On Optimal Behavior Under Uncertainty in Humans and Robots

Zur Erlangung des akademischen Grades Doktor-Ingenieur (Dr.-Ing.)
Genehmigte Dissertation von Boris Belousov aus Lipetsk
Tag der Einreichung: 3. Juni 2022, Tag der Prüfung: 18. Juli 2022

1. Gutachten: Prof. Jan Peters, Ph.D.
2. Gutachten: Prof. Dr. Marc Toussaint
Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

Computer Science Department

Intelligent Autonomous Systems Group

On Optimal Behavior Under Uncertainty in Humans and Robots

Accepted doctoral thesis by Boris Belousov

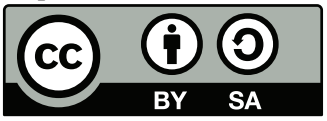
Date of submission: 3. Juni 2022
Date of thesis defense: 18. Juli 2022

Darmstadt

Bitte zitieren Sie dieses Dokument als:
URN: urn:nbn:de:tuda-tuprints-225612
URL: http://tuprints.ulb.tu-darmstadt.de/225612

Dieses Dokument wird bereitgestellt von tuprints,
E-Publishing-Service der TU Darmstadt
http://tuprints.ulb.tu-darmstadt.de
tuprints@ulb.tu-darmstadt.de

Die Veröffentlichung steht unter folgender Creative Commons Lizenz:
Namensnennung – Weitergabe unter gleichen Bedingungen 4.0 International
https://creativecommons.org/licenses/by-sa/4.0/
This work is licensed under a Creative Commons License:
Attribution–ShareAlike 4.0 International
https://creativecommons.org/licenses/by-sa/4.0/



Erklärungen laut Promotionsordnung

§ 8 Abs. 1 lit. c PromO

Ich versichere hiermit, dass die elektronische Version meiner Dissertation mit der schriftlichen Version übereinstimmt.

§ 8 Abs. 1 lit. d PromO

Ich versichere hiermit, dass zu einem vorherigen Zeitpunkt noch keine Promotion versucht wurde. In diesem Fall sind nähere Angaben über Zeitpunkt, Hochschule, Dissertationsthema und Ergebnis dieses Versuchs mitzuteilen.

§ 9 Abs. 1 PromO

Ich versichere hiermit, dass die vorliegende Dissertation selbstständig und nur unter Verwendung der angegebenen Quellen verfasst wurde.

§ 9 Abs. 2 PromO

Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

Darmstadt, 3. Juni 2022

B. Belousov

Abstract

Despite significant progress in robotics and automation in the recent decades, there still remains a noticeable gap in performance compared to humans. Although the computation capabilities are growing every year, and are even projected to exceed the capacities of biological systems, the behaviors generated using current computational paradigms are arguably not catching up with the available resources. Why is that? It appears that we are still lacking some fundamental understanding of how living organisms are making decisions, and therefore we are unable to replicate intelligent behavior in artificial systems.

Therefore, in this thesis, we attempted to develop a framework for modeling human and robot behavior based on statistical decision theory. Different features of this approach, such as risk-sensitivity, exploration, learning, control, were investigated in a number of publications.

First, we considered the problem of learning new skills and developed a framework of entropic regularization of Markov decision processes (MDP). Utilizing a generalized concept of entropy, we were able to realize the trade-off between exploration and exploitation via a choice of a single scalar parameter determining the divergence function.

Second, building on the theory of partially observable Markov decision process (POMDP), we proposed and validated a model of human ball catching behavior. Crucially, information seeking behavior was identified as a key feature enabling the modeling of observed human catches. Thus, entropy reduction was seen to play an important role in skillful human behavior.

Third, having extracted the modeling principles from human behavior and having developed an information-theoretic framework for reinforcement learning, we studied the real-robot applications of the learning-based controllers in tactile-rich manipulation tasks. We investigated vision-based tactile sensors and the capability of learning algorithms to autonomously extract task-relevant features for manipulation tasks. The specific feature of tactile-based control that perception and action are tightly connected at the point of contact, enabled us to gather insights into the strengths and limitations of the statistical learning approach to real-time robotic manipulation.

In conclusion, this thesis presents a series of investigations into the applicability of the statistical decision theory paradigm to modeling the behavior of humans and for

synthesizing the behavior of robots. We conclude that a number of important features related to information processing can be represented and utilized in artificial systems for generating more intelligent behaviors. Nevertheless, these are only the first steps and we acknowledge that the road towards artificial general intelligence and skillful robotic applications will require more innovations and potentially transcendence of the probabilistic modeling paradigm.

Zusammenfassung

Trotz erheblicher Fortschritte in der Robotik und Automatisierung in den letzten Jahrzehnten besteht nach wie vor ein deutlicher Leistungsunterschied zum Menschen. Obwohl die Rechenkapazitäten von Jahr zu Jahr zunehmen und sogar die Kapazitäten biologischer Systeme übersteigen dürften, kann das mit den derzeitigen Rechenparadigmen erzeugte Verhalten wohl nicht mit den verfügbaren Ressourcen mithalten. Woran liegt das? Offenbar fehlt uns noch immer ein grundlegendes Verständnis dafür, wie lebende Organismen Entscheidungen treffen, und deshalb sind wir nicht in der Lage, intelligentes Verhalten in künstlichen Systemen nachzubilden.

Deshalb haben wir in dieser Arbeit versucht, einen Rahmen für die Modellierung des Verhaltens von Menschen und Robotern auf der Grundlage der statistischen Entscheidungstheorie zu entwickeln. Verschiedene Merkmale dieses Ansatzes, wie z. B. Risikosensitivität, Exploration, Lernen und Kontrolle, wurden in einer Reihe von Veröffentlichungen untersucht.

Zunächst haben wir das Problem des Erlernens neuer Fähigkeiten betrachtet und einen Rahmen für die entropische Regularisierung von Markov-Entscheidungsprozessen (MDP) entwickelt. Unter Verwendung eines verallgemeinerten Konzepts der Entropie konnten wir den Kompromiss zwischen Exploration und Ausnutzung durch die Wahl eines einzigen skalaren Parameters realisieren, der die Divergenzfunktion bestimmt.

Zweitens haben wir, aufbauend auf der Theorie der teilweise beobachtbaren Markov-Entscheidungsprozesse (POMDP), ein Modell des menschlichen Ballfangverhaltens vorgeschlagen und validiert. Entscheidend ist, dass das Verhalten der Informationssuche als Schlüsselmerkmal identifiziert wurde, das die Modellierung des beobachteten menschlichen Fangverhaltens ermöglicht. Es zeigte sich, dass die Entropiereduktion eine wichtige Rolle für das geschickte menschliche Verhalten spielt.

Drittens, nachdem wir die Modellierungsprinzipien aus dem menschlichen Verhalten extrahiert und einen informationstheoretischen Rahmen für das Verstärkungslernen entwickelt hatten, untersuchten wir die realen Roboteranwendungen der lernbasierten Steuerungen in taktil reichhaltigen Manipulationsaufgaben. Wir untersuchten bildverarbeitungsbasierte taktile Sensoren und die Fähigkeit von Lernalgorithmen, autonom aufgabenrelevante Merkmale für Manipulationsaufgaben zu extrahieren. Die Besonder-

heit der taktilen Steuerung, dass Wahrnehmung und Handlung am Kontaktpunkt eng miteinander verbunden sind, ermöglichte es uns, Einblicke in die Stärken und Grenzen des statistischen Lernansatzes für die Echtzeit-Robotermanipulation zu gewinnen.

Zusammenfassend lässt sich sagen, dass in dieser Arbeit eine Reihe von Untersuchungen zur Anwendbarkeit des Paradigmas der statistischen Entscheidungstheorie bei der Modellierung des menschlichen Verhaltens und bei der Synthese des Roboterverhaltens durchgeführt wurden. Wir kommen zu dem Schluss, dass eine Reihe wichtiger Merkmale im Zusammenhang mit der Informationsverarbeitung dargestellt und in künstlichen Systemen genutzt werden können, um intelligenteres Verhalten zu erzeugen. Dennoch sind dies nur die ersten Schritte, und wir erkennen an, dass der Weg zu künstlicher allgemeiner Intelligenz und geschickten Roboteranwendungen weitere Innovationen und möglicherweise eine Übersteigung des probabilistischen Modellierungsparadigmas erfordern wird.

Acknowledgement

This thesis owes its appearance thanks to multiple people. First and foremost, the unwavering support and encouragement that I received from my PhD advisor *Jan Peters* is hard to overstate. His ideas and vision are seen throughout the thesis and I am proud of having worked with such a remarkable scientist and mentor. The creative environment provided by the IAS lab was paramount for my growth as a scientist and as a human being. The freedom to explore and test new ideas that I enjoyed during my years at IAS, although at times scary and torn with many frustration, has been ultimately a path towards greater appreciation of the complexity of human psychology and control algorithms underlying human and robot behavior.

The breadth of view of my mentors *Constantin Rothkopf* and *Oliver Tessmann* together with their practical support were invaluable in bringing our joint research projects to fruition. I further thank my committee members for evaluating my thesis and participating in the defense.

I was blessed with outstanding colleagues, many of whom became my close friends. Both the projects that we did together as well as the times after work will remain a highlight of my time in Darmstadt. I am indebted to *Fabio Muratore*, *Hany Abdulsamad*, *Samuele Tossato*, *Michael Lutter*, *Joe Watson*, *Pascal Klink*, *João Carvalho*, and *Svenja Stark* for making these years together an unforgettable experience and for providing a secure and open environment to debate and challenge each other's ideas. The amount I learned from *Hany Abdulsamad* and *Joe Watson* on Bayesian thinking truly changed me forever. The work ethics and penetrative insight into human nature of *Fabio Muratore* taught me more than any textbook. Especially at the beginning, I benefited a lot from the friendship of *Filipe Veiga* and *Rudolf Lioutikov* who welcomed me into the IAS family.

It was a great fortune to meet and collaborate with *Bastian Wibranek*, who introduced me into the world of digital design and architecture and majorly influence the trajectory of my PhD. Having been surrounded by so many bright colleagues, I enjoyed every opportunity to collaborate and learn from everyone. *Georgia Chalvatzaki*, *Niklas Funk*, *Tim Schneider*, and *Yuxi Liu* especially helped me a lot during the last phase of my journey. Many discussions with *Filipe Veiga*, *Yi Zheng*, and *Roberto Calandra* helped me to streamline my thinking about tactile sensing and robotic manipulation.

More broadly, I thank my collaborators *Tuan Dam*, *Davide Tateo*, *Riad Akrour*, *Simone Parisi*, *Jia-Jie Zhu*, *Guoping Zhao*, *Diego Romeres*, *Devesh Jha* and students *Jianyang Tang*, *Honggang Gou*, *Tim Staschewski*, *Yunlong Song*, *Rong Zhi*, *Alymbek Sadybakasov*, *David Nass*, *Lennart Ebeling*, *Christian Eilers*, *Jonas Eschmann*, *Robin Menzenbach*, *Matthias Schultheis*, *Maximilian Hensel*, *Kai Cui*, *Rustam Galljamov*, *Markus Semmler*, *Leon Wietschorke*, *Jianpeng Chen*, *Jan Schneider*, *Tim Dorau*, *Jan Rathjens*, *Leon Magnus*, *Svenja Menzenbach*, *Max Siebenborn*, *Jan Emrich*, *Simon Kiefhaber*, *Theo Gruner*, *Arlene Kühn*, *Florian Wiese*, *Frederik Wegner*, *Timo Imhof* who embarked on the journey of exploring the frontiers of robot learning together with me.

I am indebted to my assistants *Christian Betschinske* without whom the tactile sensor would not be possible, *Frederik Wegner* who substantially contributed to the tactile manipulation robot setup, *Axel Patzwahl* who helped me with carrying out the ball catching experiments, *Si Jun Kwon* who provided invaluable support in processing the ball catching data, and to *Boris Feldman* together with the *Darmstadt Whippets* baseball team who taught me a great deal about real-world ball catching and supported the ball catching experiments at all stages.

Finally, and most importantly, I thank my love *Lena Pirner* for her support and encouragement during the hardest times and for enriching my life beyond what I could imagine. My sincere acknowledgements also go to my *friends and family*, without whom I would not have come this far at all.

Contents

Abstract vi

Zusammenfassung viii

Acknowledgement ix

1. Introduction 1
- 1.1. Motivation . . . 1
- 1.2. Contributions . . . 2
  - 1.2.1. Entropic Regularization of Markov Decision Processes . . . 2
  - 1.2.2. Human Optimality in Ball Catching . . . 3
  - 1.2.3. Robotic Architectural Assembly with Tactile Skills . . . 3
- 1.3. Outline . . . 4

2. Entropic Regularization of Markov Decision Processes 5
- 2.1. Introduction . . . 5
- 2.2. Background . . . 6
  - 2.2.1. Policy Gradient Methods . . . 7
  - 2.2.2. Entropic Penalties . . . 7
- 2.3. Entropic Proximal Policy Optimization . . . 8
  - 2.3.1. Fighting Covariate Shift via Trust Regions . . . 9
  - 2.3.2. Policy Optimization with Entropic Penalties . . . 10
  - 2.3.3. Value Function Approximation . . . 11
  - 2.3.4. Sample-Based Algorithm for Dual Optimization . . . 11
  - 2.3.5. Parametric Policy Fitting . . . 11
  - 2.3.6. Temperature Scheduling . . . 12
  - 2.3.7. Practical Algorithm for Continuous State-Action Spaces . . . 12
- 2.4. High- and Low-Temperature Limits; Analytic Solutions and Asymptotics . . . 13
  - 2.4.1. KL Divergence ($\alpha = 1$) and Pearson χ^2-Divergence ($\alpha = 2$) . . . 13
  - 2.4.2. High- and Low-Temperature Limits . . . 15

2.5. Empirical Evaluations . . . 16
2.5.1. Experiments on Stochastic Multi-Armed Bandit Problems . . . 17
2.5.2. Empirical Evaluations on Ergodic MDPs . . . 19
2.6. Related Work . . . 21
2.7. Conclusions . . . 21

3. Human Optimality in Ball Catching 23
3.1. Introduction . . . 23
3.2. Related Work . . . 25
3.3. Modeling Ball Catching Under Uncertainty as an Optimal Control Problem 26
3.3.1. Optimal Control Under Uncertainty . . . 27
3.3.2. A Computational Model of the Catching Agent . . . 28
3.3.3. Implementation Details . . . 30
3.4. Simulated Experiments and Results . . . 30
3.4.1. Continuous Tracking of an Outfielder—Heuristics Hold . . . 30
3.4.2. Interrupted Tracking During Long Passes—Heuristics Break . . . 32
3.4.3. Switching Behaviors When Uncertainty and Reaction Time are Varied 34
3.5. Real-World Experiments With Human Catchers . . . 37
3.5.1. Selection of the Experiment Participants . . . 37
3.5.2. Experimental Setup . . . 39
3.5.3. Data Acquisition . . . 40
3.5.4. Experimental Protocol . . . 44
3.5.5. Statistical Description of the Results . . . 45
3.5.6. Comparison of Simulated and Real Trajectories . . . 48
3.5.7. Conclusions From the Experimental Results . . . 48
3.6. Discussion and Conclusion . . . 54

4. Robotic Architectural Assembly with Tactile Skills: Simulation and Optimization 57
4.1. Introduction . . . 57
4.2. Related Work . . . 59
4.3. Reinforcement Learning for Contact-Rich Modular Assembly . . . 63
4.3.1. Reinforcement Learning Problem Setting . . . 64
4.3.2. Rhino 6 Integration . . . 65
4.3.3. Learning Algorithm Description . . . 66
4.3.4. Reinforcement Learning for Part Stacking . . . 69
4.3.5. Simulation of the Tactile Sensor DIGIT . . . 75
4.3.6. Learning on the Real Robot . . . 75
4.4. Conclusion and Future Work . . . 78

5. Conclusion **81**
5.1. Summary of Contributions 81
5.2. Critical Discussion of Contributions 82
5.3. Outlook 83
5.3.1. Open Problems 83
5.3.2. Future Work 84

A. Supplementary Material **87**
A.1. Technical Background on f-Divergence 87
A.2. Parameter Settings for Entropic Proximal Policy Optimization 89

List of Acronyms **91**

List of Figures **94**

List of Tables **95**

Bibliography **97**

1. Introduction

A movement of an arm when we want to grab a glass of water happens effortlessly. We seem to be hardly aware of how the intent of reaching for the water is transformed into the action of stretching the arm. The underlying mechanisms of motor control appear to be outside of our conscious awareness and we are not able to introspect into how the action is generated. Therefore, if we want to endow robots with the ability to move and act in the physical world, we are compelled to create models of perception and action from first principles, rather than relying on introspection.

Optimal control theory provides a rational framework for analysis and synthesis of behaviors of dynamical systems [1]. From an outside perspective and at a fixed level of analysis, a living organism and a robot can be both viewed as dynamical systems [2]. Therefore, optimal control theory can be applied to model the observed behavior of animals as well as for generating behaviors of artificial systems [3].

If we consider the mechanism that can modulate phase portraits of a dynamical system as a separate entity called *agent*, then one important question is how does the agent transform observations into actions given a certain objective function. This, in a nutshell, is the classical setting of optimal control theory expressed in plain language.

The object of study in this thesis is the design of the agent. We assume that the agent has a probabilistic model of the environment and acts optimally with respect to that model, inferring unobserved variables when necessary. This perspective allows us to account for ball catching behavior in humans, design a novel policy optimization procedure, and learn representations in a tactile-based manipulation environment.

1.1. Motivation

It is of scientific as well as practical interest to develop models of human behavior. If it turns out that the decision making agent can be described by a mathematical function, then we can fit this function to the data to predict and influence the behavior of the agent.

From the modeling perspective, the key conceptual framework for describing the behavior of dynamical systems that emerged in sciences is optimization. We postulate a

probabilistic model and an optimization criterion and find the variables of interest as a minimizer of the objective function. This provides both an intellectually satisfying explanation of behavior and a practical approach for controller synthesis.

However, there is a number of open questions with regards to the implementation of this approach to agent modeling. Among others, we focus on the three following questions: i) does this model account for observed human behavior? ii) how can we utilize probabilistic modeling to design a system that *learns* a new behavior through interaction with the environment? iii) can we apply reinforcement learning with learned representations to challenging tactile-rich manipulation tasks on real robots? The thesis elaborates on these questions and provides directions for future work.

1.2. Contributions

This thesis advances the fields of cognitive science and robotics by shedding light on the important questions of behavior modeling, optimization, and learning. First, we developed an entropic regularization framework for Markov decision processes [4], [5], where the policy update is constrained by a general f-divergence. Such constraint enables learning of policies with different exploration-exploitation trade-off profiles depending on the choice of the divergence function. Second, we presented a model of behavior that captures salient characteristics of human ball catching [6]. In field study experiments we demonstrated that the observed human trajectories correspond to the belief-optimal control strategies. Third, we investigated information-theoretic representation learning in a real-robot tactile-based assembly scenario [7]. Here a compact representation of a high-dimensional visual sensor input was learned and evaluated on a challenging contact-rich insertion task. It was shown that reinforcement learning with such learned representations is a feasible approach for tactile manipulation albeit future research is needed to reduce the computational load and improve real-time capabilities of such methods.

1.2.1. Entropic Regularization of Markov Decision Processes

An optimal feedback controller for a given Markov decision process (MDP) can in principle be synthesized by value or policy iteration. However, if the system dynamics and the reward function are unknown, a learning agent must discover an optimal controller via direct interaction with the environment. Such interactive data gathering commonly leads to divergence towards dangerous or uninformative regions of the state space unless additional regularization measures are taken. Prior works proposed bounding the information loss measured by the Kullback–Leibler (KL) divergence at every policy improvement step

to eliminate instability in the learning dynamics. In this paper, we consider a broader family of f-divergences, and more concretely α-divergences, which inherit the beneficial property of providing the policy improvement step in closed form at the same time yielding a corresponding dual objective for policy evaluation. Such entropic proximal policy optimization view gives a unified perspective on compatible actor-critic architectures. In particular, common least-squares value function estimation coupled with advantage-weighted maximum likelihood policy improvement is shown to correspond to the Pearson χ^2-divergence penalty. Other actor-critic pairs arise for various choices of the penalty-generating function f. On a concrete instantiation of our framework with the α-divergence, we carry out asymptotic analysis of the solutions for different values of α and demonstrate the effects of the divergence function choice on common standard reinforcement learning problems.

1.2.2. Human Optimality in Ball Catching

Two seemingly contradictory theories attempt to explain how humans move to intercept an airborne ball. One theory posits that humans *predict* the ball trajectory to *optimally* plan future actions; the other claims that, instead of performing such complicated computations, humans employ *heuristics* to reactively choose appropriate actions based on *immediate* visual feedback. In this paper, we show that interception strategies appearing to be heuristics can be understood as computational solutions to the optimal control problem faced by a ball-catching agent acting under uncertainty. Modeling catching as a continuous partially observable Markov decision process and employing stochastic optimal control theory, we discover that the four main heuristics described in the literature are optimal solutions if the catcher has sufficient time to continuously visually track the ball. Specifically, by varying model parameters such as noise, time to ground contact, and perceptual latency, we show that different strategies arise under different circumstances. The catcher's policy switches between generating reactive and predictive behavior based on the ratio of system to observation noise and the ratio between reaction time and task duration. Thus, we provide a rational account of human ball-catching behavior and a unifying explanation for seemingly contradictory theories of target interception on the basis of stochastic optimal control.

1.2.3. Robotic Architectural Assembly with Tactile Skills

Construction is an industry that could benefit significantly from automation, yet still relies heavily on manual human labor. Thus, we investigate how a robotic arm can be used to assemble a structure from predefined discrete building blocks autonomously. Since

assembling structures is a challenging task that involves complex contact dynamics, we propose to use a combination of reinforcement learning and planning for this task. In this work, we take a first step towards autonomous construction by training a controller to place a single building block in simulation. Our evaluations show that trial-and-error algorithms that have minimal prior knowledge about the problem to be solved, so called model-free deep reinforcement learning algorithms, can be successfully employed. We conclude that the achieved results, albeit imperfect, serve as a proof of concept and indicate the directions for further research to enable more complex assemblies involving multiple building elements.

1.3. Outline

The thesis consists of 5 chapters. Chapters $2-4$ provide the core contributions of this thesis, while Chapter 1 and Chapter 5 serve to introduce the background and context for this work, as well as to summarize the contributions and lessons learned from the conducted research. The individual chapters cover the following topics.

Chapter 1 outlines the thesis. We provide the motivation for the research undertaken withing this thesis. The scope and key findings are placed in the context of the broader perspective of the fields of cognitive science and robot learning.

Chapter 2 develops a framework of entropic regularization for Markov decision processes. We study the properties of f-divergence regularization of policy improvement on the convergence of the policy iteration scheme. Analytical properties are derived and empirical evaluations on simulated reinforcement learning problems are presented.

Chapter 3 presents a model of human ball catching behavior and describes how the experimental results demonstrate agreement with the model predictions. This result provides an argument in support of probabilistic modeling for describing human behavior in contrast to the framework of heuristic decision making.

Chapter 4 demonstrates an application of reinforcement learning with learned representations on a challenging contact-rich manipulation task with vision-based tactile sensing. We show that the approach is feasible but requires high computational power and is not yet suitable for real-time applications.

Chapter 5 summarizes the contributions of the thesis and discusses their significance as well as the directions for future work. We provide in a condensed form the main messages that can be gleaned from the investigations undertaken within this thesis.

2. Entropic Regularization of Markov Decision Processes

An optimal feedback controller for a given Markov decision process (MDP) can in principle be synthesized by value or policy iteration. However, if the system dynamics and the reward function are unknown, a learning agent must discover an optimal controller via direct interaction with the environment. Such interactive data gathering commonly leads to divergence towards dangerous or uninformative regions of the state space unless additional regularization measures are taken. Prior works proposed bounding the information loss measured by the Kullback–Leibler (KL) divergence at every policy improvement step to eliminate instability in the learning dynamics. In this paper, we consider a broader family of f-divergences, and more concretely α-divergences, which inherit the beneficial property of providing the policy improvement step in closed form at the same time yielding a corresponding dual objective for policy evaluation. Such entropic proximal policy optimization view gives a unified perspective on compatible actor-critic architectures. In particular, common least-squares value function estimation coupled with advantage-weighted maximum likelihood policy improvement is shown to correspond to the Pearson χ^2-divergence penalty. Other actor-critic pairs arise for various choices of the penalty-generating function f. On a concrete instantiation of our framework with the α-divergence, we carry out asymptotic analysis of the solutions for different values of the divergence parameter α and demonstrate the effects of the divergence type on the performance on common standard reinforcement learning problems.

2.1. Introduction

Sequential decision-making problems under uncertainty are described by the mathematical framework of Markov decision processes (MDPs) [8]. The core problem in MDPs is to find an optimal policy—a mapping from states to actions which maximizes the expected cumulative reward collected by an agent over its lifetime. In reinforcement learning (RL), the agent is additionally assumed to have no prior knowledge about the environment

dynamics and the reward function [9]. Therefore, direct policy optimization in the RL setting can be seen as a form of stochastic black-box optimization: the agent proposes a query point in the form of a policy, the environment evaluates this point by computing the expected return, after that the agent updates the proposal and the process repeats [10]. There are two conceptual steps in this scheme known as policy evaluation and policy improvement [11]. Both steps require function approximation in high-dimensional and continuous state-action spaces due to the curse of dimensionality [11]. Therefore, statistical learning approaches are employed to approximate the value function of a policy and to perform policy improvement based on the data collected from the environment.

In contrast to traditional supervised learning, in reinforcement learning, the data distribution changes with every policy update. State-of-the-art generalized policy iteration algorithms [12]–[15] are mindful of this covariate shift problem [16], taking active measures to account for it. To smoothen the learning dynamics, these algorithms limit the information loss between successive policy updates as measured by the KL divergence or approximations thereof [17]. In the optimization literature, such approaches are categorized as proximal (or trust region) algorithms [18].

The choice of the divergence function determines the geometry of the information manifold [19]. Recently, in particular in the area of implicit generative modeling [20], the choice of the divergence function was shown to have a dramatic effect both on the optimization performance [21] and the perceptual quality of the generated data when various f-divergences were employed [22]. In this paper, we carry over the idea of using generalized entropic proximal mappings [23] given by an f-divergence to reinforcement learning. We show that relative entropy policy search [13], framed as an instance of stochastic mirror descent [24], [25] as suggested by [17], can be extended to use any divergence measure from the family of f-divergences. The resulting algorithm provides insights into the compatibility of policy and value function update rules in actor-critic architectures, which we exemplify on several instantiations of the generic f-divergence with representatives from the parametric family of α-divergences [26]–[28].

2.2. Background

This section provides the necessary background on policy gradients [10] and entropic penalties [23] for later derivations and analysis. Standard RL notation [29] is adopted.

2.2.1. Policy Gradient Methods

Policy search algorithms [10] commonly use the so-called score function gradient estimator [30] of the form

$$\hat{g} = \hat{E}_t \left[\nabla_\theta \log \pi_\theta \hat{A}_t^w \right] \tag{2.1}$$

where $\pi_\theta(a|s)$ is a stochastic policy and $\hat{A}_t^w(s_t, a_t)$ is an estimator of the advantage function at timestep t. Expectation $\hat{E}_t[\dots]$ indicates an empirical average over a finite batch of samples, in an algorithm that alternates between sampling and optimization. The advantage estimate $\hat{A}_t^w$ can be obtained from an estimate of the value function [31], [32], which in its turn is found by least-squares estimation. Specifically, if $V^w(s)$ denotes a parametric value function, and if $\hat{V}_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k}$ is taken as its rollout-based estimate, then the parameters w can be found as

$$w = \arg\min_{\tilde{w}} \ \hat{E}_t \left[\| V^{\tilde{w}}(s_t) - \hat{V}_t \|^2 \right]. \tag{2.2}$$

The advantage estimate $\hat{A}_t^w = \sum_{k=0}^{\infty} \gamma^k \delta_{t+k}^w$ is then obtained by summing the temporal difference errors $\delta_t^w = R_t + \gamma V^w(s_{t+1}) - V^w(s_t)$, also known as the Bellman residuals. Treating $\hat{A}_t^w$ as fixed for the purpose of policy improvement, we can view (2.1) as the gradient of an advantage-weighted log-likelihood; therefore, the policy parameters θ can be found as

$$\theta = \arg\max_{\tilde{\theta}} \ \hat{E}_t \left[\log \pi_{\tilde{\theta}} \hat{A}_t^w \right]. \tag{2.3}$$

Thus, actor-critic algorithms that use the gradient estimator (2.1) to update the policy can be viewed as instances of the generalized policy iteration scheme, alternating between policy evaluation (2.2) and policy improvement (2.3). In the following, we will see that the actor-critic pair (2.2) and (2.3), that combines least-squares value function fitting with linear-in-the-advantage-weighted maximum likelihood policy improvement, is just one representative from a family of such actor-critic pairs arising for different choices of the f-divergence penalty within our entropic proximal policy optimization framework.

2.2.2. Entropic Penalties

The term entropic penalties [23] refers to both f-divergences and Bregman divergences. In this paper, we will focus on f-divergences, leaving generalization to Bregman divergences for future work. The f-divergence [33] between two distributions P and Q with the corresponding densities p and q is defined as

$$D_f(p\|q) = E_q \left[f\left(\frac{p}{q} \right) \right]$$

where f is a convex function on $(0, \infty)$ with $f(1) = 0$, and P is absolutely continuous with respect to Q. The most well-known example of the f-divergence is the KL divergence, which corresponds to $f_1(x) = x \log x - (x - 1)$, where the formula also applies to unnormalized distributions [34]. Many common divergences lie on the curve of α-divergences [26], [27] defined by a special choice of the generator function [28]

$$f_\alpha(x) = \frac{(x^\alpha - 1) - \alpha(x - 1)}{\alpha(\alpha - 1)}, \quad \alpha \in \mathbb{R}. \tag{2.4}$$

The α-divergence $D_\alpha = D_{f_\alpha}$ will be used as the primary example of the f-divergence throughout the paper. For more details on the α-divergence and its properties, see Appendix A.1. Noteworthy is the symmetry of the α-divergence with respect to $\alpha = 0.5$, which relates reverse divergences as $D_{0.5+\beta}(p\|q) = D_{0.5-\beta}(q\|p)$.

2.3. Entropic Proximal Policy Optimization

Consider the average-reward RL setting [9], where the dynamics of an ergodic MDP are given by the transition density $p(s'|s, a)$. An intelligent agent can modulate the system dynamics by sampling actions a from a stochastic policy $\pi(a|s)$ at every time step of the evolution of the dynamical system. The resulting modulated Markov chain with transition kernel $p_\pi(s'|s) = \int_A p(s'|s, a)\pi(a|s)da$ converges to a stationary state distribution $\mu_\pi(s)$ as time goes to infinity. This stationary state distribution induces a state-action distribution $\rho_\pi(s, a) = \mu_\pi(s)\pi(a|s)$, which corresponds to visitation frequencies of state-action pairs [8]. The goal of the agent is to steer the system dynamics to desirable states. Such objective is commonly encoded by the expectation of a random variable $R\colon S \times A \to \mathbb{R}$ called reward in this context. Thus, the agent seeks a policy that maximizes the expected reward $J(\pi) = E_{\rho_\pi(s,a)}[R(s, a)]$.

In reinforcement learning, neither the reward function R nor the system dynamics $p(s'|s, a)$ are assumed to be known. Therefore, to maximize (or even evaluate) the objective $J(\pi)$, the agent must sample a batch of experiences in the form of tuples (s, a, r, s') from the dynamics and use an empirical estimate $\hat{J} = \hat{E}_t[R(s_t, a_t)]$ as a surrogate for the original objective. Since the gradient of the expected reward with respect to the policy parameters can be written as [35]

$$\nabla_\theta J(\pi_\theta) = E_{\rho_{\pi_\theta}(s,a)}[\nabla_\theta \log \pi_\theta(a|s)R(s, a)]$$

with a corresponding sample-based counterpart

$$\nabla_\theta \hat{J} = \hat{E}_t[\nabla_\theta \log \pi_\theta(a_t|s_t)R(s_t, a_t)],$$

one may be tempted to optimize a sample-based objective

$$\hat{E}_t[\log \pi_\theta(a_t|s_t)R(s_t, a_t)]$$

on a fixed batch of data $\{(s, a, r, s')_t\}_{t=1}^N$ till convergence. However, such an approach ignores the fact that sampling distribution $\rho_{\pi_\theta}(s, a)$ itself depends on the policy parameters θ; therefore, such greedy optimization aims at a wrong objective [13]. To have the correct objective, the dataset must be sampled anew after every parameter update—doing otherwise will lead to overfitting and divergence. This problem is known in statistics as the covariate shift problem [16].

2.3.1. Fighting Covariate Shift via Trust Regions

A principled way to account for the change in the sampling distribution at every policy update step is to construct an auxiliary local objective function that can be safely optimized till convergence. Relative entropy policy search (REPS) algorithm [13] proposes a candidate for such an objective

$$J_\eta(\pi) = E_{\rho_\pi}[R] - \eta D_1(\rho_\pi \| \rho_{\pi_0}) \tag{2.5}$$

with π_0 being the current policy under which the data samples were collected, policy π being the improvement policy that needs to be found, and $\eta > 0$ being a 'temperature' parameter that determines how much the next policy can deviate from the current one. The original formulation employs a relative entropy trust region constraint D_1 with radius ε instead of a penalty, which allows for finding the optimal temperature η as a function of the trust region radius ε.

Importantly, the objective function (2.5) can be optimized in closed form for policy π (i.e., treating the policy itself as a variable and not its parameters, in contrast to standard policy gradients). To that end, several constraints on ρ_π are added to ensure stationarity with respect to the given MDP [13]. In a similar vein, we can solve Problem (2.5) with respect to π for any f-divergence with a twice differentiable generator function f.

2.3.2. Policy Optimization with Entropic Penalties

Following the intuition of REPS, we introduce an f-divergence penalized optimization problem that the learning agent must solve at every policy iteration step

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & J_\eta(\pi) = E_{\rho_\pi}[R] - \eta D_f(\rho_\pi \| \rho_{\pi_0}) \\ \text{subject to} \quad & \int_A \rho_\pi(s', a') da' = \int_{S\times A} \rho_\pi(s,a) p(s'|s,a) ds da, \quad \forall s' \in S, \\ & \int_{S\times A} \rho_\pi(s,a) ds da = 1, \\ & \rho_\pi(s,a) \geq 0, \quad \forall (s,a) \in S \times A. \end{aligned} \tag{2.6}$$

The agent seeks a policy that maximizes the expected reward and does not deviate from the current policy too much. The first constraint in (2.6) ensures that the policy is compatible with the system dynamics, and the latter two constraints ensure that π is a proper probability distribution. Please note that π enters Problem (2.6) indirectly through ρ_π. Since the objective has the form of free energy [36] in ρ_π with an f-divergence playing the role of the usual KL, the solution can be expressed through the derivative of the convex conjugate function f'_*, as shown for general nonlinear problems in [23],

$$\rho_\pi(s,a) = \rho_{\pi_0}(s,a) f'_* \left(\frac{R(s,a) + \int_S V(s') p(s'|s,a) ds' - V(s) - \lambda + \kappa(s,a)}{\eta} \right). \tag{2.7}$$

Here, $\{V(s), \lambda, \kappa(s,a)\}$ are the Lagrange dual variables corresponding to the three constraints in (2.6), respectively. Although we get a closed-form solution for ρ_π, we still need to solve the dual optimization problem to get the optimal dual variables

$$\begin{aligned} \underset{V,\lambda,\kappa}{\text{minimize}} \quad & g(V,\lambda,\kappa) = \eta E_{\rho_{\pi_0}} \left[f_* \left(\frac{A^V(s,a) - \lambda + \kappa(s,a)}{\eta} \right) \right] + \lambda \\ \text{subject to} \quad & \kappa(s,a) \geq 0, \quad \forall (s,a) \in S \times A, \\ & \arg f_* \in \text{range}_{x \geq 0} f'(x), \quad \forall (s,a) \in S \times A. \end{aligned} \tag{2.8}$$

Remarkably, the advantage function $A^V(s,a) = R(s,a) + \int_S V(s') p(s'|s,a) ds' - V(s)$ emerges automatically in the dual objective. The advantage function also appears in the penalty-free linear programming formulation of policy improvement [8], which corresponds to the zero-temperature limit $\eta \to 0$ of our formulation. Thanks to the fact that the dual objective in (2.8) is given as an expectation with respect to ρ_{π_0}, it can be straightforwardly estimated from rollouts. The last constraint in (2.8) on the argument

of the convex conjugate f_* is easy to evaluate for common α-divergences. Indeed, the convex conjugate f_α^* of the generator function (2.4) is given by

$$f_\alpha^*(y) = \frac{1}{\alpha}(1 + (\alpha - 1)y)^{\frac{\alpha}{\alpha-1}} - \frac{1}{\alpha}, \quad \text{for } y(1-\alpha) < 1. \tag{2.9}$$

Thus, the constraint on $\arg f_*$ in (2.4) is just a linear inequality $y(1-\alpha) < 1$ for any α-divergence.

2.3.3. Value Function Approximation

For small grid-world problems, one can solve Problem (2.8) exactly for $V(s)$. However, for larger problems or if the state space is continuous, one must resort to function approximation. Assume we plug an expressive function approximator $V^w(s)$ in (2.8), then vector w becomes a new vector of parameters in the dual objective. Later, it will be shown that minimizing the dual when $\eta \to \infty$ is closely related to minimizing the mean squared Bellman error.

2.3.4. Sample-Based Algorithm for Dual Optimization

To solve Problem (2.8) in practice, we gather a batch of samples from policy π_0 and replace the expectation in the objective with a sample average. Please note that in principle one also needs to estimate the expectation of the future rewards $\int_S V(s')p(s'|s,a)ds'$. However, since the probability of visiting the same state-action pair in continuous space is zero, one commonly estimates this integral from a single sample [10], which is equivalent to assuming deterministic system dynamics. Inequality constraints in (2.8) are linear and they must be imposed for every (s,a) pair in the dataset.

2.3.5. Parametric Policy Fitting

Assume Problem (2.8) is solved on a current batch of data sampled from π_0 and thus the optimal dual variables $\{V(s), \lambda, \kappa(s,a)\}$ are given. Equation (2.7) allows one to evaluate the new density $\rho_\pi(s,a)$ on any pair (s,a) from the dataset. However, it does not yield the new policy π directly because representation (2.7) is variational. A common approach [10] is to assume that the policy is represented by a parameterized conditional density $\pi_\theta(a|s)$ and fit this density to the data using maximum likelihood.

To fit a parametric density $\pi_\theta(a|s)$ to the true solution $\pi(a|s)$ given by (2.7), we can minimize the KL divergence $D_1(\rho_\pi \| \rho_{\pi_\theta})$ with respect to the parameters θ. Minimization of this KL is equivalent to maximization of the weighted maximum likelihood $\hat{E}[f_*'(\dots)\log \rho_{\pi_\theta}]$.

Unfortunately, the state-action distribution $\rho_{\pi_\theta}(s,a) = \mu_{\pi_\theta}(s)\pi_\theta(a|s)$ is in general not known because the state distribution $\mu_{\pi_\theta}(s)$ not only depends on the policy but also on the system dynamics. Assuming the effect of the policy parameters on the stationary state distribution is small [10], we arrive at the following optimization problem for fitting the policy parameters

$$\theta = \arg\max_{\tilde{\theta}} \ \hat{E}_t \left[\log \pi_{\tilde{\theta}}(a_t|s_t) f'_* \left(\frac{\hat{A}^w(s_t,a_t) - \lambda + \kappa(s_t,a_t)}{\eta} \right) \right]. \tag{2.10}$$

Compare our policy improvement step (2.10) to the commonly used advantage-weighted maximum likelihood (ML) objective (2.3). They look surprisingly similar (especially if the weighting function $f'_*(y) = y$ is linear), which is not a coincidence and which will be systematically explained in the next sections.

2.3.6. Temperature Scheduling

The 'temperature' parameter η trades off reward vs divergence, as can be seen in the objective function in Problem (2.6). In practice, devising a schedule for η may be hard because η is sensitive to reward scaling and policy parameterization. A more intuitive way to impose the f-divergence proximity condition is by adding it as a constraint $D_f(\rho_\pi \| \rho_{\pi_0}) \leq \varepsilon$ with a fixed ε and then treating the temperature $\eta \geq 0$ as an optimization variable. Such formulation is easy to incorporate into the dual (2.8) by adding a term $\eta\varepsilon$ to the objective and a constraint $\eta \geq 0$ to the list of constraints. Constraint-based formulations were used before with a KL divergence constraint [13] and with its quadratic approximation [12], [14].

2.3.7. Practical Algorithm for Continuous State-Action Spaces

Our proposed approach for entropic proximal policy optimization is summarized in Algorithm 1. Following the generalized policy iteration scheme, we (i) collect data under a given policy, (ii) evaluate the policy by solving (2.8), and (iii) improve the policy by solving (2.10). In the following section, several instantiations of Algorithm 1 with different choices of function f will be presented and studied.

Algorithm 1: Primal-dual entropic proximal policy optimization with function approximation

Input: Initial actor-critic parameters (θ_0, w_0), divergence function f, temperature $\eta > 0$

while *not converged* **do**

  sample one-step transitions $\{(s, a, r, s')_t\}_{t=1}^N$ under current policy π_{θ_0};

  policy evaluation: optimize dual (2.8) with $V(s) = V^w(s)$ to obtain critic parameters w;

  policy improvement: perform weighted ML update (2.10) to obtain actor parameters θ;

end

Output: Optimal policy $\pi_\theta(a|s)$ and the corresponding value function $V^w(s)$

2.4. High- and Low-Temperature Limits; Analytic Solutions and Asymptotics

How does the f-divergence penalty influence policy optimization? How should one choose the generator function f? What role does the step size play in optimization? This section will try to answer these and related questions. First, two special choices of the penalty function f are presented, which reveal that the common practice of using mean squared Bellman error minimization coupled with advantage reweighted policy update is equivalent to imposing a Pearson χ^2-divergence penalty. Second, high- and low-temperature limits are studied, on one hand revealing the special role the Pearson χ^2-divergence plays, being the high-temperature limit of all smooth f-divergences, and on the other hand establishing a link to the linear programming formulation of policy search as the low-temperature limit of our entropic penalty-based framework.

2.4.1. KL Divergence ($\alpha = 1$) and Pearson χ^2-Divergence ($\alpha = 2$)

As can be deduced from the form of (2.10), great simplifications occur when $f'_*(y)$ is a linear function ($\alpha = 2$, see (2.9)) or an exponential function ($\alpha = 1$). The fundamental reason for such simplifications lies in the fact that linear and exponential functions are homomorphisms with respect to addition. This allows, in particular, discovery of a closed-form solution for the dual variable λ and thus eliminate it from the optimization. Moreover, in these two special cases, the dual variables $\kappa(s, a)$ can also be eliminated. They are responsible for non-negativity of probabilities: when $\alpha = 1$ (KL), $\kappa(s, a) = 0$ uniformly

for all $\eta \geq 0$, when $\alpha = 2$ (Pearson), $\kappa(s,a) = 0$ for sufficiently big η. Table 2.1 gives the corresponding empirical actor-critic optimization objective pairs. A generic primal-dual actor-critic algorithm with an α-divergence penalty performs two steps

$$\begin{aligned} &\text{(step 1: policy evaluation)} && \underset{w}{\text{minimize}} \quad \hat{g}_\alpha(w) \\ &\text{(step 2: policy improvement)} && \underset{\theta}{\text{maximize}} \quad \hat{L}_\alpha(\theta) \end{aligned}$$

inside a policy iteration loop. It is worth comparing the explicit formulas in Table 2.1 to the customarily used objectives (2.2) and (2.3). To make the comparison fair, notice that (2.2) and (2.3) correspond to discounted infinite horizon formulation with discount factor $\gamma \in (0,1)$ whereas formulas in Table 2.1 are derived for the average-reward setting. In general, the difference between these two settings can be ascribed to an additional baseline that must be subtracted in the average reward setting [9]. In our derivations, the baseline corresponds to the dual variable λ, as in classical linear programming formulation of policy iteration [8], and it is automatically gets subtracted from the advantage (see (2.8)).

Table 2.1.: Empirical policy evaluation and policy improvement objectives for the KL divergence ($\alpha = 1$) and the Pearson χ^2 divergence ($\alpha = 2$).

KL Divergence ($\alpha = 1$)
$\hat{g}_1(w) = \eta \log \left(\hat{E}_t \left[\exp \left(\frac{\hat{A}^w(s_t,a_t)}{\eta} \right) \right] \right)$
$\hat{L}_1(\theta) = \hat{E}_t \left[\log \pi_\theta(a_t \vert s_t) \exp \left(\frac{\hat{A}^w(s_t,a_t) - \hat{g}_1(w)}{\eta} \right) \right]$
Pearson χ^2-Divergence ($\alpha = 2$)
$\hat{g}_2(w) = \frac{1}{2\eta} \hat{E}_t \left[\left(\hat{A}^w(s_t,a_t) - \hat{E}_t \left[\hat{A}^w \right] \right)^2 \right]$
$\hat{L}_2(\theta) = \frac{1}{\eta} \hat{E}_t \left[\log \pi_\theta(a_t \vert s_t) \left(\hat{A}^w(s_t,a_t) - \hat{E}_t \left[\hat{A}^w \right] + \eta \right) \right]$

Mean Squared Error Minimization with Advantage Reweighting is Equivalent to Pearson Penalty

The baseline for $\alpha = 2$ is given by the average advantage $\lambda_2 = \hat{E}_t[\hat{A}^w(s_t,a_t)]$, which also equals the average return in the stationary setting [8], [9]. Therefore, to translate the formulas from Table 2.1 to the discounted infinite horizon form (2.2) and (2.3), we need to remove the baseline and add discounting to the advantage. If we denote the advantage

as $A^w(s,a) = R(s,a) + \gamma \int_S V^w(s')p(s'|s,a)ds' - V^w(s)$, then the dual objective $\hat{g}_2(w)$ is proportional to the average of the advantage squared,

$$\hat{g}_2(w) \propto \hat{E}_t \left[\left(\hat{A}^w(s_t, a_t) \right)^2 \right]. \tag{2.11}$$

Naive optimization of (2.11) leads to a family of residual gradient algorithms [37], [38]. However, if the same Monte Carlo estimate of the value function is used as in (2.2), then the dual objectives (2.11) and (2.2) are equivalent. The same holds for the Pearson actor

$$\hat{L}_2(\theta) \propto \hat{E}_t \left[\log \pi_\theta(a_t|s_t) \hat{A}^w(s_t, a_t) \right] \tag{2.12}$$

and the standard policy improvement (2.3) provided that $\eta = \hat{E}_t \left[\hat{A}^w(s_t, a_t) \right]$. That means (2.12) is equivalent to (2.3) if the weight of the divergence penalty is equal to the expected return.

2.4.2. High- and Low-Temperature Limits

In the previous subsection, we established a direct correspondence between the least-squares value function fitting coupled with the advantage-weighted maximum likelihood policy parameters estimation (2.2) and (2.3) and the dual-primal pair of optimization problems (2.11) and (2.12) arising from our Algorithm 1 for the special choice of the Pearson χ^2-divergence penalty. In this subsection, we will show that this is not a coincidence but a manifestation of the fundamental fact that the Pearson χ^2-divergence is the quadratic approximation of any smooth f-divergence about unity.

High Temperatures: All Smooth f-Divergences Tend Towards Pearson χ^2-Divergence

There are two ways to show the independence of the primal-dual solution (2.8)–(2.10) on the choice of the divergence penalty: either exactly solve an approximate problem or approximate the exact solution of the original problem. In the first case, the penalty is replaced with its Taylor expansion at $\eta \to \infty$, which turns out to be the Pearson χ^2-divergence, and then the derivation becomes equivalent to the natural policy gradient derivation [12]. In the second case, the exact solution (2.8)–(2.10) is expanded by Taylor: for big η, dual variables $\kappa(s,a)$ can be dropped if $\rho_{\pi_0}(s,a) > 0$, which yields

$$f_* \left(\frac{A^w(s,a) - \lambda}{\eta} \right) \approx f_*(0) + \frac{A^w(s,a) - \lambda}{\eta} f'_*(0) + \frac{1}{2} \left(\frac{A^w(s,a) - \lambda}{\eta} \right)^2 f''_*(0). \tag{2.13}$$

By definition of the f-divergence, the generator function f satisfies the condition $f(1) = 0$. Without loss of generality [39], one can impose an additional constraint $f'(1) = 0$ for convenience. Such constraint ensures that the graph of the function $f(x)$ lies entirely in the upper half-plane, touching the x-axis at a single point $x = 1$. From the definition of the convex conjugate $f'_* = (f')^{-1}$, we can deduce that $f'_*(0) = 1$ and $f_*(0) = 0$; by rescaling, it is moreover possible to set $f''(1) = f''_*(0) = 1$. These properties are automatically satisfied by the α-divergence, which can be verified by a direct computation. With this in mind, it is straightforward to see that substitution of (2.13) into (2.8) yields precisely the quadratic objective $\hat{g}_2(w)$ from Table 2.1, the difference being of the second order in $1/\eta$.

To obtain the asymptotic policy update objective, one can expand (2.10) in the high-temperature limit $\eta \to \infty$ and observe that it equals $\hat{L}_2(\theta)$ from Table 2.1 with the difference being of the second order in $1/\eta$. Therefore, it is established that the choice of the divergence function plays a minor role for big temperatures (small policy update steps). Since this is the mode in which the majority of iterative algorithms operate, our entropic proximal policy optimization point of view provides a rigorous justification for the common practice of using the mean squared Bellman error objective for value function fitting and the advantage-weighted maximum likelihood objective for policy improvement.

Low Temperatures: Linear Programming Formulation Emerges in the Limit

Setting η to a small number is equivalent to allowing large policy update steps because η is the weight of the divergence penalty in the objective function (2.6). Such regime is rather undesirable in reinforcement learning because of the covariate shift problem mentioned in the introduction. Problem (2.6) for $\eta \to 0$ turns into a well-studied linear programming formulation [8], [17] that can be readily applied if the model $\{p(s'|s,a), R(s,a)\}$ is known.

It is not straightforward to derive the asymptotics of policy evaluation (2.8) and policy improvement (2.10) for a general smooth f-divergence in the low-temperature limit $\eta \to 0$ because the dual variables $\kappa(s,a)$ do not disappear, in contrast to the high-temperature limit (2.13). However, for the KL divergence penalty (see Table 2.1), one can show that the policy evaluation objective $g_1(w)$ tends towards the supremum of the advantage $g_1(w) \to \sup_{s,a} A^w(s,a)$; the optimal policy is deterministic, $\pi(a|s) \to \delta(a - \arg\sup_b A^w(s,b))$, therefore $L(\theta) \to \log \pi_\theta(\bar{a}|\bar{s})$ with $(\bar{s},\bar{a}) = \arg\sup_{s',a'} A^w(s',a')$.

2.5. Empirical Evaluations

To develop an intuition regarding the influence of the entropic penalties on policy improvement, we first consider a simplified version of the reinforcement learning problem—namely

the stochastic multi-armed bandit problem [40]. In this setting, our algorithm is closely related to the family of Exp3 algorithms [41], originally motivated by the adversarial bandit problem. Subsequently, we evaluate our approach in the standard reinforcement learning setting.

2.5.1. Experiments on Stochastic Multi-Armed Bandit Problems

In the stochastic multi-armed bandit problem [40], at every time step $t \in \{1, \dots, T\}$, an agent chooses among K actions $a \in \mathcal{A}$. After every choice $a_t = a$, it receives a noisy reward $R_t = R(a_t)$ drawn from a distribution with mean $Q(a)$. The goal of the agent is to maximize the expected total reward $J = E[\sum_{t=1}^{T} R_t]$. Given the true values $Q(a)$, the optimal strategy is to always choose the best action, $a_t^* = \arg\max_a Q(a)$. However, due to the lack of knowledge, the agent faces the exploration-exploitation dilemma. A generic way to encode the exploration-exploitation trade-off is by introducing a policy π_t, i.e., a distribution from which the agent draws actions $a_t \sim \pi_t$. Thus, the question becomes: given the current policy π_t and the current estimate of action values $\hat{Q}_t$, what should the policy π_{t+1} at the next time step be? Unlike the choice of the best action under perfect information, such sampling policies are hard to derive from first principles [42].

We apply our generic Algorithm 1 to the stochastic multi-armed bandit problem to illustrate the effects of the divergence choice. The value function disappears because there is no state and no system dynamics in this problem. Therefore, the estimate $\hat{Q}_t$ plays the role of the advantage, and the dual optimization (2.8) is performed only with respect to the remaining Lagrange multipliers.

Effects of α on Policy Improvement

Figure 2.1 shows the effects of the α-divergence choice on policy updates. We consider a 10-armed bandit problem with arm values $Q(a) \sim \mathcal{N}(0, 1)$ and keep the temperature fixed at $\eta = 2$ for all values of α. Several iterations starting from an initial uniform policy are shown in the figure for comparison. Extremely large positive and negative values of α result in ε-elimination and ε-greedy policies, respectively. Small values of α, in contrast, weigh actions according to their values. Policies for $\alpha < 1$ are peaked and heavy-tailed, eventually turning into ε-greedy policies when $\alpha \to -\infty$. Policies for $\alpha \geq 1$ are more uniform, but they put zero mass on bad actions, eventually turning into ε-elimination policies when $\alpha \to \infty$. For $\alpha \geq 1$, policy iteration may spend a lot of time in the end deciding between two best actions, whereas for $\alpha < 1$ the final convergence is faster.

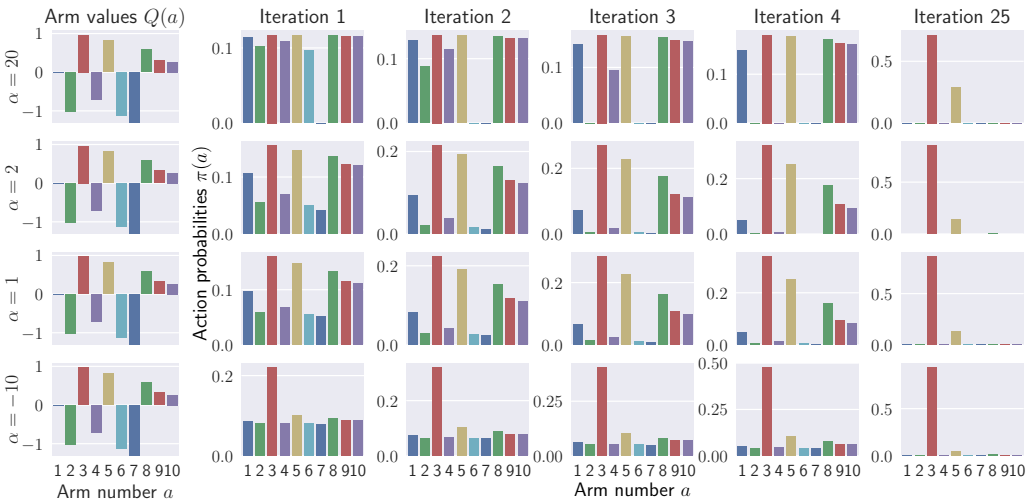
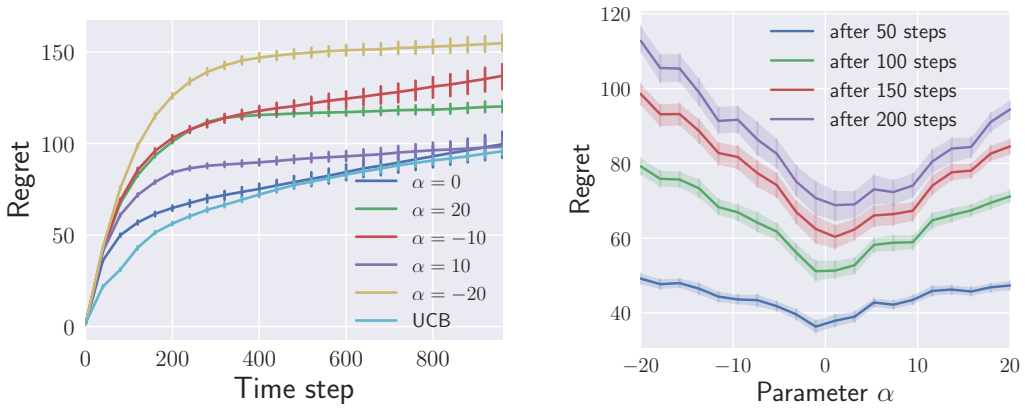


Figure 2.1.: Effects of the divergence parameter α on policy improvement. Each row corresponds to a fixed value of α. First four iterations of policy improvement together with a later iteration are shown in each row. Large positive α's eliminate bad actions one by one, keeping the exploration level equal among the rest. Small α's weigh actions according to their values; actions with low value get zero probability for $\alpha > 1$, but remain possible with small probability for $\alpha \leq 1$. Large negative α's focus on the best action, exploring the remaining actions with equal probability.

Effects of α on Regret

The average regret $C_n = nQ_{\max} - \mathbb{E}[\sum_{t=0}^{n-1} R_t]$ is shown in Figure 2.2a for different values of α as a function of the time step n, with 95% confidence error bars. The performance of the UCB algorithm [40] is also shown for comparison. The presented results are obtained in a 20-armed bandit environment where rewards have Gaussian distribution $R(a) \sim \mathcal{N}(Q(a), 0.5)$. Arm values are estimated from observed rewards and the policy is updated every 20 time steps. The temperature parameter η is decreased starting from $\eta = 1$ after every policy update according to the schedule $\eta^+ = \beta\eta$ with $\beta = 0.8$. Results are averaged over 400 runs. In general, extreme α's accumulate more regret. However, they eventually focus on a single action and flatten out. Small α's accumulate less regret, but they may keep exploring sub-optimal actions longer. Values of $\alpha \in [0, 2]$ perform



(a) Regret vs. time for different α.

(b) Regret vs. divergence type α.

Figure 2.2.: Average regret vs. time and divergence type α on a bandit problem.

comparably with UCB after 400 steps, once reliable estimates of values were obtained.

Figure 2.2b shows the average regret after a given number of time steps as a function of the divergence type α. As can be seen from the figure, smaller values of α result in lower regret. Large negative α's correspond to ε-greedy policies, which oftentimes prematurely converge to a sub-optimal action, failing to discover the optimal action for a long time if the exploration probability ε is small. Large positive α's correspond to ε-elimination policies, which may by mistake completely eliminate the best action or spend a lot of time deciding between two options in the end of learning, accumulating more regret. The optimal value of the parameter α depends on the time horizon for which the policy is being optimized. Depending on the horizon, the minimum of the curves shifts from slightly negative α's towards the range $\alpha \in [0, 2]$ with increasing time horizon.

2.5.2. Empirical Evaluations on Ergodic MDPs

We evaluate our policy iteration algorithm with f-divergence on standard grid-world reinforcement learning problems from OpenAI Gym [43]. The environments that terminate or have absorbing states are restarted during data collection to ensure ergodicity. Figure 2.3 demonstrates the learning dynamics on different environments for various choices of the divergence function. Parameter settings and further details can be found in Appendix A.2.

Each row in Figure 2.3 corresponds to a given environment. Results for different values of α are split into three subplots within each row, from the more extreme α's on the left to

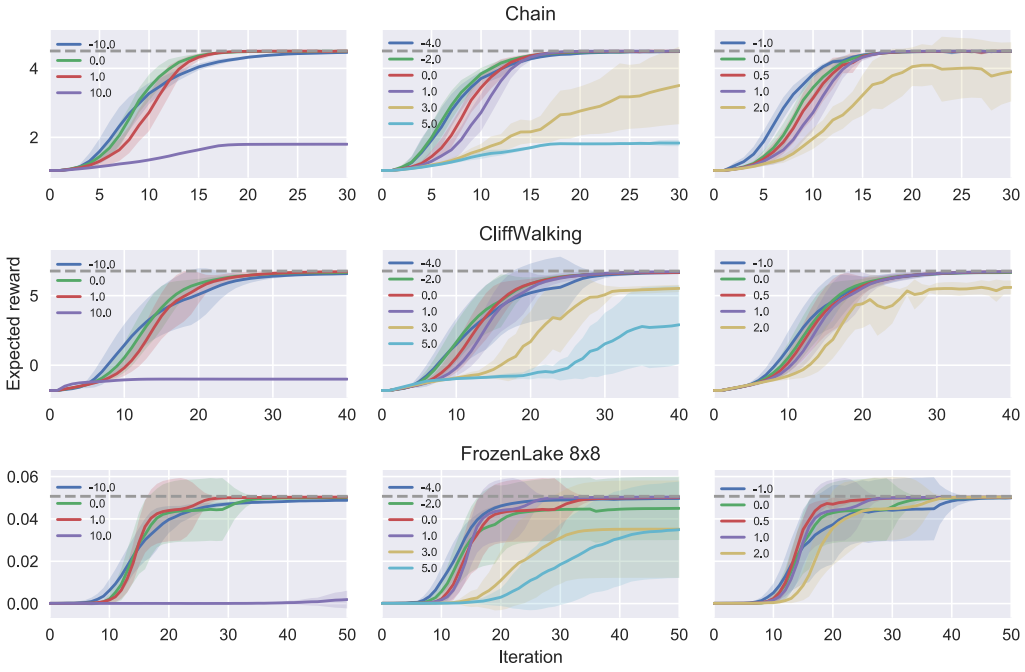


Figure 2.3.: Effects of the divergence parameter α on policy iteration.

the more refined values on the right. In all cases, more negative values $\alpha < 0$ initially show faster improvement because they immediately jump to the mode and keep the exploration level low; however, after a certain number of iterations they get overtaken by moderate values $\alpha \in [0, 1]$ that weigh advantage estimates more evenly. Positive $\alpha > 1$ demonstrate high variance in the learning dynamics because they clamp the probability of good actions to zero if the advantage estimates are overly pessimistic, never being able to recover from such a mistake. Large positive α's may even fail to reach the optimum altogether, as exemplified by $\alpha = 10$ in the plots. The most stable and reliable α-divergences lie between the reverse KL ($\alpha = 0$) and the KL ($\alpha = 1$), with the Hellinger distance ($\alpha = 0.5$) outperforming both on the FrozenLake environment.

In summary, one can either promote risk averse behavior by choosing the divergence parameter $\alpha < 0$, which may, however, result in sub-optimal exploration, or one can promote risk seeking behavior with $\alpha > 1$, which may lead to overly aggressive elimination of options. Our experiments suggest that the optimal balance should be found in the range $\alpha \in [0, 1]$. The effect of the α-divergence on policy iteration is not linear and not

symmetric with respect to $\alpha = 0.5$, contrary to what one could have expected given the symmetry of the α-divergence as a function of α. While switching from $\alpha = -3$ to $\alpha = -2$ may have little effect on policy iteration, switching from $\alpha = 3$ to $\alpha = 4$ may have a much more pronounced influence on the learning dynamics.

2.6. Related Work

Apart from computational advantages, information-theoretic approaches provide a solid framework for describing and studying aspects of intelligent behavior [44], from autonomy [45] and curiosity [46] to bounded rationality [47] and game theory [48].

Entropic proximal mappings were introduced in [23] as a general framework for constructing approximation and smoothing schemes for optimization problem. Problem formulation (2.6) presented here can be considered as an application of this general theory to policy optimization in Markov decision processes. Following the recent work [17] that establishes links between popular in reinforcement learning KL-divergence-regularized policy iteration algorithms [13], [14] and a well-known in optimization stochastic mirror descent algorithm [24], [25], one can view our Algorithm 1 as an analog of the mirror descent with an f-divergence penalty.

Concurrent works [49], [50] consider similar regularized formulations, although in the policy space instead of the state-action distribution space and in the infinite horizon discounted setting instead of the average-reward setting. The α-divergence in its entropic form, i.e., when the base measure is a uniform distribution, was used in several papers under the name Tsallis entropy [51]–[54], where its sparsifying effect was exploited in large discrete action spaces.

An alternative proximal reinforcement learning scheme was introduced in [55] based on the extragradient method for solving variational inequalities and leveraging operator splitting techniques. Although the idea of exploiting proximal maps and updates in the primal and dual spaces is similar to ours, regularization in [55] is applied in the value function space to smoothen generalized TD learning algorithms, whereas we study regularization in the primal space.

2.7. Conclusions

We presented a framework for deriving actor-critic algorithms as pairs of primal-dual optimization problems resulting from regularization of the standard expected return objective with so-called entropic penalties in the form of an f-divergence. Several examples

with different α-divergence penalties have been worked out in detail. In the limit of small policy update steps, all f-divergences with twice differentiable generator function f are approximated by the Pearson χ^2-divergence, which was shown to yield the most commonly used in reinforcement learning pair of actor-critic updates. Thus, our framework provides a sound justification for the common practice of minimizing mean squared Bellman error in the policy evaluation step and fitting policy parameters by advantage-weighted maximum likelihood in the policy improvement step.

In the future work, incorporating non-differentiable generator functions, such as the absolute value that corresponds to the total variation distance, may provide a principled explanation for the empirical success of the algorithms not accounted for by our current smooth f-divergence framework, such as the proximal policy optimization algorithm [15]. Establishing a tighter connection between online convex optimization that employs Bregman divergences and reinforcement learning will likely yield both a deeper understanding of the optimization dynamics in RL and allow for improved practical algorithms building on the firm fundament of optimization theory.

3. Human Optimality in Ball Catching

Two seemingly contradictory theories attempt to explain how humans move to intercept an airborne ball. One theory posits that humans *predict* the ball trajectory to *optimally* plan future actions; the other claims that, instead of performing such complicated computations, humans employ *heuristics* to reactively choose appropriate actions based on *immediate* visual feedback. In this paper, we show that interception strategies appearing to be heuristics can be understood as computational solutions to the optimal control problem faced by a ball-catching agent acting under uncertainty. Modeling catching as a continuous partially observable Markov decision process and employing stochastic optimal control theory, we discover that the four main heuristics described in the literature are optimal solutions if the catcher has sufficient time to continuously visually track the ball. Specifically, by varying model parameters such as noise, time to ground contact, and perceptual latency, we show that different strategies arise under different circumstances. The catcher's policy switches between generating reactive and predictive behavior based on the ratio of system to observation noise and the ratio between reaction time and task duration. Thus, we provide a rational account of human ball-catching behavior and a unifying explanation for seemingly contradictory theories of target interception on the basis of stochastic optimal control.

3.1. Introduction

Humans exhibit impressive abilities of intercepting moving targets as exemplified in sports such as baseball [56]. Despite the ubiquity of this visuomotor capability, explaining how humans manage to catch flying objects is a long-standing problem in cognitive science and human motor control. What makes this problem computationally difficult for humans are the involved perceptual uncertainties, high sensory noise, and long action delays compared to artificial control systems and robots. Thus, understanding action generation in human ball interception from a computational point of view may yield important insights on human visuomotor control. Surprisingly, there is no generally accepted model that explains empirical observations of human interception of airborne balls. McIntyre *et al.* [57] and

Hayhoe *et al.* [58] claim that humans employ an internal model of the physical world to predict where the ball will hit the ground and how to catch it. Such internal models allow for planning and potentially optimal action generation, e.g., they enable optimal catching strategies where humans predict the interception point and move there as fast as mechanically possible to await the ball. Clearly, there exist situations where latencies of the catching task require such strategies (e.g., when a catcher moves the arm to receive the pitcher's ball). By contrast, Gigerenzer & Brighton [59] argue that the world is far too complex for sufficiently precise modeling (e.g., a catcher or an outfielder in baseball would have to take air resistance, wind, and spin of the ball into account to predict its trajectory). Thus, humans supposedly extract few simple but robust features that suffice for successful execution of tasks such as catching. Here, immediate feedback is employed to guide action generation instead of detailed modeling. Policies based on these features are called *heuristics* and the claim is that humans possess a bag of such tricks, the "adaptive toolbox". For a baseball outfielder, a successful heuristic could be "Fix your gaze on the ball, start running, and adjust your running speed so that the angle of gaze remains constant" [60]. Thus, at the core, finding a unifying computational account of the human interception of moving targets also contributes to the long-lasting debate about the nature of human rationality [61].

In this paper, we propose that these seemingly contradictory views can be unified using a single computational model based on a continuous partially observable Markov decison process model (POMDP). In this model, the intercepting agent is assumed to choose optimal actions that take uncertainty about future movement into account. This model prescribes that both the catcher and the outfielder act optimally for their respective situation and uncertainty. We show that an outfielder agent using a highly stochastic internal model for prediction will indeed resort to purely reactive polices resembling established heuristics from the literature. The intuitive reason for such short-sighted behavior being optimal is that ball predictions over sufficiently long time horizons with highly stochastic models effectively become guessing. Similarly, our model will yield optimally planned actions based on predictions if the uncertainty encountered by the catcher agent is low while the latency is non-negligible in comparison to the movement duration. Moreover, we identify catching scenarios where the only strategy to intercept the ball requires to turn away from it and run as fast as possible. While such strategies cannot be explained by the heuristics proposed so far, the optimal control approach yields a plausible policy exhibiting both reactive and feedforward behavior. While other motor tasks (e.g., reaching movements [62], [63], locomotion [64]) have been explained in terms of stochastic optimal control theory, to the best of our knowledge this paper is the first to explain ball catching within this computational framework. We show that the four previously described empirical heuristics are actually optimal control policies. Moreover,

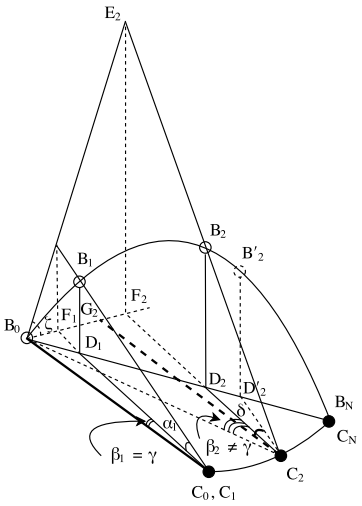


Figure 3.1.: Four well-known ball catching heuristics shown in one figure. See the description of the heuristics in text in Section 3.2.

our approach allows predictions for settings that cannot be explained by heuristics and have not been studied before. As catching behavior has previously been described as a prime example of humans not following complex computations but using simple heuristics, this study opens an important perspective on the fundamental question of human rationality.

3.2. Related Work

A number of heuristics have been proposed to explain how humans catch balls, see [65]–[67] for an overview. We focus on three theories well-supported by experiments: Chapman's theory, the generalized optic acceleration cancellation (GOAC) theory, and the linear optical trajectory (LOT) theory. Chapman [56] considered a simple kinematic problem (see Figure 3.1) where the ball B follows a parabolic trajectory $B_{0:N}$ while the agent C follows $C_{0:N}$ to intercept it. Only the position of the agent is relevant—his gaze is always directed towards the ball. Angle α is the *elevation angle*; angle γ is the *bearing angle* with respect to direction C_0B_0 (or C_2G_2 which is parallel). Due to delayed reaction, the agent starts running when the ball is already in the air. Chapman proposed two heuristics, i.e., the optic acceleration cancellation (OAC) that prescribes maintaining $\mathrm{d}\tan\alpha/\mathrm{d}t = \text{const}$, and the constant bearing angle (CBA), which requires $\gamma = \text{const}$.

However, Chapman did not explain how these heuristics cope with disturbances and observations.

To incorporate visual observations, McLeod *et al.* [67] introduced the *field of view* of the agent into Chapman's theory and coupled the agent's running velocity to the location of the ball in the visual field. Instead of the CBA heuristic, a *tracking heuristic* is employed to form the generalized optic acceleration cancellation (GOAC) theory. This tracking heuristic allows reactions to uncertain observations. In our example in Figure 3.1, the agent might have moved from C_0 to C_2 while maintaining a constant γ. To keep fulfilling this heuristic, the ball needs to arrive at B_2 at the same time. However, if the ball is already at B_2', the agent will see it falling into the right side of his field of view and he will speed up. Thus, the agent internally tracks the angle δ between CD and C_0B_0 and tries to adjust δ to γ.

In Chapman's theory and the GOAC theory, the elevation angle α and the bearing angle γ are controlled independently. As such, separate control strategies are implausible, therefore McBeath *et al.* [68] proposed the linear optical trajectory (LOT) heuristic that controls both angles jointly. LOT suggests that the catching agent runs such that the projection of the ball trajectory onto the plane perpendicular to the direction CD remains linear, which implies that $\zeta = \angle E_2B_0F_2$ remains constant. As $\tan\zeta = \tan\alpha_2/\tan\beta_2$ can be observed from the pyramid $B_0F_2C_2E_2$ with the right angles at F_2, there exist a coupling between the elevation angle α and the horizontal optical angle β (defined as the angle between CB_0 and CD), which can be used for directing the agent.

In contrast to the literature on outfielder's catching in baseball, other strands of research in human motor control have focused on predictive models [69] and optimality of behavior [62], [63]. Tasks similar to the catcher's in baseball have yielded evidence for prediction. Humans were shown to anticipate where a tennis ball will hit the floor when thrown with a bounce [58], and humans also appear to use an internal model of gravity to estimate time-to-contact when catching balls [57]. Optimal control theory has been used to explain reaching movements (with cost functions such as minimum-jerk [62], minimum-torque-change [70] and minimum end-point variance [71]), motor coordination [63], and locomotion (as minimizing metabolic energy [64]).

3.3. Modeling Ball Catching Under Uncertainty as an Optimal Control Problem

To parsimoniously model the catching agent, we rely on an optimal control formulation (Sec. 3.3.1) where the agent is described in terms of state-transitions, observations and a cost function (Sec. 3.3.2).

3.3.1. Optimal Control Under Uncertainty

In optimal control, the interaction of the agent with the environment is described by a stochastic dynamic model or system (e.g., describing ball flight and odometry). The system's state

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{\epsilon}_{k+1}, \quad k = 0 \dots N-1, \tag{3.1}$$

at the next time step $k+1$ is given as a noisy function of the state $\boldsymbol{x}_k \in \mathbb{R}^n$ and the action $\boldsymbol{u}_k \in \mathbb{R}^m$ at the current time step k. The mean state dynamics $\boldsymbol{f}$ are perturbed by zero-mean stationary white Gaussian noise $\boldsymbol{\epsilon}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q})$ with a constant system noise covariance matrix $\boldsymbol{Q}$ modeling the uncertainty in the system (e.g., the uncertainty in the agent's and ball's positions).

The state of the system is not always fully observed (e.g., the catching agent can only observe a ball when he looks at it), lower-dimensional than the system's state (e.g., only ball positions can directly be observed) and the observations are generally noisy (e.g., visuomotor noise affects ball position estimates). Thus, at every time step k, sensory input provides a noisy lower-dimensional measurement $\boldsymbol{z}_k \in \mathbb{R}^p$ of the true underlying system state $\boldsymbol{x}_k \in \mathbb{R}^n$ with $p < n$ described by

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{\delta}_k, \quad k = 1 \dots N, \tag{3.2}$$

where $\boldsymbol{h}$ is a deterministic observation function and $\boldsymbol{\delta}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_k)$ is zero-mean non-stationary white Gaussian noise with a state-dependent covariance matrix $\boldsymbol{R}_k = \boldsymbol{R}(\boldsymbol{x}_k)$. For catching, such state-dependency is crucial to modeling the effect of the human visual field. When the ball is at its center, measurements are least uncertain; whereas when the ball is outside the visual field, observations are maximally uncertain.

The agent obviously can only generate actions based on the observations collected so far, while affecting his and the environment's true next state. The history of observations allows forming probability distributions over the state at different time-steps called *beliefs*. Taking the uncertainty in (3.1) and (3.2) into account, the agent needs to plan and control in the *belief space* (i.e., the space of probability distributions over states) rather than in the state space. We approximate belief $\boldsymbol{b}_k$ about the state of the system at time k by a Gaussian distribution with mean $\boldsymbol{\mu}_k$ and variance $\boldsymbol{\Sigma}_k$. For brevity, we write $\boldsymbol{b}_k = (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, associating the belief with its sufficient statistics. Belief dynamics $(\boldsymbol{b}_{k-1}, \boldsymbol{u}_{k-1}, \boldsymbol{z}_k) \to \boldsymbol{b}_k$ is approximated by the extended Kalman filter [72, Chapter 3.3].

A cost function J can be a parsimonious description of the agent's objective. The agent will choose the next action by optimizing such a cost function with respect to all future actions at every time-step. To make the resulting optimal control computations numerically tractable, future observations need to be assumed to coincide with their most

likely values (see e.g., [73], [74]). Thus, at every time step, the agent solves a constrained nonlinear optimization problem

$$\begin{aligned} \min_{\boldsymbol{u}_{0:N-1}} \quad & J(\boldsymbol{\mu}_{0:N}, \boldsymbol{\Sigma}_{0:N}; \boldsymbol{u}_{0:N-1}) \\ \text{s.t.} \quad & \boldsymbol{u}_k \in \mathcal{U}_{\text{feasible}}, \quad k = 0 \dots N-1, \\ & \boldsymbol{\mu}_k \in \mathcal{X}_{\text{feasible}}, \quad k = 0 \dots N, \end{aligned} \tag{3.3}$$

which returns an optimal sequence of controls $\boldsymbol{u}_{0:N-1}$ minimizing the objective function J. The agent executes the first action, obtains a new observation, and replans again; such an approach is known as *model predictive control*. The policy resulting from such computations is sub-optimal because of open-loop planning and limited time horizon, but with growing time horizon it approaches the optimal policy. Reaction time τ_r can be incorporated by delaying the observations. An interesting property of this model is that the catching agent decides on his own in an optimal way when to gather information by looking at the ball and when to exploit already acquired knowledge depending on the level of uncertainty he agrees to tolerate.

3.3.2. A Computational Model of the Catching Agent

Here we explain the modeling assumptions concerning states, actions, state transitions, and observations. After that we describe the cost function that the agent has to minimize.

States and Actions. The state of the system $\boldsymbol{x}$ consists of the location and velocity of the ball in 3D space, the location and velocity of the catching agent in the ground plane, and the agent's gaze direction represented by a unit 3D vector. The agent's actions $\boldsymbol{u}$ consist of the force applied to the center of mass and the rate of change of the gaze direction.

State Transitions and Observations. Several model components are essential to faithfully describe catching behavior. First, the state transfer is described by the damped dynamics of the agent's center of mass $\ddot{\boldsymbol{r}}_c = \boldsymbol{F} - \lambda \dot{\boldsymbol{r}}_c$, where $\boldsymbol{r}_c = [x, y]$ are the agent's Cartesian coordinates, $\boldsymbol{F}$ is the applied force resulting from the agent's actions, and λ is the damping coefficient. Damping ensures that the catching agent's *velocity does not grow without bound* when the maximum force is applied. The magnitude of the maximal force and the friction coefficient are chosen to fit Usain Bolt's sprint data[1]. Second, the gaze vector's direction $\boldsymbol{d}$ is controlled through the first derivatives of the two angles that define it. These are the angle between $\boldsymbol{d}$ and its projection onto the xy-plane and the angle

[1] Usain Bolt's world record sprint data http://datagenetics.com/blog/july32013/index.html

between $\boldsymbol{d}$'s projection onto the xy-plane and the x-axis. Such parametrization of the actions allows for realistically *fast changes of gaze direction*. Third, the maximal running speed depends on the gaze direction, e.g., *running backwards is slower* than running forward or even sideways. This relationship can be incorporated through dependence of the maximal applicable force $\boldsymbol{F}_{\text{max}}$ on the direction $\boldsymbol{d}$. It can be expressed by limiting the magnitude of the maximal applicable force $|\boldsymbol{F}_{\text{max}}(\theta)| = F_1 + F_2 \cos\theta$, where θ is the angle between $\boldsymbol{F}$ (i.e., the direction into which the catcher accelerates) and the projection of the catcher's gaze direction $\boldsymbol{d}$ onto the xy-plane. The parameters F_1 and F_2 are chosen to fit human data on forward and backwards running[2]. The resulting continuous time dynamics of agent and ball are converted into discrete time state transfers using the classical Runga-Kutta method. Fourth, the *observation uncertainty depends on the state*, which reflects the fact that humans' visual resolution falls off across the visual field with increasing distance from the fovea. When the ball falls to the side of the agent's field of view, the uncertainty about ball's position grows according to $\sigma_o^2 = s(\sigma_{\text{max}}^2(1 - \cos\Omega) + \sigma_{\text{min}}^2)$ depending on the distance to the ball s and the angle Ω between gaze direction $\boldsymbol{d}$ and the vector pointing from the agent towards the ball. The parameters $\{\sigma_{\text{min}}, \sigma_{\text{max}}\}$ control the scale of the noise. The ball is modeled as a parabolic flight perturbed by Gaussian noise with variance σ_b^2.

Cost Function. The catching agent has to trade-off success (i.e., catching the ball) with effort. In other words, he aims at *maximizing the probability of catching the ball with minimal effort*. A ball is assumed to be caught if it is within reach, i.e., not further away from the catching agent than $\varepsilon_{\text{threshold}}$ at the final time. Thus, the probability of catching the ball can be expressed as $\Pr(|\boldsymbol{\mu}_b - \boldsymbol{\mu}_c| \leq \varepsilon_{\text{threshold}})$, where $\boldsymbol{\mu}_b$ and $\boldsymbol{\mu}_c$ are the predicted positions of the ball and the agent at the final time (i.e., parts of the belief state of the agent). Since such beliefs are modeled as Gaussians, this probability has a unique global maximum at $\boldsymbol{\mu}_b = \boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_N \to \mathbf{0}^+$. Therefore, a final cost $J_{\text{final}} = w_0 \|\boldsymbol{\mu}_b - \boldsymbol{\mu}_c\|_2^2 + w_1 \operatorname{tr} \boldsymbol{\Sigma}_N$ can approximate the negated log-probability of successfully catching the ball while rendering the optimal control problem solvable. The weights w_0 and w_1 are set to optimally approximate this negated log-probability. The desire of the agent to be energy efficient is encoded as a penalty on the control signals $J_{\text{energy}} = \tau \sum_{k=0}^{N-1} \boldsymbol{u}_k^T \boldsymbol{M} \boldsymbol{u}_k$ with the fixed duration τ of the discretized time steps and a diagonal weight matrix $\boldsymbol{M}$ to trade-off controls. Finally, we add a term that penalizes agent's uncertainty at every time step $J_{\text{running}} = \tau w_2 \sum_{k=0}^{N-1} \operatorname{tr} \boldsymbol{\Sigma}_k$ that encodes the agent preference of certainty over uncertainty. It appears naturally in optimal control problems when the maximum likelihood observations assumption is relaxed [75] and captures how final uncertainty distributes over the preceding time steps,

[2]World records for backwards running http://www.recordholders.org/en/list/backwards-running.html

but has to be added explicitly within the model predictive control framework in order to account for replanning at every time step. The complete cost function is thus given by the sum

$$J = \underbrace{w_0 \|\boldsymbol{\mu}_b - \boldsymbol{\mu}_c\|_2^2}_{\text{final position}} + \underbrace{w_1 \operatorname{tr} \boldsymbol{\Sigma}_N}_{\text{final uncertainity}} + \underbrace{\tau w_2 \sum_{k=0}^{N-1} \operatorname{tr} \boldsymbol{\Sigma}_k}_{\text{running uncertainty}} + \underbrace{\tau \sum_{k=0}^{N-1} \boldsymbol{u}_k^T \boldsymbol{M} \boldsymbol{u}_k}_{\text{total energy}}$$

that the catching agent has to minimize in order to successfully intercept the ball.

3.3.3. Implementation Details

To solve Problem (3.3), we used the covariance-free multiple shooting method [76] for trajectory optimization [77], [78] in the belief space. Derivatives of the cost function were computed using CasADi [79]. Non-linear optimization was carried out by Ipopt [80]. L-BFGS and warm-starts were used.

3.4. Simulated Experiments and Results

In this section, we present the results of two simulated scenarios and a comparative evaluation. First, using the optimal control approach, we show that continuous tracking (where the ball always remains in the field of view of the outfielder) naturally leads to the heuristics from literature [56], [67], [68] if the catching agent is sufficiently fast in comparison to the ball independent of whether he is running forward, backwards, or sideways. Subsequently, we show that more complex behavior arises when the ball is too fast to be caught while running only sideways or backwards (e.g., as in soccer or long passes in American football). Here, tracking is interrupted as the agent needs to turn away from the ball to run forward. While the heuristics break, our optimal control formulation exhibits plausible strategies similar to those employed by human catchers. Finally, we systematically study the effects of noise and time delay onto the agent's policy. The optimal control policies arising from our model switch between reactive and predictive behaviors depending on uncertainty and latency.

3.4.1. Continuous Tracking of an Outfielder—Heuristics Hold

To directly compare our model against empirical catching data that has been described as resulting from a heuristic, we reproduce the settings from [67] where a ball flew $15\,\mathrm{m}$ in $3\,\mathrm{s}$ and a human subject starting about $6\,\mathrm{m}$ away from the impact point had to intercept it. The optimal control policy can deal with such situations and yields the behavior observed

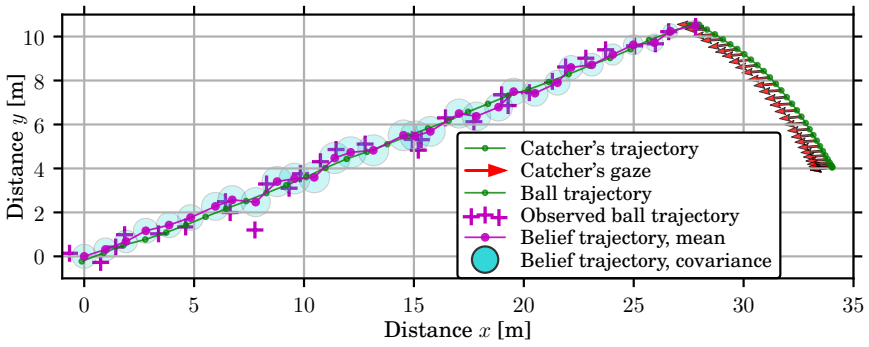


Figure 3.2.: A typical simulated trajectory of a successful catch in the continuous tracking scenario as encountered by the outfielder. The uncertainty in the belief state is kept low by the agent by fixating the ball. Such empirically observed scenarios [56], [67], [68] have led to the proposition of the heuristics which arise naturally from our optimal control formulation.

by McLeod *et al.* [67]. In fact, even when doubling all distances the reactive control policy exhibits all four major heuristics (OAC, GOAC, CBA and LOT) with approximately the same precision as in the original human experiments. Figure 3.2 shows a typical simulated catch viewed from above. The ball and the agent's true trajectories are depicted in green (note that the ball is frequently hidden behind the belief state trajectory). The agent's observations and the mean belief trajectory of the ball are represented by magenta crosses and a magenta line, respectively. The belief uncertainty is indicated by the cyan ellipsoids that capture 95% of the probability mass. The gaze vectors of the agent are shown as red arrows. The catching agent starts sufficiently close to the interception point to continuously visually track the ball, therefore he is able to efficiently reduce his uncertainty on the ball's position and successfully intercept it while keeping it in sight. Note that the agent does not follow a straight trajectory but a curved one in agreement with human experiments [67].

Figure 3.3 shows plots of the relevant angles over time to compare the behavior exhibited by human catchers to the optimal catching policy. The tangent of the elevation angle $\tan\alpha$ grows linearly with time, as predicted by the optic acceleration cancellation heuristic (OAC). The bearing angle γ remains constant (within a $5\,\text{deg}$ margin) as predicted by the constant bearing angle heuristic (CBA). The rotation angle δ oscillates around γ as predicted by the generalized optic acceleration cancellation theory (GOAC). The tangent of the horizontal optical angle $\tan\beta$ is proportional to $\tan\alpha$, as predicted by the linear

optical trajectory theory (LOT). The small oscillations in the rotation angle and in the horizontal optical angle are due to reaction delay and uncertainty; they are also predicted by GOAC and LOT. Thus, in this well-studied case, the model produces an optimal policy that exhibits behavior which is fully in accordance with the heuristics.

3.4.2. Interrupted Tracking During Long Passes—Heuristics Break

The competing theory to the heuristics claims that a predictive internal model allows humans to intercept the ball [57], [58]. Brancazio [81] points out that "the best outfielders can even turn their backs to the ball, run to the landing point, and then turn and wait for the ball to arrive". Similar behavior is observed in football and american football during long passes. To see whether predictions become necessary, we reproduced situations where the agent cannot catch the ball when acting purely reactively. For example, if the running time to interception point when running backwards (i.e., the ratio between the distance to the interception point divided by the maximal backwards running velocity) is substantially higher than the flight time of the ball, no backwards running strategy will be successful. Thus, by varying the initial conditions for the catching agent and the ball, new scenarios can be generated using our optimal control model. The agent's control policy can be tested on reliance on predictions as it is available in form of a computational model, i.e., if the computed policy makes use of the belief states on future time steps, the agent clearly employs an internal model to predict the interception point. By choosing appropriate initial conditions for the ball and the agent, we can pursue such scenarios. For example, if the ball flies over the agent's head, he has to turn away from it for a moment in order to gain speed by running forward, instead of running backwards or sideways and looking at the ball all the time. Figure 3.4 shows such an interception plan where the agent decides to initially speed up and, when sufficiently close, turn around and track the ball while running sideways. Notice that the future belief uncertainty (i.e., the posterior uncertainty Σ returned by the extended Kalman filter), represented by red ellipses, grows when the catcher is not looking at the ball and shrinks otherwise. The prior uncertainty (obtained by integrating out future observations), shown in yellow, on the other hand, grows towards the end of the trajectory because future observations are not available at planning time. Similar to [74], [82], we can show for our model predictive control law that the sum of prior and posterior uncertainties (shown as green circles) equals the total system uncertainty obtained by propagating the belief state into the future without incorporating future observations. Figure 3.5 shows that the heuristics fail to explain this catch—even in the final time steps where the catching agent is tracking the ball to intercept it. OAC deviates from linearity, CBA is not constant, the tracking heuristic wildly deviates from the prediction, and LOT is highly non-linear. GOAC and LOT are affected

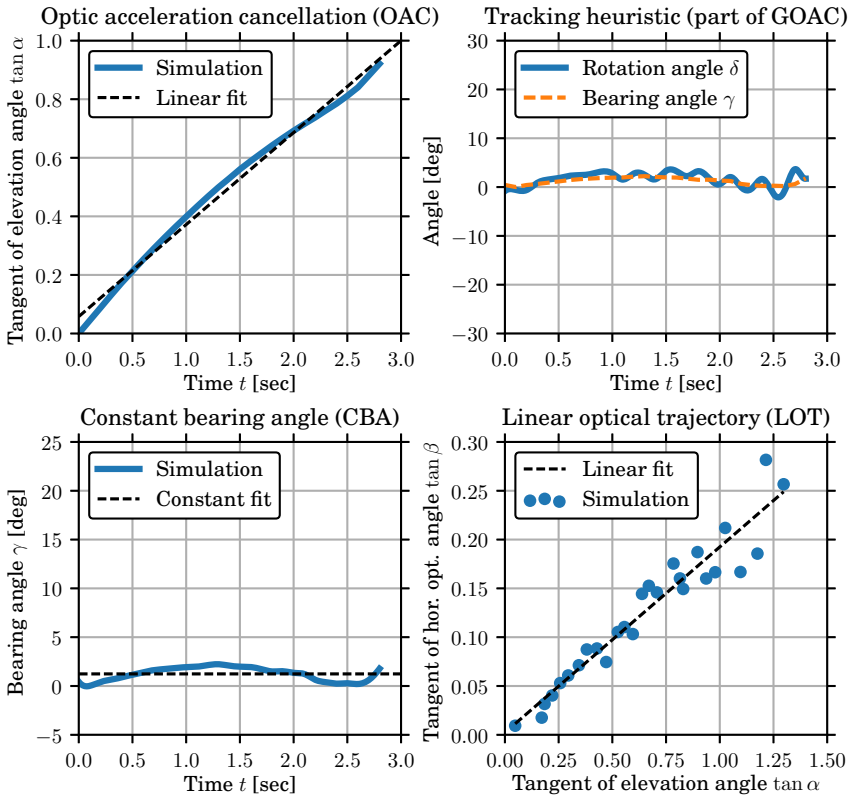


Figure 3.3.: In simulations of successful catches for the continuous tracking scenario encountered by the outfielder (shown in Figure 3.2), the policies resulting from our optimal control formulation always fulfill the heuristics (OAC, GOAC, CBA, and LOT) from literature with approximately the same precision as in the original human experiments.

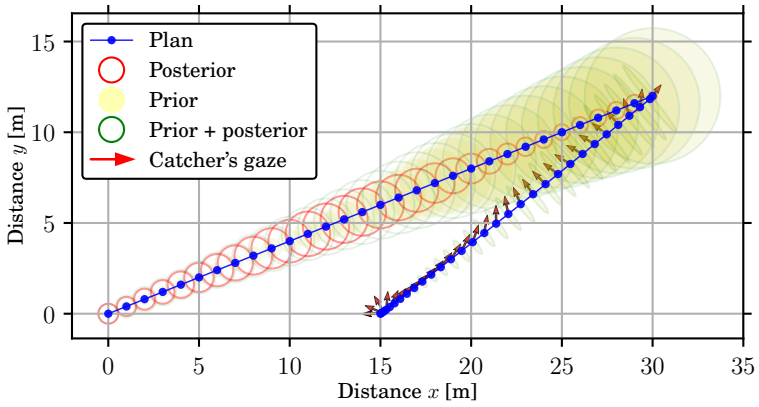


Figure 3.4.: An interception plan that leads to successful catch despite violating heuristics. Here, the agent would not be able to reach the interception point in time while running backwards and, thus, has to turn forward to run faster. The resulting optimal control policy relies on beliefs on the future generated by an internal model.

more dramatically because they directly depend on the catcher's gaze, in contrast to OAC and CBA. Since the heuristics were not meant to describe such situations, they predictably do not hold. Only an internal model can explain the reliance of the optimal policy on the future belief states.

3.4.3. Switching Behaviors When Uncertainty and Reaction Time are Varied

The previous experiment has pointed us towards policies that switch between predictive subpolicies based on internal models and reactive policies based on current observations. To systematically study what behaviors arise, we use the scenario from Section 3.4.2 and vary two essential model parameters: system to observation noise ratio $\eta_1 = \log \sigma_b^2/\sigma_o^2$ and reaction time to task duration ratio $\eta_2 = \tau_r/T$, where T is the duration of the ball flight. The system to observation noise ratio effectively determines whether predictions based on the internal model of the dynamics are sufficiently trustworthy for (partially) open-loop behavior or whether reactive control based on the observations of the current state of the system should be preferred. The reaction time to task duration ratio sets the time scale of the problem. For example, an outfielder in baseball may have about $3\,\mathrm{s}$ to catch a ball and his reaction delay of about $200\,\mathrm{ms}$ is negligible, whereas a catcher in baseball often has to act within a fraction of a second, and, thus, the reaction latency

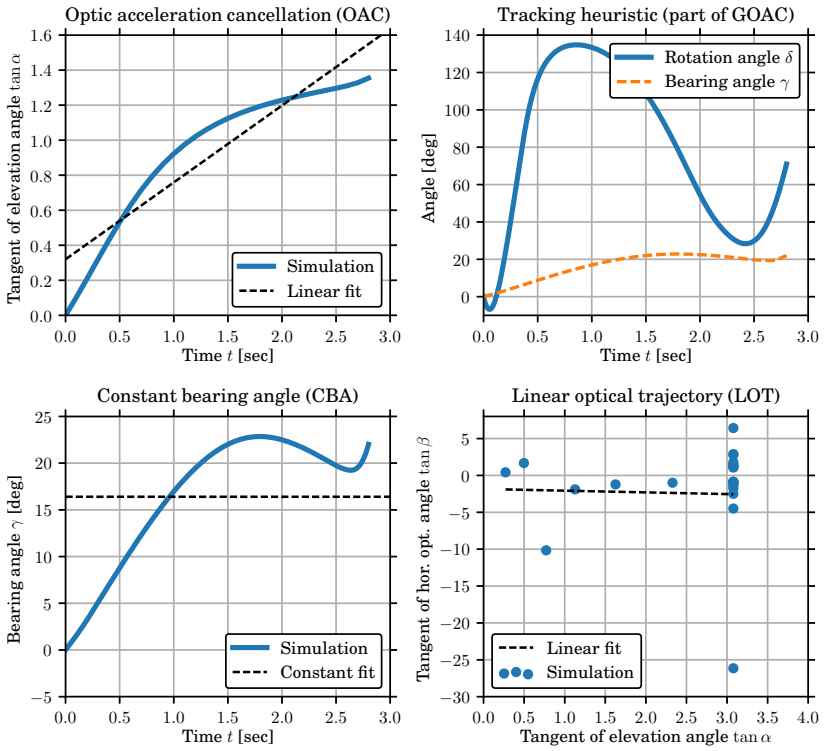


Figure 3.5.: For initial conditions (positions of the ball and the agent) which do not allow the agent to reach the interception point by running backwards or sideways, the optimal policy will include running forward with maximal velocity (as shown in Figure 3.4). In this case, the agent cannot continuously visually track the ball and, expectedly, the heuristics do not hold.

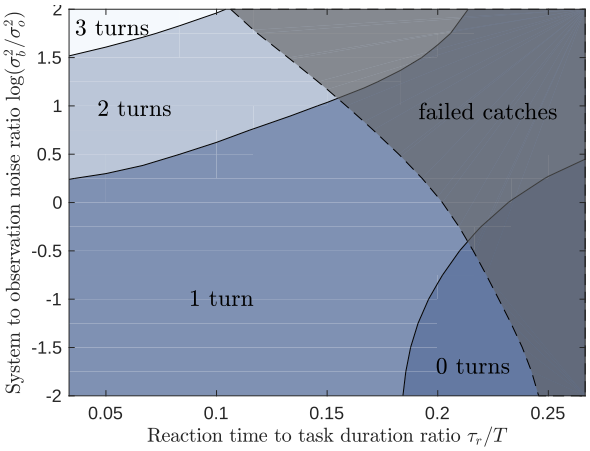


Figure 3.6.: Switches between reactive and feedforward policies are determined by uncertainties and latency.

becomes crucial. We run the experiment at different noise levels and time delays and average the results over 10 trials. In all cases, the agent starts at the point $(20, 5)$ looking towards the origin, while the ball flies from the origin towards the point $(30, 15)$ in 3 s. All parameters are kept fixed apart from the reaction time and system noise; in particular, task duration and observation noise are kept fixed. Figure 3.6 shows how the agent's policy depends on the parameters. Boundaries correspond to contour lines of the function that equals number of times the agent turns towards the ball. We count turns by analyzing trajectories for gaze direction changes and reduction of uncertainty (e.g., in Figure 3.4 the agent turns once towards the ball). When reaction delays are long and predictions are reliable, the agent turns towards the interception points and runs as fast as he can (purely predictive strategies; lower right corner in Figure 3.6). When predictions are not sufficiently trustworthy, the agent has to switch multiple times between a reactive policy to gather information and a predictive feedforward strategy to successfully fulfill the task (upper left corner). When reaction time and system noise become sufficiently large, the agent fails to intercept the ball (upper right grayed out area). Thus, seemingly substantially different behaviors can be explained by means of a single model. Note that in this figure a purely reactive strategy (as required for only using the heuristics) is not possible. However, if different initial conditions enabling the purely reactive strategy are used, the upper left corner is dominated by the purely reactive strategy.

3.5. Real-World Experiments With Human Catchers

This section presents the real-world experiments with human catchers which were carried out to validate the optimal control model of ball catching introduced in Section 3.3. The experiments were performed at a baseball field in order to faithfully reproduce the conditions encountered by outfielders in a game. Importantly, our model predicts behaviors that can only be validated at real scale, whereas previous studies on ball catching heuristics could also be performed at a smaller scale [67].

Figure 3.7 shows the baseball field that was used in our experiments. The tracking of the ball and the catcher was accomplished using a DJI Mavic 2 Pro drone. Videos in 2.7k resolution (2688x1512) at 30 FPS were captured from the height of 60m. Markers on the field were utilized to calibrate the camera. Participants wore hats to indicate the direction of the head. Standard OpenCV tools were used to extract the ball position and the position and orientation of the catcher's head. Additionally, a second drone recording the scene from the side was utilized to estimate the height of the ball trajectory and verify the direction of catcher's gaze. Furthermore, eye tracking glasses Pupil Invisible from Pupil Labs were worn by all participants in order to obtain reliable data on the direction of the gaze as opposed to the direction of the head only.

Balls were launched by the Combination Pitching Machine from Jugs Sports. Training baseballs were used, which are softer than normal balls, in order to reduce the likelihood of injury for inexperienced catchers. Fly balls with a backspin were launched to spend between 3-4s in the air and cover the distance of 50-70m. The experiments were carried out on a sunny day in August with light wind below 10km/h.

3.5.1. Selection of the Experiment Participants

Experiment participants were chiefly recruited among computer science students in the age group between 25 and 35 years old (except for one participant in the age group 35–45 years old). Only one participant had prior experience in practicing baseball, and there was one female participant. In total, 13 sessions were recorded, with each subject performing 20–30 catches from 6 different starting positions and in 2 configuration for each starting position—facing the ball and facing away from the ball. One subject terminated the session after 13 catches due to injury, and two subjects performed 50–60 catches in two sessions due to issues with the recording equipment which later was recognized as not essential and therefore all the captured data was used.

Participants of different athletic level and sport affinity were recruited. From the total number of participants, 54% regularly practiced ball games, while the remaining 46% were generally healthy subjects, with half of them regularly practicing other sports. Both

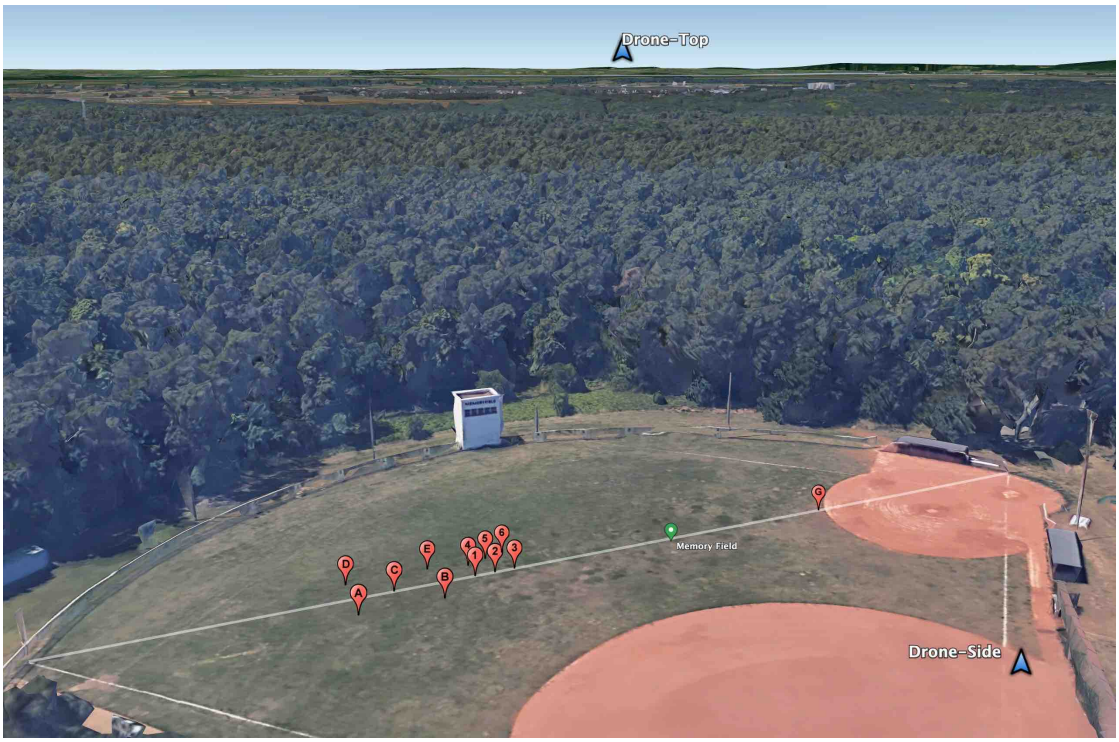


Figure 3.7.: A 3D view of the baseball stadium where real-world experiments with human catchers were carried out. A drone hovering at 60m above the ground recorded the catches from the top view. Another drone recorded the catches from the side. The ball was launched from Point G and landed in the area $ABED$ centered around C at 60m distance from the launching location G, while the catcher ran to intercept the ball starting from one of the positions $1, 2, \ldots, 6$ located 10–16m away from C.

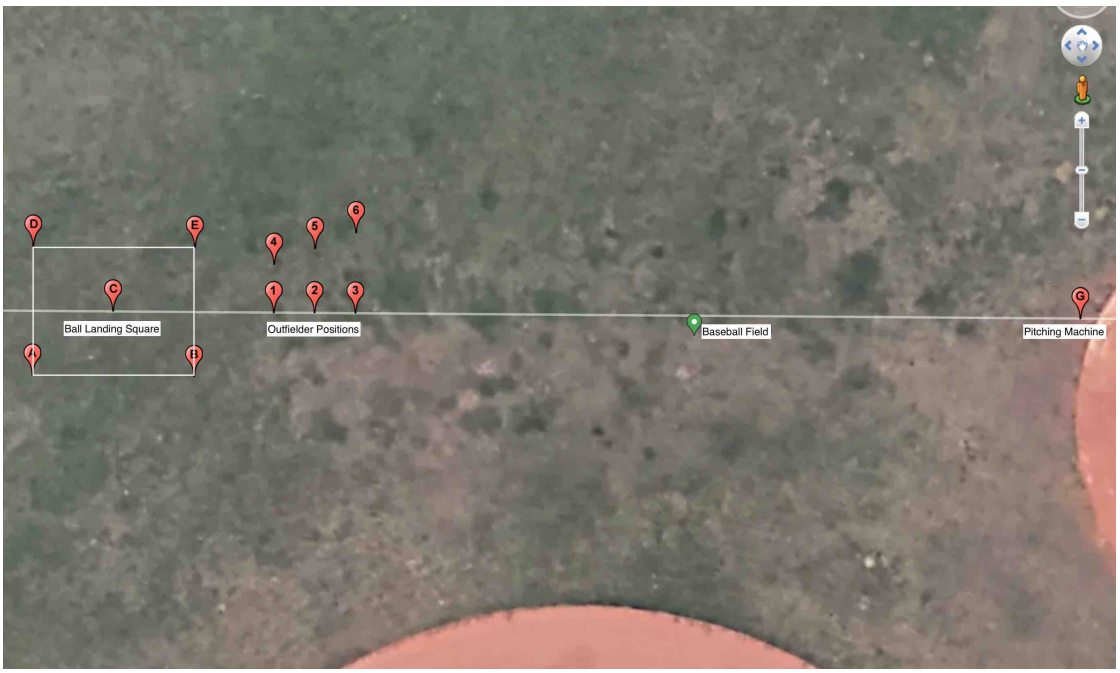


Figure 3.8.: Field setup as seen from the perspective of the drone. Point $C(0,0)$ denotes the center of the area in which the ball lands. Point $G(60,0)$, located 60m to the right from C, denotes the spot from which the ball is launched by a pitching machine. The area $ABED$ where the ball lands has the size 10x8m, where $|AB| = 10$. Six starting positions of the catcher are denoted by the numbers from 1 to 6 and their respective coordinates are given as follows $P_1(10,0), P_2(12.5,0), P_3(15,0), P_4(10,3), P_5(12.5,4), P_6(15,5)$.

right-handed (77%) and left-handed (23%) participants were present.

3.5.2. Experimental Setup

For each catcher, 6 different initial positions and 2 conditions per position were considered. Figure 3.8 shows the geometric configuration of the initial positions as well as the ball launching and landing areas. The positions $1, 2, \ldots, 6$ were chosen to cover a range of athletic conditions among human catchers. Position 1 is only 10m away from the ball landing location C, and therefore any healthy subject is expected to be able to catch the ball by lightly jogging backwards from this starting position. In contrast, Position

6 is almost 16m away, and furthermore it is shifted to the side from the line of the ball flight GC, which makes it much more challenging, and only the most sporty and trained participants are able to catch the ball starting from this position.

It is important to note that the arrangement of the cones shown in Figure 3.8 was used for right-handed catchers, who wear the glove on the left hand (the right hand is used for throwing the ball). The cones $4, 5, 6$ are located to the left when seen by the catcher looking at the pitching machine, because the catches from this side are more difficult as the glove is on the left hand. In order to accomplish the catches in a reliable manner, the participants need to overrun the landing location, such that the left hand is to the left from the body and the catcher is facing the ball. Otherwise, if the catcher does not run sufficiently fast, one is forced to stretch the left arm across the front of the body, which hinders the coordination and often leads to unsuccessful catches among inexperienced players. For the left-handed catchers, the cones $4, 5, 6$ were placed on the opposite side of the line CG to produce the same conditions for them, as they are wearing the glove on the right hand.

The two conditions per position correspond to facing the ball when it is launched and facing away from the ball when it is launched. We consider these two conditions to investigate the effect of having an initial eye contact with the ball on the catching performance. It is expected that facing away from the ball is a harder scenario because the catcher has less time to run since the ball is already high in the air by the time the catcher makes the first eye contact with it. Therefore, the catchers will start running even before looking at the ball, relying on a predictive strategy. A comparison to the case when the catchers are facing the ball can shed light on the trade-off between predictive and reactive strategies in ball catching.

3.5.3. Data Acquisition

Three data capturing devices were utilized in parallel to capture the details of each catch. First, the main data stream was obtained from the drone hovering at 60m above the ground, whose camera was pointed vertically towards the ground, providing a flat 2D image, similar to those reported for our model in the previous section. The top view from the drone above is shown in Figure 3.9. Second, another drone was hovering on the side, pointing its camera perpendicular to the vertical plane in which the ball trajectory lies. This view is shown in Figure 3.10. The side view provides additional information on the z-component of the ball position and allows for more detailed analysis of the catcher's gaze, thanks to its oblique perspective. Third, eye tracking glasses were utilized to capture the direction of the gaze of the catcher. Due to the scale of the experiment and the dynamic nature of the task, it was rendered impractical to precisely calibrate the

Figure 3.9.: Top view of the field from the camera on the drone. The ball is highlighted by a red circle for better visibility. The positions of the red cones denote the starting locations of the catcher and the landing area for the ball, as described in the figures above. This top view allows for extracting the position and orientation of the head of the catcher, as well the position of the ball.

Figure 3.10.: Side view of the ball catching setup recorded by the second drone. The ball is highlighted by a red circle for better visibility. Depending on the starting position of the catcher, the difficulty of the catch may vary. If the starting point is further away from the landing location, the catcher may need to temporarily turn away from the ball to run faster while attempting to intercept the ball. This side view was used for estimating the z-component of the ball trajectory and for verifying the direction of the catcher's gaze.

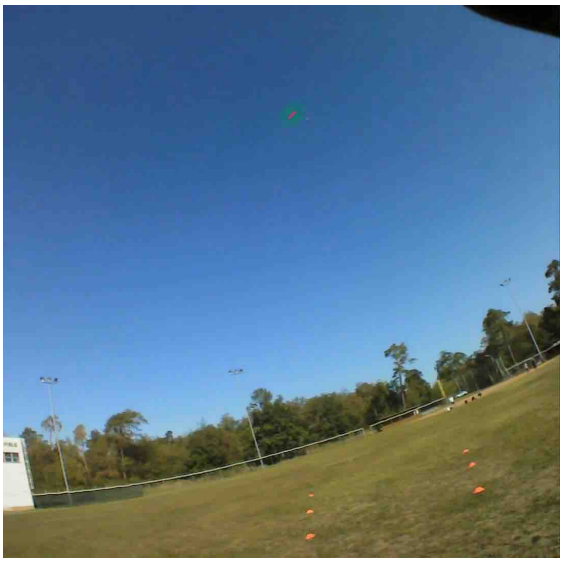


Figure 3.11.: View from the eye-tracking glasses. A built-in processing algorithm outputs the direction of gaze overlaid on the picture. The red dot in the center of the light green disk represents the estimated gaze direction and the uncertainty of that estimate. This view from the eye-tracking glasses was utilized to extract the gaze direction of the catcher, which is useful especially in ambiguous situations where the gaze is not directly observable from the drones.

glasses with the cameras of the drones. Nevertheless, the data stream obtained from the eye tracking glasses played an important role in extracting the gaze vector. As shown in Figure 3.11, the light green circle with a red dot in the middle represents the direction of the gaze, and it is pointed towards the ball. Especially when the catcher rotates the head, such information is useful for disentangling the head orientation from the gaze and determining the direction of the gaze directly.

3.5.4. Experimental Protocol

Each participant was given time to warm up and practice 15–20 catches to try every position at least once. After that, a recording session was started, which lasted 6–13 minutes per person depending on the physical condition of the catcher and the number of catches performed. Each catcher performed 20–30 catches per session, where each catch took on average 4s plus the time to prepare and return to position, giving in total 15–25s per trial. Altogether, 320 catches were recorded on the same day.

The participants were advised to try every of the 6 positions in every 2 conditions (facing the ball and facing away from the ball) at least once and decide to either drop position 1 if it is too simple for the recordings or to drop position 6 because it is too hard. At the remaining 5 positions times 2 conditions, 2 or more trials per position were recorded.

The number of trials per session was kept under 30 (plus 20 warm-up catches) for several reasons. First, the drone battery lasts for up to 20 minutes, and including the time for preparation of the two drones and the eye tracking glasses, 15 minutes of pure recording time were available per session. Second, subjects get tired over time, therefore the sessions were kept short to minimize the effects of fatigue. Finally, subjects get used to the setup, which may also affect the results over time. The selection of participants from a population mostly unfamiliar with baseball ensured that the bias towards a particular styles of catching was reduced.

The sequence in which the positions were traversed was determined by the catchers themselves depending on their physical condition. In general, each position was repeated several times before a subject would proceed to the next position. In the majority of cases, the harder positions 4, 5, 6 were intermixed with the easier positions 1, 2, 3 for the catchers to get intermittent rest.

For the backward facing catches, the participants were advised to follow the sound of the pitching machine and look at the ball as soon as it is launched but not earlier. In the discussion of the results later on, we refer to the backward facing positions as 1b, 2b, ..., 6b. Catches from these positions always start with a gaze turn towards the ball, after which the catcher may either continue running while fixating the gaze on the ball or decide to make an additional gaze turn.

Table 3.1.: Summary of the data on gaze turns in the ball catching trials. The dataset is balanced with respect to forward and backward-facing trials, as well as to successful and failed catches. Trials with 0, 1, and 2 turns were observed, including catches with short turns away from the ball of duration less than 0.5s.

Statistics of the trials	
Total trials	320
Backward trials	140
Successful catches	149
Ball remained in glove	81
No gaze turns	62
Short (under 0.5s) turn	103
One full gaze turn	136
Two gaze turns	18

An important decision that an outfielder needs to make in the beginning of a catch is through which side to make the initial turn of the body. For right-handed people, the glove is on the left hand, and when one starts on the left side from the line of the ball flight, one needs to run to the backwards right. In this situation, a natural instinct may be to turn the body to the right, i.e., make the first step with the right foot backwards. However, this strategy will require an additional turn at the catching location unless one wants to intercept the ball with a stretched arm in front of the body. Such catches with a hand on the other side of the body are discouraged in professional baseball because they are hard to control. Nevertheless, the participants were not told explicitly through which side to turn, allowing their natural tendencies to manifest in the data.

3.5.5. Statistical Description of the Results

There were 320 catches recorded in total, which were collected in 13 sessions with 11 participants. In 7 sessions, 30 catches per session were recorded; in 5 further sessions, 20 catches per session were recorded. One session was terminated after 13 catches due to an injury. In each session, catches from 6 starting positions and 2 initial conditions were recorded.

Every catch was subsequently labeled with a number of labels: Catch (the ball ended in the glove of the catcher), Done (the ball touched the glove or passed in centimeters

away from the glove), Turn (number of turns of the gaze during the catcher's trajectory), Side (whether the catcher turn through the side of the hand with the glove or through the opposite side). Furthermore, each trajectory was labeled to indicate whether it satisfies the four heuristics. As a criterion for whether a trajectory satisfies a certain heuristic, a deviation of 5% was considered within the tolerance. Thus, for each trajectory, the binary labels for CBA, OAC, GOAC, and LOT were provided.

Successful Ball Interceptions and Catches. The complete dataset contains 47% successful catches, i.e., catches in which the ball was in immediate vicinity of the glove. This is the same criterion of success as used in the model, where if the ball is within a fixed distance from the catcher at the moment of interception, the catch is considered to be successful. Out of the successful catches, in 54% of the cases, the ball stayed in the glove of the catcher. This shows that the catchers in our sample succeeded in intercepting the ball roughly every second trial, and every second trial of those they manage to hold the ball in the glove.

Statistics of Gaze Turns. Out of the total of 320 catches, in 20% of the cases the catchers made no turns of the gaze. This means the majority of the catches in the dataset are beyond the realm of applicability of the heuristics, which require the catcher to always look at the ball. Note that making no turns is only possible if the catcher is already starting the catch by looking at the ball. Among the remaining 140 backward facing catches, which amount to 44% of the total number of catches, in 57% of the cases subjects made an immediate turn to the ball and kept the gaze fixed on the ball. The total number of catches in which participants made a clear one turn of the gaze amounts to 136 catches or 42% of the total number of trials. Furthermore, in 6% of the trials, subjects made 2 turns. In 7% of the cases, the gaze was shifted away from the ball only for a short period of time less than 0.5s.

Results on the Optic Acceleration Cancellation (OAC) Heuristic. Among the most salient observations, the OAC heuristic holds in 97% of the catches if one disregards the turning of the head. However, the heuristic assumes that the catcher is constantly optically tracking the ball. In this case, if we remove the trials in which the catcher turns away from the ball for a period of time, then OAC holds in under 20% of the cases.

Results on the GOAC Heuristic. Another robust observation is that the GOAC heuristic holds only in 20% of the cases. This is explained by the fact that GOAC does not hold if the catcher turns away from the ball even for a short moment. Since only 20% of the

Table 3.2.: Summary of the data on heuristics in the ball catching trials. As shown in Table 3.1, there were 62 catches in which the catcher kept the eyes on the ball at all times. Therefore, OAC and GOAC are indicative of these scenarios, as they were designed for the setting of constant ball tracking and our experiments confirmed that. Although the CBA heuristic holds in 90 cases, there was no clear pattern as to when it would hold. The LOT held in the most number of cases, but similarly to CBA, there were no strong correlations with either the success of the catches or the number of the gaze turns.

Data on heuristics	
Total trials	320
Backward trials	140
Successful catches	149
Ball remained in glove	81
OAC holds	62
GOAC holds	62
CBA holds	90
LOT holds	132
OAC & GOAC	62
OAC & CBA	33
OAC & LOT	26
CBA & LOT	10

catches in all of the data did not involve turns of the gaze, the GOAC was holding only in those cases when the gaze was always directed at the ball.

Results on the CBA Heuristic. The CBA heuristic held in 28% of the cases. Noteworthy, these catches were not the same in which the catcher made no turns. There was no obvious pattern that would explain when the CBA heuristic holds and when not. The catches from the easier positions 1 and 4 were more likely to obey the CBA heuristic, but even for the cases of no gaze turns for which the CBA heuristic was designed there was no clear correlation between the CBA heuristic holding and the success of the catches.

Results on the LOT Heuristic. The LOT heuristic held in 59% of the cases. There was no clear correlation between the LOT heuristic and catches observed. In some cases, it held when a catch was successful, sometimes it held when a catch was unsuccessful, and similarly for when it did not hold. Furthermore, the number of gaze turns also did not correlate with the LOT heuristic.

3.5.6. Comparison of Simulated and Real Trajectories

In this section we compare simulated and real trajectories. In order to validate the model, we ran simulations with the same initial conditions as provided by the real trajectories. In total, we carried out such comparison on 16 trials, and in the following we present 4 cases for illustration. Figure 3.12 shows the trajectories themselves, which correspond to different catchers and positions. The subsequent figures show the heuristics plots for the simulated and real trajectories. We observe that the model is closely following the data, reproducing the key features, such as the number of turns of the catcher's gaze, the general trends in the plots, and the simulated trajectories are matching the heuristics when the real trajectories do. Therefore, we conclude that our model provides a succint and parsimonious representation of the observed human trajectories in ball catching.

3.5.7. Conclusions From the Experimental Results

The human ball catching experiments were aimed at evaluating the proposed model. In 80% of the catches, the participants turn the gaze away from the ball for at least 0.5s and in 48% of the cases for more than 0.5s in total. Therefore, as the heuristics are based on the assumption that the catcher is constantly looking at the ball, they are not directly applicable to 80% of the catches in our dataset. Nevertheless, we observed that on the remaining 20% of the catches, the heuristics are largely obeyed.

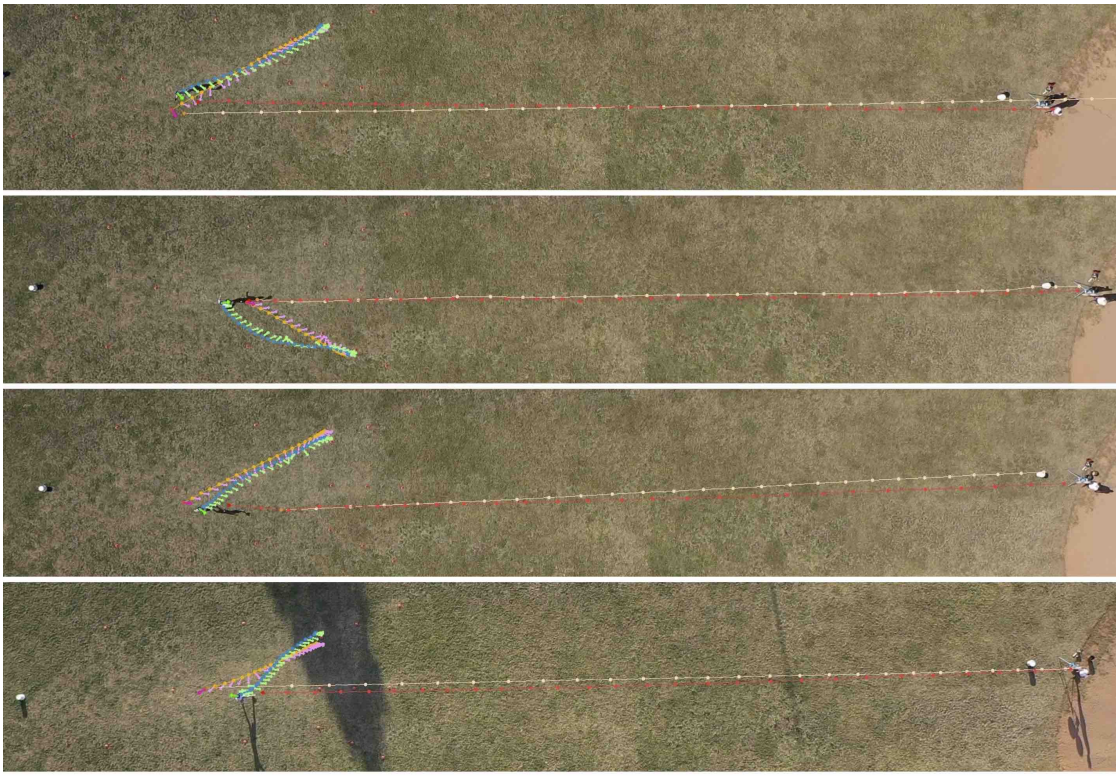


Figure 3.12.: Four example trajectories from the dataset together with the corresponding simulated trajectories. The real trajectories of the catcher are shown with blue circles and green arrows, and the real trajectories of the ball are shown with red circles. The other trajectories are simulated from the same initial conditions. The trajectories correspond to different catchers and the starting positions 5, 5b, 4, 4, respectively. All catches except the third one are successful. Figures below show the corresponding heuristics plots.

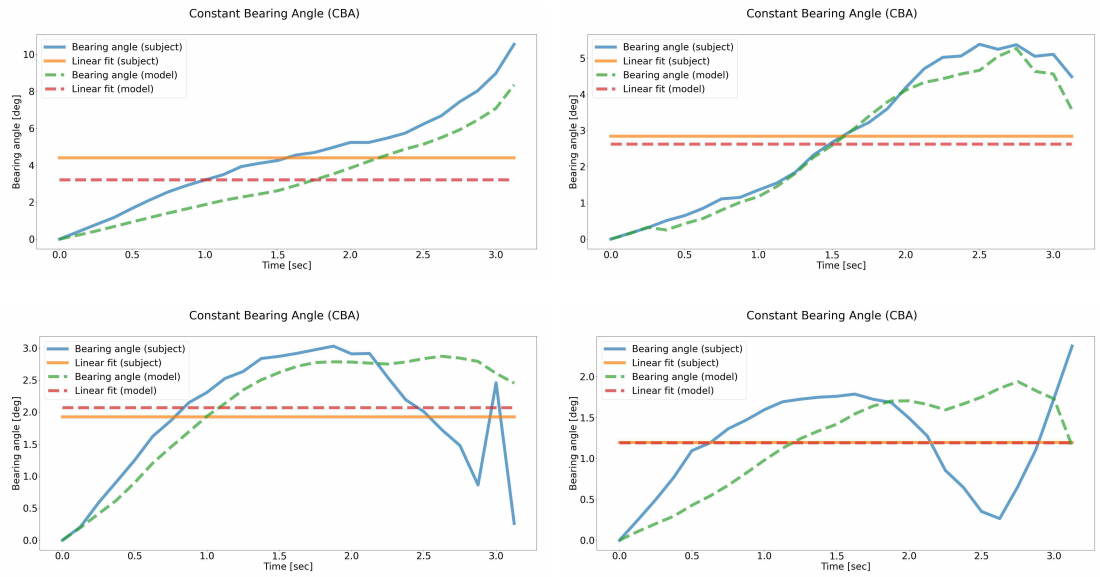


Figure 3.13.: Constant Bearing Angle (CBA) heuristic, comparing simulation and real data. Four example trajectories from different subjects and various initial positions are shown. The simulation is run with the same initial conditions as the real catches. The simulated trajectories closely follow the real trajectories, capturing the essential trends and remaining within a tight tolerance band. The CBA heuristic is not satisfied in the upper two plots and it is satisfied in the lower two plots according to our criterion that the deviation from a constant angle has to be within 1.5 degrees.

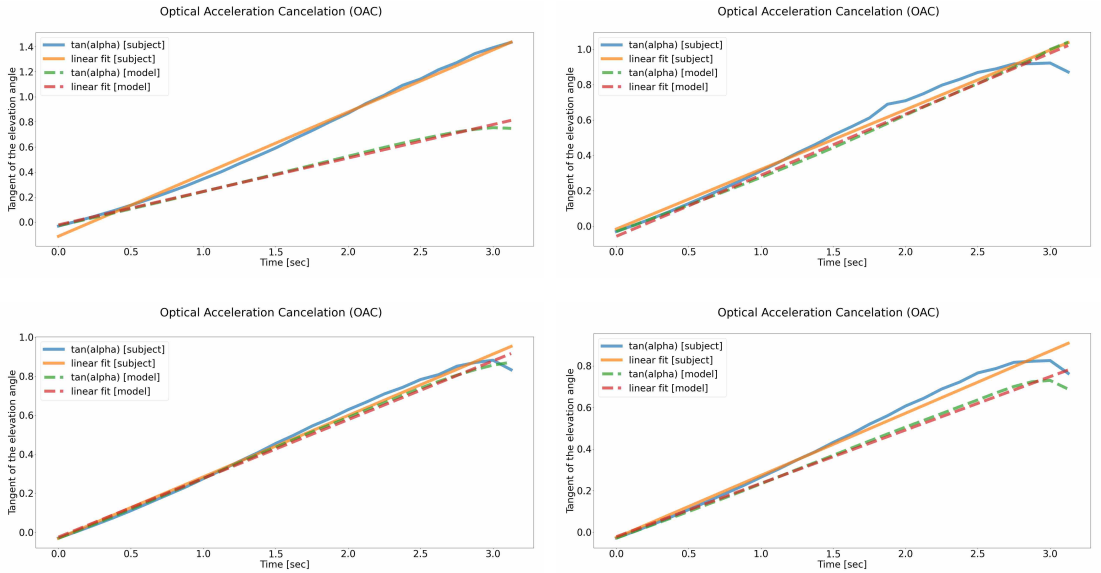


Figure 3.14.: Optic Acceleration Cancellation (OAC) heuristic, comparing simulation and real data. Four trajectories corresponding to the catches in Figure 3.13 are shown. The OAC heuristic reliably holds in 97% of the catches in the dataset, which is also reflected in this figure. However, only in 20% of the cases does the catcher continuously optically track the ball. Since the OAC is computed assuming that the gaze is always directed towards the ball, the four plots shown above demonstrate close alignment with the heuristic prediction. Nevertheless, as shown in the subsequent figure for GOAC Figure 3.15, tracking is interrupted in all four cases, and therefore we conclude that OAC does not hold in these trials. Finally, it is worth pointing out the close agreement between the simulation and the real trajectories of the OAC heuristic.

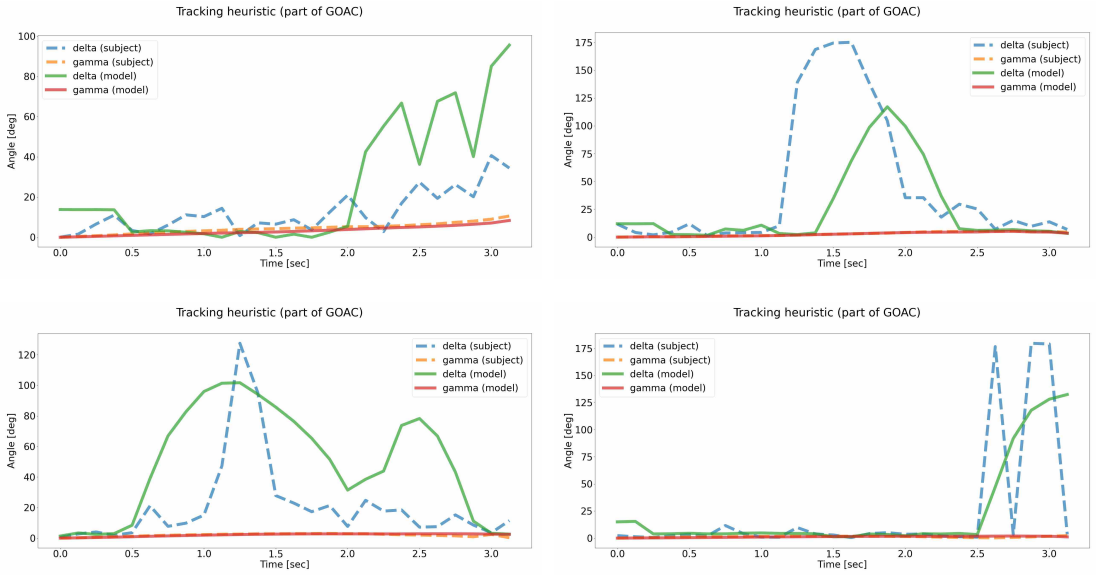


Figure 3.15.: Generalized Optic Acceleration Cancellation (GOAC) heuristic, comparing simulation and real data. Four trajectories corresponding to the catches in Figure 3.13 are shown. The simulations qualitatively match the observed trajectories. The timing of the peaks, which represent rotations of the catcher's gaze, and their shapes are not exactly aligned, but the key features of the behavior are captured. Namely, the simulated trajectories demonstrate half-turns or full turns and the location of the turns in time is aligned with up to 0.5s precision. Note that in none of the cases is the GOAC heuristic satisfied.

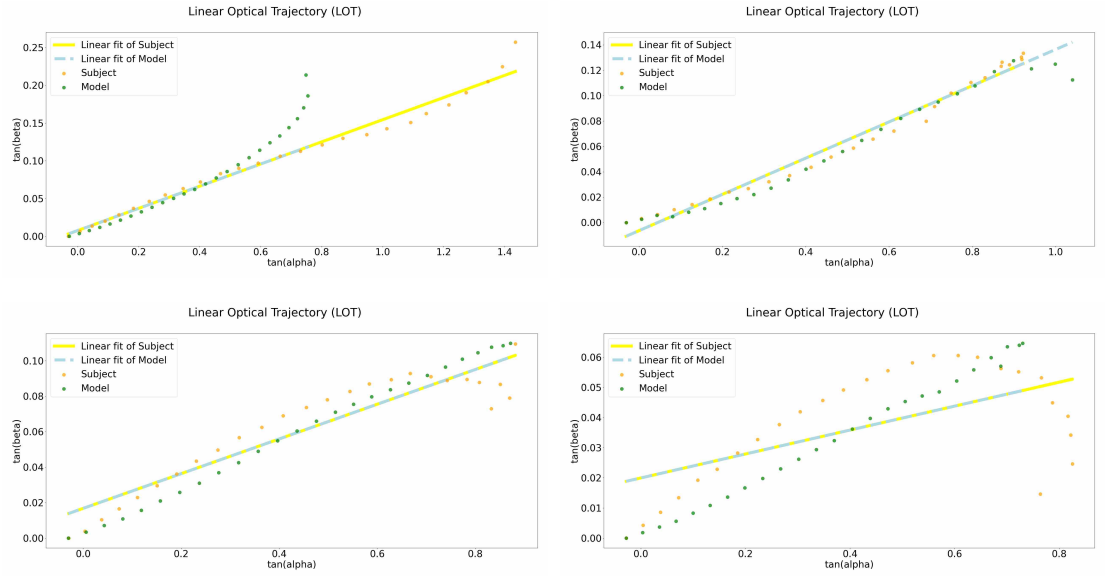


Figure 3.16.: Linear Optical Trajectory (LOT) heuristic, comparing simulation and real data. Four trajectories corresponding to the catches in Figure 3.13 are shown. The LOT heuristic in general holds in 41% of the trials. However, the LOT heuristic does not directly reveal if a catch was successful or if there were gaze turns during the catch. This figure shows that the linear fit is closely followed in all four cases, despite the fact that the bottom-left catch was unsuccessful while the top-left and bottom-right catches involved no gaze turns. The alignment of the simulation with the real data is close; if one considers the scale of the axes, the deviations are within a small tolerance bound.

Interestingly, some heuristics hold even for the catches when the catcher turns away from the ball, although they were not found to be correlated with the success of the catches. In particular, OAC was found to hold in 97% of the cases, if one disregards the fact that the catcher is not looking at the ball at some portions of the trajectory. By definition, the OAC requires a closed loop between perception and action, but we observed that humans continue following OAC even without looking at the ball. This suggest that humans may rely on a model-based prediction during the moments when they are not looking at the ball. The CBA and LOT heuristics were found to hold in 28% and 41% of the trials, respectively. However, no clear patter was observed with regards to when these heuristics hold and when they are violated. Therefore, due to such variability in the results for these heuristics, it appears that CBA and LOT do not generalize outside their initial scope, i.e., they cannot account for the catches when the subject turns the gaze away from the ball while running.

The experiments in Section 3.5.6 evaluated the model by running it with the same initial conditions as observed in the real trajectories. With the corresponding choice of the model parameters, the simulated trajectoreis were shown to match the real trajectories with a surprising degree of closeness, as demonstrated in Figure 3.12. The heuristic trajectories in Figures 3.13, 3.14, 3.15, 3.16 show that the simulated trajectories obey the heuristics when the real trajectories do, thus providing a further validation of the power of the proposed model to faithfully represent human trajectories in ball catching.

3.6. Discussion and Conclusion

We have presented a computational model of human interception of a moving target, such as an airborne ball, in form of a continuous state-action partially observable Markov decision problem. Depending on initial conditions, the optimal control solver either generates continuously tracking behavior or dictates the catching agent to turn away from the ball in order to speed up. Interception trajectories in the first case turn out to demonstrate all properties that were previously taken as evidence that humans avoid complex computations by employing simple heuristics. In the second case, we have shown that different regimes of switches between reactive and predictive behavior arise depending on relative uncertainty and latency. When the agent has sufficient time to gather observations (bottom-left in Figure 3.6), he turns towards the ball as soon as possible and continuously tracks it till the end (e.g., outfielder in baseball acts in this regime). If he is confident in the interception point prediction but the task duration is so short relative to the latency that he does not have sufficient time to gather observations (bottom-right), he will rely entirely on the internal model (e.g., catcher in baseball may

act in this regime). If the agent's interception point prediction is rather uncertain (e.g., due to system noise), the agent will gather observations more often regardless of time delays. Conclusions regarding the trade-off between reactive and predictive behaviors may well generalize beyond ball catching to various motor skills. Assuming an agent has an internal model of a task and gets noisy delayed partial observations, he has to tolerate a certain level of uncertainty; if moreover the agent has a limited time to perform the task, he is compelled to act based on prediction instead of observations. As our optimal control policy can explain both reactive heuristics and predictive feedforward strategies, as well as switches between these two kinds of subpolicies, it can be viewed as a unifying explanation for the two seemingly contradictory theories of target interception.

In this paper, we have provided a computational level explanation for a range of observed human behaviors in ball catching. Importantly, while previous interpretations of whether human catching behavior is the result of complex computations or the result of simple heuristics have been inconclusive, here we have demonstrated that what looks like simple rules of thumb from a bag of tricks is actually the optimal solution to a continuous partially observable Markov decision problem. This result therefore fundamentally contributes to our understanding of human rationality.

Acknowledgements

The stadium Memory Field was generously provided for the experiments by the Darmstadt Whippets baseball team. We especially thank Boris Feldman for his help in setting up the experiments, providing the equipment, and assisting in the initial trials. His support in organizing the experiments and his insight into the strategies that real baseball players follow and explicitly learn were invaluable for the successful execution of the real-world evaluations of our model. Furthermore, we sincerely thank the team members of Darmstadt Whippets who participated in the initial trials and who taught us the basics of ball catching and demonstrated the range of policies known to professional players for different catching scenarios.

4. Robotic Architectural Assembly with Tactile Skills: Simulation and Optimization

Construction is an industry that could benefit significantly from automation, yet still relies heavily on manual human labor. Thus, we investigate how a robotic arm can be used to assemble a structure from predefined discrete building blocks autonomously. Since assembling structures is a challenging task that involves complex contact dynamics, we propose to use a combination of reinforcement learning and planning for this task. In this work, we take a first step towards autonomous construction by training a controller to place a single building block in simulation. Our evaluations show that trial-and-error algorithms that have minimal prior knowledge about the problem to be solved, so called model-free deep reinforcement learning algorithms, can be successfully employed. We conclude that the achieved results, albeit imperfect, serve as a proof of concept and indicate the directions for further research to enable more complex assemblies involving multiple building elements.

4.1. Introduction

The global challenges of architecture and the construction industry, such as CO_2 emissions, waste production, and low productivity, are currently being approached by computational design, digital fabrication and robotics. Digitally prefabricated modular construction systems promise to improve the way we build in terms of productivity, precision, and future reuse. Computational design tools and digital fabrication allow for parametric mass customization of modules or complex combinatorics of discrete elements. However, the robotic assembly and future re-assembly of those elements on site, under messy and unpredictable circumstances is still an unsolved problem.

The digital continuity from design to production is interrupted when it comes to assembly, (dis-)assembly and (re-)assembly. However, automation is needed to make such a circular economy approach economically feasible. One possible way to achieve this ambitious goal is explored in this research project: The use of learning robots that do not need

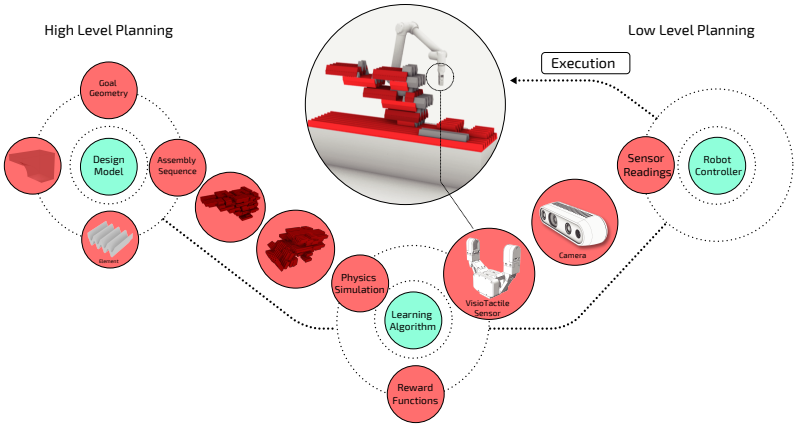


Figure 4.1.: Learning-based control architecture for autonomous assembly.

to be preprogrammed but act autonomously under closed-loop sensory feedback. We intend to develop novel robotic assembly processes and learning-based algorithms. We subdivided the assembly procedure into three steps: grasping a part at the pickup location, transporting it to the vicinity of the target location, and finally placing the part. The research then focused on trajectory planning for transportation and reinforcement learning (RL) for the part placement within the assembly process.

Recent developments in deep reinforcement learning have demonstrated the feasibility of extracting patterns from high-dimensional signals produced by visual and other sensors [83] thereby improving both the perception and actuation pipelines and enabling real robot control through reinforcement learning [84], [85]. Therefore, several tasks that were previously not feasible are now within reach.

In this paper, we are concerned with the problem of autonomously connecting building modules (Fig. 4.2). It is a challenging contact-rich manipulation task that requires coordination between perception and action, between observing the visual and tactile sensor readings and sending action commands to the robot joints and the gripper. The aspired functionalities are aimed to enable component re-use, disassembly and re-assembly of parts in the sense of a productive and effective circular economy in construction. The code is available at `https://github.com/b4be1/rl-arch-assembly`.

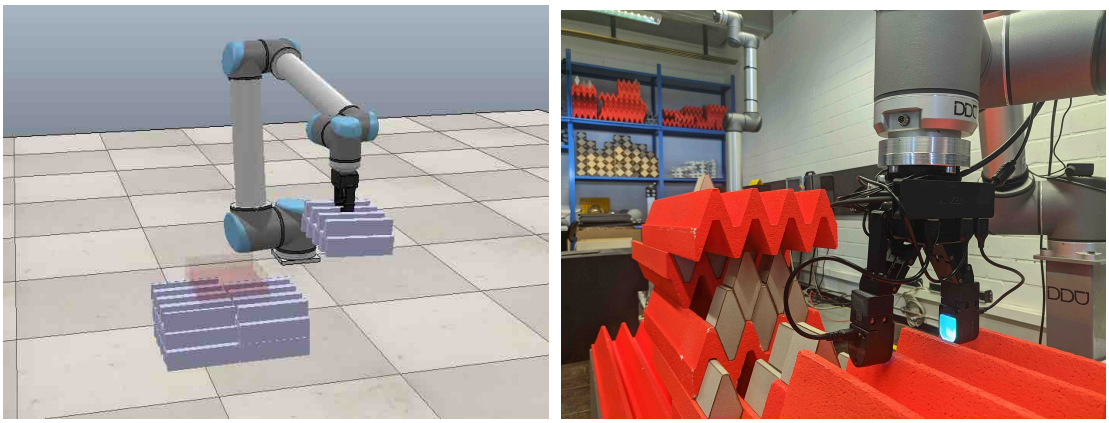


Figure 4.2.: Simulated and real-robot setups for learning block assembly. The robot is equipped with a gripper and tactile sensors to enable fine contact-rich manipulation.

4.2. Related Work

Re-assembly through in-built manipulators that change space and construction during or after its use is subject to research and design speculation [86]. Early studies date back to the 1970's. The Architecture Machine Group at MIT explored the impact of constant re-assembly of a voxelated landscape populated with gerbils. While the animals constantly changed the voxel configuration, the robotic system (SEEK) sought to keep track of the "continually changing mismatch between three-dimensional reality and the model residing in the core memory of the computer" [87].

More recently, the design team R&Sie(n) put forth the project Olzweg to speculate on the design of constant re-assembly. Here a robot that constantly repositions elements becomes an integral part of the architecture. Space is in constant flux [88]. In a similar vein, Fabio Gramazio and Matthias Kohler presented the Endless Wall at the "Scientifica 2011", where a mobile robot unit builds and rebuilds a curved brick wall and constantly controls the results by means of 3D scan technology and is furthermore able to react in real-time on any deviations and adapt during the assembling process. These projects aimed at exploring the use of robotic-based fabrication methods on construction sites, considering aspects such as complexity of construction sites and sensing capabilities of robots in regard to its own position, the surroundings and building elements [86].

In 2018, the authors designed an installation consisting of a modular structure, an

industrial robot arm and a light system as a contribution to the light festival Luminale in Frankfurt, Germany. The installation is shown in Fig. 4.3. The corrugated modules made from EPS foam were constantly re-assembled by a UR 10 robot to change the form and meaning of the installation. The robot became an integral part of the installation and the modules were designed for robotic (re-)assembly. Their zig-zag shape provided self-calibrating capacities for stacking, while the longitudinal profile accepted small imperfections during the placement of the elements. The installation carried an anamorphic image that visitors could decipher only after finding the correct point of view. Once they identified the message, the robot would deconstruct it by re-positioning the modules. The project explored the constant change of space due to changing light conditions and permanent (re-) assembly.

Further academic research to connect the material system with robotic systems was conducted in the projects Material-Robot System (MRS) [89] and Robot-oriented design (ROD) [90]. Here, the design of elements follows specific considerations of the robot's strengths and capabilities, including connections between elements and error-correction through self-alignment. In order to foster an understanding of the challenges in the robotic assembly process, designers simulate the robotic construction inside the digital design environment. Hence, assembly workflows become an integral part of a continuous design process.

The computational problems arising in robotic fabrication and assembly are challenging and therefore new methods and solutions at the interface between digital design and computer science are being explored. A methodology for using neural networks as a design tool in architecture was proposed in [91], where convolutional autoencoders were employed to predict the geometry of curved metal surfaces from a tracking pattern applied on the English wheel and vice versa. This allows the designer to make use of the trained models in both directions: as a way to design a surface and observe the resulting imprinted pattern, which is the traditional design-to-fabrication sequence, but also in a more crafty way of designing the desired fabrication marks and observing their effect on the geometry. A conceptually similar approach based on supervised learning has been applied to robotic carving [92], where human demonstrations were used as the training data to teach the robot to perform skillful timber manufacturing.

Although supervised learning is quite powerful and can be integrated in many ways into the creative design process as a regression subroutine, some types of problems require a qualitatively different approach. Namely, problems involving multi-step planning are more naturally framed in the language of reinforcement learning [93]. Such are the problems of autonomous assembly where a number of building elements need to be put together in a desired sequence or arrangement. An overview of prior research on robot-based autonomous construction is provided in [94]. In the present paper, we are framing the

Figure 4.3.: Luminale installation 2018. A centrally placed UR 10 robot constantly repositions modules. This reassembly in combination with changing light color alters form and meaning of the installation. (Image credit: Stefan Winkler)

problem of part placement as a reinforcement learning problem and thereby enabling the use of these novel computational tools in the digital design and fabrication context.

The problem of building a structure from a set of given blocks by an autonomous robot was considered in [95], where the challenge was approached by applying task-and-motion planning to calculate a path in the configuration space of the robots. However, the authors did not consider the low-level control problem and instead assumed that the robot is able to follow the plan perfectly without any deviations and that the parts can be grasped without any slippage. Optimization of the assembly sequence for curve-shaped designs based on reinforcement learning was proposed in [96], but no corresponding robot controller was developed. In [97], a hierarchical control framework was proposed that learns to sequence the building blocks to construct arbitrary 3D designs and ensures their feasibility with the robot-in-the-loop execution. A multiagent approach to the assembly problem was developed in [98], where a swarm of autonomous robots was trained with reinforcement learning to achieve a common goal.

Utilizing tactile information for robotic manipulation tasks is an area of active research [99]–[101]. Since the information from tactile sensors is typically high-dimensional and hard to interpret, learning is commonly employed to develop controllers that can use such rich feedback signals. In [85], a neural network model was trained to predict the stability of a planned grasp sensed with the GelSight tactile sensor [102]. In [103], reinforcement learning was used to stabilize an object with a robot arm based on tactile feedback from a BioTac tactile sensor [104]. In [83], reinforcement learning was used to move and rotate small objects such as marbles and dice using robotic fingers equipped with a modified GelSight sensor.

In this paper, we employ reinforcement learning to achieve part placement within a robotic architectural assembly. The advantage of the learning approach consists in the system's ability to adapt to changes in the environment, such as part positions or desired configurations. Contact between parts, friction, and deviations between planned and actual part locations require on-the-fly adaptation, which is a core strength of learning-based solutions. Furthermore, instead of specifying a path for the end-effector or providing similar low-level commands to the robot, the designer may specify a high-level objective for the robot and let the learning system figure out the optimal control sequence for the task. Finally, the use of rich high-dimensional sensors, such as depth cameras, Lidar, vision-based tactile sensors, etc. calls for scalable methods that can work with such inputs. Deep reinforcement learning is a promising approach to enable the use of such rich feedback signals for control and therefore we explore its potential in architectural assembly.

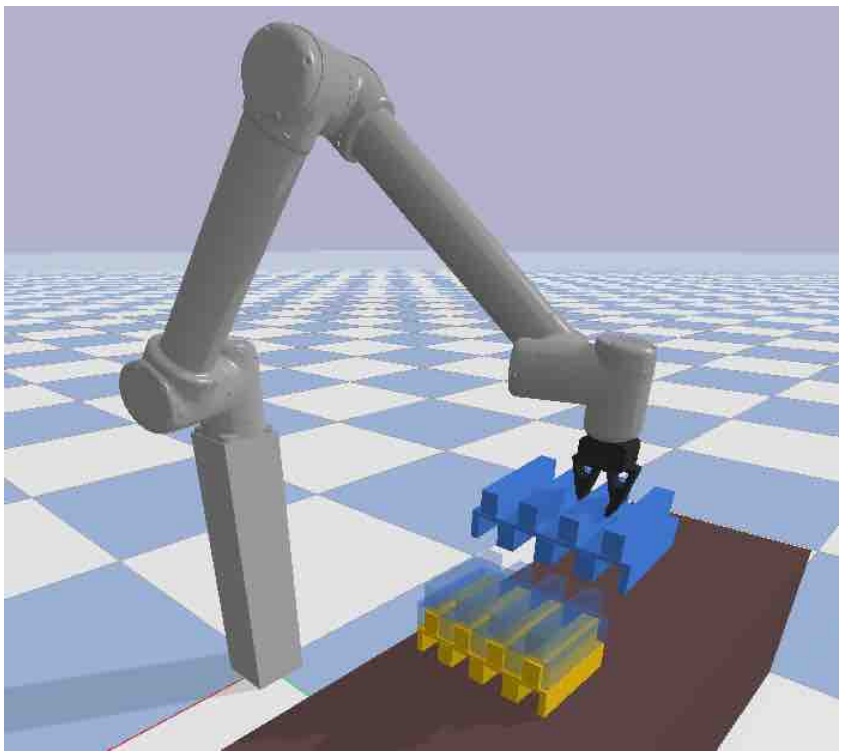


Figure 4.4.: Robot is learning to connect modules in our simulated environment (PyBullet physics engine). The robot controller is trained on 10000 randomly generated structures.

4.3. Reinforcement Learning for Contact-Rich Modular Assembly

A crucial challenge in autonomous robotic assembly is to deal with the contact between building parts. As long as the robot does not need to bring parts in contact with each other during manipulation, classical methods such as path planning and open-loop kinematic control can be applied. For example, brick laying robots can be made sufficiently precise without invoking sophisticated sensing modalities [105]. However, if parts have irregular shape or if they need to be assembled in a non-trivial manner, then more advanced robot control strategies are required. Currently, this problem is not completely solved and is a part of research in robotics on contact-rich manipulation [83], [84], [100].

Our goal is to accomplish autonomous modular assembly using a robot arm equipped with a gripper (Fig. 4.4). As the first step towards this goal, we developed a simulation

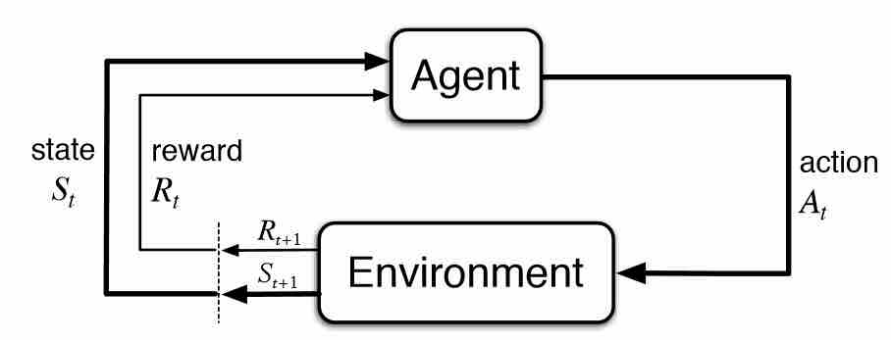


Figure 4.5.: Reinforcement learning loop. Agent performs action A_t, Environment responds with state S_t and reward R_t.

environment and let the robot learn how to place parts in that simulated environment. Initially, we explored the strategies that do not require tactile feedback and only react based on sensing the part positions and orientation. We found that a learned controller is capable of placing parts when the total tolerance is allowed to be greater or equal to 8mm. For precise manipulation, this is still too big.

Therefore, we equipped our gripper with a vision-based tactile sensor DIGIT [106] to enable more fine-grained manipulation, as exemplified in [83]. To train a reinforcement learning agent, we developed a simulation of the tactile sensor based on the model of light propagation [107]. However, this model does not include soft-body simulation and therefore we found it not suitable for contact-rich manipulation tasks that involve deformations of the gel on the sensor, as is the case in our setting. Therefore, we finish our presentation with a real-robot learning setup, where the learning is performed on the real robot, without employing a simulation environment. We give an outlook on how this method can be applied to architectural assembly tasks.

4.3.1. Reinforcement Learning Problem Setting

We formulate the problem of part stacking depicted in Fig. 4.4 as a reinforcement learning problem. We aim do develop a controller that is robust with respect to deviations in positions of the parts and can react to contact when parts are being joined together.

Reinforcement learning [93] is a general framework for developing controllers for environments which are hard to model analytically. In our scenario, it is very hard to come up with an analytical model for a controller that would react to various contact interactions between parts. Therefore, reinforcement learning is a more promising approach. Our

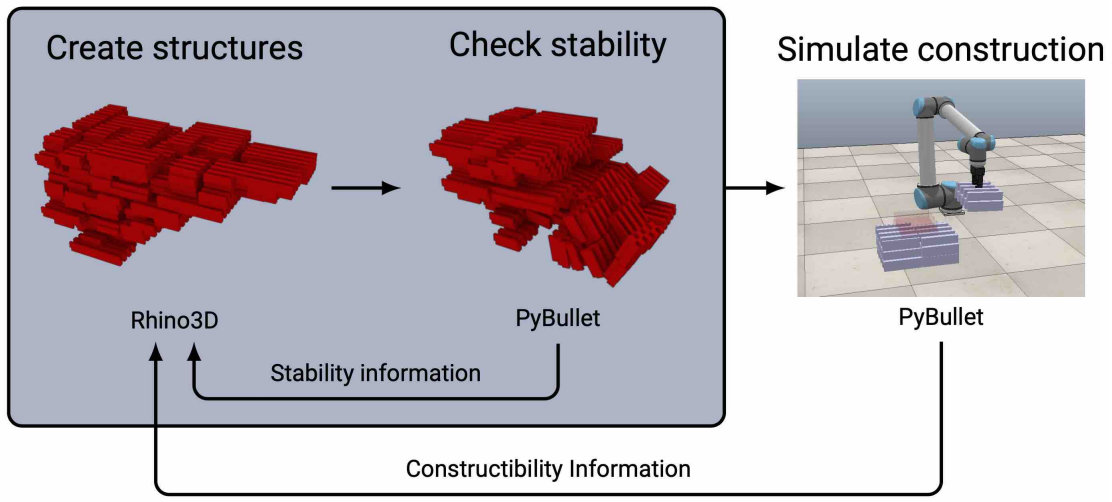


Figure 4.6.: Integration of Rhino3D with PyBullet physics engine for evaluating the stability of assembled structures within the reinforcement learning loop.

problem can be seen as a more difficult version of a classic in the control literature peg-in-a-hole task [108]. A similar setting in the context of architecture was recently considered within the robotic assembly of timber joints [109]. The difference in our case is that the parts have multiple slots that need to be aligned, which makes the problem more difficult and calls for the use of the tactile sensing technology.

To describe our problem in the language of reinforcement learning, we need to specify states, actions, and the reward function. A generic RL loop is shown in Fig. 4.5. The agent is a program that sends commands to the actuators of the robot (Cartesian joint velocity of the end-effector and gripper closure commands in our case) and receives back the state information (part positions and orientations, robot state, tactile sensor readings, etc.) together with a reward signal that evaluates how well the task is solved (e.g., position error between the desired and current part poses). The agent keeps interacting with the environment and adapting itself till it finds a successful policy $\pi\colon \mathcal{S} \to \mathcal{P}(\mathcal{A})$, a mapping from states to distributions over actions, that yields the highest expected return.

4.3.2. Rhino 6 Integration

We aim to develop a general learning-based framework for autonomous assembly of architectural modular structures, as shown in Fig. 4.1. The goal of the framework is to

allow an architect to specify a high-level design objective in Rhino/Grasshopper [110], which is then achieved by an intelligent agent. The agent first turns the design objective into a modular assembly plan, and then it turns the assembly plan into a robot controller that is able to build the structure.

The problem of finding a sequence of assembly steps that satisfies a given objective can also be framed as a reinforcement learning problem [111]. Here we assume that an assembly plan is already given and our task is to design a controller that can execute that assembly plan in a robust manner. However, although assembly in 2D has been successfully tackled in [111], extension to 3D and more complex blocks is non-trivial and is a part of our research programme.

In order to accommodate for reinforcement learning in the context of architecture, we developed several bridges between Grasshopper and PyBullet [112]. Figure 4.6 demonstrates the high-level functionality provided by the bridges. Information can flow in both direction, during the design phase (part geometry and other properties are exported from Grasshopper and imported in PyBullet; results of physical simulation are communicated back from PyBullet to Grasshopper) as well as during the construction phase (current state of construction communicated from PyBullet to Grasshopper, allowing on-the-fly changes to the design). Figure 4.7 shows how the developed PyBullet interface appears in Grasshopper.

We use PyBullet because it is a fast physics engine widely used in robotics. It not only enables learning of the block stacking controller but can also be used to check the physical properties of a structure designed in Rhino, such as stability, structural performance, and tectonics during and after assembly. The communication is implemented using TCP/IP sockets.

The integration between Rhino and PyBullet enables physical simulation of a modular structure in PyBullet and subsequent visualization of the resulting poses of all parts in Rhino. The results of this simulation can be used to compute a cost function and pass it to an optimization algorithm, such as an evolutionary algorithm or reinforcement learning. Thus, the entire simulation ecosystem of grasshopper can be used to derive cost measures for reinforcement learning from the wide range of architectural requirements such as structure, material behavior, daylight, temperature, context, etc.

4.3.3. Learning Algorithm Description

There are multiple algorithms available in the RL toolbox [113] for solving the resulting optimization problem. Depending on the problem properties—such as discrete or continuous state and action space, sparse or dense reward function, cheap or expensive simulation, etc.—one can choose a different algorithm. We found Twin Delayed DDPG (TD3) [114]

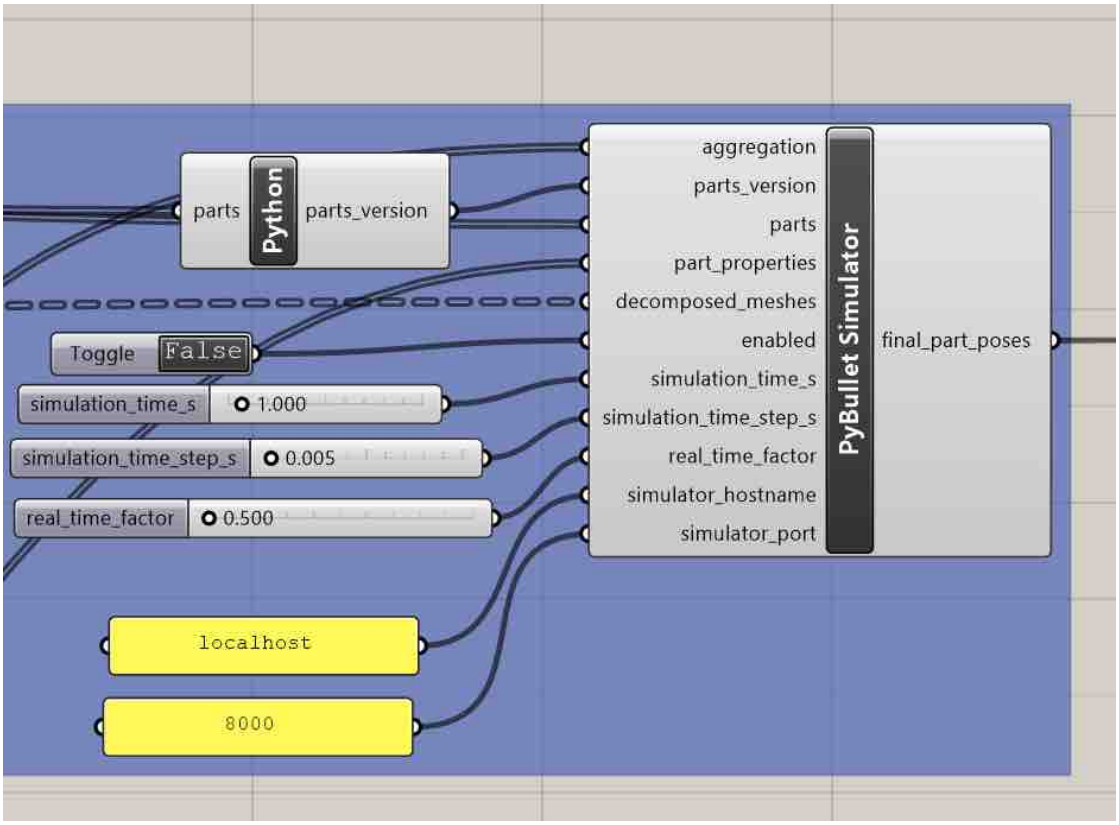


Figure 4.7.: PyBullet interface as seen in Grasshopper. Data about the scene and parts can be exported from Rhino and imported in PyBullet. The result of the simulation in PyBullet subsequently becomes available in Grasshopper.

to be most suitable for our task. TD3 is a more stable version of the well-known Deep Deterministic Policy Gradients (DDPG) algorithm [115], that extends Q-learning to continuous action spaces. DDPG for the first time showed that a Q-learning-like algorithm can be applied to solve continuous control problems, such as robot control in simulated environments.

TD3 is a model-free off-policy RL algorithm. Model-free means that the model of the environment is not known to the agent, i.e., the agent does not know a priori how its actions affect the observed states and rewards. Such ignorance may be surprisingly useful, as demonstrated by the great success of model-free RL in tasks ranging from robotic in-hand manipulation [116] to playing computer games such as StarCraft II [117]. Off-policy means that the algorithm can utilize experiences collected by another policy. In particular, past experiences can be kept in memory and reused. This allows off-policy algorithms to be more sample-efficient, requiring fewer interactions with the environment to solve the task [10].

TD3 learns two Q-functions Q_{θ_1} and Q_{θ_2} and a policy π_ψ. Both Q-functions and the policy are parameterized by neural networks with parameters $\{\theta_1, \theta_2, \psi\}$, respectively. To make learning more stable, TD3 uses separate target Q-functions $Q_{\theta'_1}$ and $Q_{\theta'_2}$ and a target policy $\pi_{\psi'}$ to update the current Q-functions Q_{θ_1} and Q_{θ_2}. The parameters of the target networks are updated to slowly track the parameters of the current networks:

$$\theta'_i \leftarrow \tau\theta_i + (1-\tau)\theta'_i,$$
$$\psi' \leftarrow \tau\psi + (1-\tau)\psi',$$

with $\tau \in (0, 1)$ being a hyperparameter. Furthermore, TD3 maintains a replay buffer in which it stores previously seen transitions as tuples (s, a, r, s'). At each update step, TD3 samples a random minibatch of N transitions (s_j, a_j, r_j, s'_j) from the replay buffer, and the Q-functions Q_{θ_1} and Q_{θ_2} are adjusted to minimize the Bellman error on this minibatch. To counter the overestimation bias present in DDPG, both target Q-functions $Q_{\theta'_1}$ and $Q_{\theta'_2}$ are evaluated and the smaller of the two values is used as the target. This results in the loss function

$$L(\theta_i) = \frac{1}{N}\sum_j (y_j - Q_{\theta_i}(s_j, a_j))^2,$$
$$y_j = r_j + \gamma \min_{i=1,2} Q_{\theta'_i}(s'_j, \tilde{a}_j),$$

where $\tilde{a}_j$ denotes the action chosen by the target policy $\pi_{\psi'}$ supplemented by Gaussian

noise,

$$\tilde{a}_j = \pi_{\psi'}(s'_j) + \epsilon,$$
$$\epsilon \sim \text{clip}(\mathcal{N}(0, \sigma), -\epsilon_{\text{max}}, \epsilon_{\text{max}}).$$

The noise is added to robustify the Q-function estimate by enforcing it to be smoother and return similar values for similar actions. The policy π_ψ is trained to maximize the agent's performance given by the expected discounted return

$$J(\psi) = \mathbb{E}\left[\sum_{t=1}^{T} \gamma^t r(s_t, \pi_\psi(s_t))\right].$$

This optimization is performed using stochastic gradient descent (SGD), plugging in the estimates of the Q-functions described above into the policy gradient expression

$$\nabla_\psi J(\psi) \approx \frac{1}{N} \sum_j \nabla_a Q_{\theta_1}(s, a)|_{s=s_j, a=\pi_\psi(s_j)} \nabla_\psi \pi_\psi(s)|_{s=s_j}.$$

For more details on the theoretical underpinnings of TD3 and its implementation, see papers [113], [114].

4.3.4. Reinforcement Learning for Part Stacking

In this section, we apply the TD3 algorithm described in Sec. 4.3.3 to solve the stacking problem described in Sec. 4.3.1. We describe the experimental setting and present the results.

Training proceeds in episodes. At the beginning of each episode, the starting block is placed at random in a square with 60cm side length. The robot then has to place a second block on top of the starting block in a given stable configuration. Examples of these stable configurations are shown in Fig. 4.8. While placing the second block, the robot has to ensure that the first block is not displaced.

The spawning location of the second block is sampled uniformly in a cube of side length 2cm that is located 30cm above the center of the starting block's spawning area. The spawning orientation is obtained by slightly perturbing the goal orientation. This perturbation is achieved by sampling Euler angles between -5 and 5 degrees and rotating the block accordingly.

Each episode, the robot starts with the block in its gripper. To avoid overfitting with respect to the exact grasp location, the position of the grasp on the block is also randomized along the ridges of the block and along the block's up-axis. Fig. 4.4 displays a possible

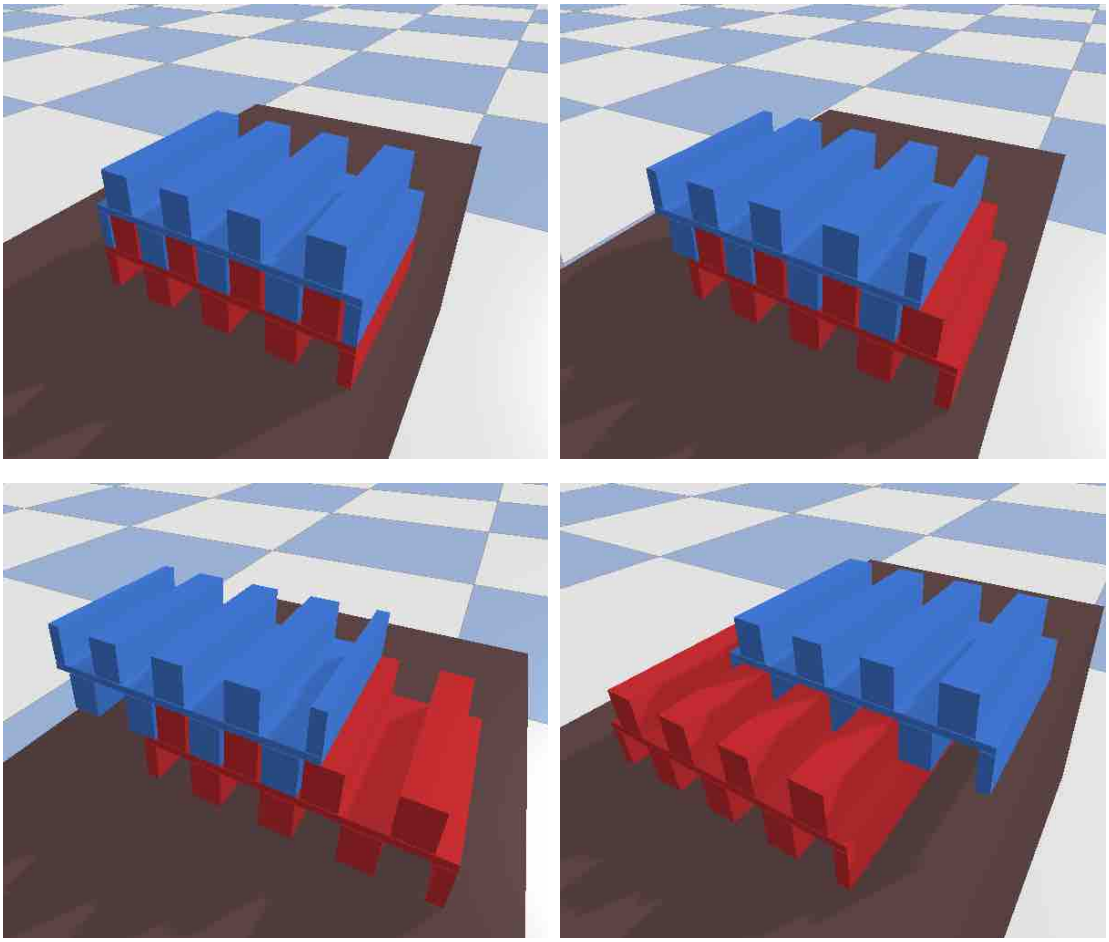


Figure 4.8.: Example goal configurations of the stacked blocks. The agent is trained on thousands of random goal configurations where the blue module is connected to the red module. By learning to solve all of these configurations, the agent develops a robust strategy.

starting configuration of the training environment. All these randomizations are included to make the controller more robust with respect to deviations in the positioning of the block and gripper that might occur on the real robot due to inaccuracies in the localization of the blocks and controller deviations.

An episode terminates if the gripper moves more than 10cm away from the current block or if the time limit of 3 seconds is reached. With the controller frequency of 20Hz, this yields a maximum sequence length of 60 steps.

Optimization Objective

The objective of the learning agent is to learn a policy $\pi_\theta\left(\boldsymbol{a} \mid \boldsymbol{o}\right)$ that minimizes the expected cost of an episode $\tau = (\boldsymbol{s}_1, \boldsymbol{a}_1, \ldots, \boldsymbol{a}_{T-1}, \boldsymbol{s}_T)$:

$$\min_\theta \ \mathbb{E}_{\tau \sim p(\tau \mid \pi_\theta)} \left[\phi_f\left(\boldsymbol{s}_T\right) + \frac{1}{T} \sum_{t=1}^{T} \phi_i\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) \right]$$

where $\boldsymbol{s}_t$ is the full system state at time step t, $\boldsymbol{o}_t$ is the observation the policy gets of $\boldsymbol{s}_t$, $\boldsymbol{a}_t$ is the chosen action, $\phi_f\left(\boldsymbol{s}_T\right)$ is the final cost, $\phi_i\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right)$ is the intermediate cost at step t, and

$$p\left(\tau \mid \pi_\theta\right) = p\left(\boldsymbol{s}_1\right) \prod_{t=1}^{T-1} p\left(\boldsymbol{s}_{t+1} \mid \boldsymbol{a}_t, \boldsymbol{s}_t\right) \pi_\theta\left(\boldsymbol{a}_t \mid \boldsymbol{o}_t\right) p\left(\boldsymbol{o}_t \mid \boldsymbol{s}_t\right)$$

is the trajectory distribution induced by policy π_θ.

The action vector $\boldsymbol{a} = (\boldsymbol{a}_{\text{arm}}, a_{\text{gripper}})$ is composed of two components: the target joint velocities $\boldsymbol{a}_{\text{arm}} \in \mathbb{R}^6$ of the 6 arm joints and the target closure state $a_{\text{gripper}} \in [0, 1]$ of the gripper. Concerning the gripper control signal, a 1 indicates that the gripper shall be fully closed, while a 0 indicates that the gripper shall be fully opened. The signal is realized by a position controller on the gripper motor.

Our main objective is that both blocks of the structure are placed in their respective target pose. Hence, we chose the final cost function as follows:

$$\phi_f\left(\boldsymbol{s}_T\right) = \alpha_{\text{pose}} \phi_{\text{pose}}\left(\boldsymbol{s}_T\right) \tag{4.1}$$

where $\alpha_{\text{pose}} \in \mathbb{R}$ is a weighting factor. ϕ_{pose} is a penalty for the pose error of the blocks. To ensure that the robot does not displace the block that is already placed at the beginning of the episode, this penalty includes the pose errors of both blocks. Note that “pose” here refers to a combination of position and orientation. It is challenging to define a distance metric on poses directly since it is not straightforward to weight position and orientation error up in a useful manner. Thus, we chose a different approach to measuring pose error.

To measure the error of the pose, we compute the average positional error of the eight vertices of the bounding box of each block:

$$d_k^2 = \frac{1}{8} \sum_{j=1}^{8} \|\boldsymbol{m}_{k,j} - \tilde{\boldsymbol{m}}_{k,j}\|_2^2$$

where $\boldsymbol{m}_{k,j}$ is the target position of vertex j of block k at the final time step and $\tilde{\boldsymbol{m}}_{k,j}$ is the actual position, respectively.

While it would be possible to use the mean of the distances d_k^2 directly as a cost function, this approach comes at a major drawback: the closer a block is to its target pose, the flatter the squared penalty becomes and the less incentive the learner has to place the block even more precisely. In construction tasks, even tiny deviations from the correct position can have a huge impact on whether a structure can be assembled properly. We therefore use a modified cost function

$$\delta_{\log}\left(d^2\right) = d^2 + 0.01\left(\ln\left(d^2 + \epsilon\right) - \ln\left(\epsilon\right)\right)$$

with a small value of $\epsilon = 10^{-5}$. Due to the function's concave shape close to 0, even small positioning errors are punished significantly.

The final pose cost is then computed as

$$\phi_{\text{pose}}\left(\boldsymbol{s}_T\right) = \min\left\{\frac{\frac{1}{B}\sum_{k=1}^{B}\delta_{\log}\left(d_k^2\right)}{\phi_{\text{pose}}^{\max}}, 1\right\}$$

where $B = 2$ is the number of blocks in the scene and $\phi_{\text{pose}}^{\max}$ is a hyperparameter that controls the clipping of this term. The intuition behind the clipping is that if the block is placed too far away from the target location, we consider the task as failed and simply assign a maximum cost. Hence, the reward becomes more sparse and the Q-function in regions far away from the target state (e.g., if the robot threw the block away from itself) becomes easier to learn. The inverse scaling by $\phi_{\text{pose}}^{\max}$ simply ensures that $\phi_{\text{pose}}\left(\boldsymbol{s}_T\right) \in [0, 1]$, which makes it easier to choose the weights α later since all cost terms will share the same domain.

The intermediate cost $\phi_i\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right)$ consists of a constant term, a penalty for movements of the current block while it is not grasped, and a penalty on the acceleration of the end-effector:

$$\phi_i\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) = \alpha_{\text{time}} + \alpha_{\text{rel}}\phi_{\text{rel}}(\boldsymbol{s}_t) + \alpha_{\text{acc}}\phi_{\text{acc}}\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) \tag{4.2}$$

The first term encourages the learner to complete the episode quickly. ϕ_{rel} is a penalty for the movement of released blocks, i.e., blocks that are not grasped, to discourage the robot from throwing the blocks at the target.

As long as the block is in the gripper, the agent should be able to move it around freely, without additional cost. The cost ϕ_{rel}, thus, is activated only if the block is released from the gripper. Whether the block is grasped is determined by checking if the distance from the block to the gripper's fingers d_{fingers} is larger than $1cm$.

$$\phi_{\text{rel}}\left(\boldsymbol{s}_t\right) = \begin{cases} \phi_{\text{vel}}\left(\boldsymbol{s}_t\right) & \text{if } d_{\text{fingers}} > 0.01m \\ 0 & \text{otherwise} \end{cases}$$

The cost for the movement of a released block is realized as a quadratic penalty on the block's linear and angular velocity.

$$\phi_{\text{vel}}\left(\boldsymbol{s}_t\right) = \min\left\{ \frac{\left\|\boldsymbol{v}_{\text{lin}}\right\|_2^2 + \left\|\boldsymbol{v}_{\text{ang}}\right\|_2^2}{\phi_{\text{vel}}^{\text{max}}}, 1 \right\}$$

Here, $\boldsymbol{v}_{\text{lin}}$ is the linear velocity of the block's center, $\boldsymbol{v}_{\text{ang}}$ the angular velocity, respectively, and $\phi_{\text{vel}}^{\text{max}}$ is again a hyperparameter.

The penalty on the accelerations of the end-effector is realized as a quadratic cost on the linear and angular velocity differences between the current and the last time step.

$$\begin{aligned} \phi_{\text{acc}}\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) = \frac{1}{\phi_{\text{acc}}^{\text{max}}} \Big(& \left\| \left(\boldsymbol{v}_{\text{lin}}^{\text{ee}}\right)_t - \left(\boldsymbol{v}_{\text{lin}}^{\text{ee}}\right)_{t-1} \right\|_2^2 \\ & + \left\| \left(\boldsymbol{v}_{\text{ang}}^{\text{ee}}\right)_t - \left(\boldsymbol{v}_{\text{ang}}^{\text{ee}}\right)_{t-1} \right\|_2^2 \Big) \end{aligned} \tag{4.3}$$

where $\left(\boldsymbol{v}_{\text{lin}}^{\text{ee}}\right)_t$ and $\left(\boldsymbol{v}_{\text{ang}}^{\text{ee}}\right)_t$ are the linear and angular velocities of the end-effector at time t, respectively. The purpose of this term is to encourage smooth movements in order to reduce energy consumption and decrease the strain on the robot's motors and gears.

Results

The controller was trained on $10{,}000$ randomly generated structures in simulation. Every $100{,}000$ steps, the controller is evaluated on $1{,}000$ unseen test structures. The learning rate starts at $10^{-4.5}$ and is decayed over time, multiplied by $1/\sqrt{10}$ after 40%, 60%, 80%, 90%, and 95% of the total training time of 48 hours. Fig. 4.9 shows the evolution of the average cost (negative return) and the position error in block placement over time. The controller reaches a block center position error of around 8mm after 12 million steps or 48 hours of training. The same accuracy is actually achieved already after 4 million steps (16 hours), therefore the training could be stopped earlier without a significant drop in task performance. While the accuracy of 8mm may be satisfactory for certain types of

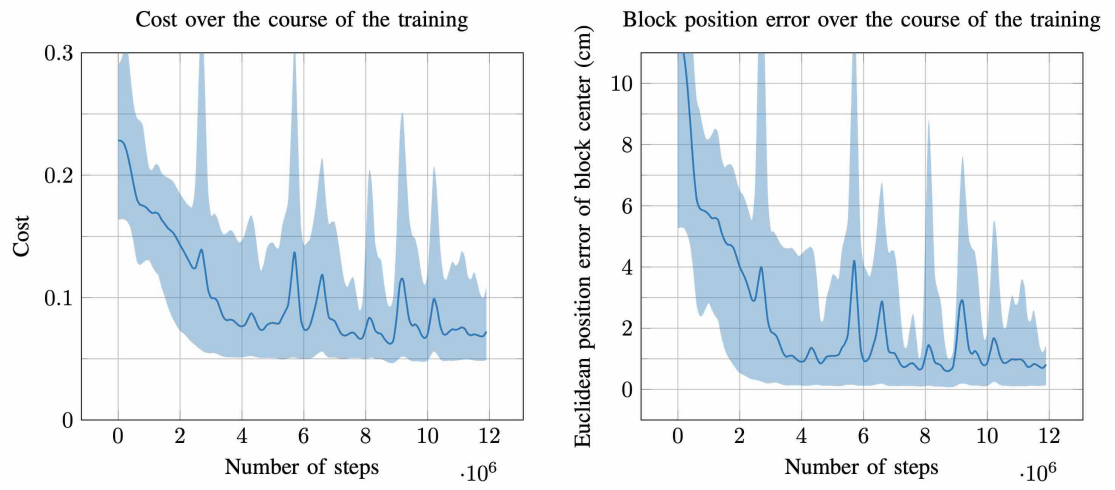


Figure 4.9.: Average cost (left) and block position error (right) over the course of TD3 training in simulation. The x-axis shows the number of agent's interactions with the simulated environment running at 20Hz. The plots show the mean and 5–95% percentiles obtained by evaluating the model on 1000 unseen test configurations every 100000 steps of training.

assemblies when part tolerances are sufficiently large, we generally would like to develop more precise controllers. Therefore, the next section introduces tactile feedback that allows for finer control.

4.3.5. Simulation of the Tactile Sensor DIGIT

DIGIT [106] is a high-dimensional vision-based tactile sensor. Its design is based on GelSight [102] but DIGIT is significantly cheaper and easier to produce. It consists of an opaquely coated layer of gel illuminated from inside the sensor by three RGB-colored LEDs. These LEDs are placed inside the casing so that they illuminate the gel from different directions. The reflections of the LEDs' light from the gel is captured by an RGB camera. If the sensor touches an object, the gel deforms according to the geometry of the object. This deformation changes how the light of the LEDs reflects from the gel and thus the geometry of the touching object can be seen in the image captured by the camera. See Fig. 4.11 and 4.10.

In order to enable reinforcement learning in simulation, the DIGIT sensor needs to be integrated into PyBullet. We considered two approaches. One is based on simulating the light propagation and reflection [107], [118]. While appealing for small objects such as the little sphere in Fig. 4.10 or dice in [83], this method does not work well for objects that cover the whole surface of the sensor, as is the case with our blocks. When large objects shift inside the gripper, this type of simulation is not able to capture any signal, whereas the real sensor does produce a signal because the silicone gel at the fingertip deforms. The other simulation approach is based on soft-body physics simulation. Modern physics engines do not yet support reliable and realistic soft-body simulation as required for robot control [119]. We were able to tune the soft-body simulation in PyBullet to resemble the real sensor, but this required very small time step, making the approach impractical for use with RL.

4.3.6. Learning on the Real Robot

Due to the aforementioned difficulties with the simulation of soft vision-based tactile sensors, we are investigating learning on the real robot, bypassing the simulation altogether. Sample sensor readings are shown in Fig. 4.11. Compared to the simulated signal in Fig. 4.10, there is much more nuance in the real data. Therefore, similar to the approaches [83] and [106], we aim to utilize these real signals directly for control by learning latent dynamics models.

Learning on the real system is different from learning in simulation. The agent can collect only a limited number of experiences and the robot can break easily if drastic actions

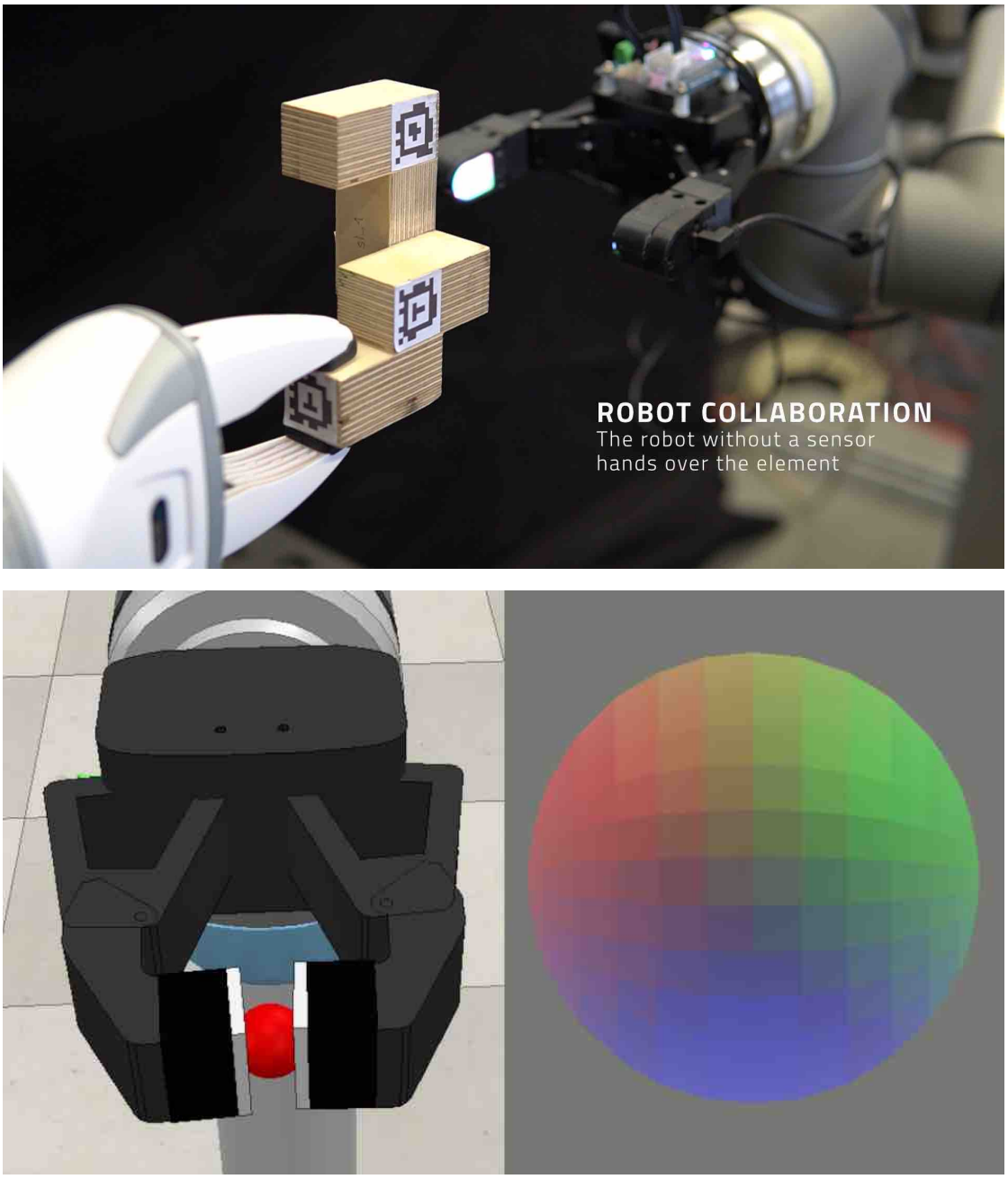


Figure 4.10.: Tactile sensor DIGIT. Top: Real gripper approaching a part. Bottom: Simulation of the gripper and RGB illumination providing detailed tactile feedback.

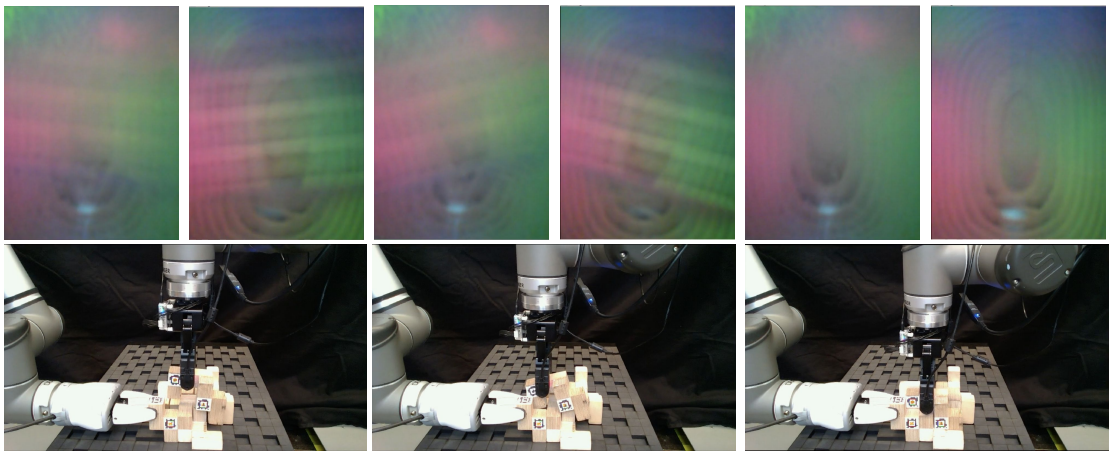


Figure 4.11.: Tactile sensor readings during learning on the real system. Compare the patterns of gel deformation on the left and on the right tactile sensors to the case where no contact is present (rightmost figure).

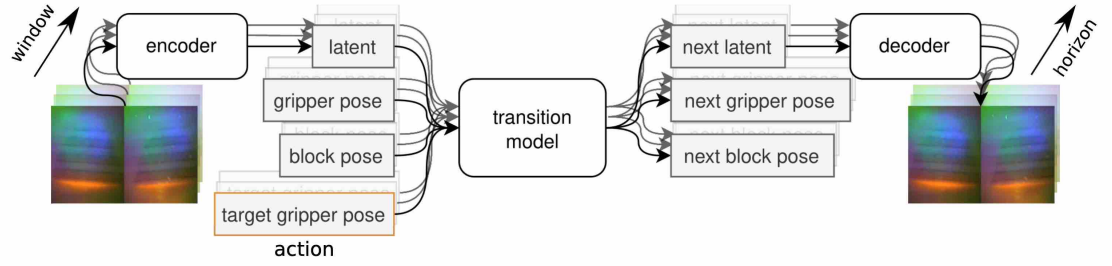


Figure 4.12.: Architecture of the model learned from the real-world data collected by the robot. As input the model receives a window of past observations, shown in the figure by the shadow going into the depth of the diagram. Each observation is multi-modal and consists of three parts: i) two tactile image (left and right gripper fingers), ii) gripper pose, iii) poses of the manipulated blocks. Raw sensor observations pass through a learnable encoder before passing into the transition model. Given a window of such observations together with a window of actions, the transition model returns a window of future observation, which is called horizon in this case.

are taken. Therefore, we pursue a model-based approach, where the agent is learning a transition model using experiences collected on the real system in a guided fashion. This is similar to how autonomous driving cars may learn to drive from human driving data. Instead of starting from scratch, we bootstrap the agent with a good initialization. The guiding procedure in our case consists in providing a few initial demonstrations (on the order of 10 trajectories) which are subsequently used by the agent to collect a larger dataset via replaying those trajectories with added perturbations. With this procedure, we build a dataset of 30000 time steps recorded at 15Hz. Each sample consists of the state of the robot and the state of the environment, as explained in Fig. 4.12. The encoder and decoder of the tactile images are trained jointly as a static autoencoder. Subsequently, the transition model is trained in a self-supervised manner.

Figure 4.13 shows the learning curves for the transition model training. A window of 7 observations was fed as input to the transition model, and a horizon of 3 time steps was used for prediction. An autoencoder with ReLU activations and 16-dimensional latent space was trained on batches of size 16 with learning rate 0.0002. Although the model is able to predict the robot state and the state of the objects rather well, as seen from the position losses in Fig. 4.13, the error in the prediction of the tactile images is high, both in the latent space and after reconstruction. This may be attributed to the agent not needing to reconstruct the whole image in order to extract task-relevant information from it.

Once the model is learned, we run model predictive control using the cross-entropy method, similar to [83], [106]. But performance varies significantly. Out of 30 trial executions on the real system, 19 finished with a successful joining of the SL-blocks, which requires the precision on the level of 1mm. The unsuccessful executions terminated prematurely due to the blocks getting jammed. The reward function is given by the position and orientation error, but it does not include any punishment on hitting other blocks. As a result, the execution may deviate from the planned trajectory when multiple points of contact are encountered simultaneously, causing the blocks to get stuck. We view these experiments as encouraging, since they shows that the proposed method is feasible and can be applied on the real robot, but further investigation is required to improve the performance and reduce the variance in the results.

4.4. Conclusion and Future Work

In this paper, we addressed the problem of autonomous construction of modular structures. The current stage focuses on the simulation and training of basic assembly skills. At a later stage, this will be complemented by combining tactile skills and physical manipulation. Reinforcement learning was shown to be effective at learning closed-loop control policies

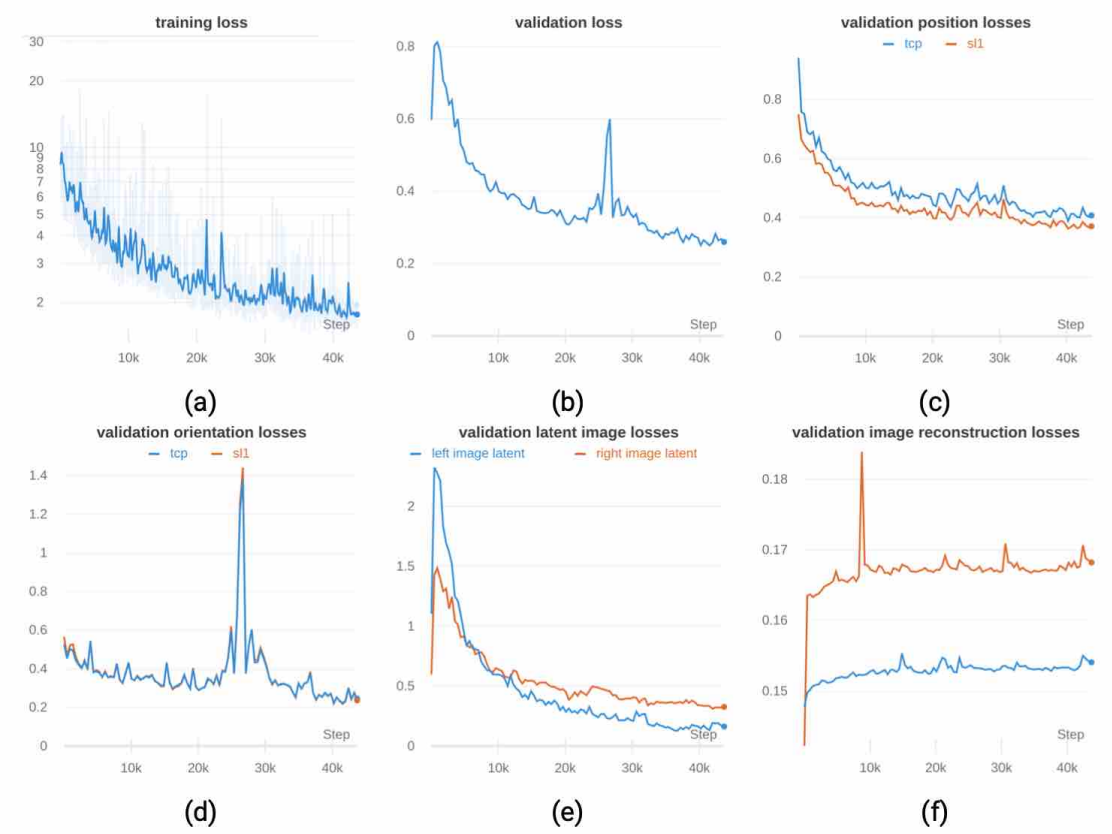


Figure 4.13.: Losses during the training of the transition model. All losses converge except the image reconstruction loss. The additional information of the other sensors could not be utilized to forward predict images well despite all other predictions matching the ground truth rather precisely.

to enable part stacking. However, the training time may become a bottleneck if more parts are added into the scene.

In order to construct arbitrary structures, a controller is needed that is able to deal with structures consisting of a variable number of blocks. Since every structure is different, the robot needs a way of sensing where other blocks of the structure are in order to place the next block without collisions. Here, the major challenge is to process an arbitrary number of block poses and achieve invariance to permutations. Both of these challenges could be tackled with Graph Convolutional Networks [120]. Alternatively, these challenges could be solved by using images of an RGB or depth camera as input to the policy instead. However, using camera images introduces a series of new challenges, as some blocks might be occluded and the robot additionally has to learn to interpret complex visual signals.

This project's long-term goal is to be able to construct structures using a real robot. Nevertheless, there is a number of challenges that need to be tackled when transferring our method to the real world. First of all, simulators can be fairly inaccurate when contact is happening, which makes transferring the policy to the real system challenging as it first has to be fine-tuned on the new system dynamics. Furthermore, some measurements that are easy to obtain in simulation are not directly available on the real system. One example is the poses of the parts that are used in the cost function and as a sensor input to the agent, which in reality require an optical tracking system to be estimated. Finally, the simulation of the DIGIT sensors is likely to be inaccurate, as the deformation of the gel is not simulated in detail. While we did some experiments with soft body simulation, we found the results to be unstable and inaccurate and hence decided to refrain from simulating deformation. Currently, the most promising approach with regards to learning based on tactile sensing is learning on the real system. However, sim-to-real transfer and other domain randomization techniques [121]–[125] may also prove useful in the future.

This paper presents first results of a research that intends to develop autonomous assembly through robots and reinforcement learning algorithms. The full integration of the tactile sensor and its feedback will allow the robot to distinguish different building elements based on surface quality, weight and form. The integration of PyBullet as robot simulation environment into Rhino allows to use architectural requirements as cost measures for reinforcement learning algorithms. Equipping robots with these capabilities provides a step towards enabling autonomously built structures made from parts of various materials and functions suitable for assembly and re-assembly, contributing to a circular economy in architecture.

5. Conclusion

In this thesis, we investigated the power of statistical decision theory in modeling the behavior of humans and deriving learning and control algorithms for robots. First, we introduced an entropic policy regularization scheme for policy iteration algorithms. Second, we showed that human ball catching behavior can be put in correspondence with optimal trajectories in a POMDP environment. Third, we evaluated non-linear dimensionality reduction for reinforcement learning of tactile-rich manipulation policies. Below one finds a more detailed summary of the contributions.

5.1. Summary of Contributions

Chapter 2 considered the problem of learning a policy through an interaction with an unknown Markov decision process. For the key issue of covariate shift which arises due to the statistical nature of the interaction with the environment, we proposed an f-divergence constrained policy update rule which punishes large deviations from the already explored areas of the state-actions space. The proposed entropic regularization framework was shown to provide a class of solutions which are parameterized by the choice of the divergence function and therefore provide different exploration-exploitation trade-offs.

Chapter 3 presented a model of human ball catching behavior and the corresponding evaluations. Through probabilistic modeling and accounting for the belief distributions, we were able to capture the essential features of human behavior in ball catching. Coupled with solid empirical evidence collected in large-scale experiments on a baseball field, this contributions provides a strong argument in support of the probabilistic modeling approach for describing human behavior, because it can reproduce earlier heuristics-based control strategies and account for catches that require interrupted information gathering which are not explained by the heuristics. We believe these results will serve as a basis for incorporating more of the statistical decision theory approaches into cognitive science modeling toolbox.

Chapter 4 investigated a real-world application of the modeling and control techniques

on a challenging problem of tactile-rich insertion which occurs in precise assembly tasks. Model-based reinforcement learning in combination with learned representations from high-dimensional vision-based tactile sensors was employed to devise an insertion policy that can adapt to displacements of the assembled parts. We demonstrated the feasibility of this method, and therefore established an avenue for further exploration of the use of such novel vision-based tactile sensors. Nevertheless, we found that the hardware requirements of the algorithms preclude its real time use at present due to high computational load. Furthermore, the accuracy of placement is dependent on the quality of the learned model and can vary depending on the dataset used for learning.

5.2. Critical Discussion of Contributions

The topics considered in this thesis are united by the desire to understand and utilize the principles of intelligent decision making as evidenced in human behavior. Therefore, one of the contributions of the thesis is the modeling of human trajectories in ball catching. However, one should note that no matter how detailed our models are, there are still many variables which are not taken into account. For example, different people may have different aptitudes towards ball games and different natural capabilities. Different levels of practice result in a qualitative difference in the catching behavior. Furthermore, physical stamina influences the repeatability of the experiments even with the same subject over time. Thus, one should be mindful of appreciating the approximate nature of modeling and recognize the value in capturing even a small set of relevant features of human behavior in a given modeling framework.

When it comes to information-theoretic bounds for policy learning, the flexibility allowed by f-divergences comes at a cost of not having a closed-form solution for some dual variables, instead relying on numerical optimization. In addition to that, a stable numerical implementation of the f-divergence is required, since the growth of the divergence function may lead to overflows in certain cases. Nevertheless, the avenue of considering other divergence functions appears promising, in particular with regards to utilizing Wasserstein distance which inherits the metric from the base space and thereby carries the geometry to the space of distributions.

The real-robot applications demonstrated the importance of respecting the real-time and hardware constraints posed by the real world. A number of components are still required for robots to achieve proficiency: better perception, better simulation environments, especially simulation of contact, efficient optimization and hierarchical action generation. In our experiments, we were able to tackle the problem of tactile-based assembly. This task involves sensing at the point of contact and agile adjustment of the robot control

commands to connect two parts together. The end-to-end approach considered in our work has advantages and drawbacks. The main advantage is the generality, which means that we can apply it to any other sensor or even other modality. However, the learned representations showed a restricted range of transferrability and generalizability. On the other hand, manual signal extraction pipeline that would rely on human-interpretable physically meaningful quantities being extracted, such as forces and pressure, could potentially transfer more easily to handling other objects and solving other tasks. Therefore, an important direction for future work is to consider and compare learning intermediate physically motivated representations vs. utilizing an end-to-end learning pipeline.

5.3. Outlook

In this section, we discuss open problems and directions for future work based on the investigations carried out within this thesis. We consider the avenues in cognitive sciences and robot learning.

5.3.1. Open Problems

Here we summarize the open problems resulting from our studies presented in Chapters 3, 2, 4. The problems refer both to technical implementation details and to more broad conceptual level questions.

Deep Learning for Entropy-Regularized Policy Improvement. Our algorithm for entropic regularization of policy improvement was derived in such a manner that linear-in-features representation of the value function resulted. Correspondingly, our evaluations employed random Fourier features and radial basis functions. However, an extension to learned deep representations would be of interest, as it would expand the range of applicability of the proposed algorithm. Although in principle such an extension is straightforward to conceive of, in practice a careful numerical implementation incorporating the latest discoveries and know-hows in neural network training needs to be worked out.

Rationality and Bounded Rationality. Our model of ball catching provides a rational explanation of the behavior. There is a POMDP model underneath and the agent finds an (approximately) optimal solution by some means. However, it appears that humans in addition need to take into account how long the computation takes and how much energy is available for it. For example, it is well known that exactly solving POMDPs is computationally prohibitive even for toy problems. Therefore, it is desirable to have a

framework that can incorporate such meta-reasoning about resource allocation. Meta-rationality is one approach, but it suffers from the infinite regress problem. Bounded rationality and active inference may potentially provide a satisfying account, but it remains to be seen if the resulting models will agree with the experimental data.

Learning Representations for Tactile Manipulation. The novel vision-based tactile sensors that were utilized in our work are quickly gaining popularity in robotics thanks to the easy availability of the cheap high-quality cameras and the advances in computer vision. Nevertheless, there are many open problems with regards to how the signals obtained from such sensors should be processed and usefully incorporated into the decision making process. On one hand, one could reach for a completely unsupervised approach and train an autoencoder-like architecture. Although this has been shown to work both in our work and in other papers, the generalization capabilities of this approach are limited by the variability of the data provided in the training set. On the other hand, employing first principles, one could model the deformation of the gel and propagation of light, and thereby obtain a more faithful physical representation of the sensor behavior in simulation. Despite usefulness for analysis, it is nevertheless not clear how helpful such an approach would be in practice, because the real environment tends to differ from simulation even on much simpler mechanical systems, and contact with soft objects involves much more effects that need to be accounted for. Therefore, a controller trained in simulation may not transfer directly to reality. This calls for research on domain adaptation and transfer learning with tactile sensors.

5.3.2. Future Work

As alluded to in the previous subsection, we need innovations both in the conceptual frameworks for modeling the perception-action loop and in the technical implementations of the resulting algorithms.

The framework of entropic regularization has been shown to enable modeling risk-sensitive behavior in policy search [126] and to account for self-paced curriculum reinforcement learning [127], [128]. Furthermore, it was demonstrated to be effective for promoting robustness against model uncertainties [129] and induce exploratory behavior [130], [131]. A related theory that aims to unify a number of decision making frameworks, including the entropic regularization, is *active inference* [2]. Although theoretically appealing, practical implementations of active inference that can compete with more specialized learning algorithms have been lacking [132]. One of the first demonstrations of active inference on real robots was presented in [133], [134]. An important avenue

for future work is to develop a complete practical framework for active inference and investigate efficient numerical implementations of the resulting algorithms.

A key issue arising in developing learning algorithms that operate on high-dimensional input data is the choice of the neural network architecture. For learning in MDPs, specialized architectures have been proposed, including deep differentiable value functions [135], [136] and deep Lagrangian networks [137], [138], among others. Recently, interest in more general architectures has intensified, fueled by the success of the Transformer architecture [139] in multi-modal learning for speech and vision. One of the first applications of Transformers to decision making is the *Decision Transformer* architecture [140] which frames offline reinforcement learning as a sequence modeling problem. A further investigation into such general models presents a promising direction for future work.

A. Supplementary Material

A.1. Technical Background on f-Divergence

This section provides the background on the f-divergence, the α-divergence, and the convex conjugate function, highlighting the key properties required for our derivations.

The f-divergence [33], [141], [142] generalizes many similarity measures between probability distributions [39]. For two distributions π and q on a finite set $\mathcal{A}$, the f-divergence is defined as

$$D_f(\pi\|q) = \sum_{a\in\mathcal{A}} q(a) f\left(\frac{\pi(a)}{q(a)}\right),$$

where f is a convex function on $(0,\infty)$ such that $f(1)=0$. For example, the KL divergence corresponds to $f_{KL}(x) = x\log x$. Please note that π must be absolutely continuous with respect to q to avoid division by zero, i.e., $q(a)=0$ implies $\pi(a)=0$ for all $a\in\mathcal{A}$. We additionally assume f to be continuously differentiable, which includes all cases of interest for us. The f-divergence can be generalized to *unnormalized distributions*. For example, the generalized KL divergence [34] corresponds to $f_1(x) = x\log x - (x-1)$. The derivations in this paper benefit from employing unnormalized distributions and subsequently imposing the normalization condition as a constraint.

The α-divergence [26], [27] is a one-parameter family of f-divergences generated by the α-function $f_\alpha(x)$ with $\alpha\in\mathbb{R}$. The particular choice of the family of functions f_α is motivated by generalization of the natural logarithm [28]. The α-logarithm $\log_\alpha(x) = (x^{\alpha-1}-1)/(\alpha-1)$ is a power function for $\alpha\neq 1$ that turns into the natural logarithm for $\alpha\to 1$. Replacing the natural logarithm in the derivative of the KL divergence $f_1' = \log x$ by the α-logarithm and integrating f_α' under the condition that $f_\alpha(1)=0$ yields the α-function

$$f_\alpha(x) = \frac{(x^\alpha - 1) - \alpha(x-1)}{\alpha(\alpha-1)}. \tag{A.1}$$

The α-divergence generalizes the KL divergence, reverse KL divergence, Hellinger distance, Pearson χ^2-divergence, and Neyman (reverse Pearson) χ^2-divergence. Figure A.1 displays well-known α-divergences as points on the parabola $y=\alpha(\alpha-1)$. For every divergence,

there is a reverse divergence symmetric with respect to the point $\alpha = 0.5$, corresponding to the Hellinger distance.

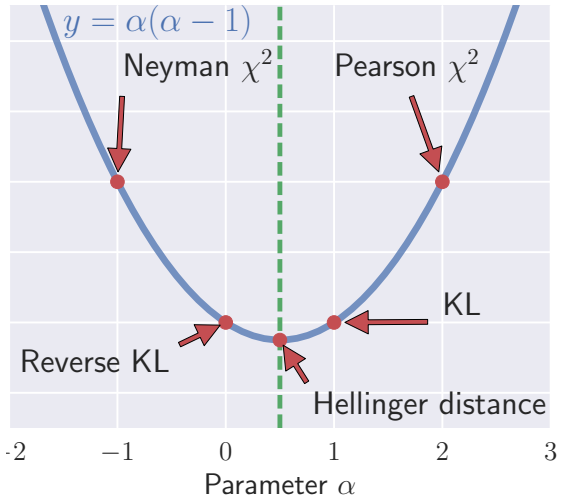


Figure A.1.: Common divergences as special cases of the α-divergence. The parabola in blue $y = \alpha(\alpha - 1)$ highlights the symmetry with respect to the axis $\alpha = 0.5$.

The convex conjugate of $f(x)$ is defined as $f^*(y) = \sup_{x \in \operatorname{dom} f}\{\langle y, x\rangle - f(x)\}$, where the angle brackets $\langle y, x\rangle$ denote the dot product. The key property $(f^*)' = (f')^{-1}$ relating the derivatives of f^* and f yields Table A.1, which lists common functions f_α together with their convex conjugates and derivatives. In the general case (A.1), the convex conjugate and its derivative are given by

$$\begin{aligned} f_\alpha^*(y) &= \frac{1}{\alpha}(1 + (\alpha - 1)y)^{\frac{\alpha}{\alpha - 1}} - \frac{1}{\alpha}, \\ (f_\alpha^*)'(y) &= \sqrt[\alpha - 1]{1 + (\alpha - 1)\, y}, \quad \text{for } \; y(1 - \alpha) < 1. \end{aligned} \tag{A.2}$$

Function f_α is convex, non-negative, and attains minimum at $x = 1$ with $f_\alpha(1) = 0$. Function $(f_\alpha^*)'$ is positive on its domain with $(f_\alpha^*)'(0) = 1$. Function f_α^* has the property $f_\alpha^*(0) = 0$. The linear inequality constraint (A.2) on the dom f_α^* follows from the requirement dom $f_\alpha = (0, \infty)$. Another result from convex analysis crucial to our derivations is Fenchel's equality

$$f^*(y) + f(x^\star(y)) = \langle y, x^\star(y)\rangle, \tag{A.3}$$

where $x^\star(y) = \arg\sup_{x \in \operatorname{dom} f}\{\langle y, x\rangle - f(x)\}$. We will occasionally put the conjugation symbol at the bottom, especially for the derivative of the conjugate function $f'_* = (f^*)'$.

Table A.1.: Function f_α(x), its convex conjugate f_α^*, and their derivatives.

Divergence	α	$f(x)$	$f'(x)$	$(f^*)'(y)$	$f^*(y)$	dom f^*
KL	1	$x \log x - (x-1)$	$\log x$	e^y	$e^y - 1$	$\mathbb{R}$
Reverse KL	0	$-\log x + (x-1)$	$-\frac{1}{x} + 1$	$\frac{1}{1-y}$	$-\log(1-y)$	$y < 1$
Pearson χ^2	2	$\frac{1}{2}(x-1)^2$	$x - 1$	$y + 1$	$\frac{1}{2}(y+1)^2 - \frac{1}{2}$	$y > -1$
Neyman χ^2	-1	$\frac{(x-1)^2}{2x}$	$-\frac{1}{2x^2} + \frac{1}{2}$	$\frac{1}{\sqrt{1-2y}}$	$-\sqrt{1-2y} + 1$	$y < \frac{1}{2}$
Hellinger	$\frac{1}{2}$	$2\left(\sqrt{x} - 1\right)^2$	$2 - \frac{2}{\sqrt{x}}$	$\frac{4}{(2-y)^2}$	$\frac{2y}{2-y}$	$y < 2$

A.2. Parameter Settings for Entropic Proximal Policy Optimization

In all experiments, the temperature parameter η is exponentially decayed $\eta_{i+1} = \eta_0 a^i$ in each iteration $i = 0, 1, \ldots$. The choice of η_0 and a depends on the scale of the rewards and the number of samples collected per policy update. Tables for each environment list these parameters along with the number of samples per policy update, the number of policy iteration steps, and the number of runs for averaging the results. Where applicable, environment-specific settings are also listed. (see the Tables A.2–A.4)

Table A.2.: Algorithm parameters for the Chain environment.

Parameter	Value
Number of states	8
Action success probability	0.9
Small and large rewards	(2.0, 10.0)
Number of runs	10
Number of iterations	30
Number of samples	800
Temperature parameters (η_0, a)	(15.0, 0.9)

Table A.3.: Algorithm parameters for the CliffWalking environment.

Parameter	**Value**
Punishment for falling from the cliff	-10.0
Reward for reaching the goal	100
Number of runs	10
Number of iterations	40
Number of samples	1500
Temperature parameters (η_0, a)	(50.0, 0.9)

Table A.4.: Algorithm parameters for the FrozenLake environment.

Parameter	**Value**
Action success probability	0.8
Number of runs	10
Number of iterations	50
Number of samples	2000
Temperature parameters (η_0, a)	(1.0, 0.8)

List of Acronyms

CBA	Constant Bearing Angle
GOAC	Generalized Optic Acceleration Cancellation]
KL	Kullback-Leibler
LOT	Linear Optical Trajectory
MDP	Markov Decision Process
OAC	Optic Acceleration Cancellation
POMDP	Partially Observable Markov Decision Process
REPS	Relative Entropy Policy Search
RL	Reinforcement Learning
TD3	Twin Delayed Deep Deterministic Policy Gradient

List of Figures

2.1. Effects of the divergence parameter α on policy improvement. 18
2.2. Average regret vs. time and divergence type α on a bandit problem. 19
2.3. Effects of the divergence parameter α on policy iteration. 20

3.1. Four well-known ball catching heuristics shown in one figure. 25
3.2. A typical simulated trajectory of a successful catch. 31
3.3. Heuristics hold in simulated successful catches. 33
3.4. A successful interception plan that violates heuristics. 34
3.5. Heuristics are violated when gaze turns are required. 35
3.6. Reactive and feedforward policies vs. uncertainty and latency. 36
3.7. A 3D view of the baseball stadium for ball catching experiments. 38
3.8. Ball catching field setup seen from above. 39
3.9. Top view of the baseball field from a drone. 41
3.10. Side view of the baseball field from a drone. 42
3.11. View from the eye-tracking glasses. 43
3.12. Four real trajectories overlayed with simulated trials. 49
3.13. CBA heuristic, simulation vs. real data. 50
3.14. OAC heuristic, simulation vs. real data. 51
3.15. GOAC heuristic, simulation vs. real data. 52
3.16. LOT heuristic, simulation vs. real data. 53

4.1. Learning-based control architecture for autonomous assembly. 58
4.2. Simulated and real-robot setups for learning block assembly. 59
4.3. Robotic reassembly at Luminale 2018. 61
4.4. Robot arm learning to connect modules in simulation. 63
4.5. Reinforcement learning loop. 64
4.6. Integration of Rhino3D with PyBullet for physics simulation. 65
4.7. PyBullet interface as seen in Grasshopper. 67
4.8. Example goal configurations of the stacked blocks. 70
4.9. Average cost and position error during training. 74

4.10. Tactile sensor DIGIT, simulation and real-robot setup. 76
4.11. Tactile sensor readings during learning on the real system. 77
4.12. Network architecture of the learned dynamics model. 77
4.13. Training loss over iterations for the dynamics model. 79

A.1. Common divergences as special cases of the α-divergence. 88

List of Tables

2.1. Policy evaluation/improvement objectives for $\alpha = 1$ and $\alpha = 2$. 14

3.1. Summary of the data on gaze turns in the ball catching trials. 45
3.2. Summary of the data on heuristics in the ball catching trials. 47

A.1. Function f_α(x), its convex conjugate f_α^*, and their derivatives. 89
A.2. Algorithm parameters for the Chain environment. 89
A.3. Algorithm parameters for the Cliff Walking environment. 90
A.4. Algorithm parameters for the FrozenLake environment. 90

Bibliography

[1] S. Schaal and N. Schweighofer, "Computational motor control in humans and robots", *Current opinion in neurobiology*, vol. 15, no. 6, pp. 675–682, 2005.

[2] T. Parr, G. Pezzulo, and K. J. Friston, *Active inference: the free energy principle in mind, brain, and behavior*. MIT Press, 2022.

[3] D. M. Wolpert, "Computational approaches to motor control", *Trends in cognitive sciences*, vol. 1, no. 6, pp. 209–216, 1997.

[4] B. Belousov and J. Peters, "Entropic regularization of markov decision processes", *Entropy*, vol. 21, no. 7, p. 674, 2019.

[5] B. Belousov and J. Peters, "F-divergence constrained policy improvement", *arXiv preprint arXiv:1801.00056*, 2017.

[6] B. Belousov, G. Neumann, C. A. Rothkopf, and J. R. Peters, "Catching heuristics are optimal control policies", *Advances in neural information processing systems*, vol. 29, 2016.

[7] B. Belousov, B. Wibranek, J. Schneider, *et al.*, "Robotic architectural assembly with tactile skills: Simulation and optimization", *Automation in Construction*, vol. 133, p. 104 006, 2022.

[8] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994, ISBN: 0471619779.

[9] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

[10] M. P. Deisenroth, G. Neumann, and J. Peters, "A survey on policy search for robotics", *Foundations and Trends®in Robotics*, vol. 2, no. 1–2, pp. 1–142, 2013.

[11] R. Bellman, *Dynamic Programming*. 1957, vol. 70, p. 342, ISBN: 978-0-691-07951-6. DOI: 10.1108/eb059970.

[12] S. M. Kakade, "A Natural Policy Gradient", in *NIPS*, 2001, pp. 1531–1538, ISBN: 9780874216561. DOI: 10.1.1.19.8165.

[13] J. Peters, K. Mülling, and Y. Altun, "Relative Entropy Policy Search", in *AAAI*, 2010, pp. 1607–1612.

[14] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization", in *International conference on machine learning*, PMLR, 2015, pp. 1889–1897.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms", *arXiv:1707.06347*, 2017.

[16] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function", *Journal of statistical planning and inference*, vol. 90, no. 2, pp. 227–244, 2000.

[17] G. Neu, A. Jonsson, and V. Gómez, "A unified view of entropy-regularized Markov decision processes", *arXiv:1705.07798*, 2017.

[18] N. Parikh, "Proximal Algorithms", *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014, ISSN: 2167-3888. DOI: 10.1561/2400000003.

[19] F. Nielsen, "An elementary introduction to information geometry", *arXiv preprint arXiv:1808.08271*, 2018.

[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial nets", *Advances in neural information processing systems*, vol. 27, 2014.

[21] L. Bottou, M. Arjovsky, D. Lopez-Paz, and M. Oquab, "Geometrical Insights for Implicit Generative Modeling", *arXiv:1712.07822*, 2017.

[22] S. Nowozin, B. Cseke, and R. Tomioka, "f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization", in *NIPS*, 2016, pp. 271–279.

[23] M. Teboulle, "Entropic Proximal Mappings with Applications to Nonlinear Programming", *Mathematics of Operations Research*, vol. 17, no. 3, pp. 670–690, 1992, ISSN: 0364-765X. DOI: 10.1287/moor.17.3.670.

[24] A. Nemirovski and D. Yudin, *Problem complexity and method efficiency in optimization*. Wiley, 1983.

[25] A. Beck and M. Teboulle, "Mirror descent and nonlinear projected subgradient methods for convex optimization", *Operations Research Letters*, vol. 31, no. 3, pp. 167–175, 2003.

[26] H. Chernoff, "A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations", *The Annals of Mathematical Statistics*, pp. 493–507, 1952.

[27] S. Amari, *Differential-Geometrical Methods in Statistics*. Springer New York, 1985. DOI: 10.1007/978-1-4612-5056-2.

[28] A. Cichocki and S. Amari, "Families of alpha- beta- and gamma- divergences: Flexible and robust measures of Similarities", *Entropy*, vol. 12, no. 6, pp. 1532–1568, 2010, ISSN: 10994300. DOI: 10.3390/e12061532.

[29] P. S. Thomas and B. Okal, "A notation for Markov decision processes", *arXiv:1512.09075*, 2015.

[30] R. S. Sutton, D. Mcallester, S. Singh, and Y. Mansour, "Policy Gradient Methods for Reinforcement Learning with Function Approximation", in *NIPS*, 1999, pp. 1057–1063, ISBN: 0-262-19450-3. DOI: 10.1.1.37.9714.

[31] J. Peters and S. Schaal, "Natural Actor-Critic", *Neurocomputing*, vol. 71, no. 7-9, pp. 1180–1190, 2008, ISSN: 09252312. DOI: 10.1016/j.neucom.2007.11.026.

[32] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High Dimensional Continuous Control Using Generalized Advantage Estimation", in *ICLR*, 2016.

[33] I. Csiszár, "Eine informationstheoretische Ungleichung und ihre Anwendung auf den Beweis der Ergodizität von Markoffschen Ketten", *Publ. Math. Inst. Hungar. Acad. Sci.*, vol. 8, pp. 85–108, 1963.

[34] H. Zhu and R. Rohwer, "Information geometric measurements of generalisation", Aston University, Tech. Rep., 1995.

[35] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning", *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992.

[36] M. J. Wainwright and M. I. Jordan, "Graphical Models, Exponential Families, and Variational Inference", *Foundations and Trends in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, 2007, ISSN: 19358237. DOI: 10.1561/2200000001.

[37] L. Baird, "Residual Algorithms: Reinforcement Learning with Function Approximation", *Proceedings of the 12th International Conference on Machine Learning*, pp. 30–37, 1995, ISSN: 00043702. DOI: 10.1.1.48.3256.

[38] C. Dann, G. Neumann, J. Peters, *et al.*, "Policy evaluation with temporal differences: A survey and comparison", *Journal of Machine Learning Research*, vol. 15, pp. 809–883, 2014.

[39] I. Sason and S. Verdu, "F-divergence inequalities", *IEEE Transactions on Information Theory*, vol. 62, no. 11, pp. 5973–6006, 2016, ISSN: 00189448. DOI: 10.1109/TIT.2016.2603151.

[40] S. Bubeck and N. Cesa-Bianchi, "Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems", *Foundations and Trends in Machine Learning*, vol. 5, no. 1, pp. 1–122, 2012, ISSN: 9781601986269. DOI: 10.1561/2200000024. arXiv: 1204.5721.

[41] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. Schapire, "The Non-Stochastic Multi-Armed Bandit Problem", *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2003.

[42] M. Ghavamzadeh, S. Mannor, J. Pineau, and A. Tamar, "Bayesian Reinforcement Learning: A Survey", *Foundations and Trends in Machine Learning*, vol. 8, no. 5-6, pp. 359–483, 2015, ISSN: 1935-8237. DOI: 10.1561/2200000049. arXiv: 1405.4980.

[43] G. Brockman, V. Cheung, L. Pettersson, *et al.*, "OpenAI Gym", *arXiv preprint arXiv:1606.01540*, 2016. arXiv: 1606.01540.

[44] N. Tishby and D. Polani, "Information theory of decisions and actions", in *Perception-action cycle*, Springer, 2011, pp. 601–636.

[45] N. Bertschinger, E. Olbrich, N. Ay, and J. Jost, "Autonomy: An information theoretic perspective", *Biosystems*, vol. 91, no. 2, pp. 331–345, 2008.

[46] S. Still and D. Precup, "An information-theoretic approach to curiosity-driven reinforcement learning", *Theory in Biosciences*, vol. 131, no. 3, pp. 139–148, 2012.

[47] T. Genewein, F. Leibfried, J. Grau-Moya, and D. A. Braun, "Bounded rationality, abstraction, and hierarchical decision-making: An information-theoretic optimality principle", *Frontiers in Robotics and AI*, vol. 2, p. 27, 2015.

[48] D. H. Wolpert, "Information theory□the bridge connecting bounded rational game theory and statistical physics", in *Complex Engineered Systems*, Springer, 2006, pp. 262–290.

[49] M. Geist, B. Scherrer, and O. Pietquin, "A theory of regularized markov decision processes", *arXiv preprint arXiv:1901.11275*, 2019.

[50] X. Li, W. Yang, and Z. Zhang, "A unified framework for regularized reinforcement learning", *arXiv preprint arXiv:1903.00725*, 2019.

[51] O. Nachum, Y. Chow, and M. Ghavamzadeh, "Path consistency learning in tsallis entropy regularized mdps", *arXiv preprint arXiv:1802.03501*, 2018.

[52] K. Lee, S. Kim, S. Lim, S. Choi, and S. Oh, "Tsallis reinforcement learning: A unified framework for maximum entropy reinforcement learning", *arXiv preprint arXiv:1902.00137*, 2019.

[53] K. Lee, S. Choi, and S. Oh, "Sparse markov decision processes with causal sparse tsallis entropy regularization for reinforcement learning", *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1466–1473, 2018.

[54] K. Lee, S. Choi, and S. Oh, "Maximum causal tsallis entropy imitation learning", in *Advances in Neural Information Processing Systems*, 2018, pp. 4403–4413.

[55] S. Mahadevan, B. Liu, P. Thomas, *et al.*, "Proximal reinforcement learning: A new theory of sequential decision making in primal-dual spaces", *arXiv preprint arXiv:1405.6757*, 2014.

[56] S. Chapman, "Catching a baseball", *American Journal of Physics*, vol. 36, no. 10, p. 868, 1968. DOI: 10.1119/1.1974297.

[57] J. McIntyre, M. Zago, A. Berthoz, and F. Lacquaniti, "Does the brain model Newton's laws?", *Nature neuroscience*, vol. 4, no. 7, pp. 693–694, 2001. DOI: 10.1097/00001756-200112040-00004.

[58] M. M. Hayhoe, N. Mennie, K. Gorgos, J. Semrau, and B. Sullivan, "The role of prediction in catching balls.", *Journal of Vision*, vol. 4, no. 8, pp. 156–156, 2004, ISSN: , 1534-7362. DOI: 10.1167/4.8.156.

[59] G. Gigerenzer and H. Brighton, "Homo Heuristicus: Why Biased Minds Make Better Inferences", *Topics in Cognitive Science*, vol. 1, no. 1, pp. 107–143, 2009. DOI: 10.1111/j.1756-8765.2008.01006.x.

[60] G. Gigerenzer, *Gut feelings: The intelligence of the unconscious*. Penguin, 2007.

[61] H. A. Simon, "A Behavioral Model of Rational Choice", *The Quarterly Journal of Economics*, vol. 69, no. 1, pp. 99–118, 1955. DOI: 10.2307/1884852.

[62] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model.", *The Journal of Neuroscience*, vol. 5, no. 7, pp. 1688–1703, 1985.

[63] E. Todorov and M. I. Jordan, "Optimal feedback control as a theory of motor coordination", *Nature neuroscience*, vol. 5, no. 11, pp. 1226–1235, 2002. DOI: 10.1038/nn963.

[64] F. C. Anderson and M. G. Pandy, "Dynamic optimization of human walking.", *Journal of biomechanical engineering*, vol. 123, no. 5, pp. 381–390, 2001.

[65] M. Zago, J. McIntyre, P. Senot, and F. Lacquaniti, *Visuo-motor coordination and internal models for object interception*, 2009.

[66] P. W. Fink, P. S. Foo, and W. H. Warren, "Catching fly balls in virtual reality: a critical test of the outfielder problem.", *Journal of vision*, vol. 9, no. 13, pp. 1–8, 2009.

[67] P. McLeod, N. Reed, and Z. Dienes, "The generalized optic acceleration cancellation theory of catching.", *Journal of experimental psychology. Human perception and performance*, vol. 32, no. 1, pp. 139–48, 2006. DOI: 10.1037/0096-1523.32.1.139.

[68] M. McBeath, D. Shaffer, and M. Kaiser, "How baseball outfielders determine where to run to catch fly balls", *Science*, vol. 268, no. 5210, pp. 569–573, 1995. DOI: 10.1126/science.7725104.

[69] R. C. Miall and D. M. Wolpert, *Forward models for physiological motor control*, 1996.

[70] Y. Uno, M. Kawato, and R. Suzuki, "Formation and control of optimal trajectory in human multijoint arm movement. Minimum torque-change model.", *Biological cybernetics*, vol. 61, no. 2, pp. 89–101, 1989.

[71] C. M. Harris and D. M. Wolpert, "Signal-dependent noise determines motor planning.", *Nature*, vol. 394, no. 6695, pp. 780–4, 1998.

[72] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. 2005.

[73] R. Platt, R. Tedrake, L. Kaelbling, and T. Lozano-Perez, "Belief space planning assuming maximum likelihood observations", *Robotics: Science and Systems*, 2010.

[74] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty", *Proceedings - IEEE ICRA*, pp. 723–730, 2011. DOI: 10.1109/ICRA.2011.5980508.

[75] J. van den Berg, S. Patil, and R. Alterovitz, "Motion planning under uncertainty using iterative local optimization in belief space", *The International Journal of Robotics Research*, vol. 31, no. 11, pp. 1263–1278, 2012, ISSN: 0278-3649. DOI: 10.1177/0278364912456319.

[76] S. Patil, G. Kahn, M. Laskey, and J. Schulman, "Scaling up Gaussian Belief Space Planning through Covariance-Free Trajectory Optimization and Automatic Differentiation", *Algorithmic Foundations of Robotics XI*, pp. 515–533, 2015.

[77] M. Diehl, H. G. Bock, H. Diedam, and P. B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control", in *Lecture Notes in Control and Information Sciences*, vol. 340, 2006, pp. 65–93.

[78] J. T. Betts, "Survey of Numerical Methods for Trajectory Optimization", *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[79] J. Andersson, J. Åkesson, and M. Diehl, "CasADi: A symbolic package for automatic differentiation and optimal control", in *Recent Advances in Algorithmic Differentiation*, Springer, 2012, pp. 297–307.

[80] A. Wächter and L. T. Biegler, "On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming", *Mathematical Programming*, vol. 106, pp. 25–57, 2006. DOI: 10.1007/BF01582568.

[81] P. J. Brancazio, "Looking into Chapman's homer: The physics of judging a fly ball", *American Journal of Physics*, vol. 53, no. 9, p. 849, 1985, ISSN: 00029505. DOI: 10.1119/1.14350.

[82] M. P. Vitus and C. J. Tomlin, "Closed-loop belief space planning for linear, Gaussian systems", in *Proceedings - IEEE ICRA*, 2011, pp. 2152–2159.

[83] S. Tian, F. Ebert, D. Jayaraman, *et al.*, "Manipulation by feel: Touch-based control with deep predictive models", in *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, IEEE, 2019, pp. 818–824. DOI: 10.1109/ICRA.2019.8794219.

[84] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search", in *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*, IEEE, 2015, pp. 156–163. DOI: 10.1109/ICRA.2015.7138994.

[85] R. Calandra, A. Owens, D. Jayaraman, *et al.*, "More than a feeling: Learning to grasp and regrasp using vision and touch", *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3300–3307, 2018. DOI: 10.1109/LRA.2018.2852779.

[86] F. Gramazio, M. Kohler, V. Helm, and S. Ercan, "In-situ robotic construction. extending the digital fabrication chain in architecture", en, in *Synthetic Digital Ecologies: Proceedings of the 32nd Annual Conference of the Association for Computer Aided Design in Architecture*, San Francisco, CA: ACADIA, 2012, pp. 169–176, ISBN: 978-1-62407-267-3.

[87] J. Rowe, "Life in a computerised environment", *Electronics Australia*, Sep. 1972, Identifier: ea197203_201912.

[88] F. Roche, S. Lavaux, and J. Navarro. "Olzweg". Retrieved September 6, 2021. (2006), [Online]. Available: https://new-territories.com/welostit.htm.

[89] B. Jenett, A. Abdel-Rahman, K. C. Cheung, and N. Gershenfeld, "Material-robot system for assembly of discrete cellular structures", *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4019–4026, 2019. DOI: 10.1109/LRA.2019.2930486.

[90] T. Bock and T. Linner, *Robot oriented design: Design and Management Tools for the Deployment of Automation and Robotics in Construction*. Cambridge university press, 2015, ISBN: 9781107076389.

[91] G. Rossi and P. Nicholas, "Re/learning the wheel: Methods to utilize neural networks as design tools for doubly curved metal surfaces", English, in *ACADIA // 2018: Recalibration. On imprecisionand infidelity*, ACADIA 2018 : Recalibration. On imprecision and infidelity ; Conference date: 18-10-2018 Through 20-10-2018, Oct. 2019, pp. 146–155.

[92] G. Brugnaro and S. Hanna, "Adaptive robotic carving", in *Robotic Fabrication in Architecture, Art and Design 2018*, J. Willmann, P. Block, M. Hutter, K. Byrne, and T. Schork, Eds., Cham: Springer International Publishing, 2019, pp. 336–348, ISBN: 978-3-319-92294-2.

[93] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018, ISBN: 9780262364010.

[94] N. Melenbrink, J. Werfel, and A. Menges, "On-site autonomous construction robots: Towards unsupervised building", *Automation in Construction*, vol. 119, p. 103 312, 2020, ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2020.103312.

[95] V. Hartmann, O. Oguz, D. Driess, M. Toussaint, and A. Menges, "Robust task and motion planning for long-horizon architectural construction planning", in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2020, Las Vegas, NV, USA, October 24, 2020 - January 24, 2021*, IEEE, 2020, pp. 6886–6893. DOI: 10.1109/IROS45743.2020.9341502.

[96] B. Wibranek, Y. Liu, N. Funk, B. Belousov, J. Peters, and O. Tessmann, "Reinforcement learning for sequential assembly of sl-blocks: Self-interlocking combinatorial design based on machine learning", in *Proceedings of the 39th eCAADe Conference*, vol. 1, CUMINCAD, 2021, pp. 27–36, ISBN: 978-94-91207-22-8.

[97] N. Funk, G. Chalvatzaki, B. Belousov, and J. Peters, "Learn2assemble with structured representations and search for robotic architectural construction", in *Proceedings of the 5th Annual Conference on Robot Learning*, 2021.

[98] G. Sartoretti, Y. Wu, W. Paivine, T. K. S. Kumar, S. Koenig, and H. Choset, "Distributed reinforcement learning for multi-robot decentralized collective construction", in *Distributed Autonomous Robotic Systems, The 14th International Symposium, DARS 2018, Boulder, CO, USA, October 15-17, 2018*, N. Correll, M. Schwager, and M. W. Otte, Eds., ser. Springer Proceedings in Advanced Robotics, vol. 9, Springer, 2018, pp. 35–49. DOI: 10.1007/978-3-030-05816-6_3.

[99] Z. Kappasov, J. A. Corrales, and V. Perdereau, "Tactile sensing in dexterous robot hands - review", *Robotics and Autonomous Systems*, vol. 74, pp. 195–220, 2015. DOI: 10.1016/j.robot.2015.07.015.

[100] H. Yousef, M. Boukallel, and K. Althoefer, "Tactile sensing for dexterous in-hand manipulation in robotics—a review", *Sensors and Actuators A: Physical*, vol. 167, no. 2, pp. 171–187, 2011, Solid-State Sensors, Actuators and Microsystems Workshop, ISSN: 0924-4247. DOI: https://doi.org/10.1016/j.sna.2011.02.038.

[101] B. Belousov, A. Sadybakasov, B. Wibranek, F. Veiga, O. Tessmann, and J. Peters, "Building a library of tactile skills based on fingervision", in *IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, IEEE, 2019, pp. 717–722.

[102] W. Yuan, S. Dong, and E. H. Adelson, "Gelsight: High-resolution robot tactile sensors for estimating geometry and force", *Sensors*, vol. 17, no. 12, p. 2762, 2017. DOI: 10.3390/s17122762.

[103] H. van Hoof, N. Chen, M. Karl, P. van der Smagt, and J. Peters, "Stable reinforcement learning with autoencoders for tactile and visual data", in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2016, Daejeon, South Korea, October 9-14, 2016*, IEEE, 2016, pp. 3928–3934. DOI: 10.1109/IROS.2016.7759578.

[104] J. A. Fishel and G. E. Loeb, "Sensing tactile microvibrations with the biotac — comparison with human sensitivity", in *2012 4th IEEE RAS EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2012, pp. 1122–1127. DOI: 10.1109/BioRob.2012.6290741.

[105] C. Robotics. "Semi-automated mason (sam)". Retrieved September 6, 2021. (2007-2021), [Online]. Available: https://www.construction-robotics.com/sam-2/.

[106] M. Lambeta, P.-W. Chou, S. Tian, *et al.*, "Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation", *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 3838–3845, 2020.

[107] S. Wang, M. Lambeta, L. Chou, and R. Calandra, "Tacto: A fast, flexible and open-source simulator for high-resolution vision-based tactile sensors", *Arxiv*, 2020.

[108] T. Inoue, G. D. Magistris, A. Munawar, T. Yokoya, and R. Tachibana, "Deep reinforcement learning for high precision assembly tasks", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 819–825. DOI: 10.1109/IROS.2017.8202244.

[109] A. A. Apolinarska, M. Pacher, H. Li, *et al.*, "Robotic assembly of timber joints using reinforcement learning", *Automation in Construction*, vol. 125, pp. 1–8, 2021, ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2021.103569.

[110] R. McNeel and Associates. "Rhinoceros". Retrieved September 6, 2021. (1993-2021), [Online]. Available: https://www.rhino3d.com/.

[111] V. Bapst, A. Sanchez-Gonzalez, C. Doersch, *et al.*, "Structured agents for physical construction", in *International Conference on Machine Learning (ICML)*, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97, PMLR, Sep. 2019, pp. 464–474.

[112] E. Coumans and Y. Bai, *Pybullet, a python module for physics simulation for games, robotics and machine learning*, Retrieved September 6, 2021, 2016–2021.

[113] A. Raffin, A. Hill, M. Ernestus, A. Gleave, A. Kanervisto, and N. Dormann, *Stable baselines3*, Retrieved September 6, 2021, 2019.

[114] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods", in *International Conference on Machine Learning (ICML)*, J. Dy and A. Krause, Eds., vol. 80, PMLR, Oct. 2018, pp. 1587–1596.

[115] T. P. Lillicrap, J. J. Hunt, A. Pritzel, *et al.*, "Continuous control with deep reinforcement learning", in *International Conference on Learning Representations (ICLR)*, Y. Bengio and Y. LeCun, Eds., 2016.

[116] OpenAI, I. Akkaya, M. Andrychowicz, *et al.*, "Solving rubik's cube with a robot hand", *Arxiv*, 2019.

[117] O. Vinyals, I. Babuschkin, W. M. Czarnecki, *et al.*, "Grandmaster level in starcraft II using multi-agent reinforcement learning", *Nat.*, vol. 575, no. 7782, pp. 350–354, 2019. DOI: 10.1038/s41586-019-1724-z.

[118] D. F. Gomes, P. Paoletti, and S. Luo, "Generation of gelsight tactile images for sim2real learning", *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4177–4184, 2021. DOI: 10.1109/LRA.2021.3063925.

[119] J. Matas, S. James, and A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation", in *Conference on Robot Learning (CoRL)*, vol. 87, PMLR, 2018, pp. 734–743.

[120] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks", in *International Conference on Learning Representations (ICLR)*, 2017.

[121] A. Church, J. Lloyd, N. F. Lepora, *et al.*, "Tactile sim-to-real policy transfer via real-to-sim image translation", in *Conference on Robot Learning*, PMLR, 2022, pp. 1645–1654.

[122] F. Muratore, "Randomizing physics simulations for robot learning", Ph.D. dissertation, Technische Universität Darmstadt, 2021, p. 138.

[123] F. Muratore, F. Treede, M. Gienger, and J. Peters, "Domain randomization for simulation-based policy optimization with transferability assessment", in *Conference on Robot Learning*, PMLR, 2018, pp. 700–713.

[124] F. Muratore, F. Ramos, G. Turk, W. Yu, M. Gienger, and J. Peters, "Robot learning from randomized simulations: A review", *Frontiers in Robotics and AI*, vol. 9, 2022.

[125] F. Muratore, T. Gruner, F. Wiese, B. Belousov, M. Gienger, and J. Peters, "Neural posterior domain randomization", in *Conference on Robot Learning*, PMLR, 2022, pp. 1532–1542.

[126] D. Nass, B. Belousov, and J. Peters, "Entropic risk measure in policy search", in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 1101–1106.

[127] P. Klink, H. Abdulsamad, B. Belousov, and J. Peters, "Self-paced contextual reinforcement learning", in *Conference on Robot Learning*, PMLR, 2020, pp. 513–529.

[128] P. Klink, H. Abdulsamad, B. Belousov, C. D'Eramo, J. Peters, and J. Pajarinen, "A probabilistic interpretation of self-paced learning with applications to reinforcement learning.", *J. Mach. Learn. Res.*, vol. 22, pp. 182–1, 2021.

[129] H. Abdulsamad, T. Dorau, B. Belousov, J.-J. Zhu, and J. Peters, "Distributionally robust trajectory optimization under uncertain dynamics via relative-entropy trust regions", *arXiv preprint arXiv:2103.15388*, 2021.

[130] M. Schultheis, B. Belousov, H. Abdulsamad, and J. Peters, "Receding horizon curiosity", in *Conference on robot learning*, PMLR, 2020, pp. 1278–1288.

[131] B. Belousov, H. Abdulsamad, M. Schultheis, and J. Peters, “Belief space model predictive control for approximately optimal system identification”, TU Darmstadt, 2019., Tech. Rep.

[132] L. Da Costa, P. Lanillos, N. Sajid, K. Friston, and S. Khan, “How active inference could help revolutionise robotics”, *Entropy*, vol. 24, no. 3, p. 361, 2022.

[133] T. Schneider, B. Belousov, G. Chalvatzaki, D. Romeres, D. Jha, and J. Peters, “Active exploration for robotic manipulation”, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[134] T. Schneider, B. Belousov, H. Abdulsamad, and J. Peters, “Active inference for robotic manipulation”, in *5th Multidisciplinary Conference on Reinforcement Learning and Decision Making (RLDM)*, 2022.

[135] M. Lutter, B. Belousov, K. Listmann, D. Clever, and J. Peters, “Hjb optimal feedback control with deep differential value functions and action constraints”, in *Conference on Robot Learning*, PMLR, 2020, pp. 640–650.

[136] M. Lutter, B. Belousov, S. Mannor, D. Fox, A. Garg, and J. Peters, “Continuous-time fitted value iteration for robust policies”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[137] M. Lutter, C. Ritter, and J. Peters, “Deep lagrangian networks: Using physics as model prior for deep learning”, in *International Conference on Learning Representations*, 2018.

[138] M. Lutter, K. Listmann, and J. Peters, “Deep lagrangian networks for end-to-end learning of energy-based control for under-actuated systems”, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 7718–7725.

[139] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, “Attention is all you need”, *Advances in neural information processing systems*, vol. 30, 2017.

[140] L. Chen, K. Lu, A. Rajeswaran, *et al.*, “Decision transformer: Reinforcement learning via sequence modeling”, *Advances in neural information processing systems*, vol. 34, 2021.

[141] T. Morimoto, “Markov processes and the H-theorem”, *Journal of the Physical Society of Japan*, vol. 18, no. 3, pp. 328–331, 1963.

[142] S. M. Ali and S. D. Silvey, “A General Class of Coefficients of Divergence of One Distribution from Another”, *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 28, pp. 131–142, 1966, ISSN: 0035-9246.