be tackled using MPC. The RLMPC algorithm proposed in Chapter 4 can be extended to a full critic approximation using MPC and thereby eliminating the need for a critic in DPG method for MPC completely. VILC is relatively a new area of research and our contributions to it in this thesis are rather basic. Future work on Part II of this thesis is to develop a model-based VILC framework for a wider range of manipulation tasks including contact-rich tasks with safety guarantees. Task-specific CBF functions can be integrated into such a VILC framework to guarantee safety, but this demands further research in holistically combining them to meet the real-time task requirements. Similarly, RLMPC approaches can be applied to robot control ideally to non-contact tasks where MPC can be very effective.

12.3 Summary

In summary, this thesis is a part of the research developments in the area of RL and robotic manipulation. We explored MBRL approaches to facilitate the use of RL approaches for real-world robotic control tasks. This includes combining conventional control approaches with reinforcement learning to develop novel model-based reinforcement learning methods. This thesis contributed to advancing the research on compliant control for robotic manipulation by developing and evaluating different model-based VILC approaches for real-world applications. Additionally,

this thesis address planning in robotic manipulation in the context of an uncertain environment and guarantees safety and stability in learning-based approaches relevant to robot control. The results of this have provided promising directions and insights into the area of RL and robotic manipulation in general while specifying clear future directions. With the help of further research efforts, this research is expected to contribute towards developing reliable RL methods for robotic systems and incorporating compliant control skills in those robotic systems.

Appendix A

Reinforcement Learning for MPC: Fundamentals and Current Challenges

Arash Bahari Kordabad * **Dirk Reinhardt** * **Akhil S Anand** * **Sebastien Gros** *

* *Department of Cybernetics, Norwegian University of Science and Technology (NTNU), Norway. E-mail: sebastien.gros@ntnu.no*

Abstract: Recent publications have laid a solid theoretical foundation for the combination of Reinforcement Learning and Model Predictive Control, in view of obtaining high-performance data-driven MPC policies. Early practical results, both in simulation and in experiments, have shown the potential of this combination but have also revealed certain challenges. In addition, the technical complexity of these results makes it difficult for interested readers to gather the fundamental ideas and principles behind this combination. This paper aims to provide a coherent and more accessible picture of these results and to offer significantly deeper and more mature insights into their meaning than has been proposed before. It also aims at identifying the current challenges in the field.

Keywords: MPC, Reinforcement Learning, Learning for MPC, Stability & Safety

1. INTRODUCTION

Model Predictive Control (MPC) is a successful control strategy that employs a (possibly inaccurate) model of the real system dynamics to generate input sequences that minimize a certain cost, possibly under some constraints (Rawlings et al., 2017). The MPC problem is solved at every time instant, in a receding-horizon fashion, delivering a policy for the real system. For many applications, building a model that accurately captures the real system dynamics can be challenging, especially for stochastic systems. In such applications, the performance of the MPC scheme can degrade significantly due to model inaccuracies. This effect is arguably more pronounced if the objective of the MPC scheme is not to bring the real system to a specific reference state (a.k.a. tracking objective) but rather to minimize a generic cost (a.k.a. economic objective).

Recent research focuses on alleviating this issue by integrating Machine Learning (ML) techniques in MPC. The classic approach is to use ML to improve the accuracy of the MPC models in a data-driven fashion (Wu et al., 2019). While this paradigm has a clear value, it does not eliminate the issues related to model inaccuracies. Indeed, the performance of a policy delivered by an MPC scheme integrating an ML-based model is still only as good as the ML-based model is, and therefore limited by the structure and choices made in the ML tools. Similar problems occur in model-based Reinforcement Learning, where the performance of the algorithms is limited by the accuracy of the model predictions. For many applications, a higher model accuracy can only be achieved through a higher model complexity. Because complex (e.g., nonlinear and stochastic) MPC models tend to yield complex MPC schemes, the ML-based MPC paradigm tends to bind the MPC performance to its complexity.

A core issue with model learning in MPC is that the model accuracy is not easily related to the MPC closed-loop performance when based on that model. In fact, an ML-based model is typically constructed to deliver the best possible prediction accuracy in the hope that this will yield a good MPC performance. But the construction of the ML-based model is not directly tied to the resulting closed-loop performance of the MPC.

The idea of optimizing the closed-loop performance of an MPC-based policy by tuning the *entire* MPC scheme (model, cost, and constraints) has been introduced formally in Gros and Zanon (2019a). Reinforcement Learning (RL) is then proposed to optimize the closed-loop performance of the MPC scheme using data from the real system. This approach is arguably unique in the field of learning-based MPC, as it does not focus purely on improving the MPC model accuracy but rather on improving the MPC closed-loop performance. Moreover, it ties the learning directly to the closed-loop performance of the resulting MPC policy.

From an RL perspective, this approach can be seen as a doorway to developing safe, stable, and explainable RL methods, which is not possible in classical Deep RL, where Deep Neural Networks (DNNs) are employed to generate the policies. In the RL context, using MPC as an approximation of the optimal policy can be construed as a way of introducing a strong structure in RL, as opposed to the more classical way of approximating the optimal policy via DNNs. A different perspective is to see RL as a rich toolbox for adjusting the MPC scheme from data to improve its closed-loop performance.

Note that RL is not the only way to directly optimize the MPC closed-loop performance. For instance, in Sorourifar et al. (2021), the authors have proposed the use of Bayesian optimization for black-box systems in order to achieve

the best closed-loop performance. However, Bayesian optimization approaches are difficult to use on large parameter spaces. Moreover, black-box approaches are adopted at the cost of abandoning the explainability provided by MPC and the possibility of embedding prior knowledge in the MPC scheme.

While the combination of RL and MPC has been investigated both in theory and in practice in dozens of papers, the concepts underlying this combination are not trivial and are prone to raise fundamental and practical questions. This paper aims to provide a coherent and deeper discussion on this combination than has been provided so far in the literature. The paper is structured as follows. Section 2 provides the necessary background. Section 3 presents the fundamental theoretical results on RL and MPC and insights into their consequences. Section 4 provides a discussion on RL methods for MPC, and the associated challenges. Section 5 and 6 discuss the use of RL for MPC with stability and safety requirements.

2. BACKGROUND

In this section, we provide a background on Markov Decision Processes (MDP), Reinforcement Learning (RL), and Model Predictive Control (MPC).

2.1 Markov Decision Processes

Markov Decision Processes (MDP) provide a fairly generic framework for the class of problems at the center of MPC. An MDP operates over given state and action (a.k.a. input) spaces $\mathcal{S}$, $\mathcal{A}$, respectively (Puterman, 2014). These spaces can be discrete (i.e. integer sets), continuous, or mixed. An MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, L, \gamma, \rho)$, where L is a stage cost, $\gamma \in (0, 1]$ a discount factor, and ρ a conditional probability (measure) defining the dynamics of the system considered, i.e. for a given state-action pair $\mathbf{s}$, $\mathbf{a} \in \mathcal{S} \times \mathcal{A}$, the successive state $\mathbf{s}_+$ is distributed according to

$$\mathbf{s}_+ \sim \rho(\cdot|\mathbf{s}, \mathbf{a}) . \tag{1}$$

Note that (1) is a generalization of the deterministic or stochastic dynamics model often considered in MPC, usually cast as

$$\mathbf{s}_+ = \mathbf{F}(\mathbf{s}, \mathbf{a}, \mathbf{w}) , \quad \mathbf{w} \sim W , \tag{2}$$

where $\mathbf{w}$ is a random disturbance from distribution W. In the special case $\mathbf{w} = 0$, (2) simply yields deterministic dynamics. An MDP is then the problem of finding the optimal policy $\boldsymbol{\pi}^\star : \mathcal{S} \to \mathcal{A}$ as the solution of

$$\boldsymbol{\pi}^\star = \arg\min_{\boldsymbol{\pi}} J(\boldsymbol{\pi}) , \quad \text{where} \tag{3a}$$

$$J(\boldsymbol{\pi}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{s}_k, \mathbf{a}_k) \,\middle|\, \mathbf{a}_k = \boldsymbol{\pi}(\mathbf{s}_k) \right] , \tag{3b}$$

and the expected value operator $\mathbb{E}[\cdot]$ is taken over the closed-loop trajectories of the system resulting from ρ and $\boldsymbol{\pi}$. Discussing the solution of MDPs is often best done via the Bellman equations defining implicitly the optimal value function $V^\star : \mathcal{S} \to \mathbb{R}$ and the optimal action-value function $Q^\star : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as

$$V^\star(\mathbf{s}) = \min_{\mathbf{a}} Q^\star(\mathbf{s}, \mathbf{a}) , \tag{4a}$$

$$Q^\star(\mathbf{s}, \mathbf{a}) = L(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}\left[V^\star(\mathbf{s}_+) \,|\, \mathbf{s}, \mathbf{a} \right] . \tag{4b}$$

The optimal policy then reads as

$$\boldsymbol{\pi}^\star(\mathbf{s}) = \arg\min_{\mathbf{a}} Q^\star(\mathbf{s}, \mathbf{a}) . \tag{5}$$

RL offers a set of methods that attempt to compute $\boldsymbol{\pi}^\star$ directly based on data on the real system. We provide next a very succinct introduction to RL.

2.2 Reinforcement Learning

The fundamental goal of RL is to use data to deliver an approximation of the optimal policy $\boldsymbol{\pi}^\star$ without the need to model the transition dynamics (1) (Sutton and Barto, 2018). The field can be coarsely divided into two large classes of approaches. The first class, generically labeled Q-learning, approximates the optimal action-value function $Q^\star$ via a parametrized function approximator $Q_{\boldsymbol{\theta}}$. An approximation of the optimal policy $\boldsymbol{\pi}^\star$ can then be obtained using

$$\hat{\boldsymbol{\pi}}^\star(\mathbf{s}) = \arg\min_{\mathbf{a}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) . \tag{6}$$

The second class approximates $\boldsymbol{\pi}^\star$ directly via a parameterized policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ and adjusts the parameters $\boldsymbol{\theta}$ to minimize $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$. This can either be done via estimating policy gradients $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$, or by building surrogate models of $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$.

2.3 Model Predictive Control

MPC produces control policies by solving an optimal control problem at each discrete time instant based on the current system state $\mathbf{s}$, on a finite, receding horizon. The problem is often cast as

$$\min_{\mathbf{x},\mathbf{u}} \quad T(\mathbf{x}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k, \mathbf{u}_k) , \tag{7a}$$

$$\text{s.t.} \quad \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) , \quad \mathbf{x}_0 = \mathbf{s} , \tag{7b}$$

$$\mathbf{h}(\mathbf{x}_k, \mathbf{u}_k) \le 0 , \quad \mathbf{u}_k \in \mathcal{A} , \tag{7c}$$

where N is the prediction horizon, L is the stage cost, T is the terminal cost, $\mathbf{f}$ is the dynamics and $\mathbf{h}$ is the inequality constraint. Problem (7) produces a complete profile of control inputs $\mathbf{u}^\star = \{\mathbf{u}_0^\star, \ldots, \mathbf{u}_{N-1}^\star\}$ and corresponding state predictions $\mathbf{x}^\star = \{\mathbf{x}_0^\star, \ldots, \mathbf{x}_N^\star\}$. However, only the first element $\mathbf{u}_0^\star$ of the input sequence $\mathbf{u}^\star$ is applied to the system (Rawlings et al., 2017). At the next physical sampling time, a new state $\mathbf{s}$ is received, and problem (7) is solved again, producing a new $\mathbf{u}^\star$ and a new $\mathbf{u}_0^\star$. MPC (7) hence yields a policy

$$\boldsymbol{\pi}_{\text{MPC}}(\mathbf{s}) = \mathbf{u}_0^\star , \tag{8}$$

with $\mathbf{u}_0^\star$ solution of (7) for $\mathbf{s}$ given. For $\gamma \approx 1$, policy (8) can provide a good approximation of the optimal policy $\boldsymbol{\pi}^\star$ for an adequate choice of prediction horizon N, terminal cost T and if the MPC model $\mathbf{f}$ approximates the true dynamics (1) sufficiently well. In that context, the latter is arguably the major weakness as many systems are challenging to model accurately. Furthermore, within a modeling structure, selecting the model $\mathbf{f}$ that yields the best closed-loop performance $J(\boldsymbol{\pi}_{\text{MPC}})$ is very difficult. And there is in general no guarantee that the most accurate model yields the best closed-loop performance.

3. FUNDAMENTALS OF RL AND MPC

The combination of RL and MPC focuses on the MPC closed-loop performance directly, unlike more classical

ML-based MPC approaches focusing on fitting the MPC model to the data. Furthermore, it has been recently established formally in the literature that–when focusing on closed-loop performance–the MPC scheme can be adjusted in a *holistic* fashion, allowing one to produce optimal policies without relying on a highly accurate model of the real system. In this section, we provide the central result supporting that statement. To that end, it is useful to construe MPC as a (possibly local) model of the action-value function $Q^\star$. Indeed, consider an MPC-based policy

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}) = \mathbf{u}_0^\star\,, \tag{9}$$

where $\mathbf{u}_0^\star$ is part of the solution of:

$$\mathbf{x}^\star, \mathbf{u}^\star = \arg\min_{\mathbf{x},\mathbf{u}} \quad T_{\boldsymbol{\theta}}\left(\mathbf{x}_N\right) + \sum_{k=0}^{N-1} L_{\boldsymbol{\theta}}\left(\mathbf{x}_k, \mathbf{u}_k\right), \tag{10a}$$

$$\text{s.t.} \quad \mathbf{x}_{k+1} = \mathbf{f}_{\boldsymbol{\theta}}\left(\mathbf{x}_k, \mathbf{u}_k\right), \quad \mathbf{x}_0 = \mathbf{s}\,, \tag{10b}$$

$$\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{x}_k, \mathbf{u}_k\right) \leq 0, \quad \mathbf{u}_k \in \mathcal{A}\,. \tag{10c}$$

This MPC formulation is identical to (7), but the cost, constraints, and dynamics underlying the MPC scheme are now all parameterized in $\boldsymbol{\theta}$, to the exception of the input constraint $\mathbf{u}_k \in \mathcal{A}$. Besides the MPC-based policy (9), we can define a parameterized action-value function $Q_{\boldsymbol{\theta}}$ based on the MPC scheme (10), as a model of the optimal action-value function $Q^\star$ as follows:

$$Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) = \min_{\mathbf{x},\mathbf{u}} \quad \text{(10a)}, \tag{11a}$$

$$\text{s.t.} \quad \text{(10b)} - \text{(10c)}, \quad \mathbf{u}_0 = \mathbf{a}\,, \tag{11b}$$

where a constraint $\mathbf{u}_0 = \mathbf{a}$ included in (11b) is the only difference to (10). MPC (11) is a valid model of $Q^\star$ in the sense that it satisfies the relationships (4) and (5), i.e.:

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}\right) = \arg\min_{\mathbf{a}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}), \quad V_{\boldsymbol{\theta}}(\mathbf{s}) = \min_{\mathbf{a}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a})\,, \tag{12}$$

where $V_{\boldsymbol{\theta}}(\mathbf{s})$ is the optimal cost resulting from solving MPC (10). One can verify that if the MPC parameters $\boldsymbol{\theta}$ are such that $Q_{\boldsymbol{\theta}} = Q^\star$, then MPC scheme (10) delivers the optimal policy $\boldsymbol{\pi}^\star$ through (9), i.e. $\boldsymbol{\pi}_{\boldsymbol{\theta}} = \boldsymbol{\pi}^\star$. An important question, then, is how effectively an MPC scheme can approximate $Q^\star$ at least in a neighborhood of $\mathbf{a} = \boldsymbol{\pi}^\star(\mathbf{s})$. The main concern here is arguably the MPC model $\mathbf{f}_{\boldsymbol{\theta}}$ for the reasons already raised in Sec. 2.3. In addition, $Q^\star$ is typically built from a discounted sum of the stage costs L, while undiscounted MPC formulations are typically preferred.

The Theorem reported below addresses these concerns and provides the central justification for considering the MPC parametrization (10) for learning-based MPC. It establishes that under some mild conditions, (11) is able to provide an exact model of $Q^\star$ even if the predictive model (10b) is inaccurate. This in turn entails that MPC (10) can achieve optimal closed-loop performance even if the MPC model is inaccurate.

Theorem 1. *Suppose that the parameterized stage cost, terminal cost, and constraints in* (10) *are universal function approximations (i.e., can approximate a given function accurately) with adjustable parameters $\boldsymbol{\theta}$. Then there exist parameters $\boldsymbol{\theta}^\star$ s.t. the following identities hold $\forall\gamma$:*

(1) $V_{\boldsymbol{\theta}^\star}(\mathbf{s}) = V^\star(\mathbf{s}),\ \forall \mathbf{s} \in \Omega$
(2) $\boldsymbol{\pi}_{\boldsymbol{\theta}^\star}(\mathbf{s}) = \boldsymbol{\pi}^\star(\mathbf{s}),\ \forall \mathbf{s} \in \Omega$
(3) $Q_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}) = Q^\star(\mathbf{s}, \mathbf{a}),\ \forall \mathbf{s} \in \Omega$, *for the inputs* $\mathbf{a} \in \mathcal{A}$ *such that* $|V^\star(\mathbf{f}_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}))| < \infty$

if the set

$$\Omega =: \left\{ \mathbf{s} \in \mathcal{S} \;\middle|\; |V^\star(\mathbf{x}_k^\star)| < \infty,\ \forall\, k \leq N \right\}, \tag{13}$$

is non-empty, where $\mathbf{x}_k^\star$ is the optimal state solution of (10).

Proof. We select the parameters such that the following holds:

$$T_{\boldsymbol{\theta}^\star}(\mathbf{s}) = V^\star(\mathbf{s})\,, \tag{14a}$$

$$L_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}) = \tag{14b}$$

$$\begin{cases} Q^\star(\mathbf{s}, \mathbf{a}) - V^\star(\mathbf{f}_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a})) & \text{if } |V^\star(\mathbf{f}_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}))| < \infty \\ \infty & \text{otherwise} \end{cases}$$

The proof then follows from Gros and Zanon (2019a); Kordabad et al. (2022b). Note that infinite values in the stage cost $L_{\boldsymbol{\theta}^\star}$ are, in practice, best treated through the constraints (10c), i.e. $\boldsymbol{\theta}^\star$ can be selected such that

$$\mathbf{h}_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}) > 0 \quad \text{if} \quad |V^\star(\mathbf{f}_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}))| = \infty\,, \tag{15}$$

such that stage cost $L_{\boldsymbol{\theta}^\star}$ does not need to carry infinite penalties. ■

Theorem 1 states that, for a given MDP, an MPC scheme with an inaccurate model can deliver the optimal value functions and the optimal policy of the original MDP. This can be achieved by selecting the appropriate stage cost, terminal cost, and constraints. Theorem 1 is very broad and extends, e.g., to robust MPC, stochastic MPC, mixed-integer MPC, and Economic MPC (EMPC), and is valid both for the discounted and undiscounted settings (see e.g., Zanon and Gros (2020); Kordabad et al. (2022a); Gros and Zanon (2020)). Assumption (13) can be interpreted as a form of stability condition on $\mathbf{f}_{\boldsymbol{\theta}^\star}$ under the optimal trajectory $\mathbf{x}^\star$. More specifically, this assumption requires the existence of a non-empty set such that the optimal value function $V^\star$ of the predicted optimal trajectories $\mathbf{x}^\star$ on the system model is finite for all initial states starting from this set. The assumption is thus milder than requiring stability of the MPC scheme.

3.1 Role of RL in Learning-based MPC

Many recent learning-based MPC methods focus on learning a predictive model for the MPC scheme from data, using Machine Learning (ML) or other data-driven techniques. In these methods, only (10b) is parameterized in the MPC scheme (10), and adjusted in view of providing predictions that are as accurate as possible. It is then important to clarify why an approach centered purely on adjusting the model is not necessarily sufficient for achieving the best possible performance. While adjusting the MPC model alone from data has a high practical value, two issues stand in the way of obtaining optimal policies from doing that alone.

First, if the objective of the MPC scheme is optimality in the sense of $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$, then adjusting the MPC model $\mathbf{f}_{\boldsymbol{\theta}}$ for delivering better predictions is only a proxy for minimizing $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$. Indeed, if the true system dynamics (1) do not belong to the set of dynamics that $\mathbf{f}_{\boldsymbol{\theta}}$ can represent, then there is no guarantee that adjusting the MPC model $\mathbf{f}_{\boldsymbol{\theta}}$ to better fit (1) will reduce $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$. It is straightforward to propose trivial counter-examples where model fitting

degrades the closed-loop performance, see e.g. Gros and Zanon (2019a).

Second, Theorem 1 shows that a modification of not only the model parameters but also of the cost and constraints in the MPC formulation enables obtaining the optimal policy $\boldsymbol{\pi}^\star$ from the MPC scheme, even if the MPC model cannot predict the real system dynamics accurately. In that context, learning techniques focusing on fitting the MPC model to the real system provide no indication as to how one ought to adjust the MPC cost and constraints for performance.

In light of these observations, while fitting the MPC model to the real system trajectories has obvious practical value, it may not be sufficient to achieve optimal closed-loop performance.

> Performance-oriented learning-based MPC ought to consider a full parametrization of the MPC scheme as per (10), and integrate learning tools that aim at minimizing $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$, or at achieving $Q_{\boldsymbol{\theta}} \approx Q^\star$ from the data. The role of RL in that context is to provide these learning tools. There are alternatives to RL for adjusting the MPC parameters, such as e.g. Bayesian optimization. However, RL is often regarded as the most effective technique, especially if the number of parameters to adjust is not very small.

3.2 Role of the Model in RL for MPC

Theorem 1 suggests that if using the complete parametrization (10a), modifying the MPC model is less important than previously thought. Indeed, it suggests that under some mild assumptions, cost and constraints modifications can compensate for the model error and produce the optimal policy and value functions. This has the potential benefit that simpler models can be used.

> However, this observation ought to trigger the natural question about the role of the MPC model if it does not need to be accurate. This central question has not been discussed enough in the literature so far. Here, we introduce four fundamental insights in that context.

The first and most obvious insight lies in (13), the core assumption of Theorem 1. While it is an arguably mild assumption, it forbids the use of *any model* in the MPC scheme. The requirement it imposes resembles the stability of the model under the optimal policy but in a less demanding way. We show in Sec. 5 how RL for MPC can be designed to satisfy this assumption by construction.

The second less obvious insight stems from the observation that the cost and constraints modifications (14) required by Theorem 1 can be difficult to approximate in practice and difficult to use in an MPC scheme. Indeed, these modifications can require fairly complex and possibly very non-convex functions. They may require very rich function approximations (e.g., large DNNs), and their complexity and non-convexity can make their use in an MPC scheme impractical. In that context, one can readily observe from (14b) that being able to adjust the MPC model introduces extra degrees of freedom in how the cost and constraints modifications can be shaped, allowing one, in turn, to impose certain restrictions on these modifications. These restrictions can be related to the simplicity of the function approximations used, and/or in imposing convexity in the resulting cost and constraints. This approach is used in Seel et al. (2022), and further elaborated in Sec. 5.2. Hence the purpose of the MPC model is to facilitate the construction of modified stage cost and constraints that are suitable for MPC in practice.

A third insight is in the use of Robust MPC to build safe policies in RL, see Sec. 6. In that context, the role of the model is to predict the worst-case scenario with respect to safety-critical constraints that the real system ought not to violate. Then the model must *fit* the real system in the sense of predicting these worst cases, where this *fitting* is performed via set-membership identification (Gros and Zanon, 2022b).

The last insight lies in the explainability of the policy delivered by MPC. Because MPC proposes a full prediction of the actions it plans to take on the system and the expected system response, it can be considered a more explainable policy than generic function approximations, such as DNN. In that context, if the MPC model is a very poor match for the real system, that explainability is lost. Hence, one arguably wants the MPC model $\mathbf{f}_{\boldsymbol{\theta}}$ to be as representative as possible of the real system dynamics in order to maintain the explainability of the MPC policy. This requirement has been investigated in A.B. Martinsen (2020).

These insights can be used as guidelines for design choices regarding the model when formulating an RL for MPC. The right choice will depend on the control problem the engineer is trying to solve. For example, in a non-safety-critical economic problem, there is more flexibility in changing the model. In contrast, the possible model modifications can be highly limited to a safety-critical task.

4. RL METHODS FOR MPC

RL offers two main classes of methods, (i) policy gradient methods which target a direct minimization of $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$ over the parameters $\boldsymbol{\theta}$, or (ii) Q-learning methods which search for a fitting of the action-value function model $Q_{\boldsymbol{\theta}}$ to the optimal one $Q^\star$. This section gives insights into how these two families operate in the context of RL for MPC.

4.1 Q-learning

Basic Q-learning aims to find optimal parameters of a parameterized action-value function $Q_{\boldsymbol{\theta}}$ that closely approximates $Q^\star$ via solving a fitting problem, typically in the form

$$\min_{\boldsymbol{\theta}} \mathbb{E}\left[\left(Q^\star\left(\mathbf{s},\mathbf{a}\right) - Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{a})\right)^2\right], \tag{16}$$

where the expected value $\mathbb{E}[\cdot]$ is taken over the system trajectories and actions. Because $Q^\star$ is unknown, (16) can be replaced with an approximation, e.g., based on iterating the temporal difference problem

$$\min_{\boldsymbol{\theta}_+} \mathbb{E}\left[\left(L\left(\mathbf{s},\mathbf{a}\right) + \gamma \min_{\mathbf{a}'} Q_{\boldsymbol{\theta}}(\mathbf{s}_+,\mathbf{a}') - Q_{\boldsymbol{\theta}_+}(\mathbf{s},\mathbf{a})\right)^2\right] \tag{17}$$

until convergence, i.e., $\boldsymbol{\theta}_+ \approx \boldsymbol{\theta}$. In the context of RL for MPC, $Q_{\boldsymbol{\theta}}$ is delivered by (11) and $\min_{\mathbf{a}'} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}') = V_{\boldsymbol{\theta}}(\mathbf{s})$ is delivered by (12), i.e. by the optimal cost of MPC (10) solved at state $\mathbf{s}$. The computed action $\mathbf{a}$ used in (11) then ought to (at least regularly) differ from the MPC policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ to introduce exploration. We will discuss exploration satisfying the MPC constraints by construction in Sec. 6.1. We want to emphasize the following:

> In an RL setting (online and offline), Q-learning needs to solve two MPC schemes for each state transition, i.e., at each time instant. These solutions can be computed in parallel.

4.2 Policy Gradient Methods & Direct Policy Search

An alternative to Q-learning is to adjust the parameters $\boldsymbol{\theta}$ of the MPC as a policy (9) to minimize cumulative cost $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$ directly. This is the standard approach for systems with continuous action spaces for which we can distinguish between two methods. *Policy gradient methods* estimate the policy gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$ from data to update the parameters $\boldsymbol{\theta}$ in a gradient descent fashion. Alternatively, *direct policy search methods* build a surrogate model of $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$ from data and use it to propose new policy parameters $\boldsymbol{\theta}$. However, direct policy search tends to scale poorly with the size of the policy parameters.

One example of policy gradient methods is the *deterministic policy gradient* by Silver et al. (2014) with the following expression for the gradient of the cumulative cost:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) = \mathbb{E}\left[\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}) \nabla_{\mathbf{a}} Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\mathbf{s}, \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}))\right]. \tag{18}$$

Here $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ is delivered by MPC (10). The *critic* $Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ is typically built separately using a generic function approximator and policy evaluation techniques.

> The policy gradient approach in RL employs the actor-critic scheme where an approximation structure must be selected for the critic $Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$, e.g., as a DNN. This is an inconvenience in using policy gradient methods for MPC compared to Q-learning, as designing the critic approximator is not obvious. Fortunately, MPC as a function approximator allows for alleviating this problem. We discuss the details in Anand et al. (2022).

4.3 NLP Sensitivities & Smoothness

Many RL methods, including Q-learning and policy gradient methods, require the sensitivities of the function approximators $\boldsymbol{\pi}_{\boldsymbol{\theta}}$, $Q_{\boldsymbol{\theta}}$, and $V_{\boldsymbol{\theta}}$. When approximated by an MPC scheme, these are typically continuous but only piecewise smooth. However, when the MPC achieves Linear Independence Constraint Qualification (LICQ) and Second Order Sufficient Condition (SOSC), non-smooth points correspond to weakly active constraints in the MPC. These points form a set of zero measures for a well-formulated MPC scheme. Because RL methods always use the sensitivities inside expected value operators, their contribution to the learning disappears as long as the state transition density ρ is bounded.

> The non-smoothness of MPC schemes may superficially appear as an issue for RL, but fortunately, in all practical cases, it is not. This issue can be ignored when the MPC response is continuous.

The arguments above do not necessarily hold anymore if the MPC does not satisfy SOSC and, as a result, produces discontinuous policies, e.g., a *bang-bang* response. For example, designing the MPC as a Linear Program might lead to such a situation. We can then still use Q-learning since $Q_{\boldsymbol{\theta}}$ and $V_{\boldsymbol{\theta}}$ typically remain continuous and piecewise smooth. However, the discontinuities in the policy lead to problems in policy gradient methods. We discussed this problem and an early approach to solve it in Kordabad et al. (2021a), but more work is necessary.

4.4 Feasible exploration

Exploration is a core requirement for all RL methods, i.e., regularly applying actions $\mathbf{a} \neq \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s})$ to the real system to gather the information necessary to improve the policy. Given the constraints in (10c), a natural question is how to generate exploration that does not jeopardize the MPC feasibility.

> In the context of MPC-based policies, feasible exploration can be trivially achieved without adding complexity to the RL methods or the MPC scheme.

A straightforward way to address this is by manipulating the MPC cost such that the resulting action differs from the original policy provided by (10). One possible modification is to add a gradient over the initial action $\mathbf{u}_0$, which yields the MPC scheme

$$\phi_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{d}) = \min_{\mathbf{x},\mathbf{u}} \mathbf{d}^\top \mathbf{u}_0 + T_{\boldsymbol{\theta}}(\mathbf{x}_N) + \sum_{k=0}^{N-1} L_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{u}_k) \tag{19a}$$

$$\text{s.t.} \quad (10b), (10c), \tag{19b}$$

where $\mathbf{d} \in \mathbb{R}^m$ is a vector of the size of the action, possibly selected randomly. Because only the MPC cost is modified, the solution of (19) is feasible for (10). One can then use MPC (19) to produce feasible exploration for policy gradient methods and to generate action-value functions for Q-learning, see Sec. 4.1. In that context, (19) produces a feasible action with exploration $\mathbf{a} = \mathbf{u}_0^\star$ where $\mathbf{u}_0^\star$ is the solution of (19) and depends on $\mathbf{d}$. The optimal cost $\phi_{\boldsymbol{\theta}}$ of (19) delivers the action-value function

$$Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) = \phi_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{d}) - \mathbf{d}^\top \mathbf{u}_0^\star. \tag{20}$$

This principle has been further detailed in Gros and Zanon (2019b). However, the feasibility of (19) does not guarantee that applying the resulting policy to the real system will not violate the constraints (10c). For example, consider the effect of stochastic dynamics or model errors. This issue is discussed in the context of safe exploration in Sec. 6.1.

4.5 Current Challenges

We identified that applying RL methods on MPC is challenging in the context of learning from existing *big data*. Learning from existing data is performed by taking

numerous sweeps (a.k.a. experience replay) through the dataset using the methods detailed above. This requires an enormous number of evaluations of $\boldsymbol{\pi}_{\boldsymbol{\theta}}$, $Q_{\boldsymbol{\theta}}$, $V_{\boldsymbol{\theta}}$ and of their sensitivities. Classical RL function approximators such as DNNs have dedicated computational tools such as GPUs for fast evaluation and differentiation. Hence, we can learn efficiently from big data sets when DNNs are used. Function approximations from MPC schemes are inexpensive to differentiate, but often expensive to evaluate because they require solving the MPC problem. Excellent tools exist to solve MPC schemes in real-time such that performing RL *online*, i.e. while the system is running, is not an issue. However, performing RL for MPC on *existing* big data can be impractical due to the amount of computational time required. This issue is partly solved in Sawant et al. (2023), but more work is required to address it fully.

5. LEARNING STABLE POLICIES VIA MPC

One benefit of MPC as a function approximator in RL is that properties such as safety and stability can be established theoretically. The nominal stability of a closed-loop system under a policy is considered central in the control community. The stability of MPC schemes for deterministic systems is relatively straightforward to establish if the MPC stage cost is lower-bounded by a class-$\mathcal{K}_\infty$ function, see Rawlings et al. (2017). If a generic (a.k.a. *economic*) stage cost is considered, asymptotic stability requires the extra *dissipativity* condition

$$\tilde{L}\left(\mathbf{s},\mathbf{a}\right) \coloneqq L\left(\mathbf{s},\mathbf{a}\right) - \lambda\left(\mathbf{s}\right) + \lambda\left(\mathbf{f}\left(\mathbf{s},\mathbf{a}\right)\right) \geq \alpha\left(\|\mathbf{s}-\bar{\mathbf{s}}\|\right) \tag{21}$$

to hold for some bounded storage function λ and some $\alpha \in \mathcal{K}_\infty$, where $\bar{\mathbf{s}}$ is the optimal steady-state. Condition (21) is difficult to interpret and to verify. However, it simply entails the existence of an MPC scheme

$$\min_{\mathbf{x},\mathbf{u}} \quad -\lambda\left(\mathbf{s}\right) + \tilde{T}\left(\mathbf{x}_N\right) + \sum_{k=0}^{N-1} \tilde{L}\left(\mathbf{x}_k,\mathbf{u}_k\right) \tag{22a}$$

$$\text{s.t.} \quad \text{(7b)}, \text{(7c)}\,, \tag{22b}$$

where $\tilde{L}$ is lower-bounded by a class-$\mathcal{K}_\infty$ function. By defining $\tilde{T}(\cdot) \coloneqq T(\cdot) + \lambda(\cdot)$, MPC (22) delivers the same policy and value function as (7), which allows to enforce (nominal) stability by design in RL for MPC. We detail that next.

5.1 Stability by verification vs. stability in learning

The literature typically considers dissipativity a property to verify rather than enforce. Unfortunately, this verification is relatively complex, as shown by Angeli et al. (2011). One has to incur that complexity when MPC scheme (10) is adjusted only via modifying the MPC model (10b).

> However, when combining RL and MPC in its fully parametrized form (10), the stability question can be addressed as a reasonably simple design requirement rather than a complex posterior verification.

Indeed, using a parameterized form for (22):

$$\min_{\mathbf{x},\mathbf{u}} \quad -\lambda_{\boldsymbol{\theta}}\left(\mathbf{s}\right) + T_{\boldsymbol{\theta}}\left(\mathbf{x}_N\right) + \sum_{k=0}^{N-1} L_{\boldsymbol{\theta}}\left(\mathbf{x}_k,\mathbf{u}_k\right) \tag{23a}$$

$$\text{s.t.} \quad \text{(10b)}, \text{(10c)}\,. \tag{23b}$$

and requiring the modified stage cost $L_{\boldsymbol{\theta}}$ to fulfil

$$L_{\boldsymbol{\theta}}\left(\mathbf{s},\mathbf{a}\right) \geq \alpha\left(\|\mathbf{s}-\bar{\mathbf{s}}_{\boldsymbol{\theta}}\|\right), \quad \forall\, \mathbf{s},\, \mathbf{a} = \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s})\,, \tag{24}$$

for some steady-state $\bar{\mathbf{s}}_{\boldsymbol{\theta}}$, ensures that the parametrized MPC scheme (23) is (nominally) dissipative for an adequate choice of $T_{\boldsymbol{\theta}}$. The added storage function in (23) allows the MPC scheme (10) to correctly approximate the value functions in addition to the optimal policy. Then the RL methods detailed in Sec. 4 can be applied to the MPC (23), with the additional restriction that the proposed parameter updates have to satisfy the condition (24). We showed in Kordabad and Gros (2022), that using the parameterized MPC scheme in (23) in the Q-learning method and enforcing (24) throughout the learning, results in a valid storage function that satisfies the dissipativity condition (21).

5.2 What cost function parametrization?

It is important to clarify two points for this section. First, when using a fully parametrized MPC scheme (10), a natural approach to initialize the MPC parametrization is to select $L_{\boldsymbol{\theta}} = L$ before letting RL possibly revise that choice. However, if L does not satisfy condition (24), then in the stability context discussed here, it cannot be used as an initial guess for $L_{\boldsymbol{\theta}}$, i.e., the initial MPC parameters $\boldsymbol{\theta}$ need to yield an initial $L_{\boldsymbol{\theta}}$ that is possibly very different from L. Selecting a meaningful initial MPC stage cost $L_{\boldsymbol{\theta}}$ for the learning is not obvious in this case. Fortunately, it is possible to use a simple approximate approach. Consider learning a fully parametrized MPC (10) with inaccuracies in the MPC model (10b). In that situation, it is not productive to compute a modified cost $L_{\boldsymbol{\theta}}$ and storage function $\lambda_{\boldsymbol{\theta}}$ that leaves the MPC policy and value functions unchanged. Instead, one can, for example, provide a quadratic stage cost as an initial guess for $L_{\boldsymbol{\theta}}$, which produces the same policy and value function as the original MPC scheme in the neighborhood of the closed-loop steady state, see Zanon et al. (2016). RL can then improve on that guess without jeopardizing stability.

The second point to clarify here is that requirement (24) is not necessarily easy to satisfy in practice because it yields a semi-infinite constraint on the parameters $\boldsymbol{\theta}$.

> A more straightforward approach is to adopt a parametrization of the cost $L_{\boldsymbol{\theta}}$ that satisfies (24) by construction, e.g., using a strictly convex cost parametrization as proposed in Seel et al. (2022). Such a choice also makes the MPC scheme significantly easier to solve than if using a non-convex stage cost.

Unfortunately, choosing a convex parametrization of the cost $L_{\boldsymbol{\theta}}$ is more restrictive than the original requirement (24), which can then prevent the MPC (23) from reaching the optimal policy and value functions. The ability to adjust the MPC model (10b) is then essential to alleviate this potential issue. A currently open question is how rich the model parametrization ought to be for MPC (23) to

reach the optimal policy and value functions with a convex cost parametrization.

5.3 Current Challenges

So far, we limited the discussion to the nominal stability of the MPC scheme, i.e., to the stability of the MPC if applied in closed-loop to its model. While nominal stability is highly desirable, it does not discuss closed-loop stability in the presence of model inaccuracies or stochastic effects in the dynamics (1). It is possible to address this issue through Robust MPC techniques; see Sec. 6. However, Robust MPC is based on conservative approaches, which can degrade the closed-loop performance. A potential alternative is the functional dissipativity theory presented in Gros and Zanon (2022a), which extends dissipativity to MDPs and to stochastic problems. However, it remains to explore this concept in the context of learning.

6. LEARNING SAFE POLICIES VIA MPC

The research community is increasingly considering the application of RL to safety-critical systems. A straightforward way to define safety is through a set of critical constraints which should not be violated, typically expressed as functions of the system state. A classic approach to safe RL is to learn the policy in silico and employ a *pessimistic* model of the real system, meaning that it overestimates the probability of violating critical constraints. During learning, one can then use high penalties for violations of the critical constraints, e.g. through the use of barrier functions, see (Cheng et al., 2019). RL will then naturally adjust the policy to avoid these penalties.

In MPC, using a pessimistic model of the system is common within the Robust MPC (RMPC) methods. Starting from a pessimistic model, RMPC builds a safe policy by ensuring that even the worst-case predictions satisfy the critical constraints at all future times. To that end, the deterministic model (10b) is replaced by a model that describes the evolution of sets enclosing all possible future trajectories, leading to an expression of the form

$$\mathbb{X}_{k+1} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbb{X}_k, \boldsymbol{\pi}_c(\mathbb{X}_k, \mathbf{x}_k, \mathbf{u}_k)) \oplus \mathbb{W}_{\boldsymbol{\theta}}\,. \tag{25}$$

One then designs the policy $\boldsymbol{\pi}_c$ to *manage* the growth of the sets $\mathbb{X}_k$, usually by operating on their deviation from a reference trajectory $\mathbf{x}_k$. The set $\mathbb{W}_{\boldsymbol{\theta}}$ represents process noise to capture the prediction uncertainties. It is possible to learn (25) from data through *set-membership identification*. We can then build the following MPC scheme to enforce satisfaction of constraints (10c) explicitly:

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{u},\mathbb{X}} \quad & T_{\boldsymbol{\theta}}\left(\mathbf{x}_N\right) + \sum_{k=0}^{N-1} L_{\boldsymbol{\theta}}\left(\mathbf{x}_k, \mathbf{u}_k\right) && \text{(26a)}\\
\text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{f}_{\boldsymbol{\theta}}\left(\mathbf{x}_k, \mathbf{u}_k\right), \quad (25) && \text{(26b)}\\
& \mathbf{h}_{\boldsymbol{\theta}}\left(\mathbb{X}_k, \boldsymbol{\pi}_c(\mathbb{X}_k, \mathbf{x}_k, \mathbf{u}_k)\right) \leq 0, && \text{(26c)}\\
& \boldsymbol{\pi}_c(\mathbb{X}_k, \mathbf{x}_k, \mathbf{u}_k) \in \mathcal{A} && \text{(26d)}\\
& \mathbb{X}_N \in \mathbb{T}_{\boldsymbol{\theta}}, \quad \mathbf{x}_0 = \mathbf{s}, \quad \mathbb{X}_0 = \mathbf{s}\,. && \text{(26e)}
\end{aligned}$$

MPC (26) generates a safe policy $\boldsymbol{\pi}_{\boldsymbol{\theta}} = \mathbf{u}_0^\star$ by construction for $\mathbb{T}_{\boldsymbol{\theta}}$ adequately chosen and if (25) accounts for the worst case situations observed in the data, see Rawlings et al. (2017). Application of RL to adjust RMPC schemes has been proposed in Zanon and Gros (2020). In the context of RL for RMPC-based policies, it is useful to stress that there is a *hard* separation between learning for safety and learning for closed-loop performance. Indeed, safety is learned via set-membership identification on (25) and then enforced in the RMPC scheme by construction. Closed-loop performance is optimized using RL in parallel (Zanon and Gros, 2020). We can now refine the remarks from Sec. 3.2:

> In safe RL based on RMPC, the role of the model becomes clear and plain again: it ensures the safety of the MPC policies, and one can consider "fitting" to the real system as a form of set-membership identification.

For the sake of clarity, we ought to underline that the adjustment of the constraints (26c) in the RMPC scheme and of set $\mathbb{W}_{\boldsymbol{\theta}}$ ought to be done with care in order to preserve safety. In particular, the adjustment of set $\mathbb{W}_{\boldsymbol{\theta}}$ must ensure that model (25) accounts for all past data points in the set-membership sense. Arguably, safety-critical constraints in (26c) ought not to be modified.

6.1 Safe Exploration

Safe exploration is difficult to produce without a model of the system in the form (25), which can predict the worst-case evolution of the system, and assess the impact of the exploration on the system safety. However, even with a model (25) of the system, it can be expensive to verify the safety of an input differing from the safe policy, and even more expensive to build the set of safe inputs.

> Fortunately, using RMPC as a tool to generate a safe policy offers a very accessible way to generate safe exploration, which does not require more computations than solving the RMPC itself.

It is straightforward to apply the feasible exploration approach of Sec. 4.4 to the RMPC formulation (26). Given that we only modify the RMPC cost, the resulting solution is feasible for (26), therefore enabling safe exploration if (26) yields a safe policy. This principle has been further detailed in Zanon and Gros (2020).

6.2 Safe Policies and Safe Learning

There are two ways to implement safe learning online. One is in a batch fashion, meaning that parameter updates require collecting a minimum amount of transition data. The alternative is to perform parameter updates at every time step. In safety-critical applications, safety must always be preserved in either case.

However, while taking actions from a unique, safe, and stable policy ensures the stability and safety of the system, taking actions from a sequence of safe and stable policies may not. That is because a sequence of policies does not necessarily inherit the properties of the individual policies.

> If stability and safety are critical requirements of the policy, and if the parameter updates are performed while the system is running, then the parameter updates need to satisfy a specified set of conditions in order for the learning process to be safe and stable.

We discuss these conditions in detail in Gros and Zanon (2022b).

6.3 Current challenges

RMPC is a solid methodology to formulate safe policies. It is relatively straightforward to use in practice if its model $\mathbf{f}_{\boldsymbol{\theta}}$ is linear in the states and inputs, and if set $\mathbb{W}$ is simple (e.g. polytopic or ellipsoidal). However, if a nonlinear MPC model is necessary, then RMPC remains challenging to implement, possibly making it prohibitive to generate safe policies. Possible ways to employ RMPC for generic problems include scenario-based approaches as in (Kordabad et al., 2021b), or set integrators, see (Houska and Villanueva, 2019). However, while effective in practice, the former approach does not provide formal safety guarantees, and the latter approach can be fairly complex to use. Even in the case of RMPC using a linear model, adjusting (25) for closed-loop performance while ensuring that it captures the worst-case situations observed in the data is difficult on big data sets. This difficulty is further discussed in Zanon and Gros (2020).

7. CONCLUSION

This paper provided a general and coherent review of the foundations, theories, and essential results that have been recently developed in the context of RL based on MPC. We also provided inputs on the challenges ahead. We reviewed how a parameterized MPC scheme can deliver the optimal policies and the value functions of a given MDP, even if the model used in the MPC scheme cannot capture the real system dynamics, and provided insights on the role of the MPC model in that context. We showed how RL algorithms and concepts such as exploration and sensitivity can be formulated in the context of MPC. Some advantages of the method, such as the nominal stability of the closed-loop system and safety, were summarized.

REFERENCES

A.B. Martinsen, A.M. Lekkas, S.G. (2020). Combining system identification with reinforcement learning-based MPC. *IFAC-PapersOnLine*, 53(2), 8130–8135.

Anand, A.S., Sawant, S., Reinhardt, D., Gravdahl, J.T., and Gros, S. (2022). A painless deterministic policy gradient method for MPC. *[accepted]*.

Angeli, D., Amrit, R., and Rawlings, J.B. (2011). On average performance and stability of economic model predictive control. *IEEE transactions on automatic control*, 57(7), 1615–1626.

Cheng, R., Orosz, G., Murray, R.M., and Burdick, J.W. (2019). End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3387–3395.

Gros, S. and Zanon, M. (2019a). Data-driven economic NMPC using reinforcement learning. *IEEE Transactions on Automatic Control*, 65(2), 636–648.

Gros, S. and Zanon, M. (2019b). Towards safe reinforcement learning using NMPC and policy gradients: Part ii-deterministic case. *arXiv preprint arXiv:1906.04034*.

Gros, S. and Zanon, M. (2020). Reinforcement learning for mixed-integer problems based on MPC. *IFAC-PapersOnLine*, 53(2), 5219–5224.

Gros, S. and Zanon, M. (2022a). Economic MPC of markov decision processes: Dissipativity in undiscounted infinite-horizon optimal control. *Automatica*, 146, 110602.

Gros, S. and Zanon, M. (2022b). Learning for MPC with stability & safety guarantees. *Automatica*, 146, 110598.

Houska, B. and Villanueva, M.E. (2019). Robust optimization for MPC. In *Handbook of model predictive control*, 413–443. Springer.

Kordabad, A.B., Cai, W., and Gros, S. (2021a). MPC-based reinforcement learning for economic problems with application to battery storage. In *2021 European Control Conference (ECC)*, 2573–2578. IEEE.

Kordabad, A.B., Esfahani, H.N., Lekkas, A.M., and Gros, S. (2021b). Reinforcement learning based on scenario-tree MPC for ASVs. In *2021 American Control Conference (ACC)*, 1985–1990. IEEE.

Kordabad, A.B. and Gros, S. (2022). Q-learning of the storage function in economic nonlinear model predictive control. *Engineering Applications of Artificial Intelligence*, 116, 105343.

Kordabad, A.B., Wisniewski, R., and Gros, S. (2022a). Safe reinforcement learning using wasserstein distributionally robust MPC and chance constraint. *IEEE Access*, 10, 130058–130067.

Kordabad, A.B., Zanon, M., and Gros, S. (2022b). Equivalency of optimality criteria of markov decision process and model predictive control. *arXiv preprint arXiv:2210.04302*.

Puterman, M.L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Rawlings, J.B., Mayne, D.Q., and Diehl, M. (2017). *Model predictive control: theory, computation, and design*, volume 2. Nob Hill Publishing Madison, WI.

Sawant, S., Akhil, S.A., Reinhardt, D., and Gros, S. (2023). Offline-model predictive control using reinforcement learning. *[submitted]*.

Seel, K., Kordabad, A.B., Gros, S., and Gravdahl, J.T. (2022). Convex neural network-based cost modifications for learning model predictive control. *IEEE Open Journal of Control Systems*, 1, 366–379.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*, 387–395. JMLR.org.

Sorourifar, F., Makrygirgos, G., Mesbah, A., and Paulson, J.A. (2021). A data-driven automatic tuning method for MPC under uncertainty using constrained bayesian optimization. *IFAC-PapersOnLine*, 54(3), 243–250.

Sutton, R.S. and Barto, A.G. (2018). *Reinforcement learning: An introduction*. MIT press.

Wu, Z., Tran, A., Rincon, D., and Christofides, P.D. (2019). Machine learning-based predictive control of nonlinear processes. part i: theory. *AIChE Journal*, 65(11), e16729.

Zanon, M. and Gros, S. (2020). Safe reinforcement learning using robust MPC. *IEEE Transactions on Automatic Control*, 66(8), 3638–3652.

Zanon, M., Gros, S., and Diehl, M. (2016). A tracking MPC formulation that is locally equivalent to economic MPC. *Journal of Process Control*, 45, 30–42.

Appendix B

Learning-based MPC from Big Data Using Reinforcement Learning

Shambhuraj Sawant, Akhil S Anand, Dirk Reinhardt, Sebastien Gros

***Abstract*— This paper presents an approach for learning Model Predictive Control (MPC) schemes directly from data using Reinforcement Learning (RL). The state-of-the-art learning-based MPC methods use RL to improve the performance of parameterized MPC schemes. However, these learning algorithms are often gradient-based methods that require frequent evaluations of computationally expensive MPC schemes, thereby restricting their use on big datasets. We propose to tackle this issue by using tools from RL to learn a parameterized MPC scheme directly from data in an offline fashion. Our approach derives an MPC scheme without having to solve it over the collected dataset, thereby eliminating the computational complexity of existing techniques for big data. We evaluate the proposed method on three simulated experiments of varying complexity.**

I. Introduction

Model Predictive Control (MPC) is a pervasive control methodology in modern industrial, power, and robotics applications. It is based on a well-established theory to handle multi-variate dynamics by generating sequences of optimal control inputs that minimize a cost function, possibly subject to constraints [1]. The common practice for designing MPC schemes includes identifying a model of the system dynamics using the collected data and specifying the shape of a cost function that encapsulates the control objective. Finding a model that accurately fits the true dynamics often requires much work, particularly for systems with complex dynamics. In many cases, it might be necessary to reduce model complexity such that an embedded computing platform can meet the computational demands and memory footprint of the resulting MPC scheme. Furthermore, even after finding a suitable model, tuning the cost function can be cumbersome for tracking problems and even harder for economic tasks.

With the increasing availability of big data in industries, researchers and practitioners have increasing access to large datasets for formulating or improving MPC schemes. Therefore, it would be beneficial to use such datasets to optimize the closed-loop performance of an MPC scheme. However, it can be a challenging task to formulate or improve an existing MPC scheme from such large datasets. In this regard, various learning-based MPC approaches have been proposed in the literature that leverages big data to optimize the performance of MPC controllers. There are two prominent approaches in this direction of learning-based MPC. The first data-driven approach for improving the performance of MPC schemes is the classical approach of adjusting the model to better fit the system dynamics using machine learning techniques [2], [3]. The underlying idea is that the predictive qualities of the model are critical to achieving optimal performance. However adjusting the model alone may not necessarily lead to better closed-loop performance, as this adjustment is not directly linked to performance improvement [4]. Rather than restricting the learning to the model, additionally adjusting the associated cost and constraints can further improve the closed-loop performance of the MPC.

The second approach [5] formally demonstrated this idea in the context of Reinforcement Learning (RL) for MPC by introducing MPC as a function approximator for the RL. In this approach, RL chooses MPC parameters directly using to improve the performance rather than to improve the prediction accuracy of the model. In a fully parameterized MPC formulation, RL can be used to learn the cost, model, and constraint functions, thereby providing high flexibility in function approximation. This approach can also be considered tuning MPC parameters using RL tools for improved performance. In recent times, various works were presented investigating the MPC-based RL approach and their use in multiple applications [6], [7], [8].

The first classical approach is limited in improving the closed-loop performance of MPC. Whereas the second RL-based approach is suitable for iteratively improving the MPC parameters in an online fashion, by using the system behaviour under the current MPC policy. This approach involves evaluating an MPC scheme and the associated sensitivities to compute the gradient step required for the parameter update. These required MPC and sensitivity evaluations can be computationally demanding. This problem gets further exasperated when carrying out batch parameter updates. Additionally, the existing methods require access to the system for on-policy interactions. Consequently, using the existing MPC-based RL methods over large datasets gets challenging. We provide further discussion on the involved challenges in Section II-D.

Learning an MPC scheme based on an available dataset at a low computational expense before interacting with the system should appeal to any MPC practitioner. But the drawbacks of the existing learning-based approaches limit the formulation of performance-oriented MPC schemes from big data. We aim to take a step toward enabling this in the current work avoiding the complexities of the existing approaches. The task of learning a policy from previously collected data is also referred to as *offline* RL in the RL literature [9] and several such methods have been proposed to effectively learn from offline datasets [10], [11], [12], [13].

In this paper, we introduce a novel approach for learning

Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU) Trondheim, Norway (e-mail: shambhuraj.sawant@ntnu.no, akhil.s.anand@ntnu.no, dirk.p.reinhardt@ntnu.no, sebastien.gros@ntnu.no).

a performance-oriented MPC formulation in an offline RL setting using previously collected data. Our approach builds on the work of [5], leveraging their theorem that establishes a connection between the optimal MPC parameters and the optimal value function of the underlying system. Our proposed approach utilizes this connection to learn a high-performing MPC scheme from data while avoiding the need for expensive MPC evaluations. The key contribution of this paper is: *introducing an approach to learning a performance-oriented MPC scheme from data without having to evaluate any MPC schemes*. We achieve this by essentially reducing the task of learning a high-performing MPC scheme from data to a supervised learning problem, resulting in a low computational expense compared to existing MPC-based RL methods.

The rest of the paper is organized as follows: Section II briefly introduces the necessary background knowledge and the challenges present in using the existing MPC-based RL methods on big data, while Section III details our proposed method for learning an MPC scheme from data. Section IV contains the evaluation of our approach on three different simulation experiments of varying complexity. A brief discussion of the evaluation results of our method and further challenges is presented in Section V and conclusions are discussed in Section VI.

II. Background

A. Markov Decision Process

Markov Decision Processes (MDPs) provide a generic framework for the class of problems at the core of MPC. An MDP operates over given state and action (aka input) spaces $\mathcal{S}$, $\mathcal{A}$, respectively. These spaces can be discrete (i.e. integer sets), continuous, or mixed. An MDP is defined by the tuple $(\mathcal{S},\, \mathcal{A},\, L,\, \gamma,\, \rho)$, where L is a stage cost, $\gamma \in [0, 1)$ a discount factor, and ρ denotes the conditional probability (measure) defining the dynamics of the underlying system, i.e. for a given state-action pair $\boldsymbol{s},\, \boldsymbol{a} \in \mathcal{S} \times \mathcal{A}$, the successive state $\boldsymbol{s}_+$ is distributed according to

$$\boldsymbol{s}_+ \sim \rho(\cdot|\boldsymbol{s}, \boldsymbol{a})\,. \tag{1}$$

Note that (1) is a generalization of the deterministic or stochastic dynamics model often considered in MPC, usually cast as:

$$\boldsymbol{s}_+ = \boldsymbol{F}(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{w})\,, \quad \boldsymbol{w} \sim W \tag{2}$$

where $\boldsymbol{w}$ is a random disturbance from distribution W. In the special case $\boldsymbol{w} = 0$, (2) simply yields deterministic dynamics. A solution to an MDP then involves finding an optimal policy $\boldsymbol{\pi}^\star : \mathcal{S} \to \mathcal{A}$, given as:

$$\boldsymbol{\pi}^\star = \arg\min_{\boldsymbol{\pi}} J(\boldsymbol{\pi}) \tag{3}$$

where $J(\boldsymbol{\pi})$ is a measure of the closed-loop performance which is defined as the cumulative cost, i.e.:

$$J(\boldsymbol{\pi}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\boldsymbol{s}_k, \boldsymbol{a}_k) \,\middle|\, \boldsymbol{a}_k = \boldsymbol{\pi}(\boldsymbol{s}_k)\right], \tag{4}$$

where $\gamma \in (0, 1]$ is the discount factor. The expected value operator $\mathbb{E}[.]$ is taken over the (possibly) stochastic closed loop trajectories of the system. Discussing the solution of an MDP is often best done via the Bellman equations defining implicitly the optimal value function $V^\star : \mathcal{S} \to \mathbb{R}$ and the optimal action-value function $Q^\star : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ [14] as:

$$V^\star(\boldsymbol{s}) = \min_{\boldsymbol{a}} Q^\star(\boldsymbol{s}, \boldsymbol{a}) \tag{5a}$$

$$Q^\star(\boldsymbol{s}, \boldsymbol{a}) = L(\boldsymbol{s}, \boldsymbol{a}) + \gamma \mathbb{E}\left[V^\star(\boldsymbol{s}_+) \mid \boldsymbol{s}, \boldsymbol{a}\right] \tag{5b}$$

The optimal policy then reads as:

$$\boldsymbol{\pi}^\star(\boldsymbol{s}) = \arg\min_{\boldsymbol{a}} Q^\star(\boldsymbol{s}, \boldsymbol{a}) \tag{6}$$

B. Model Predictive Control

For a given system state $\boldsymbol{s}$, an MPC produces control policies based on repeatedly solving an optimal control problem on a finite, receding horizon. Suppose $\boldsymbol{f}_{\boldsymbol{\theta}}$ denotes a model of the true dynamics (1), $L_{\boldsymbol{\theta}}$ is the stage cost function, and $\boldsymbol{h}_{\boldsymbol{\theta}}$ denotes the constraints, each parameterized by a parameter vector $\boldsymbol{\theta}$. The problem to be solved is then typically cast as:

$$\min_{\boldsymbol{x}, \boldsymbol{u}} \quad \gamma^N T_{\boldsymbol{\theta}}(\boldsymbol{x}_N) + \sum_{k=0}^{N-1} \gamma^k L_{\boldsymbol{\theta}}(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{7a}$$

$$\text{s.t.} \quad \boldsymbol{x}_{k+1} = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_k, \boldsymbol{u}_k)\,, \quad \boldsymbol{x}_0 = \boldsymbol{s} \tag{7b}$$

$$\boldsymbol{h}_{\boldsymbol{\theta}}(\boldsymbol{x}_k, \boldsymbol{u}_k) \le 0, \quad \boldsymbol{u}_k \in \mathcal{A} \tag{7c}$$

For an initial condition $\boldsymbol{x}_0 = \boldsymbol{s}$, problem (7) produces a sequence of control inputs $\boldsymbol{u}^\star = \{\boldsymbol{u}_0^\star, \ldots, \boldsymbol{u}_{N-1}^\star\}$ and corresponding state predictions $\boldsymbol{x}^\star = \{\boldsymbol{x}_0^\star, \ldots, \boldsymbol{x}_N^\star\}$. Only the first element $\boldsymbol{u}_0^\star$ of the input sequence $\boldsymbol{u}^\star$ is applied to the system. At the next sampling step, a new state $\boldsymbol{s}$ is received, and problem (7) is solved again, producing a new $\boldsymbol{u}^\star$ and a new $\boldsymbol{u}_0^\star$. MPC hence yields a policy:

$$\boldsymbol{\pi}_\theta(\boldsymbol{s}) = \boldsymbol{u}_0^\star\,, \tag{8}$$

with $\boldsymbol{u}_0^\star$ solution of (7) for a given initial state $\boldsymbol{s}$. MPC policy (8) can provide a good approximation of the optimal policy $\boldsymbol{\pi}^\star$ for an adequate choice of prediction horizon N, terminal cost $T_{\boldsymbol{\theta}}$ and if the MPC model $\boldsymbol{f}_{\boldsymbol{\theta}}$ approximates the true dynamics (1) sufficiently well.

In the context of approximating $\boldsymbol{\pi}^\star(\boldsymbol{s})$ using (8), the model is arguably the major bottleneck as many systems are challenging to model accurately. Machine learning approaches for MPC that modify the model based on mismatch to the observations, e.g. using Gaussian Process regression, are precisely tackling this problem. However, within a modeling structure, selecting the model $\boldsymbol{f}_{\boldsymbol{\theta}}$ that yields the best closed-loop performance $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$ is very difficult. Additionally, there is in general no guarantee that the most accurate model yields the best closed-loop performance.

C. Reinforcement Learning for MPC

It is essential to understand how the optimal value functions and policy can be approximated by an MPC scheme. Using

(7), we consider an approximation of the value function as

$$V_{\boldsymbol{\theta}}(\boldsymbol{s}) = \min_{\boldsymbol{x},\boldsymbol{u}} \quad \text{(7a)}, \tag{9a}$$
$$\text{s.t.} \quad \text{(7b)} - \text{(7c)} \tag{9b}$$

and the action-value function as

$$Q_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a}) = \min_{\boldsymbol{x},\boldsymbol{u}} \quad \text{(7a)}, \tag{10a}$$
$$\text{s.t.} \quad \text{(7b)} - \text{(7c)}, \quad \boldsymbol{u}_0 = \boldsymbol{a} \tag{10b}$$

with added constraint $\boldsymbol{u}_0 = \boldsymbol{a}$ on the initial input. The MPC scheme in (10) is a valid approximation of $Q^\star$ in the sense that it satisfies the relationships (5) and (6), i.e.:

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}) = \arg\min_{\boldsymbol{a}} Q_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a}), \quad V_{\boldsymbol{\theta}}(\boldsymbol{s}) = \min_{\boldsymbol{a}} Q_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a}). \tag{11}$$

For learning the approximations in (9) and (10), let us briefly state the central result by [5]. It establishes the equivalence between the optimal policy and value functions and the approximations provided by the MPC:

Theorem 1. *[5, Theorem 1] Consider the parameterized stage cost, terminal cost and constraints in* (7) *as function approximators with adjustable parameters $\boldsymbol{\theta}$. Further suppose that $\boldsymbol{x}^\star$ is an optimal state trajectory generated by the MPC in* (7) *and there exist parameters $\boldsymbol{\theta}^\star$ such that*

$$T_{\boldsymbol{\theta}^\star}(\boldsymbol{s}) = V^\star(\boldsymbol{s}) \tag{12a}$$
$$L_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a}) = \begin{cases} Q^\star(\boldsymbol{s},\boldsymbol{a}) - \gamma V^+(\boldsymbol{s},\boldsymbol{a}) & \text{If } |V^+(\boldsymbol{s},\boldsymbol{a})| < \infty \\ \infty & \text{otherwise} \end{cases} \tag{12b}$$

where, $V^+(\boldsymbol{s},\boldsymbol{a}) = V^\star(\boldsymbol{f}_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a}))$. Then the following identities hold:

1) $V_{\boldsymbol{\theta}^\star}(\boldsymbol{s}) = V^\star(\boldsymbol{s}), \forall \boldsymbol{s} \in \mathcal{S}$
2) $\boldsymbol{\pi}_{\boldsymbol{\theta}^\star}(\boldsymbol{s}) = \boldsymbol{\pi}^\star(\boldsymbol{s}), \forall \boldsymbol{s} \in \mathcal{S}$
3) $Q_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a}) = Q^\star(\boldsymbol{s},\boldsymbol{a}), \forall \boldsymbol{s} \in \mathcal{S}$, *for the inputs* $\boldsymbol{a} \in \mathcal{A}$ *such that* $|V^\star(\boldsymbol{f}_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a}))| < \infty$

if the set

$$\Omega =: \left\{ \boldsymbol{s} \in \mathcal{S} \;\middle|\; |V^\star(\boldsymbol{x}_k^\star)| < \infty,\ \forall\, k \le N \right\} \tag{13}$$

is non-empty.

The proof follows from [5], [15].

Essentially, Theorem 1 states that it is possible to compute the optimal policy $\boldsymbol{\pi}^\star(\boldsymbol{s})$ using inaccurate model dynamics. However, $L_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a})$ in (12b) is difficult to compute and requires a model of the system dynamics. To bypass difficult $L_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a})$ evaluation, [5] suggest to learn the optimal parameter $\boldsymbol{\theta}^\star$ using RL tools such as Q learning or policy gradient methods.

D. Challenges with learning MPC with RL on big data

Most RL methods involve iteratively optimizing the policy using the experience acquired by interacting with the system. Their success hinges on repeatedly combing through collected data to build better approximations of MDP value functions and policy, in particular when using Deep Neural Networks (DNNs) as function approximators. The MPC-based RL methods, similarly, iteratively optimize the closed-loop performance for learning $\boldsymbol{\theta}^*$ from observed state transitions. However, these iterative updates require computing multiple MPC solutions for each optimization step. Additionally, finding an optimal $\boldsymbol{\theta}^\star$ by scrubbing through the collected data is computationally demanding.

The classical RL methods fundamentally operate in *online* learning paradigm. Though many RL methods work with off-policy data, these methods often cannot learn effectively from entirely *offline* datasets, without additional on-policy interactions [9]. In recent times, the offline RL paradigm has garnered growing interest and many approaches have been proposed to effectively learn from offline data [10], [11], [12], [13]. However, many challenges persist in using RL with offline data as discussed in [9], [16].

III. Learning MPC from big data

Problem statement

We address the challenge of formulating an optimal MPC scheme $\boldsymbol{\pi}^\star$ parameterized by $\boldsymbol{\theta}$ from a previously collected dataset $\mathcal{D}$ containing transition data $(\boldsymbol{s},\boldsymbol{a},\boldsymbol{s}_+)$. Given the dataset $\mathcal{D}$ and a stage cost function $L(\boldsymbol{s},\boldsymbol{a})$, the aim is to find the optimal parameters $\boldsymbol{\theta}^\star$ for the MPC scheme such that it is optimal for closed-loop performance. Existing learning-based approaches have shown promise in improving MPC controllers using large datasets, but they have limitations as described in Section I. We propose to learn a performance-oriented MPC formulation in an offline RL setting using previously collected data, which avoids the need for expensive MPC evaluations and makes it possible to learn high-performing MPC schemes from data at a low computational expense. Fig. 1 provides a comparison between the conventional model learning-based approach and the proposed approach. This section describes the proposed approach to learning an MPC scheme from big data.

Learning MPC parameterizations

As stated in Theorem 1, under some conditions, an MPC scheme using a model $\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})$ and the stage cost in (12b) yields the optimal policy $\boldsymbol{\pi}^\star(\boldsymbol{s})$ for the underlying MDP, together with the associated optimal value functions (5). The optimal parameters for an MPC scheme parameterized in $\boldsymbol{\theta}$ relates to a well-defined optimal value function $V^\star(\boldsymbol{s})$, as in given in Theorem 1, as:

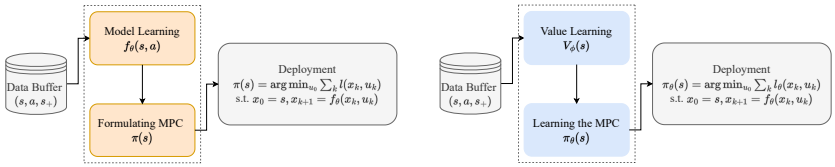
$$L_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a}) = Q^\star(\boldsymbol{s},\boldsymbol{a}) - \gamma V^\star(\boldsymbol{f}_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a})). \tag{14}$$

Using Bellman equations, we get,

$$L_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a}) = L(\boldsymbol{s},\boldsymbol{a}) + \gamma V^\star(\boldsymbol{s}_+) - \gamma V^\star(\boldsymbol{f}_{\boldsymbol{\theta}^\star}(\boldsymbol{s},\boldsymbol{a})), \tag{15}$$

where $L(\boldsymbol{s},\boldsymbol{a})$ is the stage cost of the underlying MDP. Here, variable $\boldsymbol{\theta}$ includes all the parameterizations introduced in the MPC scheme in (7).

Estimating the optimal value function $V^\star(\boldsymbol{s})$ requires knowledge of the system dynamics (1) and the stage cost $L(\boldsymbol{s},\boldsymbol{a})$. However, $V^\star(\boldsymbol{s})$ can also be approximated from a dataset with a rich enough data distribution over the state and



(a) Classical approach based on model fitting

(b) The proposed approach

Fig. 1: Comparison between the (a) classical MPC formulation approach and (b) our proposed approach. In the classical MPC formulation, an accurate model of system dynamics is learned from a given data set, and an MPC scheme is formulated using this model. Unlike the classical model learning approach, we don't focus on learning the system dynamics or cost function but rather estimate the value function from the given dataset using a value learning approach. The resulting value function is then used to learn a parametric MPC scheme $\pi_{\boldsymbol{\theta}}$. This approach helps us directly target the performance-oriented MPC scheme.

action space $\mathcal{S}, \mathcal{A}$. We propose to use Deep Neural Networks (DNNs) to approximate a value function $V_{\boldsymbol{\phi}}(\boldsymbol{s}) \approx V^*(s)$ from data. $V_{\boldsymbol{\phi}}(\boldsymbol{s})$ can be estimated using update equation given as:

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_{\tau\sim\mathcal{D}} \left[(L(\boldsymbol{s},\boldsymbol{a}) + \gamma V_{\boldsymbol{\phi}}(\boldsymbol{s}_+) - V_{\boldsymbol{\phi}}(\boldsymbol{s}))^2 \right], \tag{16}$$

where $\tau : \{\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}_+, L(\boldsymbol{s},\boldsymbol{a})\}$ is the transition sampled from the given dataset $\mathcal{D}$. With a dataset $\mathcal{D}$ generated following a greedy in the limit of infinite exploration (GLIE) policy, the learned value function $V_{\boldsymbol{\phi}}(\boldsymbol{s})$ will closely match the optimal value function $V^\star(\boldsymbol{s})$ i.e. $V_{\boldsymbol{\phi}}(\boldsymbol{s}) \sim V^\star(\boldsymbol{s})$. The estimated $V_{\boldsymbol{\phi}}(\boldsymbol{s})$ can then be integrated into (15) as:

$$L_{\boldsymbol{\theta}^*}(\boldsymbol{s},\boldsymbol{a}) \approx L(\boldsymbol{s},\boldsymbol{a}) + \gamma V_{\boldsymbol{\phi}}(\boldsymbol{s}_+) - \gamma V_{\boldsymbol{\phi}}(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})). \tag{17}$$

It is often useful to impose additional constraints on the parameterization adopted in an MPC scheme, e.g. the stage cost should preferably be convex and smooth, such that $L_{\boldsymbol{\theta}}$ may preferably be restricted to a specific class of functions. Considering possible constraints on the stage cost, we fit a structured cost $\hat{L}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})$ to the cost function $L_{\boldsymbol{\theta}^*}(\boldsymbol{s},\boldsymbol{a})$ in (17). Thus, the optimal parameters $\boldsymbol{\theta}^*$ for the parameterized MPC scheme can be derived as:

$$\begin{aligned}\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}[&(L(\boldsymbol{s},\boldsymbol{a}) + \gamma V_{\boldsymbol{\phi}}(\boldsymbol{s}_+) \\ &- \gamma V_{\boldsymbol{\phi}}(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})) - \hat{L}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a}))^2]\end{aligned} \tag{18}$$

This forms the central result of our work, which essentially reduces the problem of learning MPC parameters from data to a supervised learning problem.

An important observation regarding $\boldsymbol{\theta}^*$ from (12) and (18) is that the resulting model dynamics $\boldsymbol{f}_{\boldsymbol{\theta}^*}(\boldsymbol{s},\boldsymbol{a})$ may not match the true system dynamics. The model dynamics $\boldsymbol{f}_{\boldsymbol{\theta}^*}(\boldsymbol{s},\boldsymbol{a})$ is unlikely to be the best fit model for state predictions as it is solely adjusted for closed-loop performance. However, if it is important to have a model with a high prediction accuracy, the parameters associated with the model dynamics $\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})$ can be excluded from $\boldsymbol{\theta}$ i.e. minimizing (18) over only cost parameterizations. Indeed, according to Theorem 1, if one can accommodate a rich structure to the stage cost, then the MPC formulated by learning only the stage cost using (18) can compensate for any model inaccuracies. We will discuss further the issue of the limitations on the structure of the cost function $\hat{L}_{\boldsymbol{\theta}}$ and some possible solutions in Section V. Additionally, Theorem 1 holds for optimal value function $V^\star(\boldsymbol{s})$ which can be built given a dataset generated with a greedy in the limit of exploration (GLIE) policy [14]. However, that is a fair concern for all data-driven learning methods. In our approach, with a rich dataset, a good approximation of the value function can still be built and used for learning a high-performing MPC scheme.

To summarize, we propose a solution to formulate a high-performing MPC scheme from big data using RL tools. We reduce the problem of learning the MPC parameters to a supervised learning task by building a value function estimate from a given dataset and incorporating it into Theorem 1. To that extent, we utilize the strength of machine learning methods to extract knowledge from big data effectively and utilize it for learning the MPC scheme, thereby eliminating the enormous computational complexities faced by existing approaches for learning-based MPC. As we outsource the problem of extracting a value function estimate and learning MPC parameters to machine learning tools, the learned MPC scheme is obtained at low computational expense without any MPC evaluations. Additionally, it is assured to have a high closed-loop performance if a good approximation of $V^\star(\boldsymbol{s})$ can be built from the data and the learned MPC parameters satisfy (18). In the next section, we will evaluate the proposed approach in three different simulation experiments.

IV. Experiments

In this section, we present three simulated examples of varying complexity to evaluate our approach to learning MPC from big data.

A. Experimental Setup

1) MPC parameterization:: For all the three simulated examples, we parameterized the modified stage cost $\hat{L}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})$ in (18) as a quadratic cost given by

$$\hat{L}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a}) = (\boldsymbol{s}-\boldsymbol{s}_{ref})^T \boldsymbol{W}_{\boldsymbol{\theta}} (\boldsymbol{s}-\boldsymbol{s}_{ref}) + \boldsymbol{a}^T \boldsymbol{R}_{\boldsymbol{\theta}} \boldsymbol{a} + \theta_3 \tag{19}$$

$$\boldsymbol{W}_{\boldsymbol{\theta}} = \begin{bmatrix} \theta_{W,11} & \theta_{W,12} & \dots \\ \theta_{W,21} & \theta_{W,22} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}, \tag{20}$$

$$\boldsymbol{R}_{\boldsymbol{\theta}} = \begin{bmatrix} \theta_{R,11} & \theta_{R,12} & \dots \\ \theta_{R,21} & \theta_{R,22} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix},$$

where $\boldsymbol{W}_{\boldsymbol{\theta}}, \boldsymbol{R}_{\boldsymbol{\theta}}$ are the state and input weight matrices, and $\boldsymbol{s}_{ref}$ is the desired goal state of the tracking control problem. The terminal cost is set to be 2 norm of tracking error, $T(s) = \|\boldsymbol{s}-\boldsymbol{s}_{ref}\|$. The parameter vector $\boldsymbol{\theta}$ collects all the parameters in the MPC scheme together. In order to formulate a well-defined MPC scheme, we further include a semi-definite constraint over $\boldsymbol{W}_{\boldsymbol{\theta}}$ and $\boldsymbol{R}_{\boldsymbol{\theta}}$. Additionally, a $l2$ regularization penalty is imposed around the given initial parameter estimates. The $l2$ regularization is used to restrict the RL updates on the parameters to be closer to their initial values, thereby facilitating stable updates. We choose to fix the constraint equations for all the experiments i.e. constraints are not parameterized. The three different tracking control problems chosen to test our method are

1) Linear tracking MPC task,
2) Pendulum swing-up task,
3) Cartpole swing-up and balancing task.

In all the experiments, the variable $\boldsymbol{\theta}$ is initialized to a nominal MPC scheme formulated using an inaccurate model and a nominal stage cost. Essentially we apply our approach to improve the performance of this nominal MPC formulation directly from a dataset. The nominal MPC parameters can be seen as an initial guess for our approach to improve upon. This nominal MPC is denoted by MPC_1 throughout this section.

In all three experiments, we learn two different MPC parameterizations,

1) Only learning the stage cost parameters with a fixed model: denoted as MPC_2 where the $\boldsymbol{\theta}$ in (18) only constitute the stage cost parameters in (19).
2) Learning both stage cost and model parameters: denoted by MPC_3 where the $\boldsymbol{\theta}$ constitutes both the stage cost parameters in (19) and model parameters.

The richer parameterization in the second case allows higher flexibility for capturing $L_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})$ in (15). However, learning cost parameterization with fixed model dynamics should still capture (15) to a degree and learn a better performing MPC scheme than the nominal one. In order to evaluate our approach we compare the performance achieved by both of these learning-based MPC schemes to MPC_1 in the case of pendulum swing-up and cartpole swing-up tasks. For the linear tracking MPC task, the performance for these MPC schemes is reported relative to closed-loop optimal performance.

2) Data Generation:: A rich dataset was obtained using a popular RL method, Deep-Deterministic Policy Gradient ($DDPG$) [17]. For each example, a dataset of 500 episodes (each with 100 transitions) is collected using $DDPG$ agent. The $DDPG$ agent's learning progress ensures rich data distribution in the collected dataset.

3) Value Learning:: For all three experiments, we learn the approximations for value functions $V_{\phi}(\boldsymbol{s})$ using DNNs with 2 hidden layers (each with 256 neurons). With current machine learning tools, building a good approximation of $V^{\star}(\boldsymbol{s})$ from big data takes a fraction of computational effort compared to that of learning based MPC methods.

B. Linear Tracking MPC

We first consider a simple linear MPC to illustrate the effectiveness of our approach to learning an MPC scheme from data. The linear MPC task consists of a four-dimensional state space, $\boldsymbol{s} = [x, y, \dot{x}, \dot{y}]$ and input space, $\boldsymbol{a} = [F_x, F_y]$. The true system dynamics is of deterministic nature, defined as:

$$\boldsymbol{s}_+ = \boldsymbol{A}\,\boldsymbol{s} + \boldsymbol{B}\,\boldsymbol{a} \tag{21}$$

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0.1 & 0 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0 & 0.9 & 0 \\ 0 & 0 & 0 & 0.9 \end{bmatrix} + \alpha \begin{bmatrix} 0.68 & -1.15 & -2.29 & -2.42 \\ 1.57 & 2.06 & 0.53 & 1.15 \\ 0.22 & 2.17 & 1.58 & -2.49 \\ 1.79 & -2.33 & 1.15 & -1.62 \end{bmatrix} \times 10^{-2}$$

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} + \alpha \begin{bmatrix} 1.82 & 0.21 \\ -1.00 & -0.39 \\ -2.36 & -1.88 \\ 0.85 & 0.74 \end{bmatrix} \times 10^{-2}$$

where α scales the unmodeled part of system dynamics. Note that, by varying the value of α we can generate different system dynamics. The stage cost for the true system is given as

$$L(\boldsymbol{s},\boldsymbol{a}) = 9(x^2+y^2) + 1(\dot{x}^2+\dot{y}^2) + 0.1(F_x^2+F_y^2)\,.$$

The task is discretized with sampling time of 0.1, the discount factor γ is 0.9, and the task length is 100. The MPC scheme is defined for discretized control task with a horizon equal to the task length i.e. $N = 100$. MPC_1 is formed using $L(\boldsymbol{s},\boldsymbol{a})$ as the stage cost and nominal model dynamics with $\alpha = 0$ in (21). For MPC_2, the stage cost is parameterized by $\hat{L}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})$ in (19) and a nominal model dynamics with $\alpha = 0$ in (21). For MPC_3, the MPC scheme is formed using $\hat{L}_{\boldsymbol{\theta}}(\boldsymbol{s},\boldsymbol{a})$ in (19) as a stage cost and a model dynamics with $(\boldsymbol{A}_{\boldsymbol{\theta}}, \boldsymbol{B}_{\boldsymbol{\theta}})$, where $(\boldsymbol{A}_{\boldsymbol{\theta}}, \boldsymbol{B}_{\boldsymbol{\theta}})$ are fully parameterized matrices of corresponding sizes.

Figure 2 shows the relative performance of MPC schemes with respect to closed-loop optimal performance for the system dynamics corresponding with different values of α. As seen from fig. 2, performance of MPC_1 starts to degrade with added perturbations, as the plant-model mismatch increases. Learning cost parameterization in MPC_2 manages to find a high-performing MPC scheme for the inaccurate model dynamics, i.e. the learnt cost parameters compensate for the model inaccuracies for lower value of α. However, as α increases, MPC_2 with a quadratic cost parameterization in (19) fails to capture $L_{\boldsymbol{\theta}^*}(\boldsymbol{s}, \boldsymbol{a})$ in (15), resulting in loss of performance. Whereas, MPC_3 with learned cost and model parameterizations finds a high performing MPC scheme for all value of α i.e. with our approach, an MPC scheme with a close-to-optimal performance can be learned from the collected data.

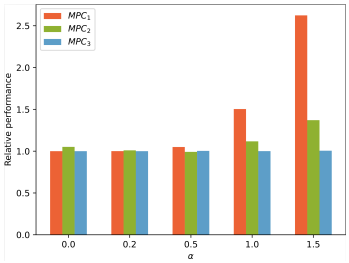


Fig. 2: Performance of the three MPC schemes relative to the closed-loop optimal performance for a linear tracking MPC task.

C. Pendulum Swing-Up

We consider a pendulum swing-up task to illustrate our method on non-linear dynamics with a complex value function. In this example, the goal of the control task is to swing up the pendulum with an under-actuated motor and maintain it at an upright angle. The system dynamics are given by

$$\ddot{\phi} = \frac{3g}{2l}\sin\phi + \frac{3}{ml^2}u \tag{22}$$

where ϕ is the angular deviation of the pole w.r.t the vertical position, gravity $g = 9.8$, u is the applied torque, $u \in [-2, 2]$, and $m = 1$, $l = 1$ are the mass and length of the pole respectively. The system state $\boldsymbol{s}$ is defined to be $\boldsymbol{s} = [\phi, \dot{\phi}]$ with ϕ normalized between $(-\pi, \pi]$. The corresponding cost function is given as

$$L(\boldsymbol{s}, u) = \phi^2 + 0.1\dot{\phi}^2 + 0.1u^2\,.$$

The task is discretized with a sampling time of 0.05, the discount factor γ is 0.9, and the task length is 100.

The MPC scheme for the discretized control task is defined over a transformed state-space $\boldsymbol{y} = [\cos\phi, \sin\phi, \dot{\phi}]$ solely for the ease of implementation. The horizon for the MPC scheme is chosen as $N = 50$ and the tracking goal corresponding to the vertical pole orientation is $\boldsymbol{y}_{ref} = [1, 0, 0]$. The nominal stage cost for the MPC scheme over observation $\boldsymbol{y}$ is

$$L_{MPC}(\boldsymbol{y}, u) = (\cos\phi - 1)^2 + \sin\phi^2 + 0.1\dot{\phi}^2 + 0.1u^2\,.$$

We use an inaccurate model with an uncertain estimate of true system properties for forming the nominal MPC scheme MPC_1. The uncertain estimates of mass and length used in the MPC model are $\hat{m}$ and $\hat{l}$ respectively, with

$$\hat{m} = m + \alpha\,\mathcal{U}[-0.5, 0.5], \quad \hat{l} = l + \alpha\,\mathcal{U}[-0.5, 0.5]$$

where α scales the uncertainty in estimates of the mass and length of the pole.

Figure 3 shows the average of relative performance for different MPC formulations. Similar to the first example, MPC_1 starts to degrade quickly with high uncertainties in the model dynamics. Whereas, the learned MPC schemes in MPC_2 and MPC_3 manage to perform better and are able to compensate for the plant-model mismatch with $\alpha \leq 1$. Even for complex value functions learned with machine learning tools, our approach can learn the stage cost parameters in MPC_2 to compensate for inaccuracies in the given model. However, for larger model errors such as $\alpha \geq 1$, the quadratic cost in (19) cannot accurately capture $L_{\boldsymbol{\theta}}(|s, a)$ in (15). Whereas additionally learning the model parameters in MPC_3 helps to further improve the MPC performance. However, we observe that since the value function approximation is built using finite data, MPC_3 still can not find a high-performing MPC scheme when initialized with highly inaccurate model dynamics.

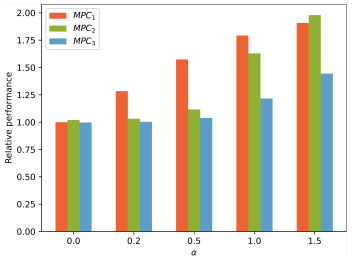


Fig. 3: Average performance of the MPC schemes relative to MPC with the ground truth model (MPC_1 with $\alpha = 0$) over 10 learning trails with different seeds for pendulum swing-up task.

D. Cartpole Swing-Up and Balancing

The cartpole system consists of a wheeled cart of mass ($m_c = 0.2$) which can freely move on a rail with a friction coefficient of ($\mu_f = 0.5$), and a pole of mass ($m_p = 0.2$) and length ($l = 0.5$) is hinged to cart on a friction-less joint. The pole can swing freely around the hinged joint. The control input, u to the system is the force exerted on the cart along

the rail with $u \in [-2, 2]$. The goal of the control task is to swing the pole to an upright position as quickly as possible and balance it in the upright position.

The dynamics equations for a cartpole system are given as

$$\begin{aligned} \ddot{\phi} &= \frac{g \sin\phi + \cos\phi \left(\frac{\mu_f \dot{x} - u - m_p l \dot{\phi}^2 \sin\phi}{m_c + m_p} \right)}{l \left(\frac{4}{3} - \frac{m_p \cos^2\phi}{m_c + m_p} \right)} \\ \ddot{x} &= \frac{u - \mu_f \dot{x} + m_p l \left(\dot{\phi}^2 \sin\phi - \ddot{\phi} \cos\phi \right)}{m_c + m_p}. \end{aligned} \tag{23}$$

where ϕ is the angular deviation of the pole w.r.t the vertical position, and x is the horizontal displacement of the cart. The system state-space $\boldsymbol{s}$ is given as $\boldsymbol{s} = [x, \dot{x}, \phi, \dot{\phi}]$ with ϕ normalized to be in $(-\pi, \pi]$. The corresponding cost function is

$$L(\boldsymbol{s}, u) = 2x^2 + \phi^2 + 0.1\dot{x}^2 + 0.1\dot{\phi}^2 + 0.1u^2 \,.$$

The task is similarly discretized with a sampling time of 0.05, the discount factor γ is set to 0.9, and the task length is 100.

The MPC scheme is formulated over the discretized task with observation $\boldsymbol{y} = [x, \dot{x}, \cos\phi, \sin\phi, \dot{\phi}]$ with the tracking goal $\boldsymbol{y}_{ref} = [0, 0, 1, 0, 0]$ and MPC horizon of $N = 50$. The stage cost of MPC scheme over observation $\boldsymbol{y}$ is

$$\begin{aligned} L_{MPC(\boldsymbol{y},u)} =& 3x^2 + 3(\cos\phi - 1)^2 + \sin\phi^2 \\ &+ 0.01\dot{x}^2 + 0.01\dot{\phi}^2 + 0.001u^2 \,. \end{aligned}$$

Similar to the pendulum swing-up task, the uncertain estimate of the system parameters used in the nominal MPC model are as follows,

$$\begin{aligned} \hat{m}_c = m_c + \alpha\, \mathcal{U}[-0.1, 0.1], \quad \hat{\mu}_f = \mu_f + \alpha\, \mathcal{U}[-0.25, 0.25] \\ \hat{m}_p = m_p + \alpha\, \mathcal{U}[-0.1, 0.1], \quad \hat{l} = l + \alpha\, \mathcal{U}[-0.25, 0.25] \,. \end{aligned}$$

Figure 4 shows the average relative performance of different MPC schemes for different inaccurate model dynamics. Learned MPC formulations in MPC_2 and MPC_3 similarly compensate for the model inaccuracies for $alpha < 1$ and finds a high performing MPC parameterizations. The learned MPC schemes' performance still degrades when initialized with highly inaccurate model dynamics. However, MPC_3 still out performs MPC_1, essentially, our approach still finds MPC parameters to improve performance over the baseline.

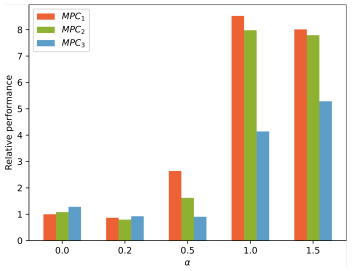


Fig. 4: Average performance of the MPC schemes relative to MPC with the ground truth model (MPC_1 with $\alpha = 0$) over 10 learning trails with different seeds for cartpole swing-up and balancing task.

V. Discussion

The experimental results demonstrate the strength of our approach to learning an MPC scheme from data. The proposed approach outsources the task of building a good value function approximation to DNN-based RL tools, thereby bypassing the computationally expensive MPC solutions over the dataset. We essentially reduce the task of learning an MPC scheme to a simple supervised learning problem. The essence of our approach can be summarized as deriving a high-performance MPC scheme directly from data without having to endure the complexities of current RL-based MPC methods.

Our approach learns a high-performing MPC formulation from the collected data in all three experiments, additionally compensating for inaccurate initial model dynamics effectively, irrespective of the value function complexity. Additionally from the results, we infer that the value function approximation built from a finite dataset is sufficient to extract a close-to-optimal MPC scheme using (18). We observe that, by only learning the stage cost parameters, the resulting MPC scheme could correct for inaccurate model dynamics. However, cost learning can not adequately capture $L_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})$ in (15) for highly inaccurate model dynamics. Whereas, learning both stage cost and model parameters in an MPC scheme provides high flexibility to minimise the supervised loss in (18), thereby obtaining a learned MPC scheme with close-to-optimal performance even while initialised with highly inaccurate models.

An important point to note here is that we directly coupled the model parameter to close-loop MPC performance through (18). This is a fundamentally different approach to learning the model parameters as compared to the classical approach of model learning where the parameters are adjusted for prediction accuracy. In the context of learning model parameters, it has to be noted that the central result in (18) holds when optimal policy $\boldsymbol{\pi}_*(\boldsymbol{s})$ can stabilize the model dynamics $f_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})$. However, further investigations need to be carried out to understand the performance of our approach when this condition fails.

In the experiments, we approximated $L_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})$ in (15) with a quadratic cost parameterization. But such simple cost functions fail to fully capture $L_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})$ for complex value functions and highly inaccurate model dynamics. This is evident from the results presented in Section IV. Limitations on cost function structure are a typical bottleneck in MPC and pose a similar challenge to our approach. However, it can be addressed by using rich function approximations as the stage cost in MPC schemes, such as convex neural networks in [18].

Additionally, we fixed the terminal cost as 2 norm of tracking error in the simulated examples and had a long MPC horizon to downplay its effect on MPC solutions. According to theorem 1, the optimal parameters $\boldsymbol{\theta}^*$ are

such that terminal cost is the optimal value function $V^*(s)$. However, a simplified value function approximation can be learned from $V_\phi(\phi)$ and further considered as a terminal cost for the learned MPC scheme. Finally, even though a rich data set is required for building a good value function approximation, a local value function estimate can be built from a sparse data distribution. Such a local value function could be used to nudge the MPC parameters toward better performance iteratively, which needs further scrutiny along the lines of the policy iteration class of methods.

VI. Conclusion

We present an approach to formulate performance-oriented MPC schemes directly from data using RL methods. We essentially reduced the problem of deriving a high-performance MPC scheme to a supervised learning problem using a value function approximated from data. The proposed approach derives an MPC scheme in an offline way from big data, bypassing the complexities of solving MPC schemes or having to interact with the real system. Evaluations of our approach to the simulated experiments show promising results by learning a near-optimal MPC scheme from the collected dataset. In further work, we aim to apply our approach to datasets from real-world applications.

References

[1] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*. Nob Hill Publishing Madison, WI, 2017, vol. 2.

[2] Z. Wu, A. Tran, D. Rincon, and P. D. Christofides, "Machine learning-based predictive control of nonlinear processes. part i: theory," *AIChE Journal*, vol. 65, no. 11, p. e16729, 2019.

[3] S. Lucia and B. Karg, "A deep learning-based approach to robust nonlinear model predictive control," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 511–516, 2018.

[4] D. Piga, M. Forgione, S. Formentin, and A. Bemporad, "Performance-oriented model learning for data-driven mpc design," *IEEE control systems letters*, vol. 3, no. 3, pp. 577–582, 2019.

[5] S. Gros and M. Zanon, "Data-driven economic nmpc using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 636–648, 2019.

[6] W. Cai, A. B. Kordabad, H. N. Esfahani, A. M. Lekkas, and S. Gros, "Mpc-based reinforcement learning for a simplified freight mission of autonomous surface vehicles," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 2990–2995.

[7] A. B. Kordabad, W. Cai, and S. Gros, "Mpc-based reinforcement learning for economic problems with application to battery storage," in *2021 European Control Conference (ECC)*. IEEE, 2021, pp. 2573–2578.

[8] W. Cai, H. N. Esfahani, A. B. Kordabad, and S. Gros, "Optimal management of the peak power penalty for smart grids using mpc-based reinforcement learning," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 6365–6370.

[9] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[10] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1179–1191, 2020.

[11] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims, "Morel: Model-based offline reinforcement learning," *Advances in neural information processing systems*, vol. 33, pp. 21 810–21 823, 2020.

[12] Y. Wu, G. Tucker, and O. Nachum, "Behavior regularized offline reinforcement learning," *arXiv preprint arXiv:1911.11361*, 2019.

[13] R. Agarwal, D. Schuurmans, and M. Norouzi, "An optimistic perspective on offline reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 104–114.

[14] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[15] A. B. Kordabad, M. Zanon, and S. Gros, "Equivalency of optimality criteria of markov decision process and model predictive control," *arXiv preprint arXiv:2210.04302*, 2022.

[16] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," *arXiv preprint arXiv:2004.07219*, 2020.

[17] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[18] K. Seel, A. B. Kordabad, J. T. Gravdahl, and S. Gros, "Convex neural network-based cost modifications for learning model predictive control," *IEEE Open Journal of Control Systems [accepted]*, 2022.

Bibliography

[1] Pieter Abbeel, Adam Coates, and Andrew Y Ng. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 29(13):1608–1639, 2010.

[2] Pieter Abbeel, Morgan Quigley, and Andrew Y Ng. Using inaccurate models in reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 1–8, 2006.

[3] Fares J Abu-Dakka, Bojan Nemec, Jimmy A Jørgensen, Thiusius R Savarimuthu, Norbert Krüger, and Aleš Ude. Adaptation of manipulation skills in physical contact with the environment to reference force profiles. *Autonomous Robots*, 39(2):199–217, 2015.

[4] Fares J Abu-Dakka and Matteo Saveriano. Variable impedance control and learning—a review. *Frontiers in Robotics and AI*, 7, 2020.

[5] Alin Albu-Schaffer and Gerd Hirzinger. Cartesian impedance control techniques for torque controlled light-weight robots. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, volume 1, pages 657–663. IEEE, 2002.

[6] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *European Control Conference (ECC)*, pages 3420–3431. IEEE, 2019.

[7] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[8] Akhil Anand, Katrine Seel, Vilde Gjærum, Anne Håkansson, Haakon Robinson, and Aya Saad. Safe learning for control using control lyapunov functions and control barrier functions: A review. *Procedia Computer Science*, 192:3987–3997, 2021.

[9] Akhil S Anand, Fares J Abu-Dakka, and Jan Tommy Gravdahl. Deep model predictive variable impedance control. *arXiv preprint arXiv:2209.09614*, 2022.

[10] Akhil S Anand, Martin Hagen Myrestrand, and Jan Tommy Gravdahl. Evaluation of variable impedance-and hybrid force/motioncontrollers for learning force tracking skills. In *2022 IEEE/SICE International Symposium on System Integration (SII)*, pages 83–89. IEEE, 2022.

[11] Akhil S Anand, Andreas Østvik, Esten Ingar Grøtli, Marialena Vagia, and Jan Tommy Gravdahl. Real-time temporal adaptation of dynamic movement primitives for moving targets. In *2021 20th International Conference on Advanced Robotics (ICAR)*, pages 261–268. IEEE, 2021.

[12] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[13] Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. Bettering operation of robots by learning. *Journal of Robotic systems*, 1(2):123–140, 1984.

[14] Zvi Artstein. Stabilization with relaxed controls. *Nonlinear Analysis: Theory, Methods & Applications*, 7(11):1163–1173, 1983.

[15] 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.

[16] Anil Aswani, Humberto Gonzalez, S Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, 2013.

[17] 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.

[18] Maciej Bednarczyk, Hassan Omran, and Bernard Bayle. Passivity filter for variable impedance control. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7159–7164. IEEE, 2020.

[19] Richard Bellman. A markovian decision process. *Journal of mathematics and mechanics*, pages 679–684, 1957.

[20] Cristian C Beltran-Hernandez, Damien Petit, Ixchel G Ramirez-Alpizar, and Kensuke Harada. Variable compliance control for robotic peg-in-hole assembly: A deep-reinforcement-learning approach. *Applied Sciences*, 10(19):6923, 2020.

[21] Cristian Camilo Beltran-Hernandez, Damien Petit, Ixchel Georgina Ramirez-Alpizar, Takayuki Nishi, Shinichi Kikuchi, Takamitsu Matsubara, and Kensuke Harada. Learning force control for contact-rich manipulation tasks with rigid position-controlled robots. *IEEE Robotics and Automation Letters*, 5(4):5709–5716, 2020.

[22] Felix Berkenkamp, Riccardo Moriconi, Angela P Schoellig, and Andreas Krause. Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes. In *Conference on Decision and Control (CDC)*, pages 4661–4666. IEEE, 2016.

[23] Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. *Advances in neural information processing systems*, 30, 2017.

[24] Felix Berkenkamp, Matteo Turchetta, Angela P Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. *arXiv preprint arXiv:1705.08551*, 2017.

[25] Dimitri Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific, 2019.

[26] Mohak Bhardwaj, Sanjiban Choudhury, and Byron Boots. Blending mpc & value function approximation for efficient reinforcement learning. *arXiv preprint arXiv:2012.05909*, 2020.

[27] Aude Billard and Danica Kragic. Trends and challenges in robot manipulation. *Science*, 364(6446):eaat8414, 2019.

[28] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer Verlag, 2006.

[29] Emilio Bizzi, Neri Accornero, William Chapple, and Neville Hogan. Posture control and trajectory formation during arm movement. *Journal of Neuroscience*, 4(11):2738–2744, 1984.

[30] Miroslav Bogdanovic, Majid Khadiv, and Ludovic Righetti. Learning variable impedance control for contact sensitive tasks. *IEEE Robotics and Automation Letters*, 5(4):6129–6136, 2020.

[31] Zdravko I Botev, Dirk P Kroese, Reuven Y Rubinstein, and Pierre L'Ecuyer. The cross-entropy method for optimization. In *Handbook of statistics*, volume 31, pages 35–59. Elsevier, 2013.

[32] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[33] Lukas Brunke, Melissa Greeff, Adam W Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5:411–444, 2022.

[34] Jonas Buchli, Freek Stulp, Evangelos Theodorou, and Stefan Schaal. Learning variable impedance control. *The International Journal of Robotics Research*, 30(7):820–833, 2011.

[35] Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. Sample-efficient reinforcement learning with stochastic ensemble value expansion. *Advances in neural information processing systems*, 31, 2018.

[36] Christof Büskens and Helmut Maurer. Sensitivity analysis and real-time optimization of parametric nonlinear programming problems. In *Online Optimization of Large Scale Systems*, pages 3–16. Springer, 2001.

[37] Wenqi Cai, Hossein N Esfahani, Arash B Kordabad, and Sébastien Gros. Optimal management of the peak power penalty for smart grids using mpc-based reinforcement learning. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 6365–6370. IEEE, 2021.

[38] Wenqi Cai, Arash B Kordabad, Hossein N Esfahani, Anastasios M Lekkas, and Sébastien Gros. Mpc-based reinforcement learning for a simplified freight mission of autonomous surface vehicles. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 2990–2995. IEEE, 2021.

[39] Edoardo Caldarelli, Adrià Colomé, and Carme Torras. Perturbation-based stiffness inference in variable impedance control. *IEEE Robotics and Automation Letters*, 7(4):8823–8830, 2022.

[40] Sylvain Calinon, Irene Sardellitti, and Darwin G Caldwell. Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 249–254. IEEE, 2010.

[41] Fernando Castañeda, Jason J Choi, Bike Zhang, Claire J Tomlin, and Koushil Sreenath. Gaussian process-based min-norm stabilizing controller for control-affine systems with uncertain input effects. *arXiv preprint arXiv:2011.07183*, 2020.

[42] Konstantinos Chatzilygeroudis, Roberto Rama, Rituraj Kaushik, Dorian Goepp, Vassilis Vassiliades, and Jean-Baptiste Mouret. Black-box data-efficient policy search for robotics. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 51–58. IEEE, 2017.

[43] Konstantinos Chatzilygeroudis, Vassilis Vassiliades, Freek Stulp, Sylvain Calinon, and Jean-Baptiste Mouret. A survey on policy search algorithms for learning robot controllers in a handful of trials. *IEEE Transactions on Robotics*, 36(2):328–347, 2019.

[44] Chien-Chern Cheah and Danwei Wang. Learning impedance control for robotic manipulators. *IEEE Transactions on robotics and automation*, 14(3):452–465, 1998.

[45] Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3387–3395, 2019.

[46] Mingshan Chi, Yufeng Yao, Yaxin Liu, and Ming Zhong. Learning, generalization, and obstacle avoidance with dynamic movement primitives and dynamic potential fields. *Applied Sciences*, 9(8):1535, 2019.

[47] S Chiaverini and L Sciavicco. Force/position control of manipulators in task space with dominance in force. *IFAC Proceedings Volumes*, 21(16):137–143, 1988.

[48] Jason Choi, Fernando Castaneda, Claire J Tomlin, and Koushil Sreenath. Reinforcement learning for safety-critical control under model uncertainty,

using control lyapunov functions and control barrier functions. *arXiv preprint arXiv:2004.07584*, 2020.

[49] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. *arXiv preprint arXiv:1805.07708*, 2018.

[50] Yinlam Chow, Ofir Nachum, Aleksandra Faust, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. Lyapunov-based safe policy optimization for continuous control. *arXiv preprint arXiv:1901.10031*, 2019.

[51] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *arXiv preprint arXiv:1805.12114*, 2018.

[52] Ignasi Clavera, Yao Fu, and Pieter Abbeel. Model-augmented actor-critic: Backpropagating through paths. In *International Conference on Learning Representations*, 2020.

[53] Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472. Citeseer, 2011.

[54] Stefan Depeweg, Jose-Miguel Hernandez-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pages 1184–1193. PMLR, 2018.

[55] Vikas Dhiman, Mohammad Javad Khojasteh, Massimo Franceschetti, and Nikolay Atanasov. Control barriers in bayesian learning of system dynamics. *arXiv preprint arXiv:2012.14964*, 2020.

[56] Fotios Dimeas and Nikos Aspragathos. Reinforcement learning of variable admittance control for human-robot co-manipulation. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1011–1016. IEEE, 2015.

[57] Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021.

[58] Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I Jordan, Joseph E Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning. *arXiv preprint arXiv:1803.00101*, 2018.

[59] Federica Ferraguti, Cristian Secchi, and Cesare Fantuzzi. A tank-based approach to impedance control with variable stiffness. In *2013 IEEE international conference on robotics and automation*, pages 4948–4953. IEEE, 2013.

[60] Randy A Freeman and James A Primbs. Control lyapunov functions: New ideas from an old source. In *Conference on Decision and Control (CDC)*, volume 4, pages 3926–3931. IEEE, 1996.

[61] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[62] Andrej Gams, Bojan Nemec, Auke Jan Ijspeert, and Aleš Ude. Coupling movement primitives: Interaction with the environment and bimanual tasks. *IEEE Transactions on Robotics*, 30(4):816–830, 2014.

[63] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[64] Jeremy H Gillula and Claire J Tomlin. Guaranteed safe online learning via reachability: tracking a ground target using a quadrotor. In *International Conference on Robotics and Automation (ICRA)*, pages 2723–2730. IEEE, 2012.

[65] Michele Ginesi, Nicola Sansonetto, and Paolo Fiorini. Overcoming some drawbacks of dynamic movement primitives. *arXiv*, pages arXiv–1908, 2019.

[66] Tobias Gold, Andreas Völz, and Knut Graichen. Model predictive interaction control for robotic manipulation tasks. *IEEE Transactions on Robotics*, 2022.

[67] Sébastien Gros and Mario Zanon. Data-driven economic nmpc using reinforcement learning. *IEEE Transactions on Automatic Control*, 65(2):636–648, 2019.

[68] Sebastien Gros and Mario Zanon. Towards safe reinforcement learning using nmpc and policy gradients: Part ii-deterministic case. *arXiv preprint arXiv:1906.04034*, 2019.

[69] Sebastien Gros, Mario Zanon, and Alberto Bemporad. Safe reinforcement learning via projection on a safe set: How to achieve optimality? *arXiv preprint arXiv:2004.00915*, 2020.

[70] Vijaykumar Gullapalli, Judy A Franklin, and Hamid Benbrahim. Acquiring robot skills via reinforcement learning. *IEEE Control Systems Magazine*, 14(1):13–24, 1994.

[71] Kevin Haninger, Christian Hegeler, and Luka Peternel. Model predictive control with gaussian processes for flexible multi-modal physical human robot interaction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6948–6955. IEEE, 2022.

[72] Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

[73] Nikolaus Hansen. Benchmarking a bi-population cma-es on the bbob-2009 function testbed. In *Proceedings of the 11th annual conference companion on genetic and evolutionary computation conference: late breaking papers*, pages 2389–2396, 2009.

[74] Nikolaus Hansen, André SP Niederberger, Lino Guzzella, and Petros Koumoutsakos. A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. *IEEE Transactions on Evolutionary Computation*, 13(1):180–197, 2008.

[75] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

[76] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. Elsevier, 1992.

[77] Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. Learning continuous control policies by stochastic value gradients. *Advances in neural information processing systems*, 28, 2015.

[78] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[79] Lukas Hewing, Kim P Wabersich, Marcel Menner, and Melanie N Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:269–296, 2020.

[80] Archibald Vivian Hill. The series elastic component of muscle. *Proceedings of the Royal Society of London. Series B, Biological Sciences*, pages 273–280, 1950.

[81] Ashley Hill, Antonin Raffin, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, Rene Traore, Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. Stable baselines. https://github.com/hill-a/stable-baselines, 2018.

[82] Heiko Hoffmann, Peter Pastor, Dae-Hyung Park, and Stefan Schaal. Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance. In *2009 IEEE International Conference on Robotics and Automation*, pages 2587–2592. IEEE, 2009.

[83] Neville Hogan. Impedance control: An approach to manipulation. In *1984 American control conference*, pages 304–313. IEEE, 1984.

[84] Neville Hogan. An organizing principle for a class of voluntary movements. *Journal of neuroscience*, 4(11):2745–2754, 1984.

[85] Neville Hogan. On the stability of manipulators performing contact tasks. *IEEE Journal on Robotics and Automation*, 4(6):677–686, 1988.

[86] H-P Huang and S-S Chen. Compliant motion control of robots by using variable impedance. *The International Journal of Advanced Manufacturing Technology*, 7(6):322–332, 1992.

[87] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

[88] Auke J Ijspeert, Jun Nakanishi, and Stefan Schaal. Learning attractor landscapes for learning motor primitives. In *Advances in neural information processing systems*, pages 1547–1554, 2003.

[89] Auke Jan Ijspeert, Jun Nakanishi, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Dynamical movement primitives: learning attractor models for motor behaviors. *Neural computation*, 25(2):328–373, 2013.

[90] Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, volume 2, pages 1398–1403. IEEE, 2002.

[91] Ryojun Ikeura and Hikaru Inooka. Variable impedance control of a robot for cooperation with a human. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, volume 3, pages 3097–3102. IEEE, 1995.

[92] Ryojun Ikeura, Tomoki Moriguchi, and Kazuki Mizutani. Optimal variable impedance control for a robot and its application to lifting an object with a human. In *Proceedings. 11th IEEE International Workshop on Robot and Human Interactive Communication*, pages 500–505. IEEE, 2002.

[93] Pushpak Jagtap, George J Pappas, and Majid Zamani. Control barrier functions for unknown nonlinear systems using gaussian processes. In *Conference on Decision and Control (CDC)*, pages 3699–3704. IEEE, 2020.

[94] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

[95] Wanxin Jin, Zhaoran Wang, Zhuoran Yang, and Shaoshuai Mou. Neural certificates for safe control policies. *arXiv preprint arXiv:2006.08465*, 2020.

[96] Yaochu Jin and Jürgen Branke. Evolutionary optimization in uncertain environments-a survey. *IEEE Transactions on evolutionary computation*, 9(3):303–317, 2005.

[97] Zhehao Jin, Andong Liu, Wen-an Zhang, and Li Yu. An optimal variable impedance control with consideration of the stability. *IEEE Robotics and Automation Letters*, 7(2):1737–1744, 2022.

[98] Lionel Jouffe. Fuzzy inference system learning by reinforcement methods. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 28(3):338–355, 1998.

[99] Mrinal Kalakrishnan, Ludovic Righetti, Peter Pastor, and Stefan Schaal. Learning force control policies for compliant manipulation. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4639–4644. IEEE, 2011.

[100] Scott D Kennedy and Andrew B Schwartz. Stiffness as a control factor for object manipulation. *Journal of Neurophysiology*, 122(2):707–720, 2019.

[101] Shahbaz Abdul Khader, Hang Yin, Pietro Falco, and Danica Kragic. Stability-guaranteed reinforcement learning for contact-rich manipulation. *IEEE Robotics and Automation Letters*, 6(1):1–8, 2020.

[102] Hassan K Khalil. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002.

[103] Hassan K Khalil and Jessy W Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002.

[104] Md Al-Masrur Khan, Md Rashed Jaowad Khan, Abul Tooshil, Niloy Sikder, MA Parvez Mahmud, Abbas Z Kouzani, and Abdullah-Al Nahid. A systematic review on reinforcement learning-based robotics within the last decade. *IEEE Access*, 8:176598–176623, 2020.

[105] S Mohammad Khansari-Zadeh, Klas Kronander, and Aude Billard. Modeling robot discrete movements with state-varying stiffness and damping: A framework for integrated motion generation and impedance control. *Proceedings of robotics: Science and systems X (RSS 2014)*, 10:2014, 2014.

[106] Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[107] Mohammad Javad Khojasteh, Vikas Dhiman, Massimo Franceschetti, and Nikolay Atanasov. Probabilistic safety constraints for learned high relative degree system dynamics. In *Learning for Dynamics and Control*, pages 781–792. PMLR, 2020.

[108] Mohammad Javad Khojasteh, Vikas Dhiman, Massimo Franceschetti, and Nikolay Atanasov. Probabilistic safety constraints for learned high relative degree system dynamics. In *Learning for Dynamics and Control*, pages 781–792. PMLR, 2020.

[109] Byungchan Kim, Jooyoung Park, Shinsuk Park, and Sungchul Kang. Impedance learning for robotic contact tasks using natural actor-critic algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(2):433–443, 2009.

[110] Mincheol Kim, Scott Niekum, and Ashish D Deshpande. Scape: Learning stiffness control from augmented position control experiences. In *Conference on Robot Learning*, pages 1512–1521. PMLR, 2022.

[111] Seungsu Kim, Elena Gribovskaya, and Aude Billard. Learning motion dynamics to catch a moving object. In *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pages 106–111. IEEE, 2010.

[112] Youngmin Kim, Richard Allmendinger, and Manuel López-Ibáñez. Safe learning and optimization techniques: Towards a survey of the state of the art. *arXiv preprint arXiv:2101.09505*, 2021.

[113] Donald E Knuth. On the lambert w function. *Advances in Computational Mathematics*, 5(1):329–359, 1996.

[114] 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.

[115] Jens Kober, Katharina Mülling, Oliver Krömer, Christoph H Lampert, Bernhard Schölkopf, and Jan Peters. Movement templates for learning of hitting and batting. In *2010 IEEE International Conference on Robotics and Automation*, pages 853–858. IEEE, 2010.

[116] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2149–2154. IEEE, 2004.

[117] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

[118] Arash Bahari Kordabad, Wenqi Cai, and Sebastien Gros. Mpc-based reinforcement learning for economic problems with application to battery storage. In *2021 European Control Conference (ECC)*, pages 2573–2578. IEEE, 2021.

[119] Arash Bahari Kordabad, Wenqi Cai, and Sebastien Gros. Multi-agent battery storage management using mpc-based reinforcement learning. In *2021 IEEE Conference on Control Technology and Applications (CCTA)*, pages 57–62. IEEE, 2021.

[120] Arash Bahari Kordabad, Mario Zanon, and Sébastien Gros. Equivalency of optimality criteria of markov decision process and model predictive control. *arXiv preprint arXiv:2210.04302*, 2022.

[121] Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. Robot motor skill coordination with em-based reinforcement learning. In *2010 IEEE/RSJ international conference on intelligent robots and systems*, pages 3232–3237. IEEE, 2010.

[122] Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. Reinforcement learning in robotics: Applications and real-world challenges. *Robotics*, 2(3):122–148, 2013.

[123] Leonidas Koutras and Zoe Doulgeri. A correct formulation for the orientation dynamic movement primitives for robot control in the cartesian space. In *Conference on Robot Learning*, pages 293–302, 2020.

[124] Leonidas Koutras and Zoe Doulgeri. Dynamic movement primitives for moving goals with temporal scaling adaptation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 144–150. IEEE, 2020.

[125] Aljaž Kramberger, Iñigo Iturrate, Miha Deniša, Simon Mathiesen, and Christoffer Sloth. Adapting learning by demonstration for robot based part feeding applications. In *2020 IEEE/SICE International Symposium on System Integration (SII)*, pages 954–959. IEEE, 2020.

[126] Aljaž Kramberger, Erfan Shahriari, Andrej Gams, Bojan Nemec, Aleš Ude, and Sami Haddadin. Passivity based iterative learning of admittance-coupled dynamic movement primitives for interaction with changing environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6023–6028. IEEE, 2018.

[127] Oliver Kroemer, Scott Niekum, and George Konidaris. A review of robot learning for manipulation: Challenges, representations, and algorithms. *J. Mach. Learn. Res.*, 22:30–1, 2021.

[128] Klas Kronander and Aude Billard. Stability considerations for variable impedance control. *IEEE Transactions on Robotics*, 32(5):1298–1305, 2016.

[129] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. *arXiv preprint arXiv:1802.10592*, 2018.

[130] Michail G Lagoudakis and Ronald Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

[131] Gustavo JG Lahr, Joao VR Soares, Henrique B Garcia, Adriano AG Siqueira, and Glauco AP Caurin. Understanding the implementation of impedance control in industrial robots. In *2016 XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR)*, pages 269–274. IEEE, 2016.

[132] Nicholas C Landolfi, Garrett Thomas, and Tengyu Ma. A model-based approach for sample-efficient multi-task reinforcement learning. *arXiv preprint arXiv:1907.04964*, 2019.

[133] Przemyslaw A Lasota, Terrence Fong, Julie A Shah, et al. *A survey of methods for safe human-robot interaction*. Now Publishers, 2017.

[134] Dongheui Lee and Christian Ott. Incremental kinesthetic teaching of motion primitives using the motion refinement tube. *Autonomous Robots*, 31(2):115–131, 2011.

[135] Sergey Levine and Vladlen Koltun. Guided policy search. In *International conference on machine learning*, pages 1–9. PMLR, 2013.

[136] C Li, Z Zhang, G Xia, X Xie, and Q Zhu. Efficient learning variable impedance control for industrial robots. *Bulletin of the Polish Academy of Sciences. Technical Sciences*, 67(2), 2019.

[137] Chao Li, Zhi Zhang, Guihua Xia, Xinru Xie, and Qidan Zhu. Efficient force control learning system for industrial robots based on variable impedance control. *Sensors*, 18(8):2539, 2018.

[138] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[139] Dennis V Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956.

[140] Winfried Lohmiller and Jean-Jacques E Slotine. On contraction analysis for non-linear systems. *Automatica*, 34(6):683–696, 1998.

[141] Jianlan Luo, Eugen Solowjow, Chengtao Wen, Juan Aparicio Ojea, Alice M Agogino, Aviv Tamar, and Pieter Abbeel. Reinforcement learning on variable impedance controller for high-precision robotic assembly. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3080–3087. IEEE, 2019.

[142] José María Manzano, Daniel Limon, David Muñoz de la Peña, and Jan-Peter Calliess. Robust learning-based mpc for nonlinear constrained systems. *Automatica*, 117:108948, 2020.

[143] Horacio J Marquez. *Nonlinear control systems: analysis and design*, volume 161. John Wiley Hobokenˆ eN. JNJ, 2003.

[144] Roberto Martín-Martín, Michelle A Lee, Rachel Gardner, Silvio Savarese, Jeannette Bohg, and Animesh Garg. Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1010–1017. IEEE, 2019.

[145] Zahra Marvi and Bahare Kiumarsi. Safe reinforcement learning: A control barrier function optimization approach. *International Journal of Robust and Nonlinear Control*, 31(6):1923–1940, 2021.

[146] Takamitsu Matsubara, Sang-Ho Hyon, and Jun Morimoto. Learning stylistic dynamic movement primitives from multiple demonstrations. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1277–1283. IEEE, 2010.

[147] Takamitsu Matsubara, Sang-Ho Hyon, and Jun Morimoto. Learning parametric dynamic movement primitives from multiple demonstrations. *Neural networks*, 24(5):493–500, 2011.

[148] Roger McFarlane. A survey of exploration strategies in reinforcement learning. *McGill University*, 2018.

[149] Youssef Michel, Christian Ott, and Dongheui Lee. Safety-aware hierarchical passivity-based variable compliance control for redundant manipulators. *IEEE Transactions on Robotics*, 38(6):3899–3916, 2022.

[150] Youssef Michel, Rahaf Rahal, Claudio Pacchierotti, Paolo Robuffo Giordano, and Dongheui Lee. Bilateral teleoperation with adaptive impedance control for contact tasks. *IEEE Robotics and Automation Letters*, 6(3):5429–5436, 2021.

[151] Maria Vittoria Minniti, Ruben Grandia, Kevin Fäh, Farbod Farshidian, and Marco Hutter. Model predictive robot-environment interaction control for mobile manipulation tasks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1651–1657. IEEE, 2021.

[152] Mayank Mittal, Marco Gallieri, Alessio Quaglino, Seyed Sina Mirrazavi Salehian, and Jan Koutník. Neural lyapunov model predictive control. *arXiv preprint arXiv:2002.10451*, 2020.

[153] Thomas M Moerland, Joost Broekens, and Catholijn M Jonker. Model-based reinforcement learning: A survey. *arXiv preprint arXiv:2006.16712*, 2020.

[154] Thomas M Moerland, Joost Broekens, Aske Plaat, Catholijn M Jonker, et al. Model-based reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 16(1):1–118, 2023.

[155] Marwan Qaid Mohammed, Lee Chung Kwek, Shing Chyi Chua, Arafat Al-Dhaqm, Saeid Nahavandi, Taiseer Abdalla Elfadil Eisa, Muhammad Fahmi Miskon, Mohammed Nasser Al-Mhiqani, Abdulalem Ali, Mohammed Abaker, et al. Review of learning-based robotic manipulation in cluttered environments. *Sensors*, 22(20):7938, 2022.

[156] Seyed Sajad Mousavi, Michael Schukat, and Enda Howley. Deep reinforcement learning: an overview. In *Proceedings of SAI Intelligent Systems Conference*, pages 426–440. Springer, 2016.

[157] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

[158] Hai Nguyen and Hung La. Review of deep reinforcement learning for robot manipulation. In *2019 Third IEEE International Conference on Robotic Computing (IRC)*, pages 590–595. IEEE, 2019.

[159] Duy Nguyen-Tuong and Jan Peters. Model learning for robot control: a survey. *Cognitive processing*, 12(4):319–340, 2011.

[160] Andreas Østvik, Lars Eirik Bø, and Erik Smistad. Echobot: An open-source robotic ultrasound system. *Proc. IPCAI*, pages 1–4, 2019.

[161] Christian Ott, Ranjan Mukherjee, and Yoshihiko Nakamura. Unified impedance and admittance control. In *2010 IEEE international conference on robotics and automation*, pages 554–561. IEEE, 2010.

[162] Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*, 2015.

[163] Dae-Hyung Park, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields. In *Humanoids 2008-8th IEEE-RAS International Conference on Humanoid Robots*, pages 91–98. IEEE, 2008.

[164] Peter Pastor, Heiko Hoffmann, Tamim Asfour, and Stefan Schaal. Learning and generalization of motor skills by learning from demonstration. In *2009*

IEEE International Conference on Robotics and Automation, pages 763–768. IEEE, 2009.

[165] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

[166] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *International conference on machine learning*, pages 5062–5071. PMLR, 2019.

[167] Theodore J Perkins and Andrew G Barto. Lyapunov design for safe reinforcement learning. *Journal of Machine Learning Research*, 3(Dec):803–832, 2002.

[168] Jan Peters and Stefan Schaal. Policy gradient methods for robotics. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2219–2225. IEEE, 2006.

[169] Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

[170] 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.

[171] Miguel Prada and Anthony Remazeilles. Dynamic movement primitives for human robot interaction. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, workshop on Robot Motion Planning: online, reactive and in Real-time, Algarve, Portugal*, 2012.

[172] Akshara Rai, Franziska Meier, Auke Ijspeert, and Stefan Schaal. Learning coupling terms for obstacle avoidance. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 512–518. IEEE, 2014.

[173] James Blake Rawlings, David Q Mayne, and Moritz Diehl. *Model predictive control: theory, computation, and design*, volume 2. Nob Hill Publishing Madison, WI, 2017.

[174] MH Reibert. Hybrid position/force control of manipulators. *ASME, J. of Dynamic Systems, Measurement, and Control*, 103:2–12, 1981.

[175] Joel Rey, Klas Kronander, Farbod Farshidian, Jonas Buchli, and Aude Billard. Learning motions from demonstrations and rewards with time-invariant dynamical systems based policies. *Autonomous Robots*, 42(1):45–64, 2018.

[176] Alexander Robey, Haimin Hu, Lars Lindemann, Hanwen Zhang, Dimos V Dimarogonas, Stephen Tu, and Nikolai Matni. Learning control barrier functions from expert demonstrations. In *Conference on Decision and Control (CDC)*, pages 3717–3724. IEEE, 2020.

[177] Muhammad Zakiyullah Romdlony and Bayu Jayawardhana. Stabilization with guaranteed safety using control lyapunov–barrier function. *Automatica*, 66:39–47, 2016.

[178] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[179] Loris Roveda, Jeyhoon Maskani, Paolo Franceschi, Arash Abdi, Francesco Braghin, Lorenzo Molinari Tosatti, and Nicola Pedrocchi. Model-based reinforcement learning variable impedance control for human-robot collaboration. *Journal of Intelligent & Robotic Systems*, 100(2):417–433, 2020.

[180] Leonel Rozo, Sylvain Calinon, Darwin Caldwell, Pablo Jiménez, and Carme Torras. Learning collaborative impedance-based robot behaviors. In *Proceedings of the AAAI conference on artificial intelligence*, volume 27, pages 1422–1428, 2013.

[181] Daniela Rus and Michael T Tolley. Design, fabrication and control of soft robots. *Nature*, 521(7553):467–475, 2015.

[182] Stuart J Russell and Peter Norvig. Artificial intelligence: a modern approach. malaysia, 2016.

[183] Manuel S Santos and Jesus Vigo-Aguiar. Analysis of a numerical dynamic programming algorithm applied to economic models. *Econometrica*, pages 409–426, 1998.

[184] Matteo Saveriano, Fares J Abu-Dakka, Aljaz Kramberger, and Luka Peternel. Dynamic movement primitives in robotics: A tutorial survey. *arXiv preprint arXiv:2102.03861*, 2021.

[185] Matteo Saveriano, Sang-ik An, and Dongheui Lee. Incremental kinesthetic teaching of end-effector and null-space motion primitives. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3570–3575. IEEE, 2015.

[186] Matteo Saveriano and Dongheui Lee. Learning barrier functions for constrained motion planning with dynamical systems. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 112–119. IEEE, 2019.

[187] Matteo Saveriano, Yuchao Yin, Pietro Falco, and Dongheui Lee. Data-efficient control policy search using residual dynamics learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4709–4715. IEEE, 2017.

[188] Stefan Schaal, Shinya Kotosaka, and Dagmar Sternad. Nonlinear dynamical systems as movement primitives. In *IEEE international conference on humanoid robotics*, pages 1–11, 2000.

[189] Stefan Schaal, Jan Peters, Jun Nakanishi, and Auke Ijspeert. Learning movement primitives. In *Robotics research. the eleventh international symposium*, pages 561–572. Springer, 2005.

[190] Marie Schumacher, Janis Wojtusch, Philipp Beckerle, and Oskar von Stryk. An introductory review of active compliant control. *Robotics and Autonomous Systems*, 119:185–200, 2019.

[191] Katrine Seel, Esten I Grøtli, Signe Moe, Jan T Gravdahl, and Kristin Y Pettersen. Neural nework-based model predictive control with input-to-state stability. In *American Control Conference (ACC)*. IEEE, 2021.

[192] Katrine Seel, Arash Bahari Kordabad, Sebastien Gros, and Jan Tommy Gravdahl. Convex neural network-based cost modifications for learning model predictive control. *IEEE Open Journal of Control Systems*, 2022.

[193] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, pages 8583–8592. PMLR, 2020.

[194] Radhe Shyam Sharma, Santosh Shukla, Hamad Karki, Amit Shukla, Laxmidhar Behera, and KS Venkatesh. Dmp based trajectory tracking for a nonholonomic mobile robot with automatic goal adaptation and obstacle

avoidance. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8613–8619. IEEE, 2019.

[195] Pranav Shyam, Wojciech Jaśkowski, and Faustino Gomez. Model-based active exploration. In *International conference on machine learning*, pages 5779–5788. PMLR, 2019.

[196] Saif Sidhik. Franka ROS Interface: A ROS/Python API for controlling and managing the Franka Emika Panda robot (real and simulated)., December 2020.

[197] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[198] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014.

[199] Bharat Singh, Rajesh Kumar, and Vinay Pratap Singh. Reinforcement learning in robotic applications: a comprehensive survey. *Artificial Intelligence Review*, 55(2):945–990, 2022.

[200] Eduardo D Sontag. A 'universal'construction of artstein's theorem on nonlinear stabilization. *Systems & control letters*, 13(2):117–123, 1989.

[201] Mohit Srinivasan, Amogh Dabholkar, Samuel Coogan, and Patricio Vela. Synthesis of control barrier functions using a supervised machine learning approach. *arXiv preprint arXiv:2003.04950*, 2020.

[202] Stefano Stramigioli. *Modeling and IPC Control of Interactive Mechanical Systems – A Coordinate-Free Approach*, volume 266 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag London, 2001.

[203] Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

[204] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[205] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

[206] Tadele Shiferaw Tadele, Theo de Vries, and Stefano Stramigioli. The safety of domestic robotics: A survey of various safety-related publications. *IEEE robotics & automation magazine*, 21(3):134–142, 2014.

[207] Andrew Taylor, Andrew Singletary, Yisong Yue, and Aaron Ames. Learning for safety-critical control with control barrier functions. In *Learning for Dynamics and Control*, pages 708–717. PMLR, 2020.

[208] Andrew J Taylor, Victor D Dorobantu, Hoang M Le, Yisong Yue, and Aaron D Ames. Episodic learning with control lyapunov functions for uncertain robotic systems. *arXiv preprint arXiv:1903.01577*, 2019.

[209] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.

[210] Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. Learning policy improvements with path integrals. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 828–835. JMLR Workshop and Conference Proceedings, 2010.

[211] Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

[212] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[213] Aleš Ude, Bojan Nemec, Tadej Petrić, and Jun Morimoto. Orientation in cartesian space dynamic movement primitives. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2997–3004. IEEE, 2014.

[214] Jonas Umlauft, Armin Lederer, and Sandra Hirche. Learning stable gaussian process state space models. In *American Control Conference (ACC)*, pages 1499–1504. IEEE, 2017.

[215] Patrick Varin, Lev Grossman, and Scott Kuindersma. A comparison of action spaces for learning manipulation tasks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6015–6021. IEEE, 2019.

[216] Luigi Villani and Joris De Schutter. Force control. In *Springer handbook of robotics*, pages 195–220. Springer, 2016.

[217] Kim P Wabersich and Melanie N Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. *arXiv preprint arXiv:1812.05506*, 2018.

[218] Li Wang, Evangelos A Theodorou, and Magnus Egerstedt. Safe learning of quadrotor dynamics using barrier certificates. In *International Conference on Robotics and Automation (ICRA)*, pages 2460–2465. IEEE, 2018.

[219] Sen Wang, Daoyuan Jia, and Xinshuo Weng. Deep reinforcement learning for autonomous driving. *arXiv preprint arXiv:1811.11329*, 2018.

[220] Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

[221] Min Wen and Ufuk Topcu. Constrained cross-entropy method for safe reinforcement learning. *Advances in Neural Information Processing Systems*, 31, 2018.

[222] Patrick M Wensing and Jean-Jacques Slotine. Sparse control for dynamic movement primitives. *IFAC-PapersOnLine*, 50(1):10114–10121, 2017.

[223] Peter Wieland and Frank Allgöwer. Constructive safety using control barrier functions. *IFAC Proceedings Volumes*, 40(12):462–467, 2007.

[224] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[225] Zhe Wu and Panagiotis D Christofides. Control lyapunov-barrier function-based predictive control of nonlinear processes using machine learning modeling. *Computers & Chemical Engineering*, 134:106706, 2020.

[226] Angeliki Zacharaki, Ioannis Kostavelis, Antonios Gasteratos, and Ioannis Dokas. Safety bounds in human robot interaction: A survey. *Safety science*, 127:104667, 2020.

[227] Mario Zanon and Sébastien Gros. Safe reinforcement learning using robust mpc. *IEEE Transactions on Automatic Control*, 66(8):3638–3652, 2020.

[228] Chao Zhai and Hung D Nguyen. Region of attraction for power systems using gaussian process and converse lyapunov function–part i: Theoretical framework and off-line study. *arXiv preprint arXiv:1906.03590*, 2019.

[229] Yueqing Zhang, Bing Chu, and Zhan Shu. A preliminary study on the relationship between iterative learning control and reinforcement learning. *IFAC-PapersOnLine*, 52(29):314–319, 2019.

[230] Hengjun Zhao, Xia Zeng, Taolue Chen, Zhiming Liu, and Jim Woodcock. Learning safe neural network controllers with barrier certificates. In *International Symposium on Dependable Software Engineering: Theories, Tools, and Applications*, pages 177–185. Springer, 2020.

[231] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 737–744. IEEE, 2020.

[232] Lei Zheng, Rui Yang, Zhixuan Wu, Jiesen Pan, and Hui Cheng. Safe learning-based gradient-free model predictive control based on cross-entropy method. *Engineering Applications of Artificial Intelligence*, 110:104731, 2022.

[233] You Zhou, Martin Do, and Tamim Asfour. Coordinate change dynamic movement primitives—a leader-follower approach. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5481–5488. IEEE, 2016.

[234] Zhuangdi Zhu, Kaixiang Lin, and Jiayu Zhou. Transfer learning in deep reinforcement learning: A survey. *arXiv preprint arXiv:2009.07888*, 2020.

ISBN 978-82-326-7003-1 (printed ver.)
ISBN 978-82-326-7002-4 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

NTNU

Norwegian University of
Science and Technology