

Feasibility-Guided Exploration and Adaptation for Constrained Robotic Manipulation

Presented on 30th April 2026

School of Engineering
Biorobotics Laboratory
Doctoral program in Electrical Engineering

for the award of the degree of Docteur ès Sciences (PhD)

by

Amirreza RAZMJOO FARD

Accepted on the jury's recommendation

Prof. C. N. Jones, jury president
Prof. A. Ijspeert, Dr S. Calinon, thesis directors
Prof. A. Ajoudani, examiner
Prof. M. Khoramshahi, examiner
Prof. J. Hughes, examiner

Wings are a constraint that makes
it possible to fly.
— Robert Bringhurst

To my mother, who taught me to hope,
and my father, who taught me to try,
and my sister, who taught me to be responsible.

Acknowledgements

I would like to take this opportunity to thank all the people who made it possible for me to reach this point.

First and foremost, I am deeply grateful to Sylvain Calinon, who offered me the chance to join his team and guided me throughout this journey. His support and encouragement have played a key role in my growth, and it has been a true pleasure to be part of the RLI group at Idiap.

I would like to sincerely thank the members of the jury — Prof. Josie Hughes, Prof. Arash Ajoudani, Prof. Mahdi Khoramshahi, Prof. Colin Jones, and Prof. Auke Ijspeert — for their valuable feedback and constructive assessment. I deeply appreciate the time and expertise they contribute to this process, and I am honored by their participation.

Among my colleagues, I owe a special debt of gratitude to Teguh Santoso Lembono (now Dr. Lembono), who mentored me from the very first days and helped me navigate both PhD life and Switzerland, and to Teng Xue, with whom I shared countless moments—from co-authoring papers to daily coffee breaks and lunches. Their friendship and support made the challenges of the PhD much more manageable, and I feel very fortunate to have had them by my side.

I also want to sincerely thank all my lab-mates in RLI—Boyang, Suhan, Jiacheng, Ante, Cem, Saray, Julius, Tobias, Martin, Marco, Mattia, João, James, and many others—for making this journey more joyful and memorable. I am especially grateful to Yiming Li and Yan Zhang, with whom I had the pleasure of collaborating on publications, and whose insights and dedication greatly enriched my research. I am equally grateful to my colleagues and friends at the Honda Research Institute, especially Dr. Michael Gienger and Dr. Fan Zhang, for providing me with an invaluable research experience that greatly enriched my PhD journey.

Outside the workplace, my deepest gratitude goes to my parents and my sister, who, as always, supported me unconditionally. They encouraged me to move forward, to not give up, and to keep hope alive—hope for a future full of uncertainty, unknowns, and, above all, love.

I also want to thank my cousins, especially Elham and Hamed, who played a decisive role in shaping my curiosity toward science and technology. They taught me the basics of thinking and have been true friends throughout my life. I am also thankful to Hadisah, Mahya, and Alireza for their constant encouragement and support.

Beyond family, I am grateful to my many friends who made life in Switzerland more enjoyable,

including Parsa, Mahdi, Amir, Zohreh, Masoumeh, and others. I owe a special gratitude to Hatef, who helped me with important life decisions, and to Pooya, whose friendship goes back to our bachelor days and who played a key role in helping me secure my PhD position.

Finally, I would like to thank my mentors and colleagues from the CEDRA lab at Sharif University of Technology—Prof. Meghdari, Dr. Taheri, Ehsan Saffari, Ehsan Ahmadi, Ehsan Zakipour, Pourya, Raman, Ali, Alireza, Amir, and many others—who laid the first bricks of my research journey, introduced me to robotics, and allowed me to experience this path with joy. In particular, I am deeply grateful to Ehsan Ahmadi, whose guidance and encouragement during my early research years played a key role in shaping my interest in robotics and motivating me to pursue this path further.

Martigny, October 2025

Amir. Razmjoo

Abstract

Robots operating in unstructured environments must safely and effectively modulate contact forces with unfamiliar objects; a requirement that underlies tasks as diverse as collaborative object handling with humans and bulky-object manipulation through pushing or pivoting. Such interactions pose significant challenges, such as multi-modal dynamics, long-horizon planning with sparse rewards. Furthermore, the constraints inherent in such tasks amplify these difficulties and make it hard even to identify a feasible motion, let alone an optimal one. Paradoxically, however, these very constraints can also serve as guiding principles, narrowing the robot's exploration and steering it toward viable solutions. Motivated by this insight, the present thesis investigates strategies for incorporating and managing constraints to simplify and accelerate robotic interactions.

First, we demonstrate that representing feasible robot configurations on a low-dimensional robust manifold enables a humanoid robot to maintain stability while allowing a human user to control its motion directly on that manifold. However, identifying such a stable manifold in the full configuration space can itself be challenging, and in many cases it is preferable to tackle the original optimization problem directly. To that end, in our second contribution, we show how incorporating expert demonstrations can seed the solvers with these demonstrations (aka, warm-start) and augment gradients in regions of sparse cost, which accelerates convergence. Practically, this "sample-then-project" approach first generates candidate trajectories from expert data and then refines them via gradient descent in the full space. Extending this, our third contribution introduces a "project-then-sample" framework that offers more explorative behavior and faster convergence. By decomposing the problem into separate feasibility and optimality subproblems, we can sample trajectories from their combined solution space, ensuring sampled candidates are simultaneously feasible and near-optimal. Our final contribution tackles the challenging scenario where robots lack explicit environmental constraints and task models, particularly in imitation learning without access to unsuccessful demonstrations. We demonstrate that by combining multiple experts learned solely from successful demonstrations, the robot can interactively refine its sampling area and proactively avoid previously failed actions. These approaches are validated in an extensive experimentation, featuring human-humanoid collaboration and diverse contact-rich manipulation scenarios, including door-opening, object pushing, and pivoting, etc.

Key words: Feasibility-Guided Exploration, Learning-from-Demonstrations, Generative Models for Control, Constrained Optimization, Products of Experts

Résumé

Les robots évoluant dans des environnements non structurés doivent traiter de manière sûre et efficace les forces de contact avec des objets inconnus, il s'agit d'une condition essentielle pour des tâches allant de la manipulation collaborative avec des humains à la manipulation d'objets volumineux par poussée ou pivotement. Ces interactions posent des défis majeurs, tels que des dynamiques multimodales et une planification sur de longs horizons avec récompenses clairsemées. De plus, les contraintes inhérentes à ces tâches amplifient ces difficultés, même pour trouver un mouvement faisable. Paradoxalement, ces contraintes peuvent aussi servir de guides, en réduisant l'espace d'exploration du robot et en l'orientant vers des solutions viables. Cette thèse explore des stratégies pour intégrer et exploiter ces contraintes afin de simplifier et d'accélérer les manipulations robotiques.

Premièrement, nous montrons que représenter les configurations faisables du robot sur une variété robuste de faible dimension permet à un humanoïde de maintenir sa stabilité tout en laissant un utilisateur humain guider son mouvement. Toutefois, identifier une telle variété dans l'espace complet des configurations est difficile, et il est souvent préférable de résoudre directement le problème d'optimisation. Notre deuxième contribution montre comment des démonstrations expertes peuvent fournir une initialisation (warm-start) et ajouter des gradients dans des régions de coût clairsemé, accélérant la convergence. Concrètement, cette approche «échantillonner-puis-projeter» génère des trajectoires candidates issues des données expertes, puis les affine par descente de gradient. Notre troisième contribution introduit une approche «projeter-puis-échantillonner» qui favorise l'exploration et la convergence rapide. En séparant faisabilité et optimalité, nous échantillonnons dans leur espace combiné, garantissant des trajectoires à la fois faisables et proches de l'optimum. Enfin, notre dernière contribution s'attaque au cas où le robot n'a pas accès à un modèle explicite, notamment en apprentissage par imitation sans démonstrations qui auraient échouées. Nous montrons qu'en combinant plusieurs experts appris uniquement à partir de démonstrations réussies, le robot peut affiner sa zone d'échantillonnage et éviter proactivement les actions déjà vouées à l'échec. Ces approches sont validées par des expérimentations variées, incluant la collaboration humain-humanoïde et des manipulations à contacts multiples telles que l'ouverture de porte, la poussée et le pivotement d'objets.

Mots-clés : Exploration guidée par la faisabilité, Apprentissage par démonstration, Modèles génératifs pour le contrôle, Optimisation sous contraintes, Produit d'experts

Contents

Acknowledgements	i
Abstract (English/Français)	iii
1 Introduction	1
1.1 Motivation	1
1.2 General Problem Formulation	3
1.3 Simplified Problem Setting for this Thesis	4
1.4 Thesis Challenges and Objectives	4
1.4.1 Slow Convergence	6
1.4.2 Inefficient Sampling	6
1.4.3 Unstable Configuration	7
1.4.4 Failure Recovery	7
1.5 Research Focus of this Thesis	8
2 Background	11
2.1 Robot Dynamics and Humanoid Stability	11
2.2 Optimal Control Approaches	13
2.2.1 Derivative-based Optimal Control	15
2.2.2 Sampling-based Optimal Control	16
2.3 Functional Representation	17
2.4 Tensor Train Decomposition	19
2.4.1 Tensor-Train Distribution	21
2.4.2 Conditional TT	21
2.4.3 TT-Sample	22
2.5 Imitation Learning	23
2.5.1 Diffusion Policy	24
2.6 Products of Experts (PoE)	24
2.6.1 Conditional diffusion policy composition.	25
3 Constraint-Aware Reference Manifold	27
3.1 Introduction	28
3.2 Related Work	28
3.3 Methodology	30

3.3.1	Simplified Robot Stability	30
3.3.2	Assumptions	30
3.3.3	Deriving a Spatial Reference	31
3.3.4	Functional Representations	32
3.4	Experiment	34
3.4.1	Finding the Spatial Reference	34
3.4.2	Results	35
3.5	Discussion	37
3.6	Conclusion	38
4	Demonstration-Guided Constrained Motion Generation	39
4.1	Introduction	40
4.2	Related Work	41
4.3	Methodology	41
4.3.1	Dynamical modeling	41
4.3.2	Task definition	42
4.3.3	Control Primitives (CPs)	43
4.3.4	Warm starting	44
4.3.5	Imitation cost	44
4.4	Simulation	45
4.5	Discussion	47
4.6	Conclusion	48
5	Products of Optimality and Feasibility for Constrained Motion Planning	49
5.1	Introduction	50
5.2	Related Work	54
5.3	Methodology	57
5.3.1	Problem Definition	57
5.3.2	Products of Experts	57
5.3.3	TT-PoE-MPPI	59
5.3.4	Decomposable feasible distribution	61
5.3.5	Data Gathering	63
5.4	Results	63
5.4.1	Obstacle Avoidance (PNGRID)	64
5.4.2	Tracking Tube (e.g., surface cleaning or painting)	65
5.4.3	Non-prehensile Manipulation (Planar-pushing)	67
5.4.4	Online Obstacle Avoidance	70
5.4.5	Higher Dimensions and Alternative Optimality Distributions	70
5.5	Discussion	72
5.5.1	TT-distribution vs Normalizing flow	72
5.5.2	Low-Rank Representation	74
5.5.3	Computation time	75
5.5.4	Limitations	75

5.6	Conclusion	77
6	CCDP: Model-free Failure Recovery via Guided Diffusion Sampling	79
6.1	Introduction	80
6.2	Related Work	81
6.3	Methodology	83
6.3.1	Composition of Diffusion Policies	83
6.3.2	Recovery Model	85
6.4	Experiments	87
6.4.1	Door Opening (DO)	90
6.4.2	Button Pressing (BP)	90
6.4.3	Object Manipulation (OM)	90
6.4.4	Objects Packing (OP)	91
6.4.5	Bartender (BT)	91
6.4.6	Ablation Study	91
6.5	Discussion	92
6.5.1	CCDP vs DP*	92
6.5.2	NOT Operation	93
6.5.3	Limitations and Future Work	93
6.6	Conclusion	93
7	Conclusion	95
7.1	Summary of contributions	95
7.2	Key Insights	96
7.3	Limitations	97
7.3.1	Chapter-specific limitations and future work	97
7.3.2	General limitations	98
7.4	Future Directions	99
7.4.1	Toward Other Algorithms	99
7.4.2	Toward Fully Integrated Systems	99
7.4.3	Common Sense Beyond Feasibility	100
7.4.4	Multi-agent, Multi-task, and Multi-scene Settings	100
7.5	Closing Remarks	100
A	Appendix	103
A.1	Appendix of chapter 5	103
A.1.1	Proj-MPPI	103
A.1.2	NFs-PoE-MPPI	103
A.1.3	MPPI on the Sphere Manifold	104
A.1.4	DIAL-mpc and TT-PoE-DIAL	105
A.1.5	Experiments	105
A.1.6	Real-robot Experiment	110

Bibliography	111
Curriculum Vitae	123

1 Introduction

1.1 Motivation

Robotic systems are increasingly deployed in unstructured and dynamic environments. In such settings, the robot must be able to adapt online and efficiently to changes in the environment. This requires advances both in hardware and software design. In this thesis, we focus on the software aspect, and more specifically on robot motion planning, where robots must interact physically with objects, humans, and their surroundings.

In unstructured environments, unlike in known and structured ones, a robot must actively decide whether to interact with objects or humans in its surroundings, or to avoid them. The first case requires the robot to perform contact-rich manipulation, where it must modulate contact forces and maintain feasibility under strict physical and task constraints. The second case corresponds to the more classical obstacle avoidance problem, which remains a challenging task in general. Examples include pushing or pivoting bulky objects while avoiding self-collisions and collisions with other objects, opening a door with uncertain hinge dynamics while avoiding contact with the door surface, or assisting a human in tasks such as a sit-to-stand transfer, where the robot must provide physical support while avoiding unintended collisions with other parts of the human body.

Such tasks are challenging for several reasons. To begin with, they involve nonlinear and hybrid dynamics, with frequent switching between different contact modes (Chavan-Dafle et al., 2018; Holladay et al., 2023). In addition, constraints are ubiquitous, including joint limits, collision avoidance (Bhardwaj et al., 2022; Ratliff et al., 2018), balance conditions (Hirukawa et al., 2006; Kajita & Tani, 1991), and task-specific force requirements (Geravand et al., 2017; Rozo et al., 2016). Moreover, rewards or costs are often sparse (Andrychowicz et al., 2017; Vulin et al., 2021) (for example, only the final success can be measured), which makes optimization and learning especially difficult over long horizons. Another important challenge is that many of these tasks are underactuated (Graesdal et al., 2024; Posa et al., 2016), meaning that not all desired motions can be achieved directly through actuation. On top of that, the state of the environment is often only partially observable (Kumar et al., 2021; Lee et al., 2020).

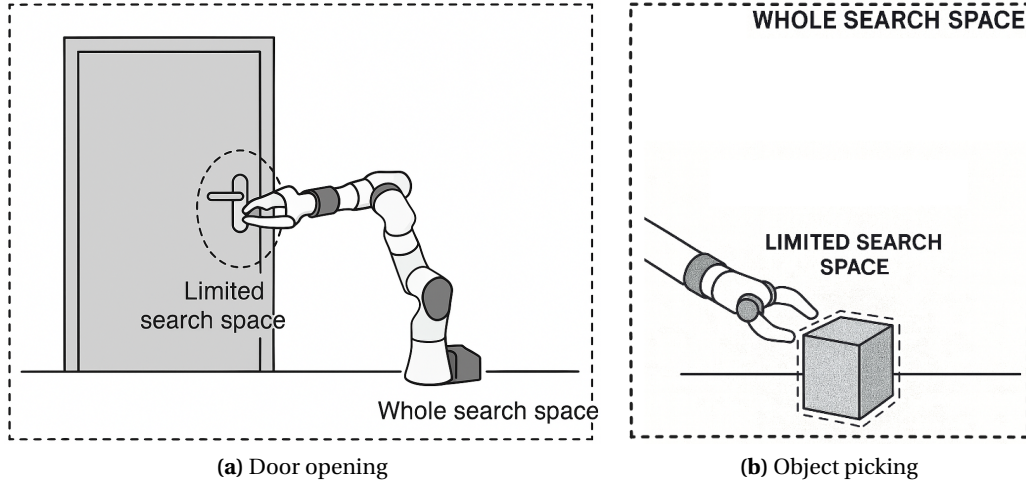


Figure 1.1: Constraining the robot to interact with the door handle (a) or the object (b) significantly limits the robot’s motion space around the handle or the object.

For instance, when manipulating an object, the robot may not know its weight distribution or friction properties. This partial observability requires the robot to make decisions with incomplete information. Finally, uncertainty is inherent (Jankowski et al., 2025; Rodriguez, 2021): unknown object properties, unmodeled dynamics, and sensor noise all add further complexity to planning and control.

Among the challenges outlined above, this thesis focuses mainly on the role of constraints. Constraints not only create difficulties on their own, but also influence many of the other challenges. For instance, stability conditions or contact constraints significantly restrict the solution space, reducing the chances of finding a feasible solution while introducing nonlinearities and local optima. However, constraints should not be seen only as obstacles. They can also serve as guiding principles: they reduce the search space, rule out irrelevant solutions, and direct exploration toward promising regions. For example, when a robot needs to interact with an object, its search can be limited to the local area around that object instead of the entire space. Similarly, as illustrated in Fig. 1.1a, when a robot is tasked with opening a door, constraining its interaction to the door handle drastically reduces the motion space. This focused approach not only can simplify the problem but also ensures that the robot’s exploration is aligned with the requirements of the task.

This thesis builds on this insight by developing methods that leverage feasibility constraints to guide exploration and adaptation in constrained robotic manipulation. The work focuses on enabling versatility across a range of tasks that require reasoning about physical interactions, stability, and safety. These include scenarios such as non-prehensile or contact-based manipulation as well as humanoid and human-assistance tasks, where maintaining balance, ensuring obstacle avoidance, and adapting to diverse constraints are essential.

1.2 General Problem Formulation

In general, robotic decision-making in motion planning and control can be described as an optimization problem. At each time step, the robot receives observations $\mathbf{y}_{t-L:t}$ from time $t-L$ to t , which are mapped to state estimates $\hat{\mathbf{x}}_t$. Depending on factors such as the quality of the mapping and the observability of the system, these estimates may differ from the true system state $\mathbf{x}_t$. The system dynamics $\mathbf{f}$ describe how states evolve given control inputs. With this in mind, the robot's goal is to find a sequence of control actions $\mathbf{u}_{t:t+H-1}^*$ over a horizon H that minimizes cumulative costs while satisfying a set of constraints. This problem can be formulated as

$$\begin{aligned} \mathbf{u}_{t:t+H-1}^* = \underset{\hat{\mathbf{u}}_{t:t+H-1}}{\operatorname{argmin}} \quad & \sum_{h=0}^{H-1} c_h(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}) + c_H(\hat{\mathbf{x}}_{t+H}) \\ \text{s.t.} \quad & \hat{\mathbf{x}}_{t+h+1} = \mathbf{f}(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}, \mathbf{w}_{t+h}), \quad \hat{\mathbf{x}}_t = \mathbf{z}(\mathbf{y}_{t-L:t}), \\ & \mathbf{h}(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}) = 0, \quad \mathbf{g}(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}) \leq 0, \quad \forall h = 0, \dots, H-1. \end{aligned} \tag{1.1}$$

Here:

- $\hat{\mathbf{u}}_{t:t+H-1} = [\hat{\mathbf{u}}_t, \dots, \hat{\mathbf{u}}_{t+H-1}]$ are the planned controls,
- $\hat{\mathbf{x}}_{t:t+H} = [\hat{\mathbf{x}}_t, \dots, \hat{\mathbf{x}}_{t+H}]$ are the predicted states,
- c_h and c_H denote stage and terminal costs,
- $\mathbf{f}$ is the (possibly nonlinear) system dynamics,
- $\mathbf{w}$ represents the stochastic (or noise) component in the dynamic equation,
- $\mathbf{z}$ maps observation histories to estimated states
- $\mathbf{h}$ and $\mathbf{g}$ represent equality and inequality constraints.

This formulation is highly general, and different research efforts in robotics have focused on different aspects of it. Each parameter or function in the formulation can itself be the subject of a dedicated line of research. For example, defining the state $\mathbf{x}$ and control $\mathbf{u}$ can be challenging in settings involving soft objects or when observations are obtained directly from vision (Ma et al., 2021; Yan et al., 2020). The mapping $\mathbf{z}$ is generally unknown, especially when the robot interacts with humans, and many works investigate learning this mapping from data using state estimation (Hartley et al., 2018; Talbot et al., 2024). The system dynamics $\mathbf{f}$ may also be unknown, or known but too complex to be suitable for online planning, leading to research on learning approximate dynamics models or using simulators (Deisenroth & Rasmussen, 2011; Orin et al., 2013). Similarly, the cost functions c may be hand-crafted or learned from data, with works in inverse reinforcement learning and inverse optimal control addressing this problem (Abbeel & Ng, 2004; Mombaur et al., 2010). Considering all these aspects is beyond

the scope of this thesis. In the following, we clarify our assumptions regarding this formulation and highlight the main focus of the thesis.

1.3 Simplified Problem Setting for this Thesis

To narrow the scope and focus on the core challenges of constrained robotic manipulation, we make the following simplifying assumptions:

- We assume direct access to the robot state ($\mathbf{x}_t = \mathbf{y}_t$).
- We assume access to either analytical dynamic models or simulators for the dynamics $\mathbf{f}$.
- We assume that the dynamic equation is deterministic.
- We assume that state/action representations are predefined. The cost functions c are also predefined, except in Chapter 6, where demonstrations implicitly represent it.

With these assumptions, the problem reduces to:

$$\begin{aligned}
 \mathbf{u}_{t:t+H-1}^* &= \underset{\hat{\mathbf{u}}_{0:H-1}}{\operatorname{argmin}} \quad \sum_{h=0}^{H-1} c_h(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}) + c_H(\hat{\mathbf{x}}_{t+H}) \\
 \text{s.t.} \quad &\hat{\mathbf{x}}_{t+h+1} = \mathbf{f}(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}), \\
 &\mathbf{h}(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}) = 0, \quad \mathbf{g}(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}) \leq 0, \quad \forall h = 0, \dots, H-1.
 \end{aligned} \tag{1.2}$$

This working formulation captures the essence of constrained motion planning: finding control sequences that optimize performance while respecting system dynamics and feasibility constraints. Nevertheless, even under these assumptions, the formulation remains general and does not by itself address the unique difficulties of robotics. These domain-specific challenges are discussed in the next section.

1.4 Thesis Challenges and Objectives

One of the core challenges in constrained robotic manipulation is the presence of uncertainty at multiple levels of the task. This uncertainty arises both in the dynamics—where the model $\mathbf{f}$ is often inaccurate despite extensive research on improving it (Aceituno-Cabezas & Rodriguez, 2020; Manchester & Kuindersma, 2020; Pang et al., 2023; Posa et al., 2014)—and in perception, where the robot lacks access to some unobservable states of the environment (Phiquepal & Toussaint, 2023). For example, when manipulating an object, the robot may not have full knowledge of its mass distribution or surface friction (T. Xue et al., 2024a). Such partial observability adds a significant layer of complexity to planning and control, since decisions must be made based on incomplete information.

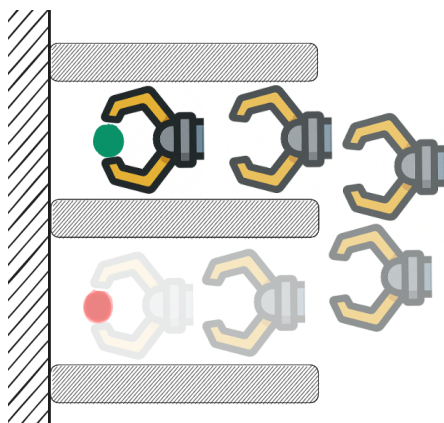


Figure 1.2: The presence of constraints (e.g., obstacles) can make the definition of cost non-intuitive. As shown, the robot needs to move away from the target at the beginning to avoid the obstacle, which is counter to the intuitive cost of minimizing distance to the target. Because of this, people may prefer to define the costs only at the goal, leading to sparse rewards, and long-horizon planning.

A common strategy to adapt to these uncertainties is to solve (1.2) repeatedly during execution (Liang et al., 2018; Pezzato et al., 2023; Phiquepal & Toussaint, 2023). This iterative re-optimization accounts for changes in the environment, limitations in observations, and the approximations made when reformulating the problem into a more tractable form. However, solving constrained optimization problems is known to be computationally hard in general (Ahmadi & Zhang, 2018; Papadimitriou, 1981; Tillmann & Pfetsch, 2014).

To improve the efficiency of these optimization problems, several challenges must be addressed, including nonlinearity, non-convexity, sparse rewards (see Fig. 1.2), long-horizon planning, and limited feasible solution spaces. The coexistence of these factors often renders constrained robotic manipulation intractable, or at least highly inefficient, under existing methods.

With these challenges in mind, the objective of this thesis is to answer the following question:

How can the constraints inherent in a manipulation problem be used not only as limitations, but also as informative structures that guide exploration and adaptation?

Rather than treating constraints purely as restrictions, this thesis investigates how they can provide geometric or logical structure that narrows the search space, directs the robot toward viable regions, and avoids unnecessary exploration. The aim is to transform constraints from a burden into a useful source of guidance for more efficient planning and control.

In the next section, we discuss some of the challenges in more detail and outline how the contributions of this thesis aim to address them by treating constraints not merely as limitations but also as guides for exploration and adaptation.

1.4.1 Slow Convergence

Optimization problems with linear dynamics or convex formulations benefit from efficient and reliable solvers (Alizadeh & Goldfarb, 2003; Karmarkar, 1984; Nesterov & Nemirovskii, 1994). For example, the Linear Quadratic Regulator (LQR) (Kalman, 1960) provides globally optimal solutions under linear dynamics and quadratic costs. More generally, convex optimization ensures that any local optimum is also a global optimum (Boyd & Vandenberghe, 2004), which allows interior-point or first-order methods to solve these problems effectively.

However, once the problem becomes non-convex and nonlinear, only local optimality can be achieved (Bertsekas, 1999). In such cases, iterative approaches such as iLQR (W. Li & Todorov, 2004) or Sequential Quadratic Programming (SQP) (Boggs & Tolle, 1995) are typically employed. This situation is common in robotics: obstacle avoidance and contact constraints are inherently non-convex, and physical interactions often involve underactuated systems that require explicit treatment of nonlinear dynamics and kinematic relations. These nonlinearities create many local minima, making solvers prone to suboptimal or even unsafe solutions. The issue is more severe in constrained problems, where constraints are often treated as penalties in the cost function (Han & Mangasarian, 1979; Nocedal & Wright, 2006). In such cases, feasibility is not guaranteed until the optimizer converges to a suitable local minimum, which can be slow and unreliable.

In this thesis, we address the challenge of slow convergence from several perspectives. Chapter 3 introduces task representations that remove certain constraints by construction, effectively turning them into unconstrained problems. Here, the feasibility constraint is used to define a manifold that guides the robot to remain within stable configurations. Chapter 4 exploits demonstrations to bias the optimizer toward promising regions of the solution space, reducing the number of iterations required. Finally, Chapter 5 proposes a feasibility expert that incorporates feasibility information before sampling, enabling the system to avoid unnecessary iterations and converge more efficiently to valid solutions.

1.4.2 Inefficient Sampling

A common strategy to accelerate optimization and mitigate the risk of local minima is to use sampling-based approaches (Bhardwaj et al., 2022; Kobilarov, 2012; Williams, Wagener, et al., 2017). However, when constraints are present, many samples turn out to be infeasible and must be discarded, making the process inefficient. Projection methods, such as RRT with tangent-space or nullspace projections (Kingston et al., 2018; Pardo et al., 2015; Suh et al., 2011), improve feasibility but remain limited: they are computationally expensive, capture only local constraint behavior, and often bias exploration toward boundaries.

An alternative is to model the feasible region directly. Recent generative approaches (Power & Berenson, 2023; Sacks & Boots, 2023) follow this idea, but they require learning complex joint distributions of feasibility and optimality, which can be data-intensive. In Chapter 5,

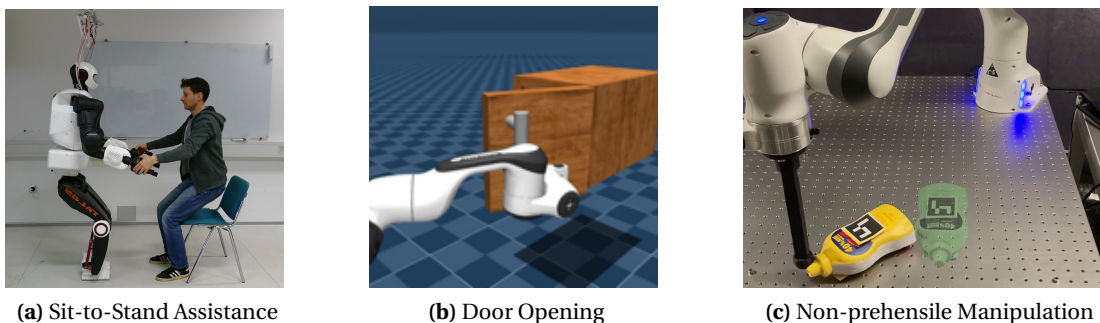


Figure 1.3: Example of experiments employed in this thesis.

we simplify the problem by learning only the feasibility distribution and then projecting the optimality distribution into it using a product-of-experts formulation. This *project-then-sample* strategy focuses exploration on feasible regions, improves efficiency, and generalizes across tasks with similar feasibility sets.

In Chapter 6, we further extend this idea by exploiting information from the robot's previous failed actions to refine the prior distribution, thereby increasing the likelihood of sampling feasible and effective actions.

1.4.3 Unstable Configuration

A critical challenge in constrained optimization arises when constraints correspond to physical safety requirements, such as robot stability. Many methods have been proposed to address these challenges (Dai et al., 2014; Kajita et al., 2003; Posa et al., 2014). In physical human-robot interaction, however, stability constraints become even more critical: if violated, the robot may enter unstable configurations that could lead to falls or injuries. Ensuring stability, therefore, requires careful control and strict constraint handling throughout the motion. For this reason, research in this domain often relies on low-dimensional representations (Shomin et al., 2015) and/or fully actuated robots (Geravand et al., 2017).

In Chapter 3, instead of using a low-dimensional and fully actuated robot, our goal was to perform a sit-to-stand task using a humanoid robot (see Fig. 1.3a). To address the challenges mentioned above, we proposed to extract a robust manifold of feasible configurations, where unstable states are removed by construction. By constraining the robot's motion to this manifold, stability is enforced implicitly, reducing the likelihood of unsafe behavior and ensuring that the robot operates within safer and more reliable regions of its configuration space.

1.4.4 Failure Recovery

In constrained environments, robot failures are more likely to occur due to unknown parameters, unmodeled dynamics, or estimation errors. When a failure happens, the robot must adapt

to the new situation rather than simply repeating the same actions, which could be dangerous or inefficient. Existing recovery approaches often rely on supplementary resources such as simulators (Lee et al., 2019), foundation models (Duan et al., 2024; Huang et al., 2022; Z. Liu et al., 2023), or expert guidance (Niekum et al., 2013). Two-level planning with hierarchical reinforcement learning or behavior cloning has also been explored (Triantafyllidis et al., 2023), but such methods increase complexity, suffer from suboptimal separation between planning levels, and face combinatorial explosion as task options grow. Moreover, in imitation learning settings, assumptions like access to models or simulators are often unrealistic.

In Chapter 6, we address recovery directly at the low-level controller by adapting the sampling distribution, so that the robot avoids repeating failed actions. Our approach enables recovery using only successful demonstrations, without requiring a simulator, explicit models, or high-level planners.

This is consistent with the broader theme of this thesis: using constraints as guidance. In this case, a constraint violation implicitly signals that the previous action was unsuccessful, prompting the robot to explore alternative strategies. For example, if the robot fails to open a door, it may recover by trying different contact points on the handle or applying force in a different direction.

1.5 Research Focus of this Thesis

To summarize our contributions, this thesis investigates how feasibility can be transformed from a restrictive constraint into a guiding principle for exploration and adaptation. The central idea is that feasibility constraints carve out a structured subset of the search space, which can be exploited to accelerate convergence, improve robustness, and enable recovery from failures. We validate this perspective through a diverse set of experiments, some of which are illustrated in Fig. 1.3.

Each contribution corresponds directly to one of the main challenges discussed in Section 1.4: configuration instability (Chapter 3), slow convergence (Chapter 4), inefficient sampling (Chapter 5), and failure recovery (Chapter 6).

Concretely, we develop and validate the following strategies:

- **Manifold-based feasibility** (Chapter 3): representing feasible robot configurations on low-dimensional robust manifolds to ensure stability and enable safe human–humanoid collaboration. Stability constraints are used to extract the manifold, restricting robot motion to this space and thereby focusing exploration on feasible regions.
- **Demonstration-guided optimization** (Chapter 4): using demonstrations, which inherently represent feasible and near-optimal solutions, to warm-start solvers and guide optimization with an additional gradient term. This improves convergence speed in

constrained optimization by directing the search toward regions with a higher likelihood of containing solutions.

- **Products-of-experts exploration** (Chapter 5): decomposing feasibility and optimality into separate distributions and recombining them, yielding efficient exploration and faster convergence in constrained spaces. This removes infeasible regions from the optimality distribution, ensuring that sampling is concentrated in feasible areas.
- **Failure-aware imitation learning** (Chapter 6): handling failures in imitation learning when only successful demonstrations are available, by dynamically refining sampling distributions to recover from failed attempts without requiring explicit models or cost functions. This final chapter emphasizes that failures are often unavoidable in constrained tasks and that robots, like humans, must be capable of recovering from them.

2 Background

This chapter provides the theoretical and methodological background necessary to understand the approaches developed in the subsequent chapters. It summarizes fundamental concepts in robot dynamics, optimal control, functional representations, tensor decomposition, imitation learning, and generative modeling. Each subsection corresponds to methods that are later used or extended in this thesis.

For transparency, parts of this chapter build upon material from my previously published papers and internal reports, where relevant methods were originally introduced. The content has been adapted and reorganized here to provide a coherent overview and to maintain consistency with the overall structure of the thesis.

2.1 Robot Dynamics and Humanoid Stability

This section recalls the elements of robot dynamics that are needed to reason about feasibility in humanoid control. Since many of the challenges addressed in this thesis arise from contacts, underactuation, or the need to keep the system within stable regions, understanding the dynamic equations is essential. These models define the boundaries of feasible motions and later support both the stability analysis and the constrained control strategies developed in the subsequent chapters.

The dynamics of a robot manipulator or humanoid robot can be described by the standard rigid-body equations of motion derived from the Euler–Lagrange formulation. For generalized coordinates $\mathbf{q} \in \mathbb{R}^{n_q}$, the equations of motion take the form

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}\boldsymbol{\tau} + \sum_i^{n_c} (\mathbf{J}^i)^\top(\mathbf{q}) \mathbf{f}_{\text{ext}}^i, \quad (2.1)$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n_q \times n_q}$ is the mass/inertia matrix, $\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{n_q}$ captures Coriolis and gravity effects, $\mathbf{S} \in \mathbb{R}^{n_q \times n_u}$ is the actuation map which will be identity matrix for fully actuated robots, $\boldsymbol{\tau} \in \mathbb{R}^{n_u}$ is the generalized force vector, $\mathbf{J}^i(\mathbf{q}) \in \mathbb{R}^{6 \times n_q}$ is the Jacobian at the i -th contact point,

and $\mathbf{f}_{\text{ext}}^i \in \mathbb{R}^6$ denotes the corresponding external forces and torques acting on the robot (e.g., ground reaction forces or human interaction).

For humanoid robots, these dynamics not only determine motion but also stability. Unlike fixed-base manipulators, humanoids are underactuated and rely on ground contact forces to maintain balance. Stability analysis is therefore often expressed in terms of the relation between the center of mass (CoM) and the support polygon, which is the convex hull of the foot contact points.

In a purely static situation where only the forces on the flat foot are considered, the robot remains balanced as long as its CoM projection falls within the support polygon. Yet, because real-world motion involves significant dynamic effects, more sophisticated stability measures have been developed.

Center of Pressure (CoP)

The *Center of Pressure* is the point location on the support surface where the resultant ground reaction force acts. It can be measured directly by force-torque sensors in the feet. While the CoP is measurable for each contact point, its interpretation is similar to that of the *Zero Moment Point (ZMP)*. In fact, the CoP remains the physically realized contact point of forces. This distinction makes CoP particularly relevant in experimental humanoid control, though in control design, the ZMP is often used due to its simpler analytical formulation.

Contact Wrench Cone (CWC)

Another approach to quantify stability is through the *Contact Wrench Cone (CWC)* (Hirukawa et al., 2006). Instead of reducing the stability condition to a point (like ZMP or CoP), CWC considers the entire feasible set of contact wrenches that the ground can exert on the robot. These wrenches, which consist of force and moment, are feasible if they lie inside the convex cone defined by the contact geometry and friction constraints. Stability is ensured if the resultant wrench required by the robot's dynamics lies inside this cone. Compared to ZMP, the CWC formulation is more general as it can handle arbitrary multi-contact scenarios and non-flat ground surfaces, but at the cost of increased computational complexity.

Zero Moment Point (ZMP)

The ZMP (Kajita et al., 2003) remains one of the most widely used criteria for humanoid stability. It is defined as the point on the ground plane where the net moment of inertial and gravitational forces has no component along the horizontal directions. In other words, the ground reaction force acts purely vertically at the ZMP.

If the ground reaction force is $\mathbf{f}_g = (f_x, f_y, f_z)$ and the net moment at the origin is $\boldsymbol{\tau}_g = (\tau_x, \tau_y, \tau_z)$,

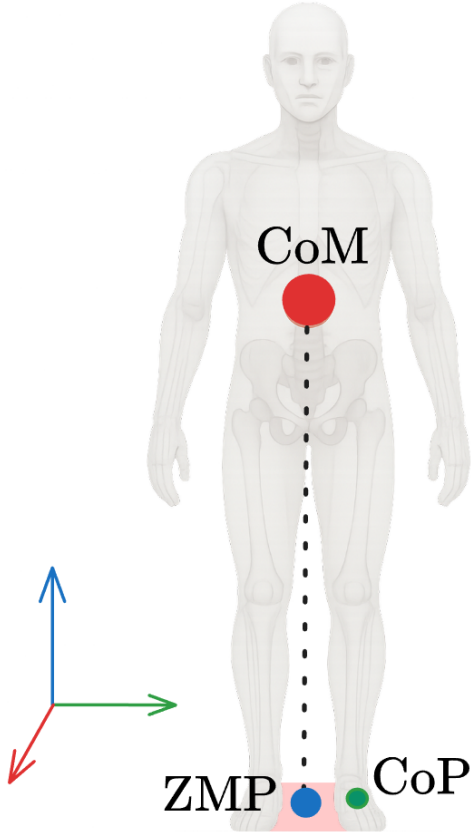


Figure 2.1: A humanoid's stability can be measured by different methods, as shown in the picture. The CoP is the point where the resultant ground reaction force acts and can be measured individually for each point; the ZMP is the point where the net moment of inertial and gravitational forces has no component along the horizontal directions, and in the static case, this point is the projection of CoM on the ground. The support polygon is shown in the pink shaded area.

then the ZMP coordinates (x_{ZMP}, y_{ZMP}) are given by

$$x_{ZMP} = \frac{-\tau_y}{f_z}, \quad y_{ZMP} = \frac{\tau_x}{f_z}. \quad (2.2)$$

If (x_{ZMP}, y_{ZMP}) lies *inside* the support polygon, the humanoid is dynamically stable, alternatively, if (x_{ZMP}, y_{ZMP}) lies *outside* the support polygon, the humanoid will lose balance unless corrective actions such as stepping, hip, or ankle strategies are taken.

In summary, CoP provides a measurable physical indicator, CWC gives a general feasibility condition in multi-contact cases, and ZMP offers a practical and computationally efficient criterion. For this reason, ZMP has been widely adopted for humanoid walking, sit-to-stand transfers, and other tasks requiring dynamic stability. A simple illustration of these stability measures is shown in Fig. 2.1.

2.2 Optimal Control Approaches

Optimal control provides a structured way to choose actions that respect system dynamics and constraints while optimizing a performance measure. In this thesis, optimal control is not used only as a classical trajectory optimization tool, but as a conceptual basis for sampling-

based model predictive control, feasibility shaping, and distribution-guided exploration. This section introduces the fundamental optimal control formulations that later support the feasibility-guided sampling strategies and the product-of-experts models presented in the thesis.

The primary aim of this methodology is to tackle an optimal control problem associated with a dynamic system described by $\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t)$ within a receding horizon H . Here, $\mathbf{x}_t \in \mathbb{R}^{n_x}$ and $\mathbf{u}_t \in \mathbb{R}^{n_u}$ denote the system state and action at time t , respectively. This objective is achieved by minimizing a predetermined cost function as

$$c(\hat{\mathbf{X}}_t, \hat{\mathbf{U}}_t) = \sum_{h=0}^{H-1} c_h(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}) + c_H(\hat{\mathbf{x}}_{t+H}). \quad (2.3)$$

Here, $\hat{\mathbf{X}}_t = [\hat{\mathbf{x}}_t^\top, \hat{\mathbf{x}}_{t+1}^\top, \dots, \hat{\mathbf{x}}_{t+H}^\top]^\top \in \mathbb{R}^{(H+1)n_x}$ represents the state trajectory, and the sequence of action commands is denoted as $\hat{\mathbf{U}}_t = [\hat{\mathbf{u}}_t^\top, \hat{\mathbf{u}}_{t+1}^\top, \dots, \hat{\mathbf{u}}_{t+H-1}^\top]^\top \in \mathbb{R}^{Hn_u}$. Additionally, c_h and c_H denote stage and terminal cost values, respectively. In many practical settings, constraints are incorporated into the cost function as soft constraints, for example, through penalty terms or augmented Lagrangian methods.

The objective is to determine a set of optimal action commands $\mathbf{U}_t^*$ that minimizes this cost function while taking into account the dynamics of the system, specifically

$$\mathbf{U}_t^* = \underset{\hat{\mathbf{U}}_t}{\operatorname{argmin}} c(\hat{\mathbf{X}}_t, \hat{\mathbf{U}}_t). \quad (2.4)$$

In this thesis, we primarily employ a shooting-based formulation, where the optimization is performed only with respect to the control variables. This choice offers a simpler problem structure and a lower-dimensional optimization space. Nevertheless, the concepts and results presented in this thesis can, without loss of generality, be extended to other optimal control formulations such as multiple shooting or direct collocation.

Since robotic systems are often nonlinear, high-dimensional, and constrained by kinematics or contacts, solving (2.4) exactly is generally infeasible. Instead, approximate solutions are employed. These methods can be broadly divided into two categories: *derivative-based* and *sampling-based* approaches (Todorov & Jordan, 2002; Williams, Wagener, et al., 2017). Derivative-based approaches exploit system derivatives, relying on first-order gradients (e.g., gradient descent) or higher-order information such as Hessians or pseudo-Hessians (e.g., Newton or Gauss–Newton methods) to refine candidate solutions. In contrast, sampling-based approaches can be regarded as zero-order optimization methods, as they do not require any derivative information.

2.2.1 Derivative-based Optimal Control

Derivative-based approaches exploit the derivatives of the dynamics and cost function to iteratively refine a candidate trajectory. Classical methods include Differential Dynamic Programming (DDP) (Tassa et al., 2012) and the Iterative Linear Quadratic Regulator (iLQR) (W. Li & Todorov, 2004). These methods rely on locally linearizing the dynamics and quadratizing the cost, allowing each iteration to be solved efficiently as an LQR problem.

Such approaches are computationally efficient when smooth derivatives are available, and they have been successfully applied to locomotion, manipulation, and model predictive control. However, they may converge only to local minima and encounter difficulties when gradients are undefined or discontinuous, as in contact-rich manipulation (Pang et al., 2023).

In this thesis, we adopted the batch version of iterative Linear Quadratic Regulator (**iLQR**) in the context of human-robot assistance (Razmjoo et al., 2021). iLQR is a second-order optimization approach which exploits Taylor expansion to quadratize the cost function c and to linearize the system's dynamics $\mathbf{f}$ around their current values w.r.t. the system variables. Then, at each iteration, it solves an LQR problem and updates the current solution. This process continues until it reaches a local optimum.

At iteration k , the system dynamics $\mathbf{x}_{t+1}^k = \mathbf{f}(\mathbf{x}_t^k, \mathbf{u}_t^k)$ can be linearly approximated around the previous nominal solution $\mathbf{x}^{k-1}$ and $\mathbf{u}^{k-1}$ as

$$\begin{aligned} \mathbf{x}_{t+1}^k &\approx \mathbf{x}_{t+1}^{k-1} + \frac{\partial \mathbf{f}}{\partial \mathbf{x}_t^{k-1}} (\mathbf{x}_t^k - \mathbf{x}_t^{k-1}) + \frac{\partial \mathbf{f}}{\partial \mathbf{u}_t^{k-1}} (\mathbf{u}_t^k - \mathbf{u}_t^{k-1}) \\ &\Rightarrow \Delta \mathbf{x}_{t+1} \approx \mathbf{A}_t \Delta \mathbf{x}_t + \mathbf{B}_t \Delta \mathbf{u}_t, \end{aligned} \quad (2.5)$$

where $\Delta \mathbf{x} = \mathbf{x}^k - \mathbf{x}^{k-1}$, $\Delta \mathbf{u} = \mathbf{u}^k - \mathbf{u}^{k-1}$, and $\mathbf{A}_t \in \mathbb{R}^{n_x \times n_x}$ and $\mathbf{B}_t \in \mathbb{R}^{n_x \times n_u}$ are the system and input matrices, respectively. This equation formulates the change of the states at each time as a linear function of the change of the inputs and states at the previous time. With this recursive formula, we can formulate the deviations of all the states as

$$\underbrace{\begin{bmatrix} \Delta \mathbf{x}_1 \\ \Delta \mathbf{x}_2 \\ \vdots \\ \Delta \mathbf{x}_H \end{bmatrix}}_{\Delta \mathbf{X}} = \underbrace{\begin{bmatrix} \mathbf{A}_0 \\ \mathbf{A}_1 \mathbf{A}_0 \\ \vdots \\ \prod_{\tau=0}^{H-1} \mathbf{A}_{H-\tau} \end{bmatrix}}_{\mathbf{S}_x} \Delta \mathbf{x}_0 + \underbrace{\begin{bmatrix} \mathbf{B}_0 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A}_1 \mathbf{B}_0 & \mathbf{B}_1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \prod_{\tau=0}^{H-2} \mathbf{A}_{H-\tau} \mathbf{B}_0 & \prod_{\tau=0}^{H-3} \mathbf{A}_{H-\tau} \mathbf{B}_1 & \cdots & \mathbf{B}_{H-1} \end{bmatrix}}_{\mathbf{S}_u} \underbrace{\begin{bmatrix} \Delta \mathbf{u}_0 \\ \Delta \mathbf{u}_1 \\ \vdots \\ \Delta \mathbf{u}_{H-1} \end{bmatrix}}_{\Delta \mathbf{U}}, \quad (2.6)$$

where $\mathbf{S}_u \in \mathbb{R}^{Hn_x \times (H-1)n_u}$ and $\mathbf{S}_x \in \mathbb{R}^{Hn_x \times n_x}$ are transformation matrices that map $\Delta \mathbf{U}$ and $\Delta \mathbf{x}_1$ to $\Delta \mathbf{X}$, respectively. Since the initial states of the system are known ($\Delta \mathbf{x}_1 = 0$), we can write

$$\Delta \mathbf{X} = \mathbf{S}_u \Delta \mathbf{U}, \quad (2.7)$$

where $\Delta \mathbf{U}$ and $\Delta \mathbf{X}$ are the concatenated vectors of variations of all the input and the state

variables, respectively.

The cost function c is a nonlinear function of all the states and inputs, so its quadratic version can be defined as

$$c(\mathbf{X}^k, \mathbf{U}^k) \approx c(\mathbf{X}^{k-1}, \mathbf{U}^{k-1}) + \Delta \mathbf{X}^\top \mathbf{g}_x + \Delta \mathbf{U}^\top \mathbf{g}_u + \frac{1}{2} \Delta \mathbf{X}^\top \mathbf{H}_x \Delta \mathbf{X} + \frac{1}{2} \Delta \mathbf{U}^\top \mathbf{H}_u \Delta \mathbf{U} + \Delta \mathbf{X}^\top \mathbf{H}_{xu} \Delta \mathbf{U}, \quad (2.8)$$

where

$$\mathbf{g}_x = \left. \frac{\partial c}{\partial \mathbf{X}} \right|_{\mathbf{X}=\mathbf{X}^k}, \quad \mathbf{g}_u = \left. \frac{\partial c}{\partial \mathbf{U}} \right|_{\mathbf{U}=\mathbf{U}^k}, \quad (2.9a)$$

$$\mathbf{H}_x = \left. \frac{\partial^2 c}{\partial \mathbf{X}^2} \right|_{\mathbf{X}=\mathbf{X}^k}, \quad \mathbf{H}_u = \left. \frac{\partial^2 c}{\partial \mathbf{U}^2} \right|_{\mathbf{U}=\mathbf{U}^k}, \quad \mathbf{H}_{xu} = \left. \frac{\partial^2 c}{\partial \mathbf{X} \partial \mathbf{U}} \right|_{\mathbf{X}=\mathbf{X}^k, \mathbf{U}=\mathbf{U}^k}. \quad (2.9b)$$

The value of $\Delta \mathbf{X}$ in (2.8) can be replaced from (2.7), so the quadratized cost would be a function of only $\Delta \mathbf{U}$. The optimizer $\Delta \mathbf{U}^*$ of (2.8) can be found by putting its gradient equal to zero, which would result in

$$\Delta \mathbf{U}^* = (\mathbf{S}_u^\top \mathbf{H}_u \mathbf{S}_u + 2 \mathbf{S}_u^\top \mathbf{H}_{xu} + \mathbf{H}_u)^{-1} (-\mathbf{S}_u \mathbf{g}_x - \mathbf{g}_u). \quad (2.10)$$

The last step only works when there is no hard constraint, so we define the constraints in this experiment as a part of the cost function and heavily penalize the system when it violates them.

After computing the optimal control updates, the control inputs are revised and the corresponding state trajectory is recalculated. The control inputs are updated as

$$\mathbf{u}_t^k = \mathbf{u}_t^{k-1} + \Delta \mathbf{u}_t, \quad (2.11)$$

and the states are updated in a forward pass as

$$\mathbf{x}_{t+1}^k = \mathbf{f}(\mathbf{x}_t^k, \mathbf{u}_t^k). \quad (2.12)$$

This procedure is repeated until the termination criterion is satisfied (e.g., $\|\Delta \mathbf{U}\| < \epsilon$).

2.2.2 Sampling-based Optimal Control

Sampling-based approaches avoid explicit differentiation. Instead, they generate many trajectory rollouts under stochastic perturbations and update the control sequence by re-weighting the sampled actions according to their costs. Examples include the Cross-Entropy Method (CEM) (Kobilarov, 2012) and the Model Predictive Path Integral (MPPI) (Williams, Wagener, et al., 2017).

These approaches are particularly attractive for tasks with discontinuous or non-smooth

dynamics, where derivative-based methods struggle. They are also naturally parallelizable, making them suitable for GPU-based acceleration (Liang et al., 2018). However, they require a large number of samples, and their efficiency depends heavily on how the sampling distribution is constructed. In constrained tasks, many samples may be rejected, reducing effectiveness (Bhardwaj et al., 2022).

In this thesis, we employed **MPPI** as a representative sampling-based method (Williams, Wagener, et al., 2017). Works by Williams et al. (Williams, Aldrich, & Theodorou, 2017; Williams, Wagener, et al., 2017) demonstrated that the solution for (2.3) can be effectively approximated using a forward approach, diverging from the conventional dynamic programming method. This discovery opened avenues for leveraging Monte Carlo estimation in approximating optimal actions through sampling-based methods. We can represent the probability of the system action commands as a Gaussian distribution, such as

$$\pi_{\theta}(\hat{\mathbf{U}}_t) = \prod_{h=0}^{H-1} \pi_{\theta_h}(\hat{\mathbf{u}}_{t+h}) = \prod_{h=0}^{H-1} \mathcal{N}(\hat{\mathbf{u}}_{t+h} | \boldsymbol{\mu}_{t+h}, \boldsymbol{\Sigma}_{t+h}). \quad (2.13)$$

Our goal is to sample N action sequences $\{\mathbf{U}_t^{(1)}, \mathbf{U}_t^{(2)}, \dots, \mathbf{U}_t^{(N)}\}$ from (2.13). These samples can be generated from the previous probability estimate such that $\mathbf{u}_{t+h}^{(i)} \sim \mathcal{N}(\hat{\mathbf{u}}_{t+h} | \hat{\boldsymbol{\mu}}_{t+h}, \hat{\boldsymbol{\Sigma}}_{t+h})$. Wagener et al., 2019 showed that in this setting, the parameters, represented by the mean of the Gaussian distribution $\boldsymbol{\mu}_{t+h}$, can be updated using these N sampled action sequences as

$$\boldsymbol{\mu}_{t+h} = (1 - \gamma_t) \hat{\boldsymbol{\mu}}_{t+h} + \gamma_t \sum_{i=1}^N w_i \hat{\mathbf{u}}_{t+h}^{(i)}, \quad (2.14)$$

where $\hat{\boldsymbol{\mu}}_t$ represents the preceding mean value, and γ_t is the step size. The term w_i is a weighting factor expressed as

$$w_i = \frac{e^{-\frac{1}{\beta} c(\hat{\mathbf{X}}_t^{(i)}, \hat{\mathbf{U}}_t^{(i)})}}{\sum_{j=0}^N e^{-\frac{1}{\beta} c(\hat{\mathbf{X}}_t^{(j)}, \hat{\mathbf{U}}_t^{(j)})}}, \quad (2.15)$$

where β is a temperature parameter. Following the adjustment of the parameters, the initial action $\mathbf{u}_t^*$ can be sampled, applied to the system, and iteratively repeated until the task is successfully completed.

2.3 Functional Representation

Many of the approaches developed in this thesis rely on representing motions in compact and structured ways. Functional representations, such as basis function expansions or movement primitives, allow trajectories to be encoded in low-dimensional parameter spaces while preserving smoothness and stability. These reduced representations are important for building manifolds and embedding smoothness, while reducing the task dimensionality. This section provides the necessary background on these representations and their role in simplifying

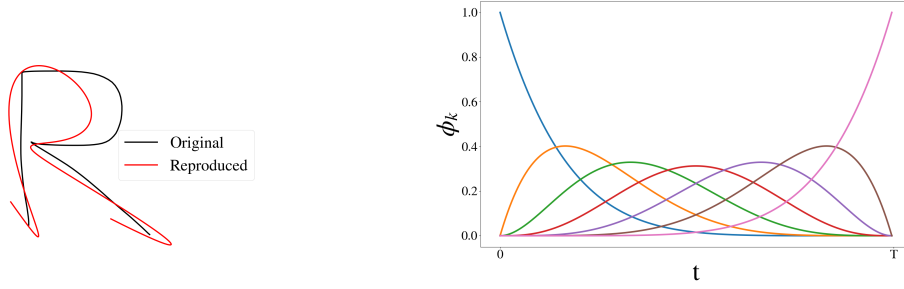


Figure 2.2: An illustrative example of using Bernstein polynomials as basis functions to model a 2D trajectory. The trajectory is represented by 7 basis functions, and the coefficients are optimized to fit the letter "R". The right figure shows the basis functions. These images are generated using the code from <https://rcfs.ch/>.

constrained motion generation.

Parameterizing a trajectory as a functional representation, such as motion primitives, has become a widely used approach in the field of Learning from Demonstration (LfD), as evidenced by the popularity of methods such as Dynamical Movement Primitives (DMP) (Ijspeert et al., 2013), Gaussian Mixture Models (GMM/GMR) (Calinon et al., 2007), or Probabilistic Movement Primitives (ProMP) (Paraschos et al., 2013), see (Calinon, 2019) for an overview of using basis functions for motion parameterization. This technique can reduce the number of parameters to describe a movement while ensuring its smoothness, which is often difficult to achieve otherwise. Due to these advantages, they have also proven to be useful in optimal control problem formulation (Razmjoo et al., 2021).

Consider a d -dimensional variable $\mathbf{q}(s) \in \mathbb{R}^d$, where $s \in [0, 1]$ is a mapping variable that indicates the relative position of the point on the 1-D manifold. It is assumed that each element $q_i(s)$ of $\mathbf{q}(s)$ where $i \in \{1, 2, \dots, d\}$ can be expressed as a weighted combination of K basis functions $q_i(s) = \boldsymbol{\phi}(s) \mathbf{w}_i$, where $\boldsymbol{\phi}(s) \in \mathbb{R}^{1 \times K}$ is the vector of basis functions at the location s , and $\mathbf{w}_i$ is the corresponding weighting vector.

By concatenating all dimensions, we can obtain a similar formulation $\mathbf{q}(s) = \boldsymbol{\Psi}(s) \mathbf{w}$ with $\boldsymbol{\Psi}(s) = \boldsymbol{\phi}(s) \otimes \mathbf{C}$, where $\otimes$ is the Kronecker product, $\mathbf{C} \in \mathbb{R}^{d \times r}$ is a coordination matrix that defines r independent variables to represent a d dimensional space, and $\mathbf{w} \in \mathbb{R}^K$ is the concatenated weighting vector. Often, $\mathbf{C} = \mathbf{I}_d$ is used as a coordination matrix to decouple the d dimensions. However, alternative selections can be made to establish correlations among the variables.

There are several ways to define the vector of basis functions, such as Radial Basis Functions (RBFs) or Bernstein polynomials (Farouki, 2012). The choice of basis functions directly affects the structure of the matrix $\boldsymbol{\Psi}$.

By selecting the number of basis functions K , k -th column of Ψ for RBFs can be defined as

$$\phi_k = \exp\left(-\lambda_k(\mathbf{s} - \mathbf{1}_T \mu_k) \odot (\mathbf{s} - \mathbf{1}_T \mu_k)\right), \quad (2.16)$$

where $\mathbf{s} = [s_1, s_2, \dots, s_T]^\top$ is a vector of T discrete points uniformly distributed over the interval $[0, 1]$, $\mathbf{1}_T$ is a vector of ones of length T , and $\odot$ denotes element-wise multiplication. Here, λ_k and μ_k represent the k -th bandwidth and center parameters, respectively.

For Bernstein polynomials, the basis vectors are defined as

$$\phi_k = \frac{(K-1)!}{(k-1)!(K-k)!} (\mathbf{1}_T - \mathbf{s})^{K-k} \odot \mathbf{s}^{k-1}, \quad (2.17)$$

Once Ψ is defined, the weight vector $\mathbf{w}$ can be estimated through linear regression techniques. Figure 2.2 presents an example in which Bernstein polynomials serve as basis functions for modeling a 2D trajectory.

This representation is later employed in Chapter 3 to parameterize a low-dimensional manifold for humanoid motion generation, enabling compact and smooth control of joint trajectories, and in Chapter 4 to reduce the dimensionality of the optimal control problem while improving the smoothness of the resulting motions.

2.4 Tensor Train Decomposition

Several contributions of this thesis rely on handling high-dimensional functions or distributions that arise in constrained exploration and adaptation. Tensor decomposition, and in particular the Tensor Train (TT) format, provides an efficient way to represent such functions while keeping computation tractable. TT's separable structure enables fast sampling, combination of distributions, and feasibility-driven reshaping of policies. This section introduces the TT representation and explains why it is suitable for the tensor-based models developed later in the thesis.

A d -th order tensor, denoted as $\mathcal{P} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, serves as an extension of vectors (1-D) and matrices (2-D) to higher dimensions (d -D). Its configuration is defined by a tuple of integers $\mathbf{n} = (n_1, \dots, n_d)$. Each element in the tensor, denoted as $\mathcal{P}_{\mathbf{i}}$, is identified by $\mathbf{i} = (i_1, \dots, i_d)$ as a set of d integer variables representing the index in each dimension. The index set of the tensor is expressed as $\mathcal{I} = \{\mathbf{i} = (i_1, \dots, i_d) : i_k \in \{1, \dots, n_k\}, k \in \{1, \dots, d\}\}$.

We can express a function, denoted as $pfunc : \Omega_{\mathbf{x}} \subset \mathbb{R}^d \rightarrow \mathbb{R}$, as a tensor by assuming a rectangular domain for the function, such as $\Omega_{\mathbf{x}} = \Omega_{x_1} \times \Omega_{x_2} \times \dots \times \Omega_{x_d} = \times_{k=1}^d \Omega_{x_k}$. This involves discretizing each input interval $\Omega_{x_k} \subset \mathbb{R}$ into n_k elements. The continuous set $\mathcal{X} = \{\mathbf{x} = (x^{i_1}, \dots, x^{i_d}) : x^{i_k} \in \Omega_{x_k}, i_k \in 1, \dots, n_k\}$ can be easily mapped to the corresponding index set $\mathcal{I}_{\mathcal{X}} = \{\mathbf{i} = (i_1, \dots, i_d)\}$. The $\mathbf{i}$ -th element in the tensor $\mathcal{P}$, denoted by $\mathcal{P}_{\mathbf{i}}$, corresponds to the function value with continuous variables, such that $\mathcal{P}_{\mathbf{i}} = pfunc(\mathbf{x})$. We overload the term-

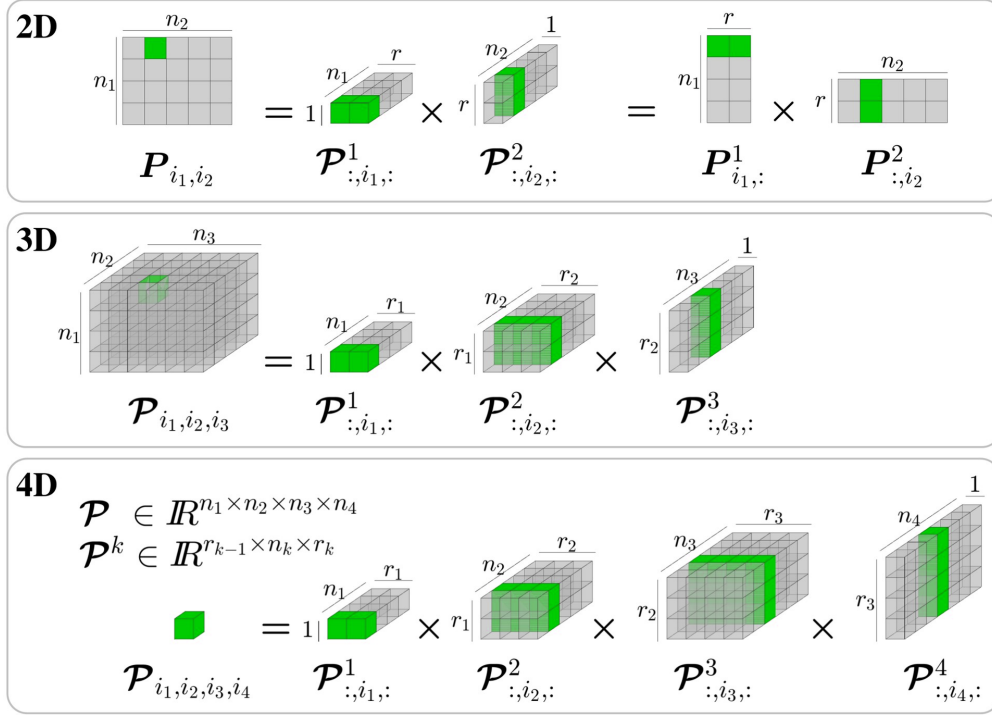


Figure 2.3: TT decomposition extends matrix decomposition techniques to higher-dimensional arrays. In the TT format, accessing an element of a tensor involves multiplying the chosen slices (represented by green-colored matrices) of the core tensors (factors). The illustration provides examples for a 2nd-order, 3rd-order, and a 4th-order tensor. This picture is adopted from (Shetty, Lembono, et al., 2024).

nology and define $\mathcal{P}_{\mathbf{x}} = \mathcal{P}_{\mathbf{i}}$ where $\mathbf{i}$ is the corresponding index of $\mathbf{x}$. Interpolation techniques between specific nodes of the tensor $\mathcal{P}$ can be used to approximate the value $pfunc(\mathbf{x})$ for any $\mathbf{x} \in \Omega_{\mathbf{x}}$.

To represent a tensor efficiently for a high-dimensional function, we employ a tensor factorization method that enhances storage efficiency by utilizing factors with a reduced number of elements. Among various methodologies, we adopt the Tensor Train (TT) decomposition approach (Oseledets, 2011) in this study, which utilizes a set of third-order tensors known as *TT-cores* to compactly encode a given tensor. In the TT format, a d -th order tensor $\mathcal{P} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ is expressed using a tuple of d third-order tensors ($\mathcal{P}^1, \dots, \mathcal{P}^d$). The dimensions of the TT-cores are denoted as $\mathcal{P}^k \in \mathbb{R}^{r_{k-1} \times n_k \times r_k}$ with $r_0 = r_d = 1$. As illustrated in Fig. 2.3, the computation of the $\mathbf{i}$ -th element of the tensor in this format involves a straightforward multiplication of matrix slices obtained from the TT-cores as

$$\mathcal{P}_{\mathbf{i}} = \mathcal{P}^1_{:, i_1, :} \mathcal{P}^2_{:, i_2, :} \dots \mathcal{P}^d_{:, i_d, :}, \quad (2.18)$$

where $\mathcal{P}^k_{:, i_k, :} \in \mathbb{R}^{r_{k-1} \times r_k}$ denotes the i_k -th matrix slice of the third-order tensor $\mathcal{P}^k$. The multiplication of these TT-cores yields a scalar variable. The determination of these cores, which have been demonstrated to exist for any given tensor consistently (Oseledets, 2011), can be achieved

through various methods, such as TT-SVD (Oseledets, 2011) and TT-cross (Savostyanov & Oseledets, 2011).

The *TT-rank* of the tensor in the TT representation is defined by the tuple $\mathbf{r} = (r_1, r_2, \dots, r_{d-1})$. We refer to $r = \max(r_1, \dots, r_{d-1})$ as the maximal rank. Similar to the matrix analogy, a smaller r leads to a less precise approximation, while a larger r necessitates greater storage capacity.

Utilizing this methodology, the ultimate outcome is the approximation of the continuous function $pfunc(\cdot)$, expressed as

$$pfunc(x_1, \dots, x_d) \approx \mathbf{P}^1(x_1) \cdots \mathbf{P}^d(x_d). \quad (2.19)$$

Here, $\mathbf{P}^k(x_k)$ with $k \in \{1, \dots, d\}$ is derived by interpolating within each of the cores. Employing advanced interpolation techniques enables a coarser discretization, thereby reducing storage requirements during the learning phase.

2.4.1 Tensor-Train Distribution

The TT approximation technique has been previously applied to optimization problems (Batsheva et al., 2023; Shetty, Lembono, et al., 2024). In this approach, the problem of minimizing a cost function is first transformed into the task of maximizing an unnormalized probability density function $pfunc(\cdot)$, represented as a tensor $\mathcal{P}$ in TT format. Subsequently, a corresponding probability distribution, referred to as the TT distribution, is constructed as

$$p(\mathbf{x}) = \frac{|\mathcal{P}_{\mathbf{x}}|}{Z}, \quad \mathbf{x} \in \Omega_{\mathbf{x}}, \quad (2.20)$$

where Z denotes the corresponding normalization constant. Leveraging the separable structure of the TT model allows for the efficient generation of exact samples from the TT distribution, eliminating the need to compute the normalization factor Z . Additional details about the sampling process can be found in Sec. 2.4.3.

2.4.2 Conditional TT

Let assume that $\mathbf{x}$ be composed of two parts as $\mathbf{x} = (\mathbf{y}_1, \mathbf{y}_2) \in \Omega_{\mathbf{x}}$, where $\mathbf{y}_1$ and $\mathbf{y}_2$ refers to the first d_1 and the last $d - d_1$ variables in $\mathbf{x}$, respectively. Our goal is to find $p(\mathbf{y}_2|\mathbf{y}_1)$ of the TT distribution given in (2.20).

For a fixed value $\mathbf{y}_1 = \mathbf{y}_t$, we define a new TT model $\mathcal{P}^{\mathbf{y}_t}$ over $\mathbf{y}_2$ by slicing the original cores:

$$\mathcal{P}_{\mathbf{y}_2}^{\mathbf{y}_t} = \mathcal{P}(\mathbf{y}_t, \mathbf{y}_2) \quad \forall \mathbf{y}_2 \in \Omega_{\mathbf{y}_2}.$$

Equivalently, its cores become

$$(\mathcal{P}^{y_t})^k = \begin{cases} \mathcal{P}_{:,y_{t_k},:,}^k & k = 1, \dots, d_1, \\ \mathcal{P}^k, & k = d_1 + 1, \dots, d. \end{cases} \quad (2.21)$$

The conditional probability then follows as

$$p(\mathbf{y}_2 | \mathbf{y}_1 = \mathbf{y}_t) = \frac{|\mathcal{P}_{\mathbf{y}_2}^{y_t}|}{Z_1} \quad \forall \mathbf{y}_2 \in \Omega_{\mathbf{y}_2}. \quad (2.22)$$

Finally, one can draw samples of $\mathbf{y}_2$ given $\mathbf{y}_1 = \mathbf{y}_t$.

2.4.3 TT-Sample

We demonstrate the application of TT-cores in the process of generating new data samples. For the sake of clarity, we omit the notation of Z , i.e. $Z = 1$, as it is not necessary for sampling from the specified distribution, thereby $p(\mathbf{x}) = |\mathcal{P}_{\mathbf{x}}|$, $\mathbf{x} \in \Omega_{\mathbf{x}}$. Using a chain of conditional distributions, we can define

$$p(x_1, \dots, x_d) = p_1(x_1) p_2(x_2 | x_1) \cdots p_d(x_d | x_1, \dots, x_{d-1}),$$

where

$$p_k(x_k | x_1, \dots, x_{k-1}) = \frac{m_k(x_1, \dots, x_k)}{m_{k-1}(x_1, \dots, x_{k-1})}$$

is the conditional distribution defined using the marginals

$$m_k(x_1, \dots, x_k) = \sum_{x_{k+1}} \cdots \sum_{x_d} p(x_1, \dots, x_d),$$

with the assumption that $m_0 = 1$. Utilizing the definitions outlined above, we can proceed to generate samples $\mathbf{x} \sim p(\cdot)$ by sampling from each conditional distribution sequentially. Each conditional distribution pertains solely to one variable, and in the discrete scenario, it can be interpreted as a multinomial distribution, wherein

$$x_k \sim p_k(x_k | x_1, \dots, x_{k-1}), \quad \forall k \in \{1, \dots, d\}.$$

This approach entails significant computational demands, as sampling x_k necessitates accessing the conditional distribution p_k , which, in turn, mandates evaluating summations across multiple variables to ascertain the marginal m_k , which escalates exponentially with the dimensionality of the problem. The function separability, facilitated by the TT-model, can be leveraged to address this concern. If the TT model is composed of cores $(\mathcal{P}^1, \dots, \mathcal{P}^d)$, then we

can express

$$\begin{aligned}
m_k(x_1, \dots, x_k) &= \sum_{x_{k+1}} \cdots \sum_{x_d} p(\mathbf{x}), \\
&\approx \sum_{x_{k+1}} \cdots \sum_{x_d} \mathcal{P}_{\mathbf{x}}, \\
&= \sum_{x_{k+1}} \cdots \sum_{x_d} \mathcal{P}_{:,x_1,:}^1 \cdots \mathcal{P}_{:,x_k,:}^k \mathcal{P}_{:,x_{k+1},:}^{k+1} \cdots \mathcal{P}_{:,x_d,:}^d, \\
&= \mathcal{P}_{:,x_1,:}^1 \cdots \mathcal{P}_{:,x_k,:}^k \left(\sum_{x_{k+1}} \mathcal{P}_{:,x_{k+1},:}^{k+1} \right) \cdots \left(\sum_{x_d} \mathcal{P}_{:,x_d,:}^d \right),
\end{aligned} \tag{2.23}$$

where $\sum_{x_k} \mathcal{P}_{:,x_k,:}^k$ is the aggregation of all matrix slices within the TT-cores. Consequently, the TT format simplifies the expensive summation into one-dimensional summations. In cases where identical summation terms recur across multiple conditionals p_k , an algorithm known as Tensor Train Conditional Distribution (TT-CD) sampling (Dolgov et al., 2020) can be employed to efficiently obtain samples from $p(\cdot)$.

The tensor-train framework introduced here forms the foundation for the feasibility representation and sampling strategy developed in Chapter 5.

2.5 Imitation Learning

Imitation learning (IL) is used throughout this thesis as a way to capture feasible behaviors demonstrated by experts. These demonstrations provide valuable priors for exploration, define distributions that reflect task constraints, and supply the structure required for later generative or sampling-based methods. This section introduces the key concepts of imitation learning that are relevant for deriving feasible motion distributions and guiding exploration according to expert behavior.

IL allows robots to acquire skills directly from demonstrations, avoiding the need for explicit system modeling or hand-crafted cost functions. Over the years, IL has evolved from functional representation methods (Ijspeert et al., 2013) to sophisticated generative models (Chi et al., 2024; Florence et al., 2022; Zhang & Gienger, 2024) that can capture complex, multimodal distributions of expert behaviors. The current state of the art in this field (Florence et al., 2022) shows that implicit representations are a key factor in achieving strong performance due to their ability to naturally model multimodality.

Recent efforts to apply energy-based models (LeCun et al., 2006) for modeling multi-modality have relied on adversarial training and sampling methods such as Langevin dynamics, which are computationally expensive. Diffusion models (Ho et al., 2020) were later proposed as a more efficient way to train and sample from energy-based models, paving the way for practical applications in robotics. These models learn to generate data by iteratively denoising noise samples toward a target distribution. Their iterative and probabilistic nature makes them particularly suitable for representing multi-modal robot policies. In this thesis, we have utilized Diffusion Policy (Chi et al., 2024), which has demonstrated state-of-the-art

performance in robot imitation learning.

2.5.1 Diffusion Policy

Let $\mathbf{u}_t \in \mathbb{R}^{n_u}$ denote the action at time t , $\mathbf{x}_t \in \mathbb{R}^{n_x}$ the state, and $\mathbf{h}_t^L = [\mathbf{u}_{t-L:t-1}^\top, \mathbf{x}_{t-L:t}^\top]^\top \in \mathbb{R}^{Ln_u+(L+1)n_x}$ the history of the previous L actions and states. Diffusion Policy models the multimodal action distribution in robot imitation learning using Denoising Diffusion Probabilistic Models (DDPMs) (Ho et al., 2020) or Denoising Diffusion Implicit Models (DDIMs) (Song et al., 2020). This approach regresses a noise prediction function $\boldsymbol{\epsilon}_\theta(\mathbf{h}_t^L, \mathbf{u}_t^k, k) = \boldsymbol{\epsilon}^k$ with a network $\boldsymbol{\epsilon}_\theta$ parameterized by θ . During training, a pair of history and actions $(\mathbf{h}_t^L, \mathbf{u}_t)$ is sampled from the demonstrated dataset. Random noise $\boldsymbol{\epsilon}^k$, conditioned on a randomly sampled denoising step k , is added to $\mathbf{u}_t$, such that

$$\mathbf{u}_t^k = \sqrt{\bar{\alpha}_k} \mathbf{u}_t^0 + \sqrt{1 - \bar{\alpha}_k} \boldsymbol{\epsilon},$$

where $\bar{\alpha}_k$ denotes a noise schedule parameter (see (Ho et al., 2020) for further details). Consequently, the loss can be expressed as

$$\mathcal{L} = \left\| \boldsymbol{\epsilon}_\theta(\mathbf{h}_t^L, \mathbf{u}_t^k, k) - \boldsymbol{\epsilon}^k \right\|^2.$$

During inference, given the history $\mathbf{h}_t^L$, Diffusion Policy executes a sequence of K denoising steps starting from random samples of actions $\mathbf{u}_t^K \sim \mathcal{N}(0, 1)$ to generate target robot actions $\mathbf{u}_t^0$. This inverse process can be defined as

$$\mathbf{u}^{k-1} = \frac{1}{\alpha_k} \left(\mathbf{u}_t^k - \frac{1 - \alpha_k}{\sqrt{1 - \bar{\alpha}_k}} \boldsymbol{\epsilon}_\theta(\mathbf{h}_t^L, \mathbf{u}_t^k, k) \right) + \sigma_k \boldsymbol{\epsilon},$$

where α_k, σ_k are the parameters of the noise schedule, and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$. This process is repeated to sample the denoised action $\mathbf{u}_t^0$.

Diffusion-based policy representations, discussed here, are extended in Chapter 6 for model-free failure recovery through conditional composition.

2.6 Products of Experts (PoE)

One of the methods frequently utilized in this thesis is the *Product of Experts* (PoE) model. This approach, originally introduced by Hinton (1999), combines multiple probabilistic models (or *experts*) in a manner analogous to a logical AND operation.

In contrast to a *Mixture of Experts* (MoE) model, which combines multiple models such that each expert explains a different subset of the data, the PoE framework integrates several experts by enforcing agreement among them. While MoE captures variability through the union of its experts' contributions, PoE captures structure through their intersection, emphasizing consistency across experts rather than specialization. This distinction is important

for manipulation tasks where we often need the action or state to satisfy several conditions at the same time (e.g., stability, contact feasibility, object alignment). With PoE, each expert can independently model one aspect of the problem, and the final distribution automatically concentrates on the region that respects all of them simultaneously. This property makes PoE an effective tool for feasibility-guided exploration, constrained sampling, and failure-aware adaptation, which are core components of this thesis.

The main idea is that, instead of modeling a single complex distribution directly, we decompose it into several simpler distributions (the experts), each capturing a different constraint or feature of the data. Namely,

$$p(\mathbf{x} | \theta_1, \dots, \theta_n) = \frac{1}{Z(\theta_1, \dots, \theta_n)} \prod_{i=1}^n p_i(\mathbf{x} | \theta_i), \quad (2.24)$$

where Z is the normalization constant ensuring that $p(\mathbf{x})$ integrates to one, and θ_i represents the parameters of i -th distribution. Each expert $p_i(\mathbf{x} | \theta_i)$ is typically simpler and easier to learn than the overall target distribution, and the product enforces that the resulting model assigns high probability only to data points that all experts agree are likely.

While this decomposition reduces the modeling burden on each expert, it introduces challenges in *sampling* and *normalization*, since Z depends on all experts. However, under certain assumptions, inference and sampling become tractable. For example, if each expert is a Gaussian distribution, then their product is also Gaussian, a property that has been widely exploited in robotics (Calinon, 2016).

2.6.1 Conditional diffusion policy composition.

The concept of policy composition in conditional diffusion models is closely related to the Product of Experts (PoE) formulation. In a diffusion model, we model a conditional distribution $p(\mathbf{u} | \mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_m)$, where each $\mathbf{c}_i$ represents a task or context condition (we omit the model parameters θ for readability). According to N. Liu et al. (2022), under the conditional independence assumption, this distribution can be expressed as

$$p(\mathbf{u} | \mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_m) \propto p(\mathbf{u}, \mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_m) = p(\mathbf{u}) \prod_{i=1}^m p(\mathbf{c}_i | \mathbf{u}), \quad (2.25)$$

where the proportionality holds because the marginal term $p(\mathbf{c}_1, \dots, \mathbf{c}_m)$ does not depend on $\mathbf{u}$. Using Bayes' rule, $p(\mathbf{c}_i | \mathbf{u}) \propto \frac{p(\mathbf{u} | \mathbf{c}_i)}{p(\mathbf{u})}$, the above expression can be rewritten as

$$p(\mathbf{u} | \mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_m) \propto p(\mathbf{u}) \prod_{i=1}^m \frac{p(\mathbf{u} | \mathbf{c}_i)}{p(\mathbf{u})}. \quad (2.26)$$

This form reveals the direct connection between conditional diffusion composition and the PoE principle: The composed conditional distribution is proportional to the product of the individual conditional experts $p(\mathbf{u} | \mathbf{c}_i)$, normalized by the unconditional prior $p(\mathbf{u})$.

Following the energy-based interpretation of diffusion models (N. Liu et al., 2022), the corresponding composition in the denoising (score) space can be written as

$$\boldsymbol{\varepsilon}_{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_m}(\mathbf{u}^k, \mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_m, k) = \boldsymbol{\varepsilon}(\mathbf{u}^k, k) + \sum_{i=1}^m w_i \left(\boldsymbol{\varepsilon}_{\mathbf{c}_i}(\mathbf{u}^k, \mathbf{c}_i, k) - \boldsymbol{\varepsilon}(\mathbf{u}^k, k) \right), \quad (2.27)$$

where $\boldsymbol{\varepsilon}_{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_m}(\cdot)$ denotes the denoiser corresponding to the composed conditional distribution $p(\mathbf{u} | \mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_m)$, $\boldsymbol{\varepsilon}(\cdot)$ is the denoiser trained for the unconditional distribution $p(\mathbf{u})$, and $\boldsymbol{\varepsilon}_{\mathbf{c}_i}(\cdot)$ are denoisers trained for the simpler conditional distributions $p(\mathbf{u} | \mathbf{c}_i)$. The weights w_i control the relative influence of each conditional expert.

The Product-of-Experts principle presented here underlies the unified formulations in Chapters 5 and 6, where feasibility, optimality, and policy models are combined in a common probabilistic framework.

3 Constraint-Aware Reference Manifold

Collaboration Note:

This chapter is a joint work with T. Brecej, K. Savevska, A. Ude, and T. Petrič, all from Josef Stefan Institute, Slovenia. They contributed to gather the data from human-humanoid experiments, and assisted me in applying the method on the Talos humanoid robot.

Publication Note:

This chapter is based on the following publication:

- A. Razmjoo, T. Brecej, K. Savevska, A. Ude, and T. Petrič, and S. Calinon, "Learning Joint Space Reference Manifold for Reliable Physical Assistance," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10412-10417, doi: 10.1109/IROS55552.2023.10342173.

One way to exploit the constraints is to use them to reduce the dimensionality of complex problems. By exploiting feasible subsets of motion, high-dimensional constrained optimization can be transformed into lower-dimensional control problems that are easier to solve in real time. This chapter explores this principle in the context of assistive sit-to-stand and stand-to-sit tasks with the Talos humanoid robot. These tasks are particularly demanding: humans exert large, fast, and unpredictable forces (100–200 N within 2–8 s), which challenge the robot's stability and adaptability. Talos, with 32 degrees of freedom, poses additional complexity, and even simplified models cannot reduce the problem below seven dimensions. To tackle this, we propose restricting the robot's motion to an interaction-force-dependent one-dimensional manifold. This manifold provides a spatial reference that simplifies control, increases the probability of stability, and allows the robot to react to human forces with a reactive controller. We parameterize the manifold using a functional representation and solve an optimization problem that accounts for stability constraints and human variability. The approach is validated in simulations and real-world experiments with Talos, demonstrating robustness and adaptability in physical human-robot interaction.

3.1 Introduction

Robots face particularly strong constraints when collaborating with humans, and these constraints become even more critical in physical assistance scenarios. Unlike in social or collaborative settings (Qin et al., 2022; Rasouli et al., 2022), close physical contact requires the robot to maintain strict safety and stability guarantees while adapting to the user’s unpredictable behavior (Sirintuna et al., 2022). This chapter focuses on sit-to-stand (STS) and stand-to-sit assistance with the Talos humanoid robot, where the user applies large and rapid interaction forces (100–200 N within 2–8 s). Such forces create a highly constrained problem: the robot must remain stable under uncertainty, while also managing its own redundancy, with 32 degrees of freedom (DoF) to coordinate.

From the perspective of this thesis, constraints here are not only limitations but also a means to simplify the problem. While directly solving the full high-dimensional constrained optimization is infeasible in real time, the redundancy of the humanoid allows us to restrict its motion to a low-dimensional subset. Specifically, we propose to encode the control problem as a one-dimensional manifold in joint space, parameterized by the human interaction force. This manifold acts as a spatial reference that naturally embeds feasibility, ensures stability, and enables the robot to follow the user’s actions with a lightweight controller. Unlike previous approaches that preplan assistance offline and treat the human as a follower (Geravand et al., 2017; J. Li et al., 2021), our method reverses the roles: the human leads the task, and the robot adapts its motion online along the manifold. A similar spirit was explored by Shomin et al., 2015, who used a ballbot to adjust its leaning angle based on interaction force.

By framing constraints as a tool for dimensionality reduction, this study demonstrates how a complex humanoid assistance problem can be reformulated into a low-dimensional, feasible subproblem. The remainder of the chapter describes how we design this manifold, validate it in simulation, and demonstrate its effectiveness in real-world experiments with Talos.

3.2 Related Work

Humanoid robots hold a unique position among robot types because of their human-like morphology, which makes them well-suited for physical assistance but also introduces significant challenges. These include underactuated dynamics, high dimensionality, and unilateral contacts with the environment. To manage these difficulties, various reduced-order models have been proposed. The linear inverted pendulum model (Kajita & Tani, 1991) and centroidal dynamics (Dai et al., 2014) are among the most widely used, as they simplify the dynamics by focusing on a few variables that are most relevant for balance.

Complementary strategies have been developed to enhance stability control. Hip and ankle strategies provide intuitive mechanisms for recovering from perturbations (Atkeson & Stephens, 2007), while optimization-based formulations extend reduced models to account for locomotion under unactuated states and constraints (Dai et al., 2014; Ibanez et al., 2017).

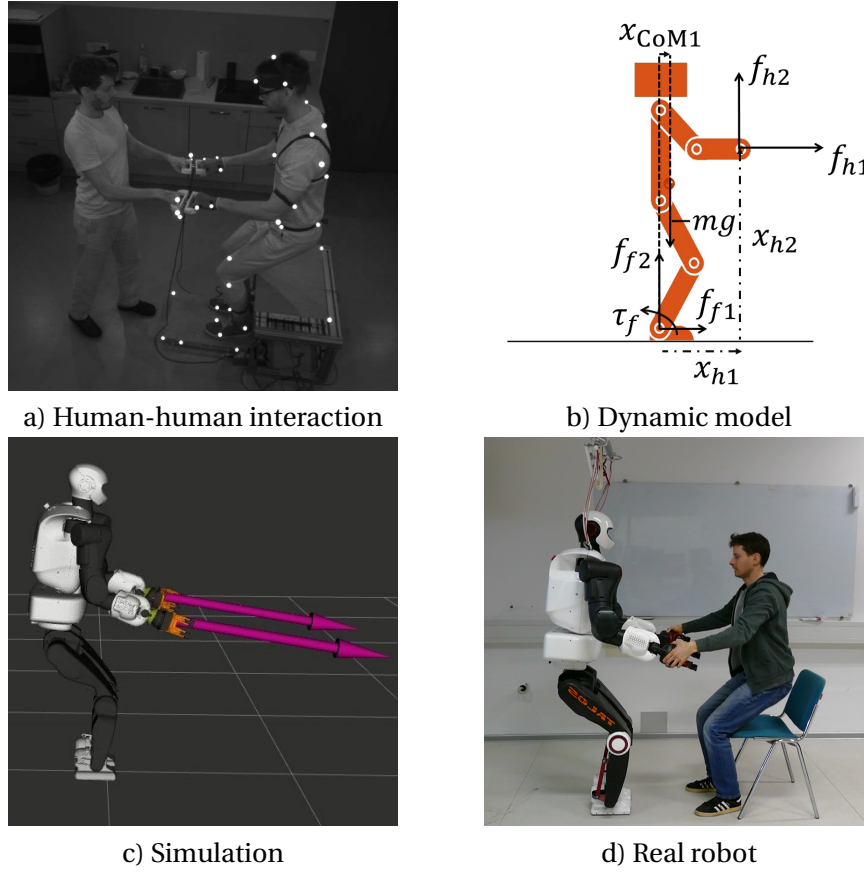


Figure 3.1: Experimental setup. (a) Force and human articulation position data collection during human-human interaction (only force data is utilized in this study), (b) Robot's dynamic model, (c) Simulation setup where the human is modeled as an external force, and (d) human-robot interaction scenario.

Another approach partitions the humanoid into separate kinematic chains, with some chains dedicated to task performance and others ensuring balance (Ibanez et al., 2012).

Despite these advances, challenges remain in mapping reduced-order representations back to full-body joint space while guaranteeing feasibility. For example, centroidal momentum matrices can be used to map CoM acceleration to joint accelerations (Orin et al., 2013). However, this regression-based approach cannot guarantee that all constraints are satisfied and can yield unintended behaviors if the human's actions are unpredictable.

In the context of physical human-robot interaction, such as STS assistance, these challenges are amplified. The task requires humans to exert large forces in short time windows (Shomin et al., 2015), forcing the robot to respond rapidly while maintaining stability. Optimization-based methods are often too slow to handle this scenario online, and they depend on accurate environment models, which are impractical when the human partner is unpredictable. Related work on STS tasks has largely focused on offline optimization for low-dimensional, fully actuated systems (Geravand et al., 2017; J. Li et al., 2021), where the robot leads and the

human adapts. In contrast, Shomin et al., 2015 explored online assistance using a ballbot, which regulated its leaning angle based on human-applied forces, illustrating the potential of reactive approaches.

These limitations highlight the need for methods that directly embed feasibility into the control formulation. Restricting motion to a low-dimensional manifold is a promising direction, as it eliminates redundant constraints and provides robustness in the face of unpredictable human behavior.

3.3 Methodology

In this section, we first derive a simplified model of the robot's stability, followed by some assumptions to reduce the complexity of the problem. Then, we formulate our main objective to construct a one-dimensional manifold in the joint space of the humanoid robot. Finally, we describe how to parameterize this manifold using a functional representation.

3.3.1 Simplified Robot Stability

Among the methods described in section 2.1, we focus on the ZMP in this work, which is the point on the ground where the horizontal moment applied on the robot is zero. For the 2D robot depicted in Fig. 3.1, this point can be calculated as

$$x_{\text{ZMP}} = \frac{\tau_f}{f_{f2}} \xrightarrow{\text{quasi-static cond.}} x_{\text{ZMP}} = \frac{f_{h1}x_{h2} - f_{h2}x_{h1} - mgx_{\text{CoM1}}}{-mg - f_{h2}}, \quad (3.1)$$

where $g = -9.81 \frac{\text{m}}{\text{s}^2}$ is the gravity acceleration. According to the data we have received from the human side (Fig. 3.3), $f_{h2}x_{h1} \ll f_{h1}x_{h2}$ and $f_{h2} \ll mg$, so (3.1) can be simplified to

$$x_{\text{ZMP}} \approx \frac{f_{h1}x_{h2} - mgx_{\text{CoM1}}}{-mg}. \quad (3.2)$$

For the robot to be stable, x_{zmp} should be located inside the support polygon. For the 2D case, this support polygon can be denoted as $\delta^- \leq x_{\text{ZMP}} \leq \delta^+$, where $\delta^\pm$ corresponds to the two ends of the foot location on the ground.

3.3.2 Assumptions

To simplify the problem, several assumptions are made:

1. The robot will not move its feet during the task, meaning that no stepping action is required.
2. When viewed from the sagittal plane, the robot's two feet coincide, and the forces

applied to its hands are identical. Consequently, the robot exhibits left–right symmetry. This relationship is captured by the coordination matrix $\mathbf{C}$ introduced in Sec. 2.3. As formally defined in (3.9), $\mathbf{C}$ enforces symmetric behavior by mapping seven input joint positions to the fourteen active joints of the robot (see Assumption 3 for justification of the dimensionality).

3. Control is applied only to the ankle, knee, hip, shoulder (three degrees of freedom), and elbow joints, collectively referred to as the active joints $\mathbf{q}^{\text{active}} \in \mathbb{R}^{14}$. All other joints of the robot, such as the waist and head, are treated as fixed throughout the experiment.
4. The robot can apply sufficient effort to achieve a desired behavior.

The first two assumptions allow the problem to be simplified by controlling only the robot CoM in the anterior/posterior direction, similar to (Shomin et al., 2015). The fourth assumption allows the problem to not consider the actuator limits during the planning phase, which is a common assumption when working with humanoids (Dai et al., 2014; Wieber et al., 2016).

3.3.3 Deriving a Spatial Reference

The problem addressed in this study can be formulated as finding a suitable configuration for the humanoid robot to maintain its balance while being subjected to external forces from a human in a collaborative task, namely

$$\begin{aligned} \ddot{\mathbf{Q}}^* &= [\ddot{\mathbf{q}}_0^*, \ddot{\mathbf{q}}_1^*, \dots, \ddot{\mathbf{q}}_T^*] = \arg \min_{\ddot{\mathbf{Q}}} \sum_{t=0}^{t=T} c_t(\mathbf{q}_t, \mathbf{f}_t, \mathbf{u}_t), \\ \text{s.t. } \mathbf{q}(0) &= \mathbf{q}_0, \mathbf{q}_t \in \mathcal{Q}^{\text{active}}, \mathbf{u}_t \in \mathcal{U}^{\text{active}}, \\ \mathbf{f}(\mathbf{q}_t, \dot{\mathbf{q}}_t, \ddot{\mathbf{q}}_t, \mathbf{f}_t, \mathbf{u}_t) &= 0, \mathbf{g}(\mathbf{q}_t, \dot{\mathbf{q}}_t, \ddot{\mathbf{q}}_t, \mathbf{f}_t, \mathbf{u}_t) \leq 0 \\ \text{ZMP}(\mathbf{q}_t, \dot{\mathbf{q}}_t, \ddot{\mathbf{q}}_t, \mathbf{f}_t, \mathbf{u}_t) &\in [\delta^-, \delta^+], \quad \forall t \in \{0, \dots, T\}, \end{aligned} \quad (3.3)$$

where for each time step t , the cost function is defined as c_t , the force applied at the robot's hands is denoted as $\mathbf{f}_t$ (please note the t subscript), and the control command as $\mathbf{u}_t$. The dynamic equation of the system is represented by the function $\mathbf{f}(\cdot)$ (without any subscript), while the ZMP point of the system is calculated using the function $\text{ZMP}(\cdot)$. Other task-specific constraints are incorporated through the function $\mathbf{g}(\cdot)$. As an example, in the present task, we require the robot's hand position to remain fixed in order to ensure safer and smoother collaboration and to reduce the influence of variations in the force sensors. The feasible set for active joint angles and their control commands is denoted by $\mathcal{Q}^{\text{active}}$ and $\mathcal{U}^{\text{active}}$, respectively.

To incorporate feedback from the environment or to adapt to different humans, the problem represented by (3.3) must be solved recurrently using a method such as a model predictive controller (MPC). However, this approach does not seem suitable for pHRI tasks as the environmental model can vary significantly at each time step. Instead of solving (3.3) for the whole trajectory, our focus is to find a desired terminal configuration $\mathbf{q}_T$, which is expected to be

static at the end of the task, i.e., $\dot{\mathbf{q}}_T = \ddot{\mathbf{q}}_T = \mathbf{0}$. Therefore, we can assume quasi-static motion and use the formula derived for the ZMP (3.2), which allows us to define the ZMP as a function of the current configuration $\mathbf{q}$ and the horizontal external force applied by the user f_{h1} , namely

$$\begin{aligned} \mathbf{q}_T^* &= \underset{\mathbf{q}_T}{\operatorname{argmin}} c(\mathbf{q}_T, f_{h1}), \\ \text{s.t. } \mathbf{q}_T &\in \mathcal{Q}^{\text{active}}, \text{ZMP}(\mathbf{q}_T, f_{h1}) \in [\delta^-, \delta^+], g(\mathbf{q}_T, f_{h1}, f_{h2}) \leq 0. \end{aligned} \quad (3.4)$$

For the sake of enhancing readability, the notation of T is omitted hereafter. The cost function c should be defined to consider the goal of the task, which is the stability of the robot. One straightforward way to formulate the objective is to define the cost function as

$$c_{\text{std}}(\mathbf{q}, f_{h1}) = \|\text{ZMP}(\mathbf{q}, f_{h1})\|^2. \quad (3.5)$$

This formulation implicitly drives ZMP toward zero. Such a choice is reasonable when no prior knowledge about the external interaction is available, for example, in the case of random external disturbances. However, we have access to human–human interaction data, which shows that the applied forces lie within a predictable range. The desired configuration should therefore be adapted to this range of forces, rather than optimized for arbitrary inputs. This consideration is crucial, yet often overlooked in previous studies (Geravand et al., 2017; Shomin et al., 2015), which focused on low-dimensional or fully actuated systems.

To address this limitation, we propose an alternative cost function

$$c_{\text{rob}}(\mathbf{q}) = \sum_{f_{\text{applied}}=0}^{f_{\text{max}}} \left\| \text{ReLU}(\text{ZMP}(\mathbf{q}, f_{\text{applied}}) - \delta^+) \right\|^2 + \left\| \text{ReLU}(\delta^- - \text{ZMP}(\mathbf{q}, f_{\text{applied}})) \right\|^2, \quad (3.6)$$

where f_{max} is the maximum force that we expect to receive from the user. This cost penalizes the system quadratically according to how much it has violated the boundaries. In contrast to (3.5), the formulation in (3.6) explicitly accounts for a range of interaction forces and therefore promotes configurations that remain robust under variability in human input.

3.3.4 Functional Representations

Combining (3.6) and (3.4) will help us obtain a robust configuration of the robot that specifies where the robot should be according to different forces (e.g., using a lookup table). However, solving this problem in its current form can have some drawbacks. For example, we need to solve (3.4) for each value of f_{h1} , which can result in a jump between different solutions, see Fig. 3.2a. The aforementioned behavior can potentially influence the quality of human-robot interaction, in addition to affecting the sensory information, particularly the force values, if the robot experiences shaking during its operation. To overcome this limitation, we deploy the idea of a functional representation, as it can correlate the solution for different force values. Specifically, we decompose the problem into two parts by introducing an intermediate

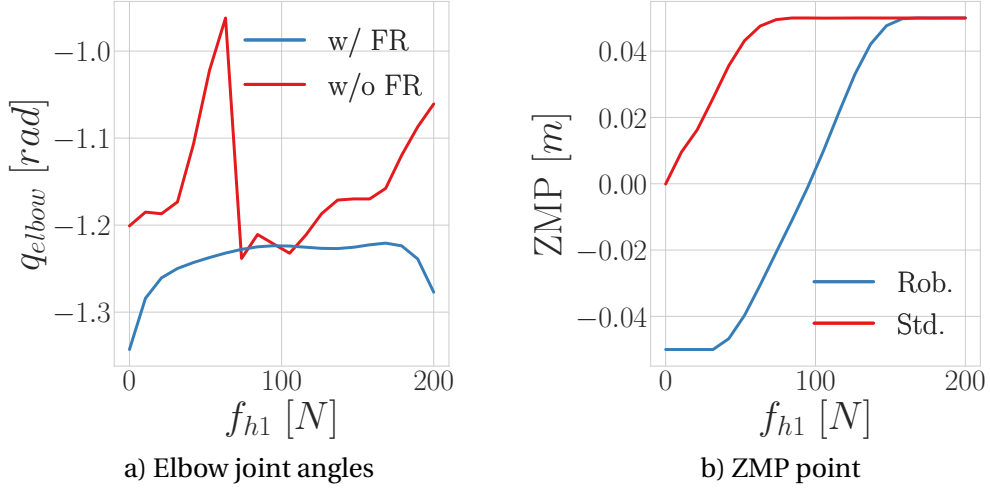


Figure 3.2: Exploring the effect of step variation on force magnitudes. (a) Incorporating the function representation yields smoother final results (FR). Without it, the red curve—shown for the elbow joint as an example of other active joints—exhibits a sharp jump around approximately 80 N. (b) Using a robust cost function enables the robot to exploit the full support polygon. In contrast, the standard cost (std.) remains regulated around zero, which is appropriate for random external forces but does not account for the task-specific range of human interaction forces. Since the maximum force is assumed to be 200 N, the diagrams may also be interpreted as functions of the phase variable s , varying from 0 to 1, noting that the mapping from force f to phase variable s is linear.

variable $s \in [0, 1]$. As outlined in Sec. 2.3, we assume that the desired configuration can be represented using basis functions that depend on s , such that

$$\mathbf{q}(s) = \Psi(s)\mathbf{w}, \text{ with } s = \ell(f_{h1}), \quad (3.7)$$

where a function $\ell(\cdot)$ maps the force to the value of s . In this formulation, $\mathbf{q}$ can be regarded as a spatial reference, while the mapping variable s can be regarded as a latent variable that determines the position along the reference. In this work, we assume a simple linear relationship between s and the applied force, namely $s = \frac{f_{h1}}{f_{\max}}$, and thus $f_{h1}(s) = sf_{\max}$. Developing an improved controller that operates effectively on this one-dimensional manifold is left for future work. Now, the problem consists of finding the desired coefficients $\mathbf{w}$ for the entire task, which can be solved by minimizing the optimization problem

$$\begin{aligned} \mathbf{w}^* &= \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=0}^{i=D} c_{\text{rob}}(\mathbf{q}(s_i) \cdot f_{h1}(s_i)), \\ \text{s.t. } \mathbf{q}(s) &\in \mathcal{Q}^{\text{active}}, \quad \text{ZMP}(\mathbf{q}(s), f_{h1}(s)) \in [\delta^-, \delta^+], \\ f_{h1}(s) &= sf_{\max}, \quad \mathbf{g}(\mathbf{q}(s), f_{h1}(s)) \leq 0, \end{aligned} \quad (3.8)$$

where $s_i = \frac{i}{D}$ for $i \in \{0, 1, 2, \dots, D\}$. The summation in (3.8) is going to be solved for $D+1$ discrete points on the manifold. Note that under this formulation, no smoothness requirement is

imposed on $\mathbf{w}$, as these variables remain fixed. Smoothness and continuity of the joint angles $\mathbf{q}$ are ensured instead through their construction as a superposition of smooth basis functions.

Another important point is that, because both force and motion variables appear in this formulation, the proposed approach may exhibit behaviors reminiscent of methods such as admittance control. It is important to emphasize, however, that our method differs in several key aspects. In typical admittance-based schemes, the robot motion is first determined at the contact point and subsequently mapped to whole-body motion through inverse kinematics. In contrast, our formulation integrates these steps into a single optimization procedure, which increases reliability and provides additional capabilities. These include identifying an optimal behavioral response without the need to select gains such as k_p or k_d , as well as incorporating additional requirements, such as constraining the hand to remain fixed during the interaction.

3.4 Experiment

This section analyses the efficacy of the proposed method through both simulation and real-world experimentation by conducting two experiments. The first experiment, which we refer to as the *robust case*, has utilized c_{rob} to find a robust solution for a range of force values, while the second experiment considers only the current force values, i.e., $c = c_{\text{std}}$, and is referred to as the *standard case*. Videos of all experiments are available on our group's YouTube channel¹.

3.4.1 Finding the Spatial Reference

The initial step in this study is to solve (3.8) for the robotic system. Based on the data gathered from human-human interaction (Fig. 3.3), it has been determined that a force of approximately 90 N is required at each hand to enable standing up. Thus, the robot must be capable of withstanding forces up to 180 N to replicate the motion. In consideration of safety concerns, a maximum value of $f_{\text{max}} = 200$ N has been selected. Although the ZMP for Talos can fall within a range of ± 10 cm around the foot, a conservative margin of $\delta = 5$ cm has been incorporated to account for unmodeled factors such as friction and to accommodate the simplifying assumptions utilized in this study. Additionally, to ensure safety, the system has been constrained to not exceed a hand position displacement of 10 cm from its default configuration.

Bézier curves with 11 control points are utilized in this study (number selected empirically). Moreover, to exploit the symmetrical characteristics of the 14 active joints, the coordination matrix $\mathbf{C}$ has been defined as

$$\mathbf{C}_{14 \times 7} = \begin{bmatrix} 1 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 & 1 \end{bmatrix}^T. \quad (3.9)$$

¹<https://www.youtube.com/watch?v=GQAad6GFPIE>

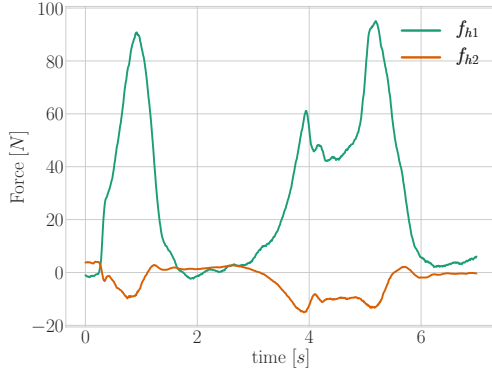


Figure 3.3: The force that the user applied at one of his/her hands when interacting with another user.

We have solved (3.8) offline by discretizing s into 20 equally spaced points. The resulting elbow joint and ZMP values are illustrated in Fig. 3.2a. We have used the SLSQP optimization method, available in the SciPy package (“SciPy 1.0: fundamental algorithms for scientific computing in Python”, 2020) for solving the optimization problem.

3.4.2 Results

Simulation

The simulation was executed using a ROS interface on the Docker file provided by PAL Robotics. Gazebo was utilized for physical simulation, with a PD position controller applied to the robot’s motor current. It should be noted that the method described here is not restricted to a position controller and that alternative controllers could also be utilized. The control system reads the force values applied to the robot, calculates the desired latent position on the manifold and its corresponding active joint configurations, and publishes the desired values at 20 Hz.

To replicate the human behaviors in the simulation, a sinusoidal force was applied for $2h$ seconds, reaching its maximum value of $M \in [100, 200]$ N within $h \in [1, 4]$ seconds. The results of this simulation are presented in Fig. 3.4. The findings suggest that the robust version of the system demonstrates a significantly higher success rate in handling the challenges inherent to the STS task. It is worth noting that the spatial reference generated for this experiment assumed quasi-static motion, which may not hold true during the experiment. This underlying assumption may have contributed to the system failure in certain instances. However, the system developed by taking into account the unpredictability of human behavior demonstrates greater resilience and produces more robust spatial references.

Real Robot

Following the verification of results in the simulation, a real-world experiment was conducted to validate the system’s performance in a practical setting. Participants were asked to perform

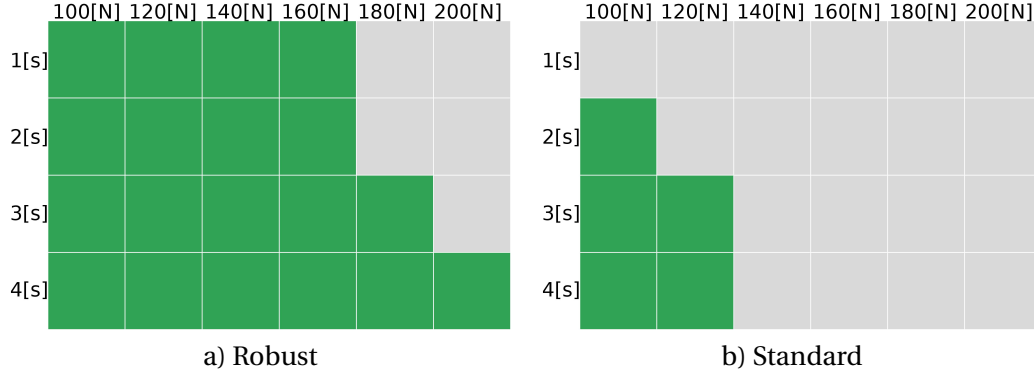


Figure 3.4: Examining the performance of the two systems operating under different force magnitudes M [N] (columns), and its rising time h [s] (rows). The force is applied for $2h$ seconds. Green and gray blocks represent successful and failed motions, respectively.

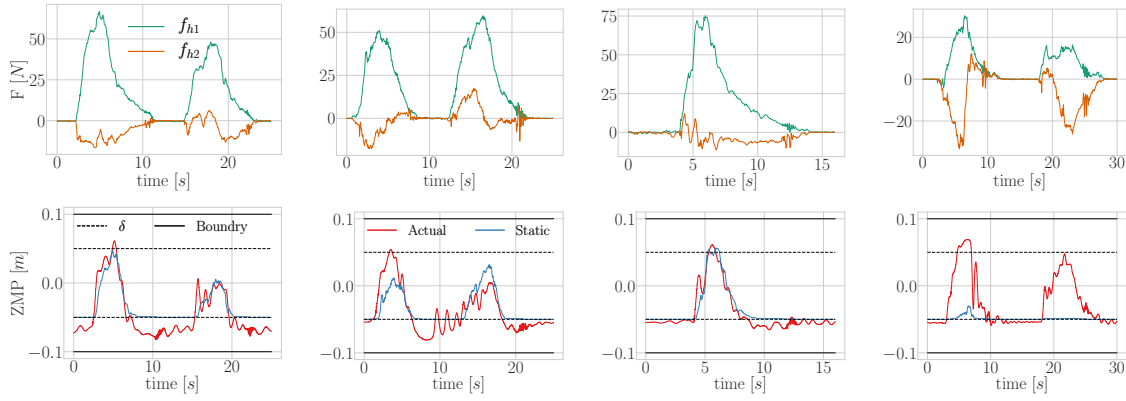


Figure 3.5: For each robot hand (top row), user-applied forces are shown: green for horizontal force and orange for vertical force. The bottom row shows the resulting ZMP behavior: the true support boundary (solid black), the planning margin (shaded black), the sensor-based ZMP (red), and the quasi-static ZMP estimate (blue). Each column corresponds to a different user.

the STS task on the robot, and the same setup was employed as in the simulation, with the only difference being the replacement of the Gazebo simulator with the real robot.

The force values applied by the participants on the robot are depicted in Fig. 3.5, indicating that the maximum assumed force was not exceeded. Figure 3.5 also illustrates how the ZMP point on the robot varied during the real-world experiment. Although the controller was derived under the quasi-static assumption, the system attempted to remain within the specified boundary. Nevertheless, in some instances, the ZMP deviated from the marginal boundary δ while remaining within the stable region. Although the control mechanism itself was not capable of precisely controlling the ZMP point, the manifold derived by the robust case displayed appropriate resilience in preventing the system from going beyond the marginal boundary and stable region.

3.5 Discussion

This section discusses the effects of the steps taken to solve the STS task using the proposed method. Without considering the use of a functional representation, there could be big discontinuities in the joint space, as shown in Fig. 3.2a with jumps up to 0.2 rad (≈ 11.46 degrees) for the elbow joint. We observed that a functional representation eliminates the jump in the spatial reference by correlating the optimization solution for all force values using the location-independent variables $\boldsymbol{w}$.

We incorporated human unpredictability into the cost function by optimizing it over the range of the expected human behavior. The results, as illustrated in Fig. 3.2b, demonstrate that our approach allows the robot to utilize the marginal boundary more effectively. Specifically, at the beginning of the task, the robot anticipates the possibility of a force being applied by the human and adjusts its ZMP point accordingly to provide a larger safety margin. Conversely, the standard method focuses solely on regulating the ZMP point of each force around zero. The advantages of this robustness are demonstrated in Fig. 3.4. To further enhance this approach, future studies may consider weighting each potential human behavior by its occurrence probability. This would enable the system to better adapt to more likely scenarios, and potentially lead to more personalized and efficient task performance.

The method that we propose has some remaining limitations. Fig. 3.5 shows that the ZMP points slightly violate the marginal boundaries δ . This is because the method does not have direct control over the ZMP point and relies on the robustness of the manifold. This error can also be related to unmodeled dynamics such as the friction effect, especially when there is no external force, where ZMP stays around -0.07 cm instead of -0.05 cm. In future work, adding feedback on the ZMP point to control it more directly on the manifold can be investigated. Another idea could be to solve (3.3) on the manifold, removing the need to consider some constraints such as joint limits, and optimizing a one-dimensional variable $\tilde{s}$ over a horizon. The efficiency of these methods should be investigated in future studies.

One observation from the experimental results is that for the user shown in the rightmost column in Fig. 3.5, the estimated ZMP point using a quasi-static assumption differs significantly from the actual one, despite the successful completion of the task. This phenomenon is also evident in other experiments (not depicted here), particularly when the user applies forces below 40 N on each hand (80 N on the robot). Our preliminary hypothesis is that, as shown in Fig. 3.2b, when the force is below 40 N (20 N at each hand), the robot's movement is concentrated on maintaining the ZMP on the boundary, which requires agile motion of CoM to compensate for the external force. Consequently, the quasi-static assumption is more likely to be violated in this phase. This can also be seen by comparing the amount of motion in Fig. 3.2a at the beginning of the task versus the remainder of the task. Additionally, since the robot's motion is controlled through a position controller, the accuracy of the calculated ZMP depends on the controller's precision in tracking the reference motion, which is expected to be less accurate for agile motions. To address this issue, we can either modify the underlying controller or

incorporate an online controller. Furthermore, unlike the others, this user has applied force of comparable or greater magnitude in the vertical direction, as opposed to the horizontal direction. We initially assumed that the impact of this force would be negligible compared to the effect of the horizontal force. However, for this particular user, this assumption may not hold true. This outcome may be attributed to the non-adaptive initial pose of the robot's hand, which might not have been comfortable for certain users, resulting in non-intuitive force patterns. While this force was insufficient to cause the robot to fall, we are motivated to consider both forces in our future formulation, which will result in higher-dimensional manifolds.

Another limitation arises from using the ZMP as the stability metric. This was acceptable under our assumptions, such as quasi-static motion and symmetric behavior, but for more dynamic interactions, and given that the robot's feet may occupy different heights, a more advanced stability measure is required, such as the contact wrench cone (Hirukawa et al., 2006).

3.6 Conclusion

We demonstrated how the concept of a manifold can be leveraged to exploit redundancy and construct an interpretable low-dimensional latent space for humanoid assistance. The manifold can be seen as a local coordination structure, correlating all joints according to the robot's position on the manifold. This correlation naturally embeds feasibility by eliminating the need to explicitly enforce certain constraints, such as joint limits or stability, and allows the robot to adapt to human behavior in real time.

In this study, we employed a simple controller that mapped the measured interaction force to a position on the manifold, without addressing online controller design. Future work could explore more advanced controllers that operate directly on the manifold, such as fast model predictive controllers (MPC) that update the desired position based on stability indicators like the zero moment point. Extending the concept beyond one-dimensional manifolds may also open new directions for richer interactions.

A limitation of our approach lies in the simplifying assumptions made to derive the manifold, which may not generalize to all scenarios. Designing task-specific low-dimensional representations that remain interpretable, especially for physical human-robot interaction, is an important research challenge. At the same time, alternative approaches that tackle the problem in its original high-dimensional space remain valuable, as they require fewer assumptions. In Chapter 4, we will investigate one such approach.

4 Demonstration-Guided Constrained Motion Generation

Collaboration Note:

This chapter is the result of joint work with Teguh Santoso Lembono, who contributed through discussions and by exploring alternative approaches with me, and also assisted in improving the writing of the paper.

Publication Note:

This chapter is based on the following publication:

- A. Razmjoo, T. S. Lembono, and S. Calinon, "Optimal Control Combining Emulation and Imitation to Acquire Physical Assistance Skills," in *20th International Conference on Advanced Robotics (ICAR)*, 2021, pp. 338-343, doi: 10.1109/ICAR53236.2021.9659381.

In constrained robotic assistance, solving optimal control problems directly in their original space avoids restrictive assumptions but is often difficult due to nonlinearity and non-convexity. In this chapter, we investigate how previous experiences, examples of feasible and near-optimal behavior, in the form of demonstrations can guide the optimization process. Demonstrations can support optimal control in three complementary ways: (i) by defining task-related cost functions, similar to goal-level emulation in animals; (ii) by providing warm starts that initialize the optimization near promising solutions; and (iii) by acting as an imitation prior inside the cost function. While the first two strategies are well studied, we show that the third one, although less explored, plays a key role in accelerating convergence. Specifically, we augment the cost function with an imitation term that attracts the solution toward demonstrations, while still allowing generalization beyond them. We evaluate this approach in a physical assistance scenario, where a humanoid robot supports a human in sit-to-stand tasks. Our results demonstrate that the method reduces the number of solver iterations by more than 70% compared to standard formulations without demonstrations, thereby highlighting how demonstrations can serve as a form of feasibility-guided adaptation.

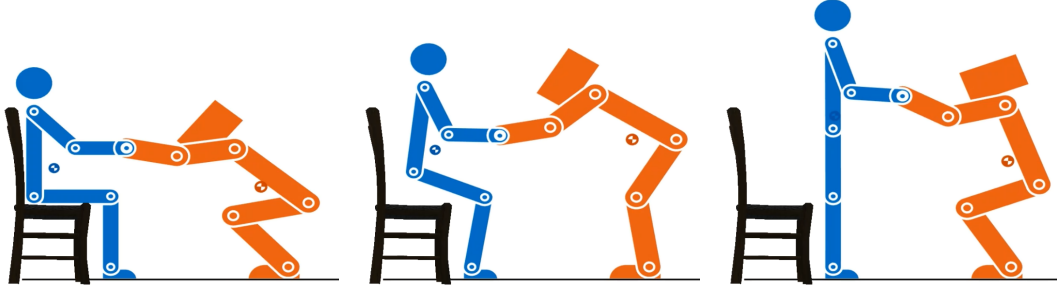


Figure 4.1: Models of the human (blue) and the robot (orange). The corresponding CoM positions are presented as semi-filled discs.

4.1 Introduction

This chapter investigates how to improve a baseline optimal control problem without introducing additional modeling assumptions by exploiting prior experience in the form of demonstrations available at execution time. Demonstrations can be viewed as samples from the feasible and near-optimal region of the state–action space. They can be used in three complementary ways. First, one can infer the underlying cost or reward from demonstrations—known as inverse optimal control (IOC) or inverse reinforcement learning (IRL) (Abbeel & Ng, 2004; Mombaur et al., 2010); we refer to this goal-driven perspective as emulation. In this chapter, we assume the cost is known and focus on leveraging demonstrations in other ways. Second, demonstrations can serve as a memory to accelerate convergence by initializing the solver with the closest example, i.e., warm-starting (Forte et al., 2012; T. S. Lembono, Paolillo, et al., 2020; C. Liu & Atkeson, 2009; Mansard et al., 2018; Stolle & Atkeson, 2006). Warm starts are particularly useful because nonlinear optimal control typically relies on iterative methods such as DDP (Tassa et al., 2012) and iLQR (W. Li & Todorov, 2004), whose convergence is sensitive to initialization.

However, warm-starting typically influences only the initial iterate; subsequent updates are guided solely by the main cost. To keep the optimization close to relevant prior experience throughout the iterations, we use the third approach which augments the cost with an imitation term that penalizes deviations from nearby demonstrations. We thus combine imitation (trajectory-level proximity) with emulation (objective-driven optimality). Evidence from social learning suggests these strategies are complementary (Whiten et al., 2009): emulation supports generalization by reasoning about goals, whereas imitation enables rapid acquisition by copying and later refining.

We evaluate this approach on a simulated sit-to-stand (STS) task (see Fig. 4.1) formulated as two optimal control problems (OCPs). The first OCP predicts the user’s desired assistance (human-assistance prediction), and the second computes the robot torques required to deliver it (robot controller). We compare convergence with and without the imitation term. Our formulation accounts for user-specific variability (mass, height) and potential joint impairments (ankle, knee, hip), and aims to solve both OCPs rapidly by leveraging successful demonstrations.

In the remainder of this chapter, we describe the related work, and we explain the methodology in Section 4.3. The simulation results are presented and discussed in Section 4.4 and 4.5, respectively. Finally, we conclude the chapter in Section 4.6.

4.2 Related Work

As introduced in Sec. 3.2, one of the challenging assistance tasks studied in the literature is STS. Here, we review works that highlight the role of demonstrations in STS. El-Hussieny et al., 2017 employed inverse optimal control (IOC) to identify a quadratic cost function that generates human-like hip motion during STS, with terms for state and input regularization. Geravand et al., 2017 proposed a more elaborated cost function considering criteria such as the human’s final posture, CoM position, and joint limits. J. Li et al., 2021 divided the STS task into sit-to-perch (STP) and perch-to-stand (PTS) phases. Using human demonstrations, they observed consistent motion patterns across participants and defined phase-specific cost functions that minimize human torque. In our work, we adopt a simplified cost formulation similar to (Geravand et al., 2017) as the basis of the task definition (see Section 4.3.2). Our contribution then focuses on the other two aspects of using demonstrations—warm-starting the optimization and adding imitation terms—both of which we show to significantly improve convergence.

4.3 Methodology

We formulate the STS task as two optimal control problems with two separate cost functions to describe the human assistance and the robot controller. These cost functions are minimized by considering the human and robot’s dynamical model using iLQR. To improve the convergence rate of the OCPs, we propose to use previous demonstrations to predict the optimal trajectories which are then used to warm start the OCP solver as well as to modify the cost function with an imitation term. In addition, we take advantage of *control primitives* (CPs) to generate smoother behaviors and reduce the dimension of the system states and inputs.

4.3.1 Dynamical modeling

We model the human as a planar inverted pendulum with five links, which is inspired from STS literature such as (Geravand et al., 2017; J. Li et al., 2021). We assume that the robot assisting the human is also a five-link planner pendulum. Fig. 4.1 illustrates the two agents during the STS task. In this modeling, the mass and the length of each body segment are calculated according to their proportion to the agent’s total mass and height, respectively. Also, the mass of each link is assumed to be located at its CoM position. For each segment, its L_{ratio} (length ratio to the human’s height) (Tözeren, 1999), M_{ratio} (mass ratio to the human’s total mass) (Tözeren, 1999), I (the moment of inertia) (Tözeren, 1999), and $\text{CoM}_{\text{ratio}}$ (distance of CoM position of each link from its previous link normalized with the length of each link) (Drillis

Table 4.1: Dynamical parameters of a human body segments.

The segment	L_{ratio}	M_{ratio}	$I \text{ (kg.m}^2\text{)}$	$\text{CoM}_{\text{ratio}}$
Foot	0.042	0.0145	0.0038	-
Shank	0.250	0.0465	0.0505	0.532
Thigh	0.240	0.0988	0.1502	0.500
Trunk	0.300	0.5080	1.3080	0.530
Upper arm	0.183	0.0270	0.0213	0.430
Forearm	0.160	0.0160	0.0760	0.410

et al., 1964) are presented in Table 4.1. Note that in the simulation, the values of M_{ratio} and I should be doubled for all of the segments, except for the trunk, as there are two of them in human bodies. While the human can have different total mass and height among different experiments to simulate different subjects, The robot always has 80 kg mass and 1.8 m height.

4.3.2 Task definition

Human assistance prediction

At this step, the robot tries to find what is the optimal way to help the human by solving an optimization problem whose outputs are the optimal desired external forces and the human reaction during the task. The cost function defined for this step has two main terms. The first one specifies the human to be at the standing position at the end of the task ($\|\mathbf{q}_T - \mathbf{q}_{\text{stand}}\|^2$) and the other term penalizes high input commands and external forces ($\|\mathbf{U}_{\text{Hum.}}^T \mathbf{R} \mathbf{U}_{\text{Hum.}} + \mathbf{F}_{\text{int}}^T \mathbf{R}_f \mathbf{F}_{\text{int}}\|^2$), where $\mathbf{U}_{\text{Hum.}}$ and $\mathbf{F}_{\text{int}}$ are the human torque commands and interaction forces over the task, respectively. The standing position is defined as the human's shanks, thighs and trunk are in the straight-up direction with zero velocity. By adjusting the cost weight associated with each of the human joints, we regulate the corresponding torque values and thereby simulate different levels of impairment. Moreover, by controlling the ratio of these weights to $\mathbf{R}_f$, we can specify the amount of assistance required by the user. For example, when $\mathbf{R}_f$ is set to the identity matrix, assigning a large weight to the human knee joint (e.g., 1000) indicates that this joint should exert minimal torque, thereby modeling a limitation. Conversely, assigning a small weight (e.g., 0.01) represents a healthy joint capable of generating torque with little need for assistance. In this experiment, we only change the cost weights of the ankle, knee, and hip joints.

There are two other terms in the cost function of the human assistance prediction to make the total CoM position of the human to be at the foot support at the end of the task and to consider the human's joints limits during the task ($\mathbf{x}_{\text{CoM}} \in \delta^\pm$ and $\mathbf{q}_t \in [\mathbf{q}^{\min}, \mathbf{q}^{\max}]$ for all $t \in [0, T]$). These two costs penalize the system when it goes out of the feasible region. This step can be defined mathematically as

$$\mathbf{U}_{\text{Hum.}}^*, \mathbf{F}_{\text{int}}^* = \underset{\mathbf{U}_{\text{Hum.}}, \mathbf{F}_{\text{int}}}{\operatorname{argmin}} c_{\text{Hum.}}(m, h, \mathbf{R}, \mathbf{U}_{\text{Hum.}}, \mathbf{F}_{\text{int}}), \quad (4.1)$$

where $c_{\text{Hum.}}$ is the cost function. m and h are the user's total mass and height, respectively, and $\mathbf{R}$ is the joints cost weight. The human's optimal joint trajectory $\mathbf{X}_{\text{Hum.}}^*$ and its hand trajectory can also be calculated with the outputs of (4.1) and considering the human's dynamic model.

The robot controller

The robot should follow the trajectory of the human's hand while it is expected to receive $\mathbf{F}_{\text{int}}^*$ alongside this trajectory. For the first one, we put a term in the robot's cost function to move its end-effector at the planned human's hand trajectory ($\|\mathbf{X}_{\text{Rob.}} - \mathbf{X}_{\text{Hum.}}^*\|^2$), and for the second one, we regularize the robot interaction forces $\mathbf{F}_{\text{int}}'$ around the desired interaction forces $\mathbf{F}_{\text{int}}^*$ ($\|\mathbf{F}_{\text{int}}' - \mathbf{F}_{\text{int}}^*\|^2$). Like the human assistance prediction, the cost function of the robot should be minimized w.r.t. the robot's torques and the interaction forces. Note that interaction forces are functions of the human's and robot's torques and configurations. Thus, by considering them as separate variables, we violate a physical constraint, which is that the end-effector of the robot and the hand of the human should be at the same place every time. We consider this constraint in our cost function. In this way, we do not need to solve the closed-chain kinematic problem of the two systems to find the interaction forces, so we can consider the two agents as two separate systems. The robot's joint limits and the stability of both the human (with the new interaction forces) and the robot at the end of the task are also considered in the robot controller. That said, the robot controller can be modeled as

$$\mathbf{U}_{\text{Rob.}}^*, \mathbf{F}_{\text{int}}'^* = \underset{\mathbf{U}_{\text{Rob.}}, \mathbf{F}_{\text{int}}'}{\operatorname{argmin}} c_{\text{Rob.}}(\mathbf{X}_{\text{Hum.}}^*, \mathbf{F}_{\text{int}}^*, \mathbf{U}_{\text{Hum.}}^*, \mathbf{U}_{\text{Rob.}}, \mathbf{F}_{\text{int}}'), \quad (4.2)$$

where $\mathbf{U}_{\text{Rob.}}$ is the robot's torque commands and $\mathbf{F}_{\text{int}}'$ is the interaction forces. If (4.2) converges well, we expect $\mathbf{F}_{\text{int}}'^*$ to be equal or at least be close to $\mathbf{F}_{\text{int}}^*$. The two cost functions mentioned above are the emulation part of the STS task. In this study, we propose to speed up the convergence rate of these two OCPs by exploiting warm starting and imitation.

4.3.3 Control Primitives (CPs)

We assume that the input commands in (4.1) and (4.2) are made up of some basis functions, each of which is called a *control primitive (CP)*. Here, we used radial basis functions (RBFs), but other basis functions such as Bernstein polynomials or Fourier series can also be used (Calinon, 2019). The basis functions implicitly enforce the smoothness and the continuity of the control inputs and help the human and the robot behave more naturally. We assume that the input commands can be defined as

$$\mathbf{U} = \Psi \mathbf{w}_u \Rightarrow \Delta \mathbf{U} = \Psi \Delta \mathbf{w}_u, \quad (4.3)$$

where Ψ is the transformation matrix made up of the basis functions (CPs) and $\mathbf{w}_u$ is the vector of the CPs' coefficients. With this assumption, (2.10) should be modified as

$$\Delta \mathbf{w}_u^* = \left(\Psi^\top (\mathbf{S}_u^\top \mathbf{H}_u \mathbf{S}_u + 2\mathbf{S}_u^\top \mathbf{H}_{xu} + \mathbf{H}_u) \Psi \right)^{-1} \Psi^\top (-\mathbf{S}_u \mathbf{g}_x - \mathbf{g}_u). \quad (4.4)$$

We use a different number of basis functions for the two OCPs, i.e., five basis functions for the human assistance prediction and eight basis functions for the robot controller.

4.3.4 Warm starting

Although the use of CPs reduces the dimensionality of the task, solving the resulting problem can still be computationally demanding. A common strategy to mitigate this, as discussed in the literature (T. S. Lembono, Paolillo, et al., 2020), is warm starting. The key idea is that the closer the initial guess for (4.1) and (4.2) is to the local optimum of the system, the fewer iterations are required by iLQR to converge. The system can guess a good initial point by exploiting its previous experiences, which are, in this study, some demonstrations gathered by solving the OCPs without considering the imitation term. This could be done either by lookup tables or any regression method that could map the situations to the initial guess. Here, we present the idea with the former one, in the form of k -nearest neighbors (k -NN) with $k = 1$.

4.3.5 Imitation cost

Warm starting can be effective, but its influence is limited to the initialization phase; the subsequent evolution of the solution is governed by the gradient of the cost function. The cost function c defined in (4.1) and (4.2) can be interpreted as the emulation component of the task. Although it can generalize the task to any situation, it often requires lots of iterations to converge to a solution. To address this problem, we modify c as

$$c' = c + \lambda (\mathbf{y} - \mathbf{y}_d)^\top (\mathbf{y} - \mathbf{y}_d). \quad (4.5)$$

In (4.5), $\mathbf{y}$ is a feature vector we want to imitate, $\mathbf{y}_d$ is its desired value extracted from the demonstrations, and λ is a positive constant. This imitation term has a minimizer that is expected to be close to the minimizer of the main cost function c , so the optimization problem would more likely iterate toward an area that is emphasized by the main cost and the imitation term. We can be more confident about the closeness of the minimizers if the desired value of the chosen feature $\mathbf{y}_d$ can be predicted for the new situation with sufficient accuracy. Because of this, this feature should be chosen with some caution. Note that it does not need to be very accurate: just accurate enough to give the system some guidance toward its local optima. In this work, we chose the joint angles as the features, and we predict its values from the demonstrations by using k -NN with $k = 1$, but any applicable method could be used. The value of λ should be small as compared to other values in c in order to not significantly affect

Table 4.2: Comparing the mean value of number of iterations and convergence time with their standard deviation for different approaches.

		exp1		exp2	
	method	# iterations	conv. time (s)	# iterations	conv. time (s)
Human assistance prediction	STD	80.84 ± 103.77	10.63 ± 18.50	80.24 ± 68.55	10.07 ± 11.15
	Only warm start	53.93 ± 119.67	7.22 ± 19.32	48.62 ± 41.06	5.35 ± 5.77
	Only imitation	14.28 ± 3.28	1.66 ± 0.30	14.31 ± 2.75	1.24 ± 0.65
	Combined	7.26 ± 3.30	1.27 ± 0.58	9.33 ± 2.87	0.77 ± 0.37
The robot controller	STD	89.58 ± 71.58	25.23 ± 37.79	101.63 ± 111.81	31.42 ± 53.86
	Only warm start	56.53 ± 74.77	13.72 ± 25.38	57.97 ± 45.00	13.78 ± 16.67
	Only imitation	43.39 ± 27.59	9.30 ± 9.41	46.99 ± 36.97	10.64 ± 13.86
	Combined	17.90 ± 7.43	2.79 ± 1.79	28.73 ± 23.78	5.32 ± 8.82

the main optimization problem, but it should be large enough to affect the convergence rate. Hence, λ should be adjusted for each task.

4.4 Simulation

In this section, we apply our method to a simulated STS task. To gather the demonstrations, we solve the OCPs without the imitation term for $N = 15$ times and record the human's dynamical characteristics (i.e, m , h , and $\mathbf{R}$) and the coefficients of the CPs for the human assistance prediction and the robot controller. We also record the joint angles of the human and the robot to predict the desired trajectory for the imitation part; however, we map them to a lower dimension as

$$\mathbf{Q} = \Psi \mathbf{w}_q \Rightarrow \mathbf{w}_q = \Psi^\dagger \mathbf{Q}, \quad (4.6)$$

where $\mathbf{Q} = [\mathbf{q}_0^\top, \mathbf{q}_1^\top, \dots, \mathbf{q}_T^\top]^\top$ is the trajectory of T joint angles, $\Psi^\dagger$ is the pseudoinverse of Ψ . Ψ used for mapping each agent's joint angles is the same as its corresponding CPs.

We then test our method on 100 random situations. The evaluation criteria are the convergence rates and the cost values at different iterations. We compare our method with three other baselines, i.e., the standard problem without any modification (STD), using only the warm start method, and using only the imitation term.

We consider two experiments for the STS task. In Experiment 1, we only change the human's disability type and keep the mass and height constant for all demonstrations and test points. In Experiment 2, we change the mass and the height of the human beside its disability type. The variation of the cost weights is considered only for the ankle, knee, and hip joints. The robot is the same for all of the experiments.

Experiment 1: Changing only the human's disability

In this experiment, we assume that we have the same human for the whole experiment whose type of disability is changing. We simulate this by changing the values of $\mathbf{R}$, while keeping the

values of m and h constant ($m = 80$ kg and $h = 1.8$ m). For the demonstrations, we choose elements in $\mathbf{R}$ corresponding to ankle, knee and hip joints randomly among $\{1, 10, 100, 1000, 10000\}$. We compute $N = 15$ demonstrations by solving (4.1) and (4.2). During the test phase, we try different situations by initializing the problem with random joint cost weights. Unlike the demonstrations, the system can choose any value between 1 to 10^4 (not limited to the discrete values). We utilize the logarithmic Euclidean distance for k -NN, which is used for warm starting and extracting desired features $\mathbf{y}_d$. Table 4.2 shows the results of this experiment for both the human assistance prediction and the robot controller. The proposed method outperforms other approaches; comparing it with the STD method, the combined method reduces the number of iterations by around 90% and 80% for the human assistance prediction and the robot controller, respectively. The data in Table 4.2 show that even adding only the imitation term without any warm starting can result in a better convergence rate than the other two methods. The reason why this method affects the human assistance prediction more than the robot controller is that in the former, the cost function is sparse. Namely, it specifies the behavior of the system at a particular time (joint angles and velocities are only relevant at the end of the task, i.e., $\mathbf{q}_T = \mathbf{q}_{\text{stand}}$, $\dot{\mathbf{q}}_T = 0$, and $\ddot{\mathbf{q}}_T = 0$) or when it goes out of the feasible region (joint limits). Adding the imitation term to this problem provides a trajectory reference to follow and makes the problem less sparse. For the robot controller, however, the robot should already follow a trajectory (the human's hand, i.e., $\mathbf{x}_{\text{rob.}} = \mathbf{x}_{\text{Hum.}}$), and adding another trajectory to follow could be less effective. Nevertheless, the proposed method still has remarkable effects on the number of iterations compared to the other three ones.

We present the value of the cost on a logarithmic scale at different iterations for all of the methods in Fig. 4.2. To make a rational comparison between different test points, we normalized all the costs w.r.t. the lowest value achieved from the baseline methods at the convergence. As it is illustrated in this figure, adding the imitation term increases the convergence rate, which is more obvious for the points that start far from the local optimum. On the other hand, warm starting helps the system to be initialized at a lower cost. The combined method takes advantage of both methods and outperforms the other ones. The cost value presented in this figure is just for the emulation part c , as the imitation cost is only introduced to help the emulation and is not part of the original task.

Experiment 2: Changing mass, height and the human's disability

In this experiment, we repeat Experiment 1, but with varying values for the human's mass and height. We assume that the human's mass varies between 60 to 100 kilograms and the height of the human ranges from 1.6 to 2 meters. To generate $N = 15$ demonstrations, we randomly pick among $\{60, 70, 80, 90, 100\}$ and $\{1.6, 1.7, 1.8, 1.9, 2\}$ to describe the human's mass and height, respectively. However, we do not limit the system to these discrete values for the test points, and the system can choose any value in the corresponding range. Since the data are not in the same range, we define closeness for the k -NN method differently from Experiment 1. We first get the logarithms of the cost weights $\mathbf{R}$, then normalize these data as well as the human's

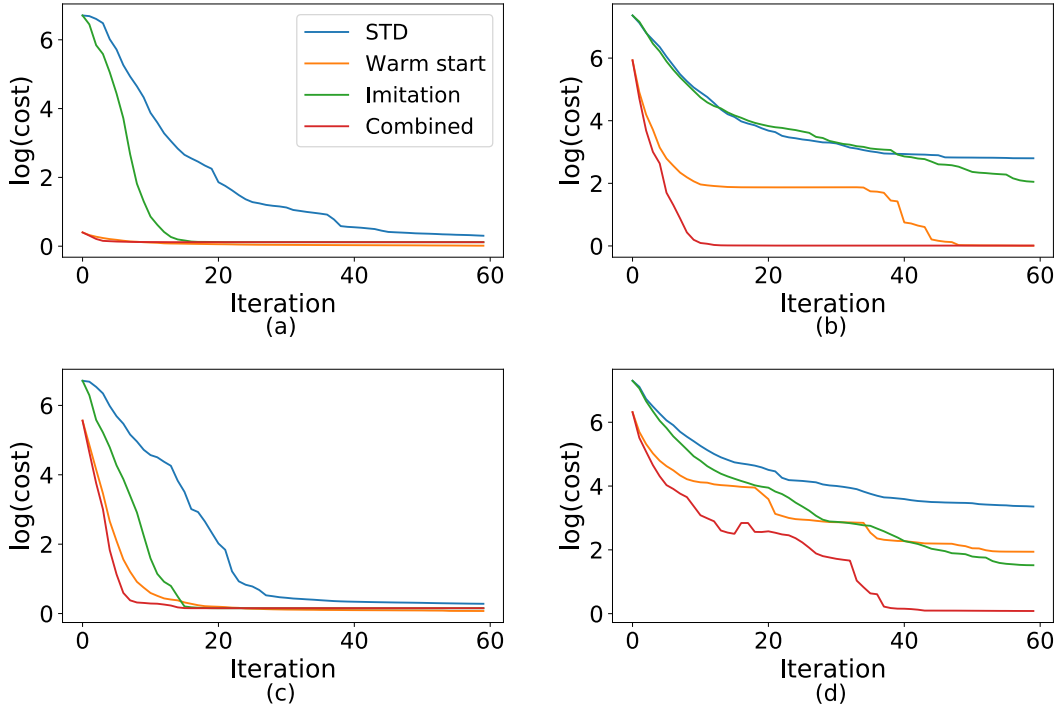


Figure 4.2: Comparing the mean values of the normalized costs in logarithmic scale for Experiment 1 (first row) and Experiment 2 (second row). *Left:* The human assistance prediction. *Right:* The robot controller.

mass and height with their mean values and standard deviations. The distances between the demonstrations and the test points are defined with the Euclidean distance of the normalized data. The results of this experiment are presented in Table 4.2 and Fig. 4.2. Comparing with the STD method, the combined method has reduced the number of iterations by around 90% and 70% in the human assistance prediction and the robot controller, respectively. Although using only the imitation term may initialize iLQR with a higher cost value, it can converge even faster than using only the warm starting method, as shown in Fig. 4.2d.

4.5 Discussion

We can evaluate the proposed method with two criteria: 1- how it affects the convergence rate, and 2- how it affects the cost value. Regarding the convergence rate, as it is already discussed, the combined method has notably reduced the number of iterations. As it is also shown in Fig. 4.2, this method does not have a sensible effect on the cost value; however, it would slightly increase the value of the emulation cost. This behavior is expected, as the system compromises between the emulation part and the imitation part. We can solve this issue by decreasing the value of λ as the number of iterations increases, which is often done in other optimization methods. In this study, however, we keep the value of λ constant.

In some methods, such as direct optimization approaches (Posa et al., 2014), it is possible to warm start the controller with the input commands and state variables simultaneously.

As studied in (T. S. Lembono, Mastalli, et al., 2020), initializing with both the control input commands and the state variables is better than predicting just one of them. Our experiments support this observation. Moreover, our method provides a framework to use any feature that can describe the task more accurately. Hence, one can feed the solver (4.5) with more elaborated features and is not limited only to the optimization variables.

In this experiment, we used k -NN to predict the warm starts or to find the imitation trajectory y_d . It worked pretty well in this experiment, but it is not the case for all of the tasks. One can use any algorithms proposed in the Learning from Demonstration (LfD) field, such as dynamical movement primitives (DMPs) or task parametrized-GMM (TP-GMM) (Calinon & Lee, 2019).

We presented the idea with the manually-adjusted cost functions, but the idea proposed is independent of a specific cost function and should work for most optimization problems. Moreover, one of the reasons to work with the manual cost function was that it gave us the freedom to gather random demonstrations. For real applications, it is not possible, and we are limited to the available options. Therefore, we may not have access to a diverse and rich demonstration set; for example, we may not be able to gather data from a human whose ankle, knee, and hip joints are severely paralyzed. In such situations, we can program the robot to play the role of the assisted human and ask an expert caregiver to simulate the desired help. This technique, known as switching roles between the robot and the human, presents an interesting direction for future studies.

4.6 Conclusion

In this chapter, we showed how imitation can benefit optimal control without introducing additional, non-generalizable modeling assumptions. We augmented the cost with an imitation term that tracks features extracted from demonstrations. Unlike warm-starting, which primarily affects initialization, this term shapes the optimization throughout and can be defined over arbitrary task features rather than only the solver’s decision variables. On a simulated assistive task, the method substantially reduced the number of solver iterations with negligible impact on the final cost value. Conceptually, the pipeline samples experience to obtain an informed initial guess and then projects it to a feasible, optimal solution—a sample-then-project strategy. The next chapter explores a complementary project-then-sample approach.

5 Products of Optimality and Feasibility for Constrained Motion Planning

Collaboration Note:

This chapter is the result of joint work with Teng Xue, who contributed through discussions, experimental design, and paper writing, as well as Suhan Shetty, who supported the use of tensor codes required in this work.

Publication Note:

This chapter is based on the following publication:

- A. Razmjoo, T. Xue, S. Shetty, and S. Calinon, “Sampling-Based Constrained Motion Planning with Products of Experts,” *International Journal of Robotics Research*, conditional acceptance, 2025.

Codes for this chapter are available at: https://github.com/idiap/smpc_poe

In the previous chapter, we combined optimal control with imitation learning, where demonstrations provided richer data and optimal control refined the motion for the task. This can be seen as a sample-then-project approach: samples are drawn first, and then mapped onto the feasible and optimal manifold. However, this wastes many samples in regions that are neither feasible nor optimal, since sampling is performed blindly and feasibility is evaluated only afterward. To address this limitation, we introduce a more proactive sampling strategy that incorporates feasibility during the sampling process. Specifically, we present a project-then-sample strategy, where the optimality distribution is first projected into the feasible space and then sampled. Here, feasibility is interpreted in a more general sense: while in the previous chapter the feasible set corresponded to demonstrations, in this chapter it represents the full constraint-valid set of the task. To realize this, we leverage products of experts, with one expert modeling optimality and the other feasibility. Their combination yields a distribution that promotes efficient exploration and reduces boundary accumulation. We implement this idea using tensor train decomposition, which provides a non-parametric, composable, and efficient sampling framework. Experiments in obstacle avoidance, non-prehensile manipulation, and restricted-volume tasks confirm that the proposed method consistently outperforms existing baselines.

5.1 Introduction

In the previous chapter, we studied how demonstrations can be used to improve constrained motion planning. There, demonstrations were not treated as the exact solution distribution but rather as a source of guidance, either through warm-starting or additional shaping terms. The key assumption was that the solution of a new scenario would remain close to previously observed experiences, even if the demonstration distribution itself varied due to robot embodiment, task parameters, or environment. In other words, demonstrations acted as a heuristic prior, not as a strict feasibility condition.

This approach followed what we called a *sample-then-project* strategy. Samples were first drawn from demonstrations (or other naive distributions) without considering feasibility or optimality, and only afterwards projected toward valid and more optimal solutions. While this strategy proved effective, it suffered from inefficiency: many samples could be wasted in regions that were neither feasible nor useful for optimization.

In this chapter, we take a different perspective. Instead of relying on demonstrations as a guide, we explicitly model the **feasibility distribution**. Here, feasibility is defined not only by demonstrations but by the entire set of states or actions that satisfy the constraints of the task. We then combine this feasibility distribution with an **optimality distribution** through a *Products of Experts (PoE)* formulation (Hinton, 1999). This leads to a *project-then-sample* strategy: the optimality distribution is projected into the feasible region before any sampling takes place. Compared to the previous chapter’s sample-then-project approach, this strategy ensures that samples are inherently feasible and near-optimal, thereby improving efficiency and avoiding wasted effort.

This shift is especially important for **sampling-based motion planning** (Orthey et al., 2023) and **sampling-based Model Predictive Control (SMPC)** (Bhardwaj et al., 2022; Williams, Wagener, et al., 2017). These methods are powerful in scenarios such as contact-rich manipulation (Pezzato et al., 2023) and obstacle avoidance (Bhardwaj et al., 2022), where gradients are difficult to compute. However, their performance is often limited by the lack of efficient sampling mechanisms, especially under constraints or sparse reward signals. For example, as shown in Fig. 5.1a, some samples can cause the robot to move away from the object of interest or even collide with obstacles (Fig. 5.1b), resulting in minimal useful information. By incorporating feasibility directly into the sampling process, our method improves both reliability and sample efficiency in these challenging conditions.

A basic approach to incorporating constraints into sampling-based methods is to generate samples from a distribution naively without considering constraints and subsequently reject those that violate the constraints. This *sample-then-reject* strategy can be highly inefficient in tightly constrained problems, leading to a large number of discarded samples. This inefficiency has motivated the development of numerous alternative approaches in the literature that improve the sampling process to reduce the number of rejected samples. These methods can be categorized into three main approaches. The first category includes projection-based

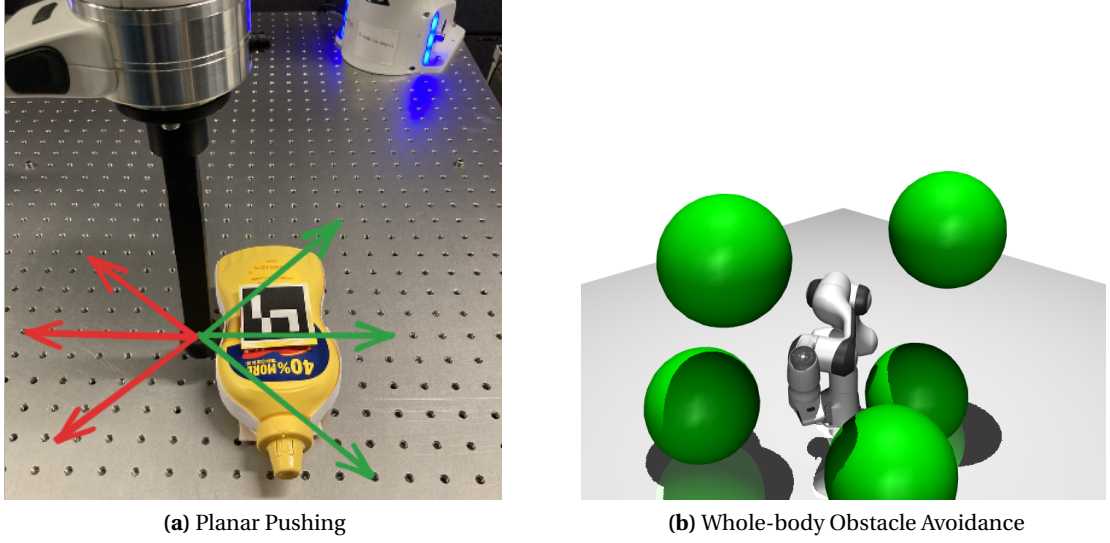


Figure 5.1: In sampling-based Model Predictive Control (SMPC), many samples can be discarded due to inefficiencies or infeasibility. In (a), only the green samples effectively influence the movement of the object, while the actions indicated by the red arrows have no impact on the environment. (b) illustrates the importance of selecting samples that do not collide with obstacles.

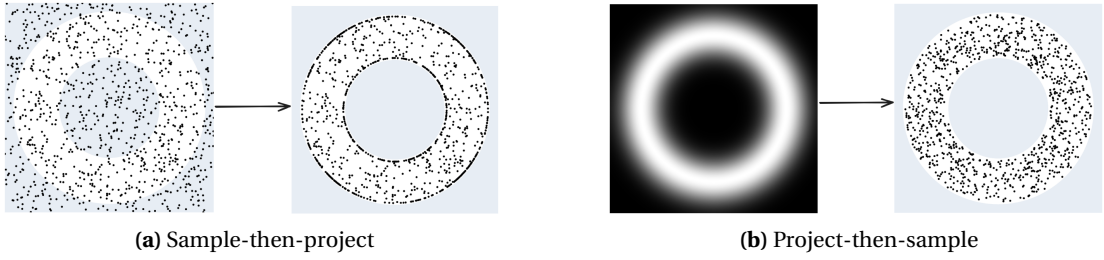


Figure 5.2: An illustrative example highlighting the difference between two sampling strategies from a uniform distribution that meets the task constraints outlined in Fig. 5.3: (a) *sample-then-project* and (b) *project-then-sample*. In (a), many samples cluster on the boundaries, leaving significant areas within the feasible set unexplored, whereas (b) promotes a more thorough exploration. Note that the difference between the two approaches would become more pronounced if the feasible space were narrower.

methods (Jinkyu Kim & Park, 2016; Mirabel et al., 2016), which sample first and then project into the feasible region. However, projection is not always straightforward: some methods (Pardo et al., 2015; Suh et al., 2011) require iterative computation of system gradients to perform the projection, which can be problematic when gradients are difficult to obtain—such as in non-prehensile manipulation. As shown in Fig. 5.2, even with an efficient projection mechanism (often difficult to implement), samples can accumulate near the boundary of the feasible region, leading the system to potentially converge toward that boundary. The second approach involves penalizing constraint violations (Bhardwaj et al., 2022), serving as a data-driven, iterative alternative to projection. However, this method does not guarantee feasible solutions before convergence and may require multiple iterations to achieve feasibility.

The third approach, similar to Power and Berenson, 2023 and Sacks and Boots, 2023, learns a distribution that promotes sampling of feasible and optimal actions or states. While this approach can be effective, it is often complex to implement and computationally expensive (similarly to reinforcement learning methods).

One idea to solve these complexities can be to use several experts. Leveraging multiple expert models to simplify complex tasks is a widely adopted strategy in the machine-learning community. This is commonly implemented either through *mixture of experts* (MoE) (Jordan & Jacobs, 1993), which combines outputs in a manner analogous to an *OR* operation, or *products of experts* (PoE) (Hinton, 1999), which integrates outputs similarly to an *AND* operation. In our framework, since the objective is to identify solutions that are both optimal *AND* feasible, we adopt the *products of experts* paradigm. Beyond their inherent conceptual differences, PoE offers distinct advantages that make it particularly well-suited to our framework. Specifically, PoE enables each expert to focus on their own decision variables and objectives independently, simplifying and streamlining the learning process for each expert. For instance, in the example presented in this study, the expert tasked with optimizing for action optimality can concentrate solely on action variables (as in methods like MPPI), while another expert can focus on both action and state variables, or even exclusively on state variables, prioritizing feasibility without concern for optimization. By combining their outputs, PoE identifies the intersection of their respective distributions, yielding solutions that satisfy all variables and objectives simultaneously. In contrast, MoE typically combines solutions as a weighted average, potentially failing to account for such considerations.

Similarly, in other works on SMPC, such as (Pan et al., 2024), researchers have employed multiple experts to define the overall distribution. In these approaches, each expert specifies a distribution over the full trajectory, and the experts are combined first, either through a summation of cost terms or a product of distributions, before jointly solving the resulting problem. In contrast, our method decomposes the problem differently. The feasibility expert is learned independently and then combined with the optimality distribution. By taking the product of these two distributions, we reduce the probability of the optimality distribution in the infeasible area, thereby improving the efficiency of the sampling process. The optimality distribution can be modeled as a Gaussian distribution using methods such as Model Predictive Path Integral (MPPI), so this part of the problem is computationally efficient and does not require any learning. By introducing a feasibility expert and integrating it with optimality considerations, we obtain the *project-then-sample* strategy, which efficiently focuses the search on the feasible region of the optimization problem, thereby improving overall efficiency.

Fig. 5.3 illustrates our proposed approach, which splits the main problem into two sub-problems: the **optimality problem**, focusing on optimization, and the **feasibility problem**, focusing on constraints. We compute distributions for both subproblems and then combine them with the products of experts approach to create a distribution that is both feasible and optimal. This framework effectively projects the optimality distribution into the feasible region. A key remaining challenge is how to learn these typically complex, multimodal distributions

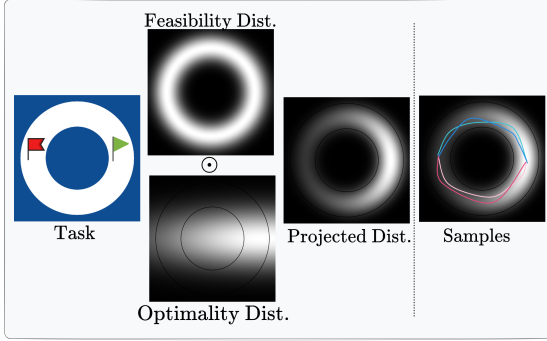


Figure 5.3: An illustrative example of using products of experts. The agent must navigate from the red flag to the green flag while remaining in the white area and avoiding the blue one. The problem is divided into two parts: one models the feasibility distribution, and the other models the optimality distribution. These are combined to project the optimality distribution into the feasible area (Project first), from which trajectories are then sampled (Then sample).

and to sample from them efficiently.

There are three main paradigms for combining experts at the solution level in the literature: Gaussian experts (Calinon, 2016; Hinton, 1999), energy-based models (Gkanatsios et al., 2023), and diffusion-based models (Wang et al., 2024; Yang et al., 2023). Gaussian experts offer closed-form inference but are inherently restrictive in the kinds of multimodal distributions they can represent. Energy-based models are more flexible but impose heavy sampling burdens (e.g., Langevin dynamics or MCMC). Diffusion-based approaches significantly reduce sampling complexity, yet typically require large demonstration datasets. Critically, these models often fail to consider feasibility when they are performing sampling during the denoising process, leading to the rejection of many samples at each step. In this chapter, we show that by integrating a dedicated feasibility expert, we can improve its reliability before sampling and boost per-step sampling efficiency throughout the diffusion process.

In this study, we demonstrate how different experts can be effectively integrated using tensor train (TT) decomposition. Our previous work demonstrated the potential of this method for learning and sampling from desired distributions (Shetty, Lembono, et al., 2024; T. Xue et al., 2024b). We extend these findings by applying TT in the context of products of experts. We compare it to a baseline relying on normalizing flows (NFs) (Dinh et al., 2016), inspired by (Power & Berenson, 2023), to show the advantages of TT over NFs, as well as to discuss the differences between a *project-then-sample* strategy compared to a *sample-then-project* or pure *sample-then-reject* strategies.

In summary, our contributions are threefold:

- Enhancing the sampling efficiency of SMPC methods by introducing the feasibility expert and combining it with the optimality distribution using products of experts (i.e., *project-then-sample* strategy). Particularly, the feasibility distribution is learned separately and then combined with the optimality one before sampling. This enables more effective exploration of the feasible set and extends applicability to problems where gradient calculation is challenging.
- Introducing an effective approach to leverage products of experts through TT density estimation, a non-parametric distribution capable of learning multi-modal distributions.

This approach can also leverage existing straightforward sampling algorithms, enabling efficient exploration of complex distribution spaces.

- Conducting a series of experiments in both simulation and real-world settings to validate the effectiveness of our proposed method compared to baselines.

5.2 Related Work

Gradient-Based vs Sampling-based MPC: Analytical solutions are often impractical for optimization problems in robotics, especially with non-convex, non-linear cost functions and constraints arising from kinematic chains and dynamics. Gradient-based methods like iLQR (Diehl et al., 2005; Todorov & Jordan, 2002) are widely used, offering scalability and local solution updates. However, they face issues such as local minima, and undefined gradients in tasks like contact-rich manipulation (Dafle et al., 2014; Pang et al., 2023). These limitations are critical in Model Predictive Control (MPC), where fast solutions are essential.

To address these challenges, sampling-based methods have gained attention. They bypass the need for gradients, can exploit GPU-based simulators for parallel sampling (Liang et al., 2018), and may reduce sensitivity to local minima by approximating smoothed gradients (Pang et al., 2023). This makes them promising alternatives for complex robotic tasks involving discrete decisions (Chavan-Dafle et al., 2020).

Sampling-based MPC (SMPC): Methods such as Rapidly-Exploring Random Tree (RRT) (Kuffner & LaValle, 2000) and Probabilistic Roadmap Planner (PRM) (Kavraki et al., 1996) for trajectory optimization, as well as approaches such as Cross-Entropy Motion Planning (CEM) (Kobilarov, 2012) and Model Predictive Path Integral (MPPI) (Williams, Wagener, et al., 2017) have shown the effectiveness of sampling strategies in addressing MPC problems. Recent findings by DeepMind (Howell et al., 2022) suggest that even for a challenging task such as quadruped walking, selecting the best sample suffices to achieve the objective. However, these advantages are limited to tasks without substantial constraints.

Constrained Sampling-based Motion Planning: Given that sampling-based methods offer probabilistic completeness, they may require extensive data, posing challenges to finding constraint-free solutions for highly constrained problems. Projection methods are widely used in sampling-based methods, particularly within RRT, to ensure constraints are met and improve sampling efficiency (Kingston et al., 2018). Prior research shows that this approach enables probabilistic completeness (Berenson, 2011), ensuring that the feasible set will be fully explored over time. However, finding an analytical solution for projection remains challenging under most constraints. In specific cases, projection operators can be locally defined using the tangent space and a nullspace projector (Pardo et al., 2015) or by direct sampling within the tangent space (Suh et al., 2011). Nevertheless, this approach has limitations, as it only captures local constraint behavior and fails with non-differentiable constraints (e.g., contact-rich manipulation). To address this, sampling-based methods often use gradient-based

techniques to approximate the projection (Stilman, 2010; Yakey et al., 2001). Some works, such as (Stilman, 2010), frame projection as a secondary optimization problem. An alternative is to relax constraints by introducing penalty terms within the main objective function (Bhardwaj et al., 2022). However, both approaches face limitations, including increased computational demands and the risk of converging to infeasible regions.

This study contends that the above approaches may not be the most efficient, even with an analytical projection solution. Standard projection-based sampling may guide exploration toward regions with limited feasibility (e.g., boundaries), yielding biased solutions and insufficiently exploring the feasible set. A more effective approach could be to directly model the distribution of the feasible and optimal set and sample exclusively from this distribution, concentrating exploration solely within feasible regions. Recent advances in generative models have facilitated this, allowing feasibility-aware sampling strategies (Power & Berenson, 2023; Sacks & Boots, 2023). However, these methods, akin to RL, often require learning highly complex distributions, which can be data-intensive and computationally costly.

Our approach seeks to combine the advantages of both projection and generative methods. We propose a two-stage process: first, we train generative models to approximate the distribution of the feasible region. Rather than projecting individual samples, we then project an estimated optimality distribution into this feasible region using a product of experts, enhancing both the efficiency and focus of sampling within feasible areas. Our approach has two main benefits. Compared to learning a distribution that captures both feasibility and optimality, we focus on the simpler task of feasibility alone and allowing the learned model to be used across tasks with similar feasibility sets. Furthermore, compared to projection-based methods, our approach *project-then-sample* promotes better exploration within the feasible set.

Product of Experts: In SMPC, it is common practice to define the overall cost function or solution distribution by combining multiple terms, typically through summation of cost terms or a product of distributions (Pan et al., 2024). Each term is designed to address a specific aspect of the problem, such as optimality or constraint satisfaction, and the final solution is obtained by solving the joint problem defined by their combination.

In most existing approaches, this combination occurs at the definition level, meaning that the full problem is formed first and then solved using techniques such as gradient descent, Langevin dynamics, or other sampling-based methods. While this is effective, it tends to reduce flexibility, as the individual structures of the constituent terms are lost once they are merged.

In contrast, Hinton’s original formulation of the *product of experts* (Hinton, 1999) suggests an alternative: solving each expert individually and then combining their solutions. By *solution*, we refer to a sampleable distribution, such as a Gaussian, that does not require further optimization or inference steps. This approach preserves the modularity of each expert, enables the use of specialized solvers, and simplifies sampling from the final distribution without re-solving the full joint problem. In this regard, each expert represents a relatively

simple distribution focusing on specific parameters or tasks. These experts are subsequently combined to yield a more refined and sharper distribution. This methodology has found applications in robotics, including the fusion of sensory information (Pradalier et al., 2003), the fusion of robot motion learned in multiple coordinate systems (Calinon, 2016), and across task spaces (Mühlbauer et al., 2024; Pignat et al., 2022). Our work extends these prior approaches by incorporating the product of experts framework within an MPC paradigm.

There are three main approaches in the literature for combining multiple experts at the solution level: (1) **Gaussian experts** (Calinon, 2016; Hinton, 1999), (2) **Energy-based models** (Gkanatsios et al., 2023), and (3) **Diffusion-based models** (Wang et al., 2024; Yang et al., 2023). Each of these methods offers different advantages and trade-offs, which we discuss further to motivate our use of the TT representation.

Among these methods, Gaussian distributions are significantly more limited than TT and other baselines, primarily due to the parametric nature of Gaussian models. Energy-based models (LeCun et al., 2006), which can be viewed as a more general form of Gaussian distributions, indeed hold greater potential. However, according to our experience, sampling from these models typically relies on Langevin dynamics or methods such as MCMC, which can become a significant computational bottleneck. In contrast, TT does not suffer from this issue and allows for efficient and straightforward sampling.

Diffusion models (Ho et al., 2020) (or similar variations such as Flow-Matching (Lipman et al., 2023)) address the sampling challenges present in energy-based models to a significant extent. These models can be broadly categorized into two types: data-based (Wang et al., 2024) and model-based (Pan et al., 2024). Data-based methods require the optimality distribution to be explicitly learned in diffusion form, which typically demands a substantial number of high-quality demonstrations—a requirement not assumed in our work.

Moreover, recent works such as (Pan et al., 2024; H. Xue et al., 2024) have proposed innovative strategies to represent diffusion models in a data-free manner. Their approach, which bears resemblance to annealed MPPI methods, combines multiple experts at the function level. In their pipeline, different candidate solutions are sampled first, followed by a rejection step to discard infeasible samples at each denoising step. By contrast, our approach merges experts at the solution level. This allows us to reshape the sampling distribution before sampling, which is a more proactive method than relying solely on post-hoc sample rejection. Concretely, we inject the feasibility expert’s density into every denoising (sampling) step of the diffusion process, producing candidates that are inherently more likely to satisfy constraints.

5.3 Methodology

5.3.1 Problem Definition

In this chapter, our objective is to solve the optimal control problem defined in (1.2). To simplify notation, consider a system governed by the dynamics $\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t)$. We rewrite the problem as

$$\begin{aligned} \mathbf{U}_t^* = \operatorname{argmin}_{\hat{\mathbf{U}}_t} & \sum_{h=0}^{H-1} c_h(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}) + c_H(\hat{\mathbf{x}}_{t+H}), \\ \text{s.t. } & \mathbf{g}(\hat{\mathbf{x}}_{t+h+1}, \hat{\mathbf{u}}_{t+h}) \leq \mathbf{0}, \quad \forall h \in \{0, 1, \dots, H-1\}. \end{aligned} \quad (5.1)$$

Here, $\mathbf{g}(\cdot) \in \mathbb{R}^{n_g}$ denotes the vector of n_g constraint functions, which may include both inequality and equality constraints. Equality constraints can be represented as a pair of inequalities, for example, by expressing $f(\cdot) = 0$ as $f(\cdot) \leq 0$ and $-f(\cdot) \leq 0$. It is important to note that the constraint set may incorporate prior knowledge that we expect the system to follow. For instance, in a pushing task, one constraint could ensure that actions taken at each step influence the object and prevent the robot from moving away from it beyond a certain threshold, as illustrated in Fig. 5.1a.

5.3.2 Products of Experts

Similar to other SMPC methods, we assume that the solution to this problem can be modeled as a distribution $p(\hat{\mathbf{U}}_t | \mathbf{x}_t)$. Furthermore, we assume that this distribution can be expressed as a product of two other distributions

$$p(\hat{\mathbf{U}}_t | \mathbf{x}_t) \propto p^o(\hat{\mathbf{U}}_t | \mathbf{x}_t) p^f(\hat{\mathbf{U}}_t | \mathbf{x}_t). \quad (5.2)$$

The first distribution p^o captures the dynamic aspect of the task and ensures optimality (i.e., the optimality distribution, Expert 1), while the second distribution, p^f represents the feasible area (i.e., the feasibility distribution, Expert 2).

Expert 1: We can solve (5.1) using the MPPI method by incorporating the constraint within the cost function. This approach enables the system to gradually converge to the feasible area, although it may lead to time-consuming iterations and the rejection of numerous samples due to high-cost values. This expert addresses the following problem:

$$\begin{aligned} \mathbf{U}_t^* = \operatorname{argmin}_{\hat{\mathbf{U}}_t} & \sum_{h=0}^{H-1} c_h(\hat{\mathbf{x}}_{t+h}, \hat{\mathbf{u}}_{t+h}) + \\ & c_g(\mathbf{x}_{t+h+1}, \mathbf{u}_{t+h}) + c_H(\hat{\mathbf{x}}_{t+H}), \end{aligned} \quad (5.3)$$

where $c_g(\cdot)$ denotes the cost term associated with constraint violations in $\mathbf{g}$. Although Expert 1 is primarily responsible for modeling the optimality distribution, the inclusion of the additional cost term $c_g(\cdot)$ offers several advantages. First, by assigning significantly higher

values to this term relative to others, infeasible samples are effectively assigned zero weight via (2.15), thereby removing them from consideration. Secondly, it allows us to express the objective in a more general form, consistent with how similar formulations are structured in related literature, thereby making Expert 1 directly comparable to prior approaches. Third, this additional term introduces flexibility to account for constraints that the feasibility expert may not fully consider—since that expert may capture only some, but not all, constraints—and for others that may arise during task execution but are not explicitly represented by Expert 2 (described later). Finally, from a practical standpoint, incorporating $c_g(\cdot)$ into Expert 1 can enhance efficiency by guiding the optimization using relevant constraint information, even when a separate feasibility expert is also present. In this chapter, we follow prior works (Bhardwaj et al., 2022; Jankowski et al., 2023; Power & Berenson, 2023) by assigning a large weight to this term in order to more effectively reject infeasible solutions. Bhardwaj et al., 2021 provides a detailed analysis of how increasing this penalty improves feasibility under tight constraints.

The solution to (5.3) using the MPPI method can be formulated as

$$p^o(\hat{\mathbf{U}}_t) \propto \prod_{h=0}^{H-1} \mathcal{N}(\hat{\mathbf{u}}_{t+h} | \boldsymbol{\mu}_{t+h}, \boldsymbol{\Sigma}_{t+h}), \quad (5.4)$$

which represents a Gaussian distribution at each time step, characterized by the mean $\boldsymbol{\mu}_{t+h}$ and covariance matrix $\boldsymbol{\Sigma}_{t+h}$.

Expert 2: handling constraints exclusively through a rejection mechanism, for example by assigning high values to the c_g term, is generally inefficient. It wastes computational resources by generating and then discarding infeasible trajectories. Our proposed method addresses this by introducing a *feasibility expert* that proactively biases the sampling distribution toward feasible regions. This improves the quality of samples before they are evaluated and potentially rejected, leading to several key benefits as discussed in this chapter.

To modify the sampling distribution of equation (5.4), we introduce another unnormalized distribution $p^f(\cdot)$ that focuses on the feasible set $\mathcal{F} : \{(\hat{\mathbf{x}}, \hat{\mathbf{u}}) \mid \mathbf{g}(\hat{\mathbf{x}}, \hat{\mathbf{u}}) \leq 0\}$. We define this function as

$$p^f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) = \exp\left(-\frac{\text{ReLU}(\mathbf{g}(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t))}{\lambda}\right), \quad (5.5)$$

where λ is a temperature parameter. Various methods, including different generative models, can be employed to convert this function into a proper distribution. This study chooses the approach outlined in (Shetty, Lembono, et al., 2024). This method discretizes the function (5.5) and transforms it into an unnormalized distribution. The resulting distribution can be represented using tensor cores (as defined in Sec. 2.4), enabling a lower memory footprint and seamless integration with the other experts, along with a straightforward sampling method from the new distribution.

After converting the constraint function (5.5) into a distribution in TT format, we can express

it as

$$p^f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) \rightarrow \mathcal{P}_f(\hat{\mathbf{u}}_t, \hat{\mathbf{x}}_t) = \prod_{n=0}^{d_x} \mathcal{P}_{:,i_{\hat{\mathbf{x}}_n},:}^n \prod_{m=0}^{d_u} \mathcal{P}_{:,i_{\hat{\mathbf{u}}_m},:}^{d_x+m}, \quad (5.6)$$

where $\hat{x}_n$ and $\hat{u}_m$ are respectively n -th and m -th value of the state $\mathbf{x}_t$ and action $\mathbf{u}_t$ vectors. As discussed in (Shetty, Lembono, et al., 2024), it is possible to derive the conditional distribution in TT format $\mathcal{P}_f(\hat{\mathbf{u}}_{t+h}|\hat{\mathbf{x}}_{t+h})$ from the joint distribution obtained in (5.6). Consequently, assuming that this distribution is time-independent (though it can be readily extended to a time-dependent version by introducing time as an additional variable into the system), we can define the second expert as

$$p^f(\hat{\mathbf{U}}_t|\mathbf{x}_t) \propto \prod_{h=0}^{H-1} \mathcal{P}_f(\hat{\mathbf{u}}_{t+h}|\hat{\mathbf{x}}_{t+h}). \quad (5.7)$$

5.3.3 TT-PoE-MPPI

By substituting the values derived in (5.7) and (5.4) into (5.2), we can express the distribution as

$$p(\hat{\mathbf{U}}_t|\mathbf{x}_t) \propto \prod_{h=0}^{H-1} \mathcal{N}(\hat{\mathbf{u}}_{t+h}|\boldsymbol{\mu}_{t+h}, \boldsymbol{\Sigma}_{t+h}) \mathcal{P}_f(\hat{\mathbf{u}}_{t+h}|\hat{\mathbf{x}}_{t+h}). \quad (5.8)$$

Next, we focus on a specific time step h , where the sampling can be performed from the distribution

$$p^h(\hat{\mathbf{u}}_{t+h}|\mathbf{x}_t) \propto \mathcal{N}(\hat{\mathbf{u}}_{t+h}|\boldsymbol{\mu}_{t+h}, \boldsymbol{\Sigma}_{t+h}) \mathcal{P}_f(\hat{\mathbf{u}}_{t+h}|\hat{\mathbf{x}}_{t+h}). \quad (5.9)$$

We can rewrite the conditional distribution as

$$p^h(\hat{\mathbf{u}}_{t+h}|\hat{\mathbf{x}}_{t+h}) \propto \mathcal{N}(\hat{\mathbf{u}}_{t+h}|\boldsymbol{\mu}_{t+h}, \boldsymbol{\Sigma}_{t+h}) \mathcal{P}_f(\hat{\mathbf{u}}_{t+h}, \hat{\mathbf{x}}_{t+h}). \quad (5.10)$$

This product can be computed efficiently by introducing a new tensor, $\mathcal{P}_{\mathcal{N}}(\hat{\mathbf{u}}_{t+h})$, which represents the tensor form of the $\mathcal{N}(\cdot)$ function. Consequently, the product can be expressed as

$$p^h(\hat{\mathbf{u}}_{t+h}|\hat{\mathbf{x}}_{t+h}) \propto \mathcal{P}_{\mathcal{N}}(\hat{\mathbf{u}}_{t+h}) \mathcal{P}_f(\hat{\mathbf{u}}_{t+h}, \hat{\mathbf{x}}_{t+h}). \quad (5.11)$$

Although this product can be computed for any distribution, if we assume the covariance matrix is diagonal (i.e., independent actions in different dimensions), $\boldsymbol{\Sigma}_{t+h} = \text{diag}(\boldsymbol{\sigma}_{t+h}^2)$, the computation becomes more efficient, allowing it to be performed directly at the core level. This assumption is common (Power & Berenson, 2023; Sacks & Boots, 2023) and is crucial as it enables efficient multiplication of these distributions.

It is important to note that the cost value is calculated via (5.1), and thus the sample weights in (2.15) are computed by considering the actions across all times and dimensions collectively.

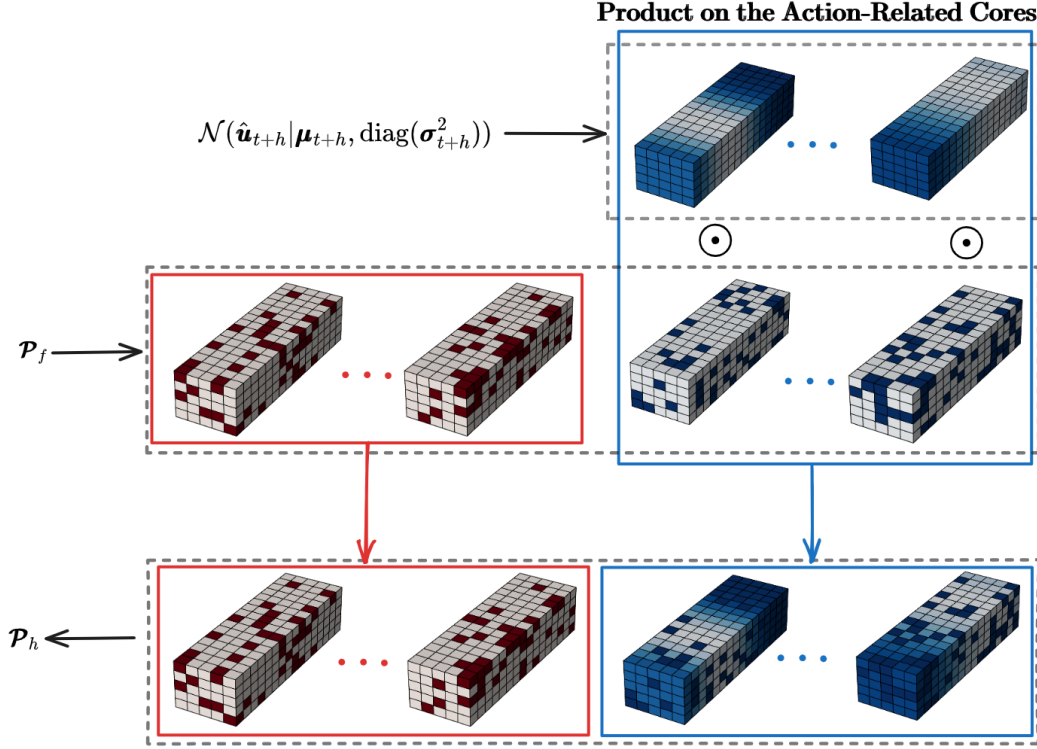


Figure 5.4: The pipeline employed in this chapter integrates two distributions. Red, and blue colors indicate the cores related to the state and action variables, respectively. Initially, the optimality distribution is transformed into TT-cores, followed by elementwise multiplication with the feasibility distribution cores corresponding to the action space. Subsequently, a new distribution is generated using the updated cores. In the illustration, white boxes represent elements equal to 1, while darker boxes denote elements equal to 0. It is important to note that this visualization is simplified for clarity: in actual scenarios, the cores would not strictly be 0 or 1.

Consequently, their influence on one another is captured in the desired action command. However, the independence assumption simplifies the learning and sampling process. Given this assumption, the elementwise multiplication can be independently performed at each TT-core as

$$p^h(\hat{\mathbf{u}}_{t+h} | \hat{\mathbf{x}}_{t+h}) \propto \mathcal{P}_h \approx \prod_{n=0}^{d_x} \mathcal{P}_{:,i_{\hat{x}_n}}^n \cdot \prod_{m=0}^{d_u} \left(\mathcal{P}^{d_x+m} \odot \mathcal{P}_{\mathcal{N}_m} \right)_{:,i_{\hat{u}_m}}, \quad (5.12)$$

where $\odot$ denotes elementwise multiplication, and $\mathcal{P}_{\mathcal{N}_m}$ has the same dimension as $\mathcal{P}^{d_x+m}$, and its elements can be described by

$$\mathcal{P}_{\mathcal{N}_m, i_{\hat{u}_m}} = \frac{1}{\sigma_{t+h}^m \sqrt{2\pi}} \exp \left(-\frac{1}{2} \left(\frac{\hat{u}_m - \mu_{t+h}^m}{\sigma_{t+h}^m} \right)^2 \right), \quad (5.13)$$

where μ_{t+h}^m and σ_{t+h}^m represent the values of $\boldsymbol{\mu}_{t+h}$ and $\boldsymbol{\sigma}_{t+h}$, respectively, corresponding to the m -th dimension. The schematic process for this step at time $t+h$ is presented in Fig. 5.4.

Algorithm 1 TT-PoE-MPPI

Require: $\mathbf{x}_t, \boldsymbol{\mu}_{t,\dots,t+H-1}, \boldsymbol{\sigma}_{t,\dots,t+H-1}$
Return: $\mathbf{u}_t^*$

- 1: $\hat{\mathbf{x}}_t \leftarrow \mathbf{x}_t$
- 2: **for** h from 0 to $H-1$ **do**
- 3: Calculate $\mathcal{P}_{\mathcal{N}_m}$ ▷ Eq. (5.13)
- 4: $\mathcal{P}_h \leftarrow$ Product of the distributions ▷ Eq. (5.12)
- 5: $\hat{\mathbf{u}}_{t+h}^0, \dots, \hat{\mathbf{u}}_{t+h}^N \leftarrow$ Sample N actions
- 6: $\hat{\mathbf{x}}_{t+h+1}^0, \dots, \hat{\mathbf{x}}_{t+h+1}^N \leftarrow$ Forward Dynamics
- 7: **end for**
- 8: $c^0, \dots, c^N \leftarrow$ Calculate cost values
- 9: $\boldsymbol{\mu}_t \leftarrow$ update the mean value ▷ Eq. (2.14)
- 10: $\mathbf{u}_t^* \leftarrow \boldsymbol{\mu}_t$

Once we determine the desired distribution, we employ the same procedure as in SMPC methods. We sample a set of actions, evaluate their associated cost values through a forward path, update the mean values accordingly, and iterate through this process until the task is completed. The complete procedure is outlined in Algorithm 1.

5.3.4 Decomposable feasible distribution

The same idea can be applied to facilitate the learning of more complex feasibility distributions from simpler ones. For instance, without loss of generality, consider a scenario where the feasible area can be defined by two constraints, $g_1(\cdot)$ and $g_2(\cdot)$. Instead of learning the feasible set for both constraints simultaneously, one could learn them separately as $\mathcal{P}^{f_1}$ and $\mathcal{P}^{f_2}$. This approach not only simplifies the problem but also increases efficiency. In some parts of the task, we may only need to consider one of the constraints, eliminating the need to relearn both.

The feasible set for the entire problem can be easily computed by combining the two separate (TT) models as

$$\mathcal{P}^f = \mathcal{P}^{f_1} * \mathcal{P}^{f_2}, \quad (5.14)$$

where the symbol $*$ denotes the product of the two TT models. To calculate this product, we do not need access to the entire tensor, as it can be efficiently computed at the core level. Libraries such as tntorch (Usvyatsov et al., 2022) can facilitate this process.

As a side note, when the feasible regions of two constraints overlap, we can use a relaxed combination via summation rather than multiplication as

$$\mathcal{P}^f = \mathcal{P}_1^f + \mathcal{P}_2^f, \quad (5.15)$$

where $+$ denotes arithmetic operations between two TT models. While a product-of-experts formulation is often preferred, the additive combination retains the same high-probability

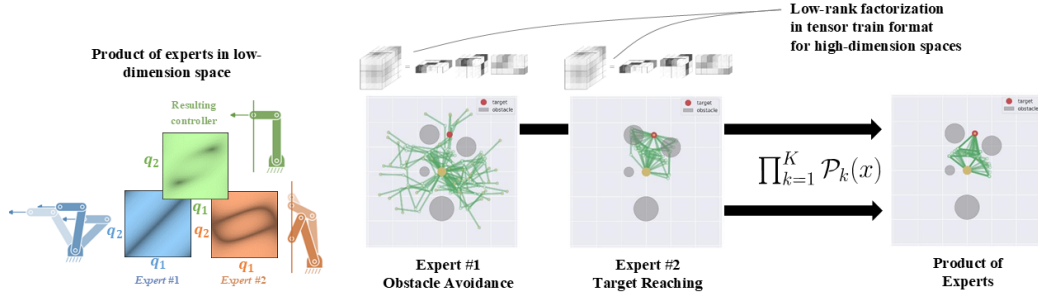


Figure 5.5: Example of utilizing multiple experts to extract the feasible set. *Left:* Combining one expert that aligns the end-effector horizontally and another that positions the end-effector on a desired vertical line. The resulting distribution satisfies both constraints. The matrices show the corresponding configuration distribution with the darker values showing the desired configurations. *Right:* Similarly, combining models for obstacle avoidance and target reaching generates configurations (green lines) that meet both goals.

overlap of feasible regions and thus remains effective at guiding the sampler toward the feasible set. Moreover, this relaxed *sum-of-experts* requires fewer ranks and can be more practical in high-dimensional settings.

Fig. 5.5 demonstrates two examples of leveraging multiple expert models to address complex tasks. On the left, one expert models the joint angles for achieving a specific orientation, while another focuses on positions along a desired vertical line. The resulting combined distribution respects both constraints and is more sparse compared to the individual models, as evident from the corresponding matrices shown in the figure. On the right, one expert focuses on obstacle avoidance, while another models the task of reaching the target. The combined distribution ensures that the target is reached while avoiding obstacles, effectively balancing both objectives. This approach highlights the power of combining simpler expert models to manage complex scenarios. Furthermore, the modularity of this method allows constraints to be selectively applied. For instance, in the right example, the obstacle avoidance model can be prioritized for intermediate trajectory points, while the target-reaching distribution is only necessary for the endpoint.

Moreover, this concept extends beyond simple AND operations. Indeed, we can also consider other logical behaviors such as OR operations. Without delving into further details, we can express the distribution that satisfies at least one of the constraints g_1 or g_2 as

$$\mathcal{P}^f = \mathcal{P}^{f_1} + \mathcal{P}^{f_2} - \mathcal{P}^{f_1} * \mathcal{P}^{f_2}, \quad (5.16)$$

These operations can further assist in solving more complex problems by focusing on simpler ones, allowing us to efficiently utilize the learned models instead of relearning them for each situation. Moreover, logical operations provide a high-level language to compose and orchestrate controllers, enabling the cost function and task constraints to be systematically constructed, modified, and combined.

5.3.5 Data Gathering

To learn the TT representation of the feasibility distribution, we must first collect data that captures how the system behaves under different configurations and constraints. This data can be gathered either once at the beginning or incrementally during the TT-cross procedure, depending on the chosen learning strategy.

A key requirement is a cost function that evaluates whether a sampled point is feasible or not. This evaluation is carried out using the constraint cost defined in (5.5). For constraints that depend on the next state of the system, a forward dynamics model is also required. Depending on the task, two approaches can be used:

- **Simplified analytical dynamics**, such as a single-integrator model. This is used in tasks like the PNRGRID example, where the goal is to study feasibility over a low-dimensional configuration space without requiring a high-fidelity simulator.
- **Physics-based simulators**, when contact interactions or friction effects matter. In such cases, we use Isaac Gym to generate data efficiently, as its parallelization capabilities allow thousands of evaluations per iteration.

The resulting dataset therefore contains: (i) state–action pairs covering a wide range of configurations, and (ii) their corresponding feasibility values, computed with the cost function or obtained through forward simulation. This produces a structured dataset that allows the TT model to learn both feasible and infeasible regions, including the sharp transitions created by hard constraints.

5.4 Results

We applied our methodology to several tasks, verifying the effectiveness of the proposed approach by conducting a comparative analysis with other baselines, namely MPPI, Proj-MPPI (when possible) and NFs-PoE-MPPI. In the MPPI baseline, constraints are integrated directly into the main cost function (i.e., as relaxed constraints), which aligns with the most widely accepted approach in the literature employing this method. In Proj-MPPI, each sample generated by the MPPI method was initially projected into the feasible region before being processed in the MPPI pipeline. The NFs-PoE-MPPI method leverages normalizing flows (NFs) (Dinh et al., 2016) to learn the feasibility distribution, taking inspiration from FlowMPPI (Power & Berenson, 2023). To learn this distribution with NFs, we used the same amount of data as for TT, ensuring that all data points were positive. We also maintained consistency in the number of data points across methods to enable a fair comparison. Details regarding the hyperparameters used in various learning methods across different tasks are provided in Table A.1.

The core problem, particularly the optimality distribution, is identical for all baselines. Specifically, each method employs a high-penalty term with the same value within the cost function to reject infeasible samples. This mechanism is critical for the constrained tasks considered in this chapter; without it, even a few infeasible samples that score well on the unconstrained cost could degrade performance. This approach is consistent with prior work (Bhardwaj et al., 2022; Jankowski et al., 2023; Power & Berenson, 2023). When comparing baselines, we therefore focus on evaluating the unique components of each method, such as the feasibility expert in PoE-MPPI, rather than the shared rejection mechanism. This separation allows us to independently assess the contribution of these additional components.

The comparison is based on three metrics: the success rate, the total cost, and the number of steps required to reach the goal. To ensure fair comparison despite the varying complexities of different task parameters, we normalize the results of each baseline method to those of the MPPI method. We then report the mean for the logarithms of these normalized values. Thus, negative values indicate better performance relative to the MPPI method.

Additionally, we conduct experiments with different sampling regimes, specifically with 16 samples for a low-sampling regime, 64 samples for a medium-sampling regime, and 512 samples for a high-sampling regime. Further information regarding the experiments, including the definition of the cost function and hyperparameter details, is available in the Appendix.

Finally, we demonstrate that our framework scales to high-dimensional problems and can seamlessly integrate with alternative optimality distributions—such as the diffusion-based DIAL-MPC (H. Xue et al., 2024).

5.4.1 Obstacle Avoidance (PNGRID)

The PNGRID task, as utilized in prior studies like (Sacks & Boots, 2023), is an instance of the widely studied obstacle-avoidance problem in the SMPC field. This task, illustrated in Fig. 5.6, involves a 2D environment where an agent must navigate from a randomly selected start point to a randomly assigned target while avoiding static obstacles marked in orange. For this task, the feasibility distribution captures the obstacle-free areas, directing sampling toward viable paths.

Table 5.1 presents a comparative analysis of system performance across baseline approaches, based on averages from 100 trials with random start and target points. All Proj-MPPI, NF-PoE-MPPI and TT-PoE-MPPI demonstrate performance improvements over the MPPI method, indicated by negative values, consistent with prior findings (Power & Berenson, 2023; Sacks & Boots, 2023). Notably, the TT-PoE-MPPI method consistently outperforms the other methods, particularly in terms of success rate within the medium to low sampling regime. However, as the number of samples increases, both NFs-PoE-MPPI and TT-PoE-MPPI exhibit improved performance in terms of step efficiency and cost values. In contrast, Proj-MPPI’s performance aligns more closely with MPPI under high-sampling conditions than in the lower and medium

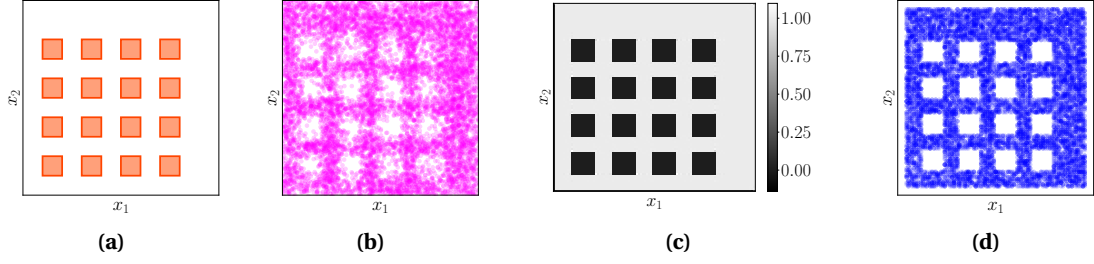


Figure 5.6: PNGRID Configuration. (a) Depiction of the primary environment, with orange rectangles representing obstacles. (b) Samples generated from a GMM model using $k = 400$ Gaussian components, trained with expectation-maximization (EM) algorithm. (c) Distribution learned through the TT-SVD method, and the samples generated from this distribution are depicted in (d). As evident from the results, the TT-SVD method precisely captures the feasible set, with the generated samples exhibiting a higher concentration within the feasible area compared to those generated by the GMM model.

Table 5.1: Comparison of various methods across different tasks employing different sampling regimes. Values are normalized to the MPPI method. Hence, the number of steps and cost values for the MPPI method are not provided. Negative values indicate superior performance compared to the MPPI method, with bold formatting indicating superior results.

Task	#samples	MPPI succ.	Proj-MPPI			NFs-PoE-MPPI			TT-PoE-MPPI (proposed)		
			succ.	$\log(\#steps_n)$	$\log(cost_n)$	succ.	$\log(\#steps_n)$	$\log(cost_n)$	succ.	$\log(\#steps_n)$	$\log(cost_n)$
PNGRID	16	46%	70%	-0.07	-0.19	74%	-0.03	-0.04	96%	-0.81	-0.35
	64	81%	92%	-0.06	-0.09	90%	-0.10	-0.16	100%	-0.69	-0.90
	512	93%	96%	-0.06	-0.09	97%	-0.13	-0.16	100%	-0.65	-0.78
Sphere. Man.	16	73%	73%	-0.11	-0.09	75%	0.01	-0.57	100%	-0.32	-1.08
	64	93%	77%	0.03	-1.57	87%	0.20	-0.68	100%	-0.39	-4.38
	512	100%	78%	0.05	-1.89	100%	-0.26	-1.97	100%	-0.89	-3.08
Sin. Man.	16	31%	38%	-0.09	-1.08	38%	-0.15	-0.08	100%	-1.58	-1.86
	64	44%	44%	-0.09	-0.87	50%	-0.41	-0.40	100%	-2.37	-2.03
	512	55%	68%	-0.30	-0.68	65%	-0.52	-0.14	100%	-1.92	-2.40

sampling regimes. This trend highlights the competitive advantage of the TT-PoE-MPPI and NFs-PoE-MPPI approaches across varying sampling densities.

5.4.2 Tracking Tube (e.g., surface cleaning or painting)

The proposed method is also applied to a task where the robot's end-effector must traverse a restricted volume (e.g., a spherical or a sinusoidal manifold with some margins, as illustrated in Fig. 5.7). Such tasks are prevalent in robotics. For instance, when a robot needs to clean or paint a surface or follow a specific path. In this setup, we control the robot's end-effector position in the task space, ensuring it stays within a designated shaded area. However, the feasible region for this task is substantially restricted relative to the entire workspace, making it challenging to sample feasible actions. We employed two different volumes: one between two spheres and another between two sinusoidal waves (Fig. 5.7). The evaluation results, shown in Table 5.1, reveal that the proposed method significantly improves the task success rate, especially in low-sampling scenarios. Moreover, our method achieves faster convergence with lower associated costs. As shown in the table, Proj-MPPI exhibits behavior similar to

MPPI in terms of success rate (and even lower in the sphere case) and number of steps, while achieving significantly lower cost values. This outcome is likely due to the projection step, which ensures that nearly all samples remain within the designated area, leading to reduced costs. However, this projection mechanism appears to make the system more inclined to stay within its current location rather than actively explore new tasks, which may contribute to its lower success rate on the sphere manifold compared to MPPI. In contrast, NFs-PoE-MPPI shows markedly lower performance than TT-PoE-MPPI, for reasons that are discussed in detail in Section 5.5.

In this experiment, our objective was to evaluate the performance of our systems on equality constraints, which in this case correspond to a manifold. However, we acknowledge that the TT method, which is a volumetric representation, cannot directly enforce strict equality constraints. To address this limitation, we evaluated the performance of our approach on a relaxed version of the problem by introducing margin boundaries around the manifold, effectively creating a "soft" constraint region for analysis. We then evaluate how the pipeline behaves as this margin is progressively reduced, and benchmark it against an intrinsic-sampling approach (i.e., Riemannian log/exp maps), which guarantees constraint satisfaction intrinsically by sampling in the tangent space first, then mapping to the task space. For certain manifolds, like a sphere, these mappings can be computed analytically (see Appendix for details).

Table 5.2 presents the results of this comparison. We evaluate our proposed method against the Riemannian log/exp maps approach (specifically for a spherical manifold) and a vanilla-MPPI method. As shown in Table 5.2, when the margin exceeds the TT discretization (2 cm), our TT-PoE-MPPI method outperforms both standard MPPI and the Riemannian mapping approach. We attribute this to the fact that intrinsic sampling in the tangent space incurs inefficiencies, particularly when shifting between time steps and averaging, whereas the TT representation captures the volumetric structure more globally. As the margin shrinks, however, the intrinsic method ultimately yields the best performance, although our method remains superior to vanilla MPPI.

These results suggest that TT-PoE-MPPI can accelerate the dynamic sampler under equality constraints, but that pure intrinsic sampling may be preferable for very tight margins. Moreover, TT-PoE can be combined with intrinsic approaches: once an intrinsic transform renders the problem unconstrained, additional constraints (e.g., joint limits or other inequalities) can still be enforced by defining the TT experts in the transformed space. Note that the optimality distributions for the intrinsic sampler and for other methods inhabit different spaces. Hence, as the margin shrinks to zero, the unconstrained intrinsic sampler can serve as the ground-truth solution to the constrained problem. Furthermore, our product-of-experts framework consistently improves sampling efficiency without imposing additional structural assumptions, such as that the constraint space forms a Riemannian manifold or that the constraints are affine in the action variables, as required by Carius et al., 2021.

Table 5.2: Comparison of different methods on sphere tube-tracking task across varying margin sizes. The intrinsic MPPI method (MPPI solved directly on the spherical manifold using log/exp maps) reformulates the problem as unconstrained and, in the limit as the margin $\rightarrow 0$, serves as the *ground truth* for the other baselines. Values are normalized to the MPPI method. Hence, the number of steps and cost values for the MPPI method are not provided. Negative values indicate superior performance compared to the MPPI method, with bold formatting indicating superior results.

Margin size	MPPI	TT-PoE-MPPI			Intrinsic MPPI		
	succ. rate	succ. rate	$\log(\#steps_n)$	$\log(cost_n)$	succ. rate	$\log(\#steps_n)$	$\log(cost_n)$
5 [cm]	98%	100%	-0.98	-2.94	100%	-0.72	-2.55
3 [cm]	91%	100%	-1.06	-4.03	100%	-0.81	-3.78
1 [cm]	78%	92%	-0.02	-4.91	100%	-0.37	-5.73

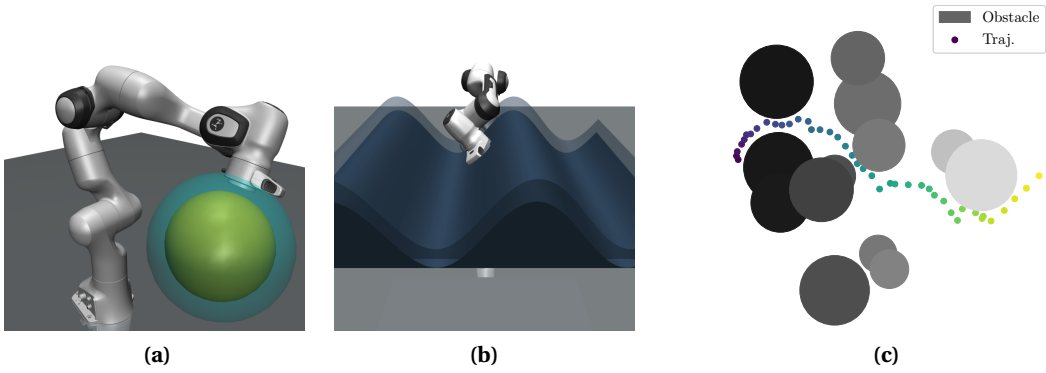


Figure 5.7: Different environmental setups: (a) sphere tube tracking, (b) sinusoidal wave tube tracking, and (c) Online obstacle-avoidance. In (c), the obstacles become more apparent as the robot approaches them; lighter colors indicate obstacles that appear later in the task.

5.4.3 Non-prehensile Manipulation (Planar-pushing)

An other task that we consider is a pushing task (Mason, 1986; T. Xue et al., 2023), a type of non-prehensile, contact-rich manipulation. These tasks are inherently challenging as they require the robot to handle a range of contact models, with hybrid dynamics that complicate the use of gradient-based methods. This hybrid nature arises from various factors, including the geometry of objects, the contact modes, as well as the task level. In this work, we focus specifically on the challenges related to geometry and contact mode, while task-level hybridization has been more extensively explored within task and motion planning (TAMP) frameworks (Holladay et al., 2023).

Previous studies have attempted to address the hybrid nature of contact through unified contact modes (Anitescu, 2006; Manchester & Kuindersma, 2020; Posa et al., 2014; Yunt & Glocker, 2006) or by employing mixed-integer programming (MIP) to jointly optimize hybrid and continuous variables (Aceituno & Rodriguez, 2022). However, these approaches struggle with geometric hybridness, and solving complex non-linear, non-convex problems with MIP can be computationally intensive. SMPC offers an effective alternative (Pezzato et al., 2023), as it avoids reliance on gradients. Furthermore, as demonstrated by Pang et al., 2023, the inherent

Table 5.3: Comparison of various methods across different tasks employing different sampling regimes. Values are normalized to the MPPI method; hence, the number of steps and cost values for the MPPI method are not provided. Negative values indicate superior performance compared to the MPPI method, with bold formatting indicating superior results.

Task	#samples	MPPI	NFs-PoE-MPPI			TT-PoE-MPPI		
		succ. rate	succ. rate	$\log(\#steps_n)$	$\log(cost_n)$	succ. rate	$\log(\#steps_n)$	$\log(cost_n)$
Push. Cir.	16	89%	96%	-0.19	-0.22	95%	-0.35	-0.41
	64	96%	99%	-0.27	-0.30	100%	-0.38	-0.45
	512	97%	100%	-0.24	-0.27	100%	-0.38	-0.45
Push. Box	16	100%	100%	0.29	0.28	100%	-0.17	-0.21
	64	100%	100%	-0.06	-0.08	100%	-0.25	-0.29
	512	100%	100%	0.25	0.23	100%	-0.01	-0.03
Push. Mus.	16	70%	73%	0.12	0.11	89%	-0.21	-0.25
	64	75%	90%	-0.01	-0.02	91%	-0.33	-0.34
	512	75%	90%	0.01	-0.01	91%	-0.26	-0.28

ability of sampling-based methods to smooth functions proves advantageous in these hybrid and contact-rich scenarios.

Using the method developed in this work, the optimality expert can focus on pushing an object toward a goal while adapting to environmental changes, such as the introduction of obstacles that were not present in the training data. The feasibility distribution is designed to model the efficiency of the actions taken. As mentioned earlier, certain actions are inconsequential if they lead the robot away from the object or otherwise fail to move the object meaningfully. In an offline phase, we gather data on various state-action combinations, labeling them as efficient if the robot moves toward the object or if the action causes the object to relocate. Specifically, $p^f(\mathbf{x}_t, \mathbf{u}_t)$ in (5.5) returns 1 if the action $\mathbf{u}_t$ in state $\mathbf{x}_t$ is effective and 0 otherwise. This approach is similar to embedding a cost function that encourages the robot to approach the object. However, we demonstrate that our method is more efficient than relying on a penalty term within the cost function.

In this experimental setup, we employed three objects: a cylinder, a box, and a mustard bottle sourced from the YCB dataset (Calli et al., 2017). Additionally, for the cylinder scenario, we introduced an obstacle into the environment to assess the efficacy of the methods in navigating around obstacles. It is important to note that the feasibility distribution is unaware of the obstacle in the environment (not considered in the training), as addressing this situation falls within the responsibility of the optimality distribution.

The results for these experiments are shown in Table 5.3. As can be seen, our approach exhibits enhanced performance when compared to both NFs-PoE-MPPI and MPPI methodologies, demonstrated through superior cost values and a reduced number of steps. Moreover, the proposed method attains a superior success rate compared to other baseline methods.

The proposed method was further applied to the real robot for pushing a mustard bottle using only 32 samples which is very low compared to the commonly used number of samples in other SMPC methods. We chose this low number deliberately to show the capability of our

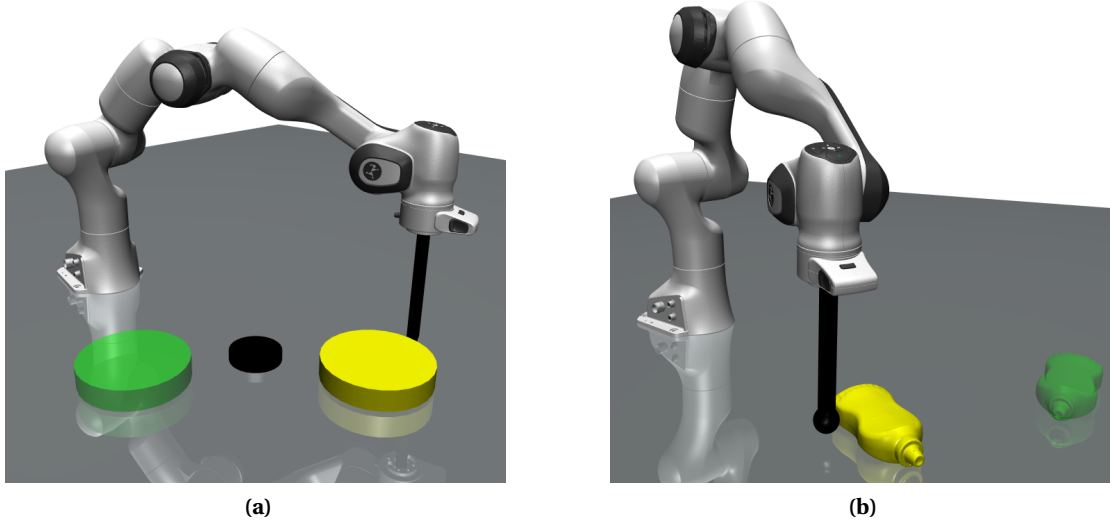


Figure 5.8: Different environment setups for pushing various objects: (a) a cylinder and (b) a mustard bottle. The obstacle is shown in black, while the green objects represent the target locations.

Table 5.4: Comparison of TT-PoE-MPPI with MPPI in an online obstacle avoidance task across 100 random obstacle locations. Values are normalized to the MPPI method. Thus, steps and cost values for MPPI are omitted. Negative values indicate superior performance, with bold formatting highlighting the best results.

#samples	MPPI	TT-PoE-MPPI		
	succ. rate	succ. rate	$\log(\# \text{ steps}_n)$	$\log(\text{cost}_n)$
16	73%	84%	-0.09	-0.13
64	73%	86%	-0.31	-0.44
512	69%	89%	-0.46	-0.66

method in increasing the efficiency of samples. Selected frames from this experiment are illustrated in Fig. 5.9. The rationale behind selecting the mustard bottle lies in its complex shape compared to other objects, thereby serving as a demonstrative example that the shapes of the objects involved do not constrain our method. The objective in this scenario is to guide the object back to the goal location (transparent shape in Fig. 5.9), while the object undergoes a 90-degree rotation. The receding horizon controller employed allows the system to effectively handle external disturbances, as demonstrated in the accompanying video¹. Disturbances were introduced to the system between Fig. 5.9c and 5.9d. Notably, the planner adeptly manages this situation, guiding the object towards the target.

¹<https://drive.google.com/file/d/1ZxxRaX7cgLk6bfGMrD2HJNrzVG9-gMey/view?usp=sharing>

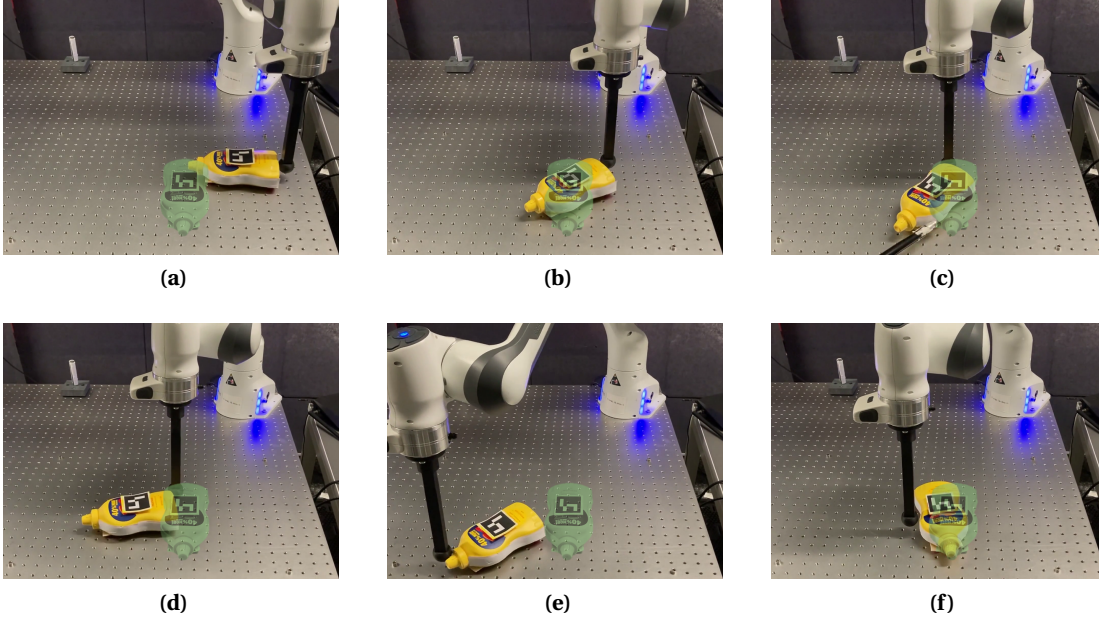


Figure 5.9: Frames selected from the real robot experiment showing the process of pushing a mustard bottle towards the target (shown as the semi-transparent picture). An external disturbance was introduced to the system at frame (c). However, the planner effectively navigates this scenario, successfully guiding the object toward the target.

5.4.4 Online Obstacle Avoidance

One significant advantage of using TT-SVD to learn the TT distribution is its capability to process the entire dataset in milliseconds if all data points are available. This is especially beneficial when the robot leverages point cloud data online to identify free space in the environment. This rapid processing allows the system to adapt its distribution quickly if new environmental changes are detected, enabling it to sample updated, obstacle-free trajectories efficiently. To evaluate this capability, we designed a task in which a point-mass robot moves from the left to the right side of the environment. As it progresses, new obstacles appear, shown as lighter regions in Fig. 5.7c. Table 5.4 presents the results of this experiment, demonstrating that the proposed method consistently outperforms the MPPI method across all measured criteria. Notably, the performance gap between the two methods widens as the number of samples increases.

5.4.5 Higher Dimensions and Alternative Optimality Distributions

Previous experiments addressed low-dimensional problems using TT-SVD and benchmarked against vanilla MPPI. Here, we demonstrate that our method scales to higher-dimensional settings and can be combined with alternative optimality distributions. We evaluate our method on a task in which a Franka Emika robot must traverse an environment with whole-body obstacle avoidance, where obstacle locations vary across scenes (as illustrated Fig. 5.1b).

Since TT-SVD becomes impractical for the large tensors required, we instead employ TT-cross with a fixed maximum rank $r_{\max} = 20$. This approach reconstructs an approximation of the full tensor while querying only $\mathcal{O}(ndr^2)$ entries of the cost function, rather than the $\mathcal{O}(n^d)$ entries required by TT-SVD. The resulting low-rank approximation recovers the full tensor from a selected subset of its entries. Our approach can be applied in two ways:

1. Augmented State Conditioning: Augment the state with all obstacle positions and, at inference time, condition on the current robot state $\mathbf{x}_t$ and obstacle locations $\mathbf{x}_{1:N_o}^o$. We define the joint feasibility distribution as

$$p^f(\mathbf{x}_{1:N_o}^o, \mathbf{x}_t, \mathbf{u}_t) = (1 - \exp(-\text{dist}(\mathbf{x}_t)/0.5))(1 - \exp(-\text{dist}(\mathbf{x}_{t+1})/0.5)), \quad (5.17)$$

where $\text{dist}(\cdot)$ is the minimum distance between the robot body and obstacles computed via the Robot Distance Field (RDF) in (Y. Li, Zhang, et al., 2024). By using a smooth, distance-based likelihood rather than a binary collision indicator, we favor configurations that keep the robot at a safe distance and produce a lower-rank function amenable to TT-cross learning.

2. Additive Feasibility Mixture: Learn a separate feasibility tensor $\mathcal{P}^{f_i}$ for only one obstacle, then combine them as

$$\mathcal{P}^f = \sum_{i=1}^{N_o} \mathcal{P}^{f_i}.$$

The additive mixture remains smoother with a lower rank representation, which is more amenable to the TT-cross method. We denote this variant as *TT-PoE-MPPI-prod*.

We further evaluate our framework against DIAL-MPC (H. Xue et al., 2024), an annealed MPPI variant that adapts its covariance using diffusion-inspired updates (Chi et al., 2024; Ho et al., 2020). Although DIAL-MPC is significantly more efficient than vanilla MPPI, it still relies on rejection sampling to eliminate infeasible trajectories—because its experts are fused at the function level and have no direct awareness of the feasible solution space. By contrast, our solution-level fusion lets us steer the sampler toward valid regions before any draws are discarded. To highlight this advantage, we embed DIAL-MPC within our TT-PoE pipeline (denoted *TT-PoE-DIAL*) and show that this hybrid consistently outperforms both standalone DIAL-MPC and our original TT-PoE-MPPI in terms of sampling efficiency and overall task success.

Table 5.5 reports success rate, total cost, and steps-to-goal for each method. Integrating TT with both MPPI and DIAL-MPC increases success rates while reducing cost and required steps. Notably, the TT-PoE-MPPI-prod variant matches or exceeds the performance of TT-PoE-MPPI, validating the effectiveness of learning and summing individual feasibility distributions within a product-of-experts framework. We define task success as reaching the target in fewer than 25 steps with total cost below 10^6 . Further experimental details are provided in the Appendix.

Table 5.5: Comparison of different methodologies in whole body obstacle avoidance task across 100 random obstacle locations and initial and target configurations using 1024 sample size. Values are normalized to the MPPI method. Thus, steps and cost values for MPPI are omitted. Negative values indicate superior performance, with bold formatting highlighting the best results.

Method	succ. rate	$\log(\# \text{ steps}_n)$	$\log(\text{cost}_n)$
MPPI	30%	0.00	0.00
TT-PoE-MPPI	53%	-0.10	-0.09
TT-PoE-MPPI-prod	50%	-0.14	-0.11
DIAL	88%	-0.35	-0.36
TT-PoE-DIAL	89%	-0.50	-0.53

5.5 Discussion

The primary objective of this study is to enhance the sampling distribution employed in SMPC through the application of products of experts in the solution space. As anticipated, all methods with TT-PoE pipeline (NFs-PoE-MPPI, TT-PoE-MPPI, and TT-PoE-DIAL) exhibit superior performance compared to their conventional counterparts. However, the performance of NFs-PoE was not significantly better than that of Proj-MPPI, and in all experiments, TT-PoE consistently outperformed NFs-PoE by a substantial margin. In the subsequent sections, we will delve deeper into the reasons behind this performance, explaining the advantages of using TT-distribution over NFs models and identifying its limiting factors.

5.5.1 TT-distribution vs Normalizing flow

The rationale behind choosing normalizing flows (NFs) over other generative approaches like autoencoders (Schmidhuber, 2015) or generative adversarial networks (GANs) (Goodfellow et al., 2014) lies in the NFs' unique ability to map samples from the latent space to the main space and vice versa. Such bidirectional mapping is crucial when combining multiple distributions in a product, ensuring they operate within the same space.

Nevertheless, similar to other latent space learning methods, the acquired latent space lacks *intuitiveness* (Losey et al., 2022). In this context, *intuitive* refers to the observed relation between data points in the latent space, not necessarily aligning with their distance in the main spaces. For instance, when two proximate data points exhibit similar distances to an objective point in latent space, while in task space, one is positioned near the nominal solution and the other is distant. This phenomenon is illustrated in Fig. 5.10, obtained from a model trained for the PNRGRID task, where the nominal action is presumed to be $[-0.5, -0.5]$ at state $[-1, -0.9]$, and the Gaussian model utilized for the MPPI method possesses identical variance in both directions. At this point, only velocity constraints (ranging between -1 and 1) must be active, with no other constraints in effect.

To generate samples from normalizing flows (NFs), the process involves mapping the nominal state-action back into the latent space (referred to as nominal-latent point), assuming a

Gaussian distribution around it, combining it with the learned feasible distribution for the task, and sampling from the resulting distribution. The nominal latent point is depicted in dark blue in Fig. 5.10a, with other samples represented in distinct colors corresponding to their distance from the nominal latent point (darker blue indicates proximity, while darker red signifies greater distance). These samples are then mapped back into the task space, shown in Fig. 5.10b, where the colors in both images correspond, i.e. each sample maintains the same color in both representations. Notably, all samples fall within the -1 to 1 range in both directions, representing the action limits (see Appendix for experiment details).

Observing the task space samples, it is evident that they are not entirely clustered around the nominal action $[-0.5, 0.5]$. Additionally, the distribution of samples in both directions appears dissimilar. This discrepancy arises due to the nonlinear function employed to revert the data back to the task space, which may have varying effects in different directions. This suggests that the covariance matrix in the latent space must be meticulously chosen to ensure the desired distribution upon reverting the data to the task space. From the perspective of obtaining random samples, this might not seem like a significant issue, as there are indeed samples close to the nominal solutions. However, the bias introduced in NFs can render the MPPI solution inefficient, which we believe is a key factor contributing to the inferior performance of this method compared to our proposed method, as well as to the MPPI method in certain cases.

In contrast, samples obtained from the TT-distribution, as illustrated in Fig. 5.10c, are centered around the desired nominal point and exhibit a consistent distribution in both directions. This is attributed to the fact that in the TT-distribution, the product of the two distributions is executed in the task space without the need to map the data to the latent space and back. This preserves the original distribution shape, contributing to increased efficiency. An alternative methodology involves the application of techniques like GMMs directly in the task space. While this approach holds promise in addressing the previously mentioned issue, a challenge arises from its parametric nature, a topic that will be further explored in subsequent discussions.

Another challenge associated with normalizing flows is that, despite enhancing the probability of positive samples in the latent space, it does not entirely eliminate the probability of negative samples. This stems from the method relying on the Gaussian distribution, a parametric distribution wherein the probability values decrease smoothly without abruptly reaching zero. This issue is also prevalent in other distribution learning methods, such as GMMs. To illustrate this point, we model the distribution of the feasible set of the PNGRID area using GMM. Fig. 5.6b displays approximately 10,000 samples generated from the GMM model, which employed 400 Gaussian components. The results show that the model fails to efficiently capture the boundaries, compared to TT-distribution samples depicted in Fig. 5.6d. Additionally, it is noteworthy that the training process for the GMM model required significantly more time: approximately 21 minutes, compared to around 5 minutes for normalizing flows and just milliseconds for TT-SVD.

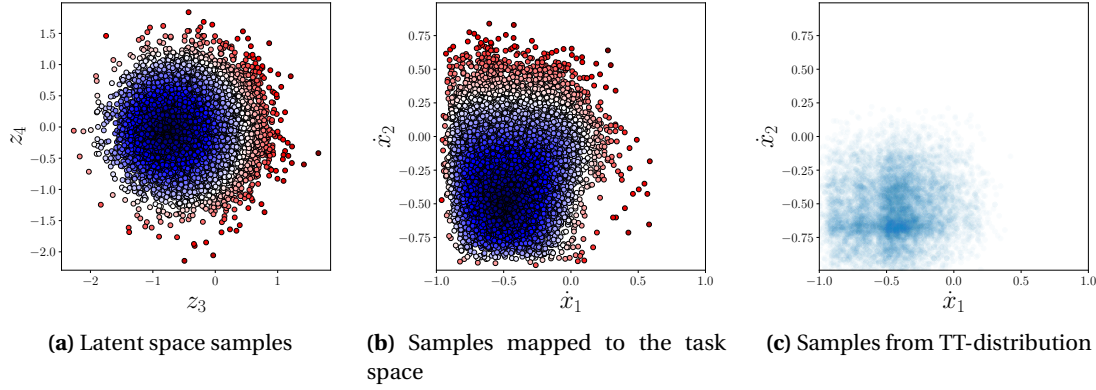


Figure 5.10: Samples gathered with different methods. In (a) and (b), the color gradient represents the proximity of the samples to the mean action of MPPI, with darker blue indicating closer proximity and darker red indicating greater distance. It is evident that samples generated in the latent space are distorted when translated back to the task space: they are not centered around the desired action and exhibit varying spans across dimensions despite having the same variance. Conversely, samples generated via the TT method are centered around the desired action and exhibit consistent spans in both dimensions.

This issue also affects the product distribution when combining it with the one obtained from MPPI. If the two distributions (optimality and feasibility) do not have overlapping high-probability areas, such that the nominal trajectory is close to regions of infeasibility (low-probability areas), the system will consider the low-probability areas of both distributions as possible candidates.

In contrast, the TT approach mitigates this problem to a greater extent. As a non-parametric distribution model, TT does not require probabilities to change smoothly, allowing it to capture sharp transitions in the distribution more effectively. Furthermore, the TT model takes both positive and negative samples into account, aiming to assign a probability of zero to the infeasible areas and one to the feasible areas. Consequently, it acts as a filter to eliminate the possibility of sampling from an infeasible distribution. This means that after combining the two distributions, the infeasible areas will ideally have zero probability, regardless of the characteristics of the optimality distribution.

5.5.2 Low-Rank Representation

As highlighted in prior works such as (Shetty, Lombono, et al., 2024), one of the features that enhance the efficiency of TT methods is their ability to represent high-dimensional tensors in low-rank formats. The rank of a tensor can be understood analogously to that of a matrix, defined as the number of independent fibers in the tensor. This low-rank representation plays a pivotal role in our process for several reasons.

First, it allows the entire tensor to be stored in a significantly more memory-efficient manner.

Second, techniques such as TT-cross (Savostyanov & Oseledets, 2011) exploit this property to learn the TT representation of a tensor without requiring access to all its elements, reducing computational overhead. Third, this low-rank structure enables faster and more efficient sampling strategies (compared to sampling directly from the whole tensor), an advantage leveraged extensively in our proposed method. Finally, it allows operations like tensor products to be executed directly at the core level, bypassing the need to reconstruct or manipulate the full tensor.

To clarify why low-rank representations make such operations efficient, it is essential to examine the relationship between tensor rank and the separability of a function’s variables. When a function is converted into a tensor representation, each edge in the resulting tensor corresponds to one argument of the function. Without delving into excessive detail, it can be demonstrated that the rank of a tensor representing a function is inversely related to the separability of the function’s variables. For instance, a fully separable function, such as $g(x, y) = g_x(x)g_y(y)$, results in a rank-1 tensor, irrespective of the specific forms or complexity of $g_x(\cdot)$ and $g_y(\cdot)$. This separability is also evident at the core level, where each tensor core corresponds to a distinct argument of the function. This property was crucial to our approach. When combining the optimality and feasibility distributions, we manipulated only the cores associated with the action commands, leaving other cores unchanged.

5.5.3 Computation time

As outlined in Alg. 1, our method adds only two extra operations (lines 3–4) and modifies the sampling step (line 5) compared to MPPI. Table 5.6 shows how these changes affect computation time. Although MPPI achieves faster overall runtimes, our TT-sampling times are on par with other baselines, making it suitable for online planning.

In its current Python implementation, TT-sampling scales linearly with the number of action variables—e.g., the whole-body obstacle-avoidance task uses 3.5× more—because it relies on a Python library (i.e., `tntorch`) with a for-loop inside, and lacks multithreading. By contrast, MPPI can leverage highly optimized native libraries and its state-independent Gaussian distribution to improve the computation time significantly. By migrating performance-critical TT sections to C/C++ and enabling multithreading, we estimate TT’s sampling time could drop below 1 ms. Until those optimizations are implemented, the timings reported here reflect the current implementation, which remains sufficient for real-time control. All measurements were recorded using an NVIDIA RTX 3090 GPU and a 12-core CPU.

5.5.4 Limitations

Despite its strengths, our approach has scalability limits: while we have demonstrated performance on moderately high-dimensional systems (which can be sufficient for a wide range of robotic tasks), it does not extend readily to very high dimensions (e.g., over 50 decision

Table 5.6: Average computation times (in milliseconds) for each method on the PNGRID (P) and whole-body obstacle avoidance (W) tasks. MPPI’s runtimes were virtually identical across both tasks and are therefore reported once. Note that computation times for related methods (e.g., DIAL and TT-PoE-DIAL) can be inferred directly from this table. Although our proposed method is slower than MPPI, it is on par with the other techniques and, as shown in the performance tables, yields a significant improvement over MPPI.

Method	Product [ms]	Conditioning [ms]	Sampling [ms]	Mapping [ms]	Projection [ms]	Total [ms]
NFs-PoE-MPPI (P)	0.03 ± 0.00	-	0.48 ± 0.02	$2.50 \pm 0.07(\times 2)$	-	5.51 ± 0.16
MPPI	-	-	0.48 ± 0.02	-	-	0.48 ± 0.02
Proj_MPPI (P)	-	-	0.48 ± 0.02	-	5.02 ± 26.46	5.50 ± 26.48
TT-PoE-MPPI (P)	0.02 ± 0.00	0.04 ± 0.00	2.67 ± 0.02	-	-	2.73 ± 0.02
TT-PoE-MPPI (W)	0.13 ± 0.01	1.23 ± 0.10	5.92 ± 0.31	-	-	7.28 ± 0.42

variables). Several avenues could address this. First, a multi-resolution discretization scheme (e.g., using coarse grids when the robot is far from obstacles and finer grids when it nears them) can dramatically shrink the effective discretization size and enable higher-dimensional problems. Second, leveraging cumulative learning to inform the TT-cross algorithm may accelerate convergence and improve accuracy. Finally, integrating TT models with complementary learning techniques to discover compact latent representations could further reduce dimensionality and allow TT methods to operate more efficiently in even larger state spaces.

Alternatively, recent advances in generative modeling techniques, such as diffusion-based models (Ho et al., 2020) and energy-based models (LeCun et al., 2006), present opportunities for further exploration. These methods could be extended for use within the MPC framework discussed in this work, as our PoE-MPPI approach is independent of TT methods and can be adapted to incorporate these generative approaches. In this work, we have already demonstrated the mutual benefits of combining our pipeline with DIAL-MPC (a data-free, diffusion-inspired sampler), and future work could similarly integrate other generative approaches to further improve sampling efficiency and constraint satisfaction.

It is also worth noting that while generative methods typically rely on demonstration data to train individual experts, this requirement can be challenging when exploring the entire feasibility region, as demonstrations may not sufficiently cover all possible scenarios. By contrast, the TT approach does not depend on demonstration data and instead operates on functions. This design choice has its advantages for our purpose but may also introduce limitations for certain applications. In some applications, a hybrid strategy that combines data-driven and function-based experts could offer the best of both worlds.

There are two potential avenues for enhancing the proposed method. Firstly, the system’s capability for long-horizon planning could be improved by incorporating an additional expert that considers environmental contexts. These methods can be regarded as supplementary experts and seamlessly integrated with other methodologies through the products of experts method elucidated in this chapter, potentially leading to increased efficiency. Secondly, recent findings (Shetty, Xue, & Calinon, 2024; T. Xue et al., 2024a) demonstrate the applicability of TT models for solving approximate dynamic programming and deriving policy models for

robots in TT form. Their approach involves selecting the most optimal or robust action from the model, but we can modify the pipeline by sampling from the TT model and incorporating additional experts, as discussed in this chapter. This modification enables a more flexible pipeline, distributing tasks among different experts and mitigating the reliance on achieving convergence to the most optimal solutions.

5.6 Conclusion

In this chapter, we showed how incorporating the *Products of Experts* (PoE) framework leads to *optimality-guided, feasibility-aware sampling* for model predictive control. Sampling-based methods are powerful in robotics, particularly for contact-rich manipulation tasks, because they are gradient-free, naturally parallelizable, and capable of smoothing irregular cost functions. However, their efficiency drops significantly in constrained environments, where many sampled trajectories are rejected due to infeasibility.

By learning a feasibility distribution and combining it with the optimality distribution through PoE, we obtained a principled *project-then-sample* strategy. This combination acts as a filter, removing infeasible regions before sampling, and results in more efficient exploration of the solution space. We demonstrated this idea in different SMPC methods, such as MPPI and DIAL-MPC, across several frameworks including obstacle avoidance and non-prehensile manipulation. In all cases, the proposed approach improved both sampling efficiency and overall task performance compared to standard baselines.

Despite these advantages, an important limitation remains: the method increases success rates but does not provide guidance when a task ultimately fails. Deciding *what the robot should do in the presence of persistent failures* is a critical open challenge in robotics. We address this issue in the next chapter, where we shift the focus from efficiency of sampling to the broader problem of failure recovery.

6 CCDP: Model-free Failure Recovery via Guided Diffusion Sampling

Collaboration Note:

This chapter is the result of joint work with Michael Gienger, and Fan Zhang from the Honda Research Institute Europe (HRI-EU), who helped me through discussions and provided the robot platform for the real-world experiments.

Publication Note:

This chapter is based on the following publication:

- A. Razmjoo, S. Calinon, M. Gienger, and F. Zhang, “CCDP: Composition of Conditional Diffusion Policies with Guided Sampling,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

Website and codes: <https://hri-eu.github.io/ccdp/>

In the previous chapters, we improved constrained motion planning using feasibility distributions, guided costs, warm-starting, and robust manifolds. Yet robots may still fail due to modeling inaccuracies or unforeseen environmental changes. In this chapter, we address the problem of **failure recovery**. We consider scenarios where a variation of successful demonstrations are available for training, as is common in imitation learning. During inference, the robot must choose the right variation, but sampled actions may still fail, and naive re-sampling is inefficient. We propose an enhanced sampling strategy that refines the distribution by steering it away from previously unsuccessful actions. Our method infers recovery behaviors solely from successful demonstrations, without additional failure data, exploration, or high-level controllers. To handle failures that depend on long action histories, we employ *diffusion model decomposition*, which breaks the recovery problem into smaller subproblems in learning, data collection, and inference. This enables a low-level controller that dynamically adapts its sampling space, improving efficiency and success rates when prior samples fail. We validate our approach on door opening with unknown directions, object manipulation, and button-searching tasks, showing consistent improvements over baseline methods.

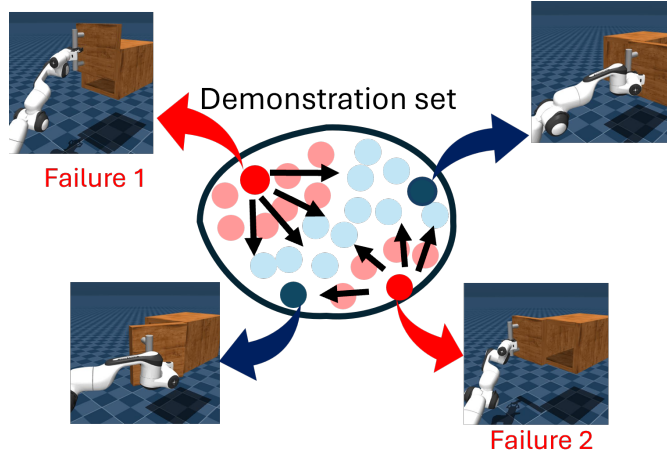


Figure 6.1: A diverse demonstration set, featuring multiple task variations, is provided to the robot. In the event of a failure, the robot is guided to alternative variations rather than repeatedly sampling a similar action.

6.1 Introduction

Another important aspect of constrained motion planning is that failures are inevitable, even if we reduce their probability through better methods. Robots must therefore be able to recover from such situations. In previous chapters, we introduced approaches that improved feasibility handling and reduced the chance of failure. Yet, failures due to modeling errors, unexpected obstacles, or environmental changes remain unavoidable. This chapter focuses on the problem of **failure recovery**.

Many existing recovery approaches assume access to supplementary resources, such as simulated environments (Lee et al., 2019), advanced reasoning or foundational models (Duan et al., 2024), or expert guidance (Niekum et al., 2013). Other works employ two-level planning, where a high-level planner—such as a foundation model (Duan et al., 2024; Huang et al., 2022; Z. Liu et al., 2023) or a hierarchical RL/BC controller (Lee et al., 2019; Triantafyllidis et al., 2023)—selects action primitives and task parameters. While promising, these approaches suffer from added complexity, suboptimal separation between planning levels, and *combinatorial explosion* as the number of options grows. For these reasons, we target failure recovery directly within the low-level controller, by modifying the prior distribution (e.g., feasibility or warm-start distributions) used for action sampling. Our goal is to develop an imitation learning approach that uses only successful demonstrations—typically the case in practice—so that the robot can recover from failures without access to a simulator, model, or high-level planner.

Recent advances in imitation learning, such as Implicit Behavior Cloning (Florence et al., 2022) and diffusion/flow-matching policies (Chi et al., 2024; Zhang & Gienger, 2024), have shown a strong ability to reproduce complex distributions from demonstrations. In inference, however, they often rely on repeated resampling when an action fails, which is inefficient and may repeat the same action. A more effective strategy is to avoid failed actions and explore alternative variations seen in demonstrations, including infrequent ones. This mirrors how

humans recover from small failures—like fumbling for a light switch or opening a door the wrong way—by trying different variations until success. An illustration of this behavior is given in Fig. 6.1, where the robot, after failing to lift a door handle and sliding left, can attempt other variations to complete the task.

In line with imitation learning, we train a policy that updates its decisions based on past outcomes (Lee et al., 2020; Yu et al., 2017). A key challenge is that demonstrations usually contain only successes, so collecting failure data or exploring failure cases autonomously would be costly and unsafe. To address this, we relax the recovery definition to select an action dissimilar to the failed ones. We conduct an offline analysis to identify mutually dissimilar actions and train a model that, upon failure, guides the controller to sample actions sufficiently *far* from the failed attempt. Despite its simplicity, this approach leads to a significant improvement in performance.

A second challenge is capturing failures that unfold over long histories. To handle this, we leverage the idea of composing models (Hinton, 1999; N. Liu et al., 2022), also known as the product of experts. We independently learn recovery distributions for individual failures and combine them at inference to form a distribution that avoids all observed failures. This approach relies on the assumption that the underlying cause of failure is static, making it particularly well-suited for environments with unknown but fixed constraints.

In this chapter, we introduce the Composition of Conditional Diffusion Policies (CCDP), which builds on a diffusion policy framework (Chi et al., 2024) but conditions the policy on detected failure cases. Unlike standard conditional models that attract the policy toward desired behaviors, CCDP uses failures as *negative guidance*, steering sampling away from unsuccessful regions. By decomposing the recovery task into smaller subproblems, we enable adaptive low-level control without labeling variations or relying on external models.

In summary, our contributions are:

- A decomposed diffusion policy framework that enhances modularity and controllability of standard diffusion policies (Chi et al., 2024).
- A failure recovery strategy that infers recovery actions solely from successful demonstrations, without labeling or modeling requirements.
- A validation across several manipulation tasks, demonstrating improved performance compared to state-of-the-art baselines.

6.2 Related Work

Imitation Learning: Imitation learning is a powerful technique for teaching robots to perform tasks by directly imitating expert behavior (Osa et al., 2018). This approach offers several advantages, such as eliminating the need for explicit system modeling, cost function design,

and extensive manual coding. Moreover, this method has been successfully integrated with reinforcement learning (Nair et al., 2018) and optimal control techniques (Razmjoo et al., 2021), enhancing their performance in complex tasks. In our work, we focus exclusively on leveraging expert demonstrations, assuming no access to an explicit cost function or physical model (even simulation environment) during the training phase. Experts provide a distribution of possible solutions for a given task, which motivates using generative models to capture this distribution and sample actions during execution. Early approaches, such as Probabilistic Movement Primitives (ProMP) (Paraschos et al., 2013) and Task-Parameterized Gaussian Mixture Models (TP-GMM) (Calinon, 2016), laid the groundwork for this framework. More recent methods, including Implicit Behavior Cloning (Florence et al., 2022), diffusion policies (Chi et al., 2024), and flow-matching policies (Zhang & Gienger, 2024), have further advanced the field by modeling demonstration distributions more effectively. Despite these advances, it is not guaranteed that a single sampled action will succeed. A common strategy to address this issue is to repeatedly sample from the same distribution until a successful action is obtained. However, this approach can be inefficient as it does not account for the fact that certain regions of the action space have already been identified as unsuccessful. In contrast, our work seeks to explicitly guide the system away from regions associated with failure, thereby improving both efficiency and overall success rates.

Guided Policy Inference: Several approaches have been proposed to guide the inference process. One class of methods conditions policy parameters on system characteristics, updating these parameters through the robot’s interactions with the environment (Adetola et al., 2009; T. Xue et al., 2025). These techniques leverage system identification or adaptive control strategies to adjust the sampling process. However, because they depend on an underlying model, they may not be directly applicable in imitation learning scenarios where such models are unavailable. An alternative strategy involves using high-level decision variables controlled by an external module, such as foundational models (Huang et al., 2022; Xie et al., 2024) or controllers trained via hierarchical reinforcement learning (HRL) (Lee et al., 2019) or hierarchical behavior cloning (HBC) (Triantafyllidis et al., 2023). Although effective in some contexts, this approach can lead to an explosion in the number of decision variables for the high-level planner—a problem commonly referred to as *combinatorial explosion*—which makes decision-making computationally expensive. A further approach is to employ rejection sampling, a Monte Carlo method where failed samples are discarded and new ones are generated. In our work, rather than rejecting individual samples, we reject entire sampling regions associated with failure. This modification reduces the likelihood of repeatedly sampling from unproductive areas. We achieve this goal through the composition of multiple diffusion models, enabling efficient and adaptive exploration of the action space without relying on a high-level decision maker.

Composition of Multiple Models: The concept of Products of Experts (PoE), introduced by Hinton (Hinton, 1999), suggests that instead of learning a single complex distribution over many variables, one can decompose the problem into several simpler subproblems (i.e., “experts”), each focusing on a specific objective and/or variables. As discussed in the

previous chapter, this framework has been successfully applied in robotics, primarily using Gaussian models (Calinon, 2016), though its effectiveness has been limited by this reliance. The emergence of more expressive models such as energy-based models (LeCun et al., 2006) and later denoising models, such as those in (Ho et al., 2020; Lipman et al., 2023), has further popularized the use of PoE, often referred to in these contexts as the composition of models. These approaches are particularly adept at handling high-dimensional problems with complex distributions. Recent works (Gkanatsios et al., 2023; Yang et al., 2023) have demonstrated how combining multiple constraint models via a products-of-experts approach can effectively address challenging task and motion planning problems. In this work, we extend this idea to the low-level controller, guiding the inference by composing multiple diffusion models to avoid the regions associated with failure.

6.3 Methodology

Given a dataset of M successful demonstrations $\mathcal{D} = \{(\mathbf{u}_t, \mathbf{x}_t, \mathbf{h}_t^L)\}_{i=1}^M$, our objective is to learn a diffusion policy that models the conditional distribution $p_\pi^{\mathcal{D}}(\mathbf{u}_t | \mathbf{x}_t, \mathbf{h}_t^L)$, where $\mathbf{h}_t^L = [\mathbf{u}_{t-L}^\top, \mathbf{x}_{t-L}^\top, \dots, \mathbf{u}_{t-1}^\top, \mathbf{x}_{t-1}^\top]^\top$. Hence, desired action can be generated as $\mathbf{u}_t \sim p_\pi^{\mathcal{D}}(\mathbf{u}_t | \mathbf{x}_t, \mathbf{h}_t^L)$ using the DDPM approach described in Sec. 2.5.1 (we omit the zero superscript for clarity).

However, this formulation requires a very long history ($L \gg 0$) to capture all previous failures. To address this limitation, we extend the formulation by conditioning on a set of failure features as

$$\mathbf{u}_t \sim p_\pi(\mathbf{u}_t | \mathbf{x}_t, \mathbf{h}_t^L, \mathbf{z}_{1:N}^f), \quad (6.1)$$

where $\mathbf{z}_i^f = \mathbf{z}(\mathbf{u}_i^f, \mathbf{x}_i^f)$ extracts key features from the i -th failure's action $\mathbf{u}_i^f$ and state $\mathbf{x}_i^f$, and N denotes the total number of previous failures. These features may include a learned latent-space representation, key measurements such as forces, or discrete mode indicators. For the purpose of this work, we assume that the function $\mathbf{z}(\cdot)$ is known, and we discuss different possibilities in Sec. 6.3.2.

With this formulation, a long history is not required since key features effectively summarize past information. When failures occur, the system extracts these features and conditions future policies on them. However, the resulting input can be high-dimensional, depending on the dimensions of the actions, states, key feature space, and the variable number of failures (which may range, for example, from 2 to 10). This high dimensionality complicates both model learning and data collection, as it must account for all possible failure combinations.

6.3.1 Composition of Diffusion Policies

To mitigate these challenges and introduce modularity, inspired by (N. Liu et al., 2022), we propose decomposing (6.1) into several simpler sub-problems. Hence, (6.1) can be rewritten

as

$$p_\pi(\mathbf{u}_t | \mathbf{x}_t, \mathbf{h}_t^L, \mathbf{z}_{1:N}^f) \propto p_u(\mathbf{u}_t) \frac{p_x(\mathbf{u}_t | \mathbf{x}_t)}{p_u(\mathbf{u}_t)} \frac{p_h(\mathbf{u}_t | \mathbf{h}_t^L)}{p_u(\mathbf{u}_t)} \prod_{i=1}^N \frac{p_z(\mathbf{u}_t | \mathbf{z}_i^f)}{p_u(\mathbf{u}_t)}. \quad (6.2)$$

Using the diffusion model framework, we initialize with a noisy sample and iteratively denoise through the reverse diffusion process

$$p(\mathbf{u}_t^{k-1} | \mathbf{u}_t^k) = \mathcal{N}\left(\frac{1}{\alpha_k} \left(\mathbf{u}_t^k - \frac{1 - \alpha_k}{\sqrt{1 - \bar{\alpha}_k}} \boldsymbol{\varepsilon}_\theta(\mathbf{h}_t^L, \mathbf{x}_t, \mathbf{z}_{1:N}^f, \mathbf{u}_t^k, k)\right), \sigma_k^2 \mathbf{I}\right). \quad (6.3)$$

According to (6.2), we decompose the denoising term $\boldsymbol{\varepsilon}_\theta(\mathbf{h}_t^L, \mathbf{x}_t, \mathbf{z}_{1:N}^f, \mathbf{u}_t^k, k)$ as follows:

$$\begin{aligned} \boldsymbol{\varepsilon}_\theta(\mathbf{u}_t^k, \mathbf{x}_t, \mathbf{h}_t^L, \mathbf{z}_{1:N}^f, k) = & \boldsymbol{\varepsilon}_u(\mathbf{u}_t^k, k) + w_x \left(\boldsymbol{\varepsilon}_x(\mathbf{u}_t^k, \mathbf{x}_t, k) - \boldsymbol{\varepsilon}_u(\mathbf{u}_t^k, k) \right) + w_h \left(\boldsymbol{\varepsilon}_h(\mathbf{u}_t^k, \mathbf{h}_t^L, k) - \boldsymbol{\varepsilon}_u(\mathbf{u}_t^k, k) \right) \\ & + \sum_{i=1}^N w_z^i \left(\boldsymbol{\varepsilon}_z(\mathbf{u}_t^k, \mathbf{z}_i^f, k) - \boldsymbol{\varepsilon}_u(\mathbf{u}_t^k, k) \right). \end{aligned} \quad (6.4)$$

Here, w_x , w_h , and w_z^i are positive coefficients associated with the state, history, and failure key features, respectively. Note that these coefficients are not necessarily bounded by 1, nor do they sum to 1 (see (N. Liu et al., 2022) for details). In the special case where $w_x = 0$ and $w_z^i = 0$ for all i while $w_h = 1$ (with $\mathbf{x}_t$ incorporated into $\mathbf{h}_t^L$), the formulation reduces exactly to the standard diffusion policy. This modular decomposition offers additional flexibility, enabling the system to accommodate a variable number of failure cases. Consequently, regardless of the number of failures, we need to learn only one model—namely, $p(\mathbf{u}_t | \mathbf{z}^f)$ —to capture the failure behavior, thereby making the learning process more tractable.

In the following, we discuss the effects of the different terms in (6.4) and how to control their corresponding weights in our scenarios. Figure 6.2 also provides an illustrative example of the roles of the different components.

- $\boldsymbol{\varepsilon}_u(\mathbf{u}_t^k, k)$: This term encourages sampling actions that are similar to those observed in the demonstrations. It does not incorporate information from the current state, history, or previous failures; rather, it guides the sampling process toward the demonstrated actions.
- $w_x \left(\boldsymbol{\varepsilon}_x(\mathbf{u}_t^k, \mathbf{x}_t, k) - \boldsymbol{\varepsilon}_u(\mathbf{u}_t^k, k) \right)$: This component steers the actions toward those that match the current state of the robot and its environment. It is analogous to a diffusion policy with zero history. We separate this term from the history component to avoid bias from past failures—especially if a previous action caused a failure—while still promoting meaningful actions considering the current state of the robot. Although relying solely on the current state may lead to less smooth behavior, combining it with the history term allows us to balance responsiveness and smoothness.

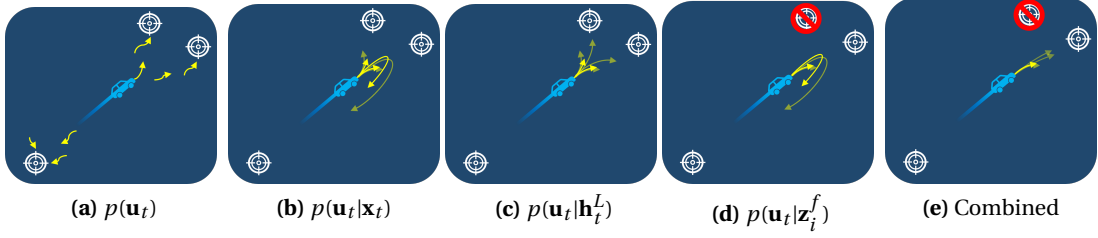


Figure 6.2: An illustrative example showing samples generated from multiple distributions learned from the same demonstration set. The blue car represents the agent, with its past trajectory depicted as a shaded blue line. The car aims to reach one of the white goal positions. Yellow arrows indicate sampled actions from the corresponding distributions. This figure demonstrates how conditioning on different factors, such as state, history, and failure, affects the resulting distributions.

- $w_h(\boldsymbol{\varepsilon}_h(\mathbf{u}_t^k, \mathbf{h}_t^L, k) - \boldsymbol{\varepsilon}_u(\mathbf{u}_t^k, k))$: This term promotes temporal continuity by encouraging the system to follow the action history. In scenarios where a failure occurs, reducing w_h prevents the system from rigidly adhering to a detrimental history; conversely, in failure-free situations, maintaining w_h at a level comparable to w_x ensures smooth motion.
- $w_z^i(\boldsymbol{\varepsilon}_z(\mathbf{u}_t^k, \mathbf{z}_i^f, k) - \boldsymbol{\varepsilon}_u(\mathbf{u}_t^k, k))$: This term allows the system to consider distinct failure cases and, based on the extracted failure features, steer away from regions that led to failures. Unlike the other terms that guide the system toward a desired area, this term acts repulsively. By handling each failure separately, we avoid the need to learn a model for every possible combination or ordering of failures, thus reducing the data collection burden. For details on learning this model, please refer to Sec. 6.3.2.

6.3.2 Recovery Model

Since we assume that the provided dataset does not include failure cases, we need to synthesize another dataset to train failure recovery. In this regard, we synthesize an auxiliary dataset $\mathcal{R}$ using distributions $p(\mathbf{u}_t)$, and $p(\mathbf{u}_t | \mathbf{x}_t)$.

We first relax the recovery definition slightly by considering all actions dissimilar to the failed one as potential recoveries. If we further assume that the cause of failure is static (i.e., repeating the same action in the same state will fail again, as the door hinge location or button location remains unchanged by external factors), then we define the set of recovery actions as

$$\mathbf{u} \in \mathcal{R}(\mathbf{z}^f) \quad \text{if} \quad \begin{cases} \|\mathbf{z}(\mathbf{u}, \mathbf{x}) - \mathbf{z}(\mathbf{u}^f, \mathbf{x}^f)\|^2 > \delta_z, \\ \|\mathbf{x} - \mathbf{x}^f\|^2 < \delta_x, \end{cases} \quad (6.5)$$

where $\mathbf{u}^f$ is the action that led to failure, $\mathbf{x}^f$ is the state where the failure occurred, δ_z is a threshold defining sufficient dissimilarity in the key feature space, and δ_x is a threshold defining similarity in the state space. Intuitively, if the state of the system has not changed

significantly, we expect the same action to result in failure again; therefore, recovery actions must differ substantially in the failure feature space.

Now we can construct the recovery dataset by traversing the available data and accepting every pair that satisfies (6.5). However, doing so directly on the main dataset can be problematic because the dataset is limited and actions are collected in various states, resulting in a sparse recovery dataset. To address this issue, we perform data synthesis: we randomly select states observed in the demonstrations and, for each state, sample multiple actions such that

$$\mathcal{D}_s(\mathbf{x}_s) = \left\{ (\mathbf{u}, \mathbf{x}_s) \mid \mathbf{u} \sim \tilde{p}^{\mathcal{D}}(\mathbf{u} \mid \mathbf{x}), \mathbf{x} \in \mathbf{x}_s + \boldsymbol{\xi}_x, \boldsymbol{\xi}_x \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}) \right\}. \quad (6.6)$$

Here, $\boldsymbol{\xi}_x$ denotes random noise added to the state. The corresponding noise estimator is defined as

$$\tilde{\boldsymbol{\epsilon}}(\mathbf{u}^k, \mathbf{x}, k) = \boldsymbol{\epsilon}_u(\mathbf{u}^k, k) + w_x \left(\boldsymbol{\epsilon}_x(\mathbf{u}^k, \mathbf{x}, k) - \boldsymbol{\epsilon}_u(\mathbf{u}, k) \right), \quad (6.7)$$

where the key difference between (6.7) and (6.4) is the exclusion of history in (6.7). This omission is crucial for two reasons: (1) including history could bias the system toward past actions, thereby reducing the diversity of recovery solutions (see Fig. 6.2), and (2) it would lead to a higher-dimensional problem. Additionally, setting $w_x \leq 1$ encourages the system to explore a broader range of actions by allowing sampling from the entire action space—even those actions that may not be perfectly suited to the current state—while still guiding the samples toward the current state. We found that this diversity is beneficial for learning robust avoidance behavior. During inference, when using (6.4), an appropriate value of w_x is chosen to ensure that the primary actions applied to the robot remain suitable for the current state.

After synthesizing the dataset $\mathcal{D}_s(\mathbf{x}_s)$ for states sampled from demonstrations, we extract the pairs that satisfy (6.5). With this recovery dataset in hand, we then apply the DDPM method to learn the corresponding distribution. Note that constructing this dataset does not require any additional model or simulator; we simply sample actions and check whether they are sufficiently separated to be considered potential recovery actions in cases where one of them fails.

Failure Key Features

The function $\mathbf{z}(\cdot)$ captures the key features of failed actions. While learning these features autonomously via latent-space methods would be highly beneficial, we leave this for future work. Instead, we discuss three intuitive and practical choices for $\mathbf{z}(\cdot)$:

1. **Directly using the failed actions:** $\mathbf{z}(\mathbf{u}^f, \mathbf{x}^f) = \mathbf{u}^f$. In this approach, the robot directly avoids the failed action, which is particularly useful when the state of the system remains largely unchanged after a failure.
2. **Using the final state:** $\mathbf{z}(\mathbf{u}^f, \mathbf{x}^f) = \mathbf{x}_T^f$, where $\mathbf{x}_T^f$ is the state reached if $\mathbf{u}^f$ were executed at $\mathbf{x}^f$; this can be readily extracted from demonstrations.

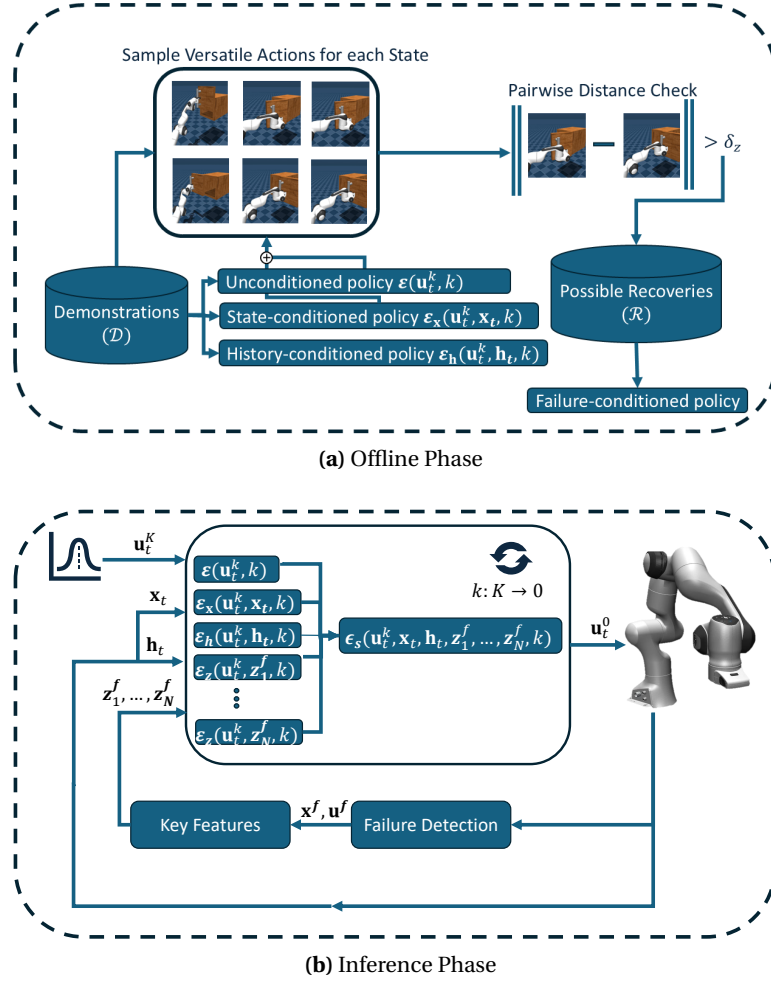


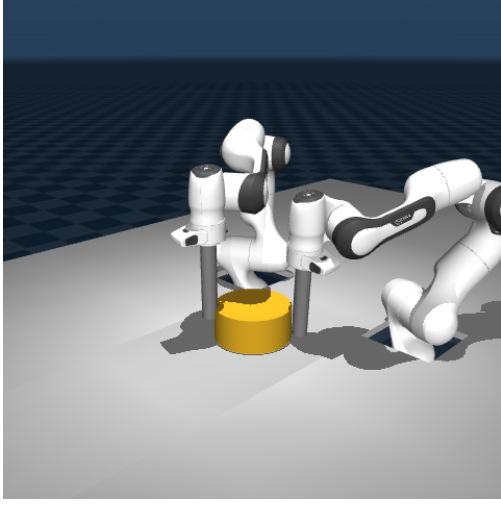
Figure 6.3: Schematic overview of the proposed method illustrating the offline and inference phases.

3. **Action primitive:** $z(u^f, x^f) = m$, where m is a discrete label indicating the action primitive applied to the system. In this case, the data must be appropriately labeled, or skill discovery methods such as (Zhu et al., 2022) can be used to obtain these labels.

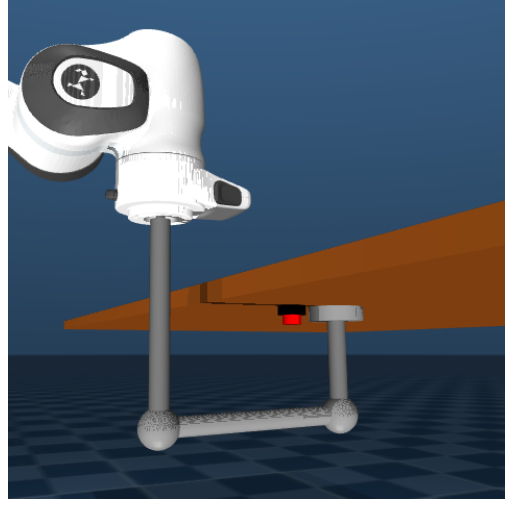
Fig. 6.3 summarizes the overall process in both the offline and online phases. In the online phase, the robot’s behavior is monitored, and when a failure is detected, the corresponding case is added to the failure set. Subsequently, (6.4) is used to sample a new action guided by previous failures.

6.4 Experiments

We validated our approach on tasks where some samples failed or underperformed, comparing it with two baselines: a standard diffusion policy (DP) and an enhanced version (DP*), which partitions the solution space and uses rejection sampling to select a new region upon failure.



(a) Object Manipulation



(b) Button Pressing



(c) Object Packing

Figure 6.4: Different experimental setups used in this study

Our approach does not rely on a high-level planner, so region selection is performed randomly. Additionally, our pipeline requires a failure detection system, which can be implemented using any appropriate method (e.g., (Cornelio & Diab, 2024)). In our study, demonstrations are collected in a simulated environment using a predefined motion planning method that has access to true parameters, allowing it to perform the task successfully. However, these parameters remain inaccessible to both our approach and the baselines. Demonstrations are collected as end-effector via point positions distributed uniformly along each trajectory, along with task-relevant variables such as the door-handle location, object location, and others. The demonstrations cover different task variations (such as cabinet locations in the door-opening task) as well as variations in constraint parameters (such as the door-opening direction). Importantly, this use of simulation does not violate our assumptions, as the models are not employed during the training phase.

Performance is evaluated based on both task success rate and the extent to which an implicit

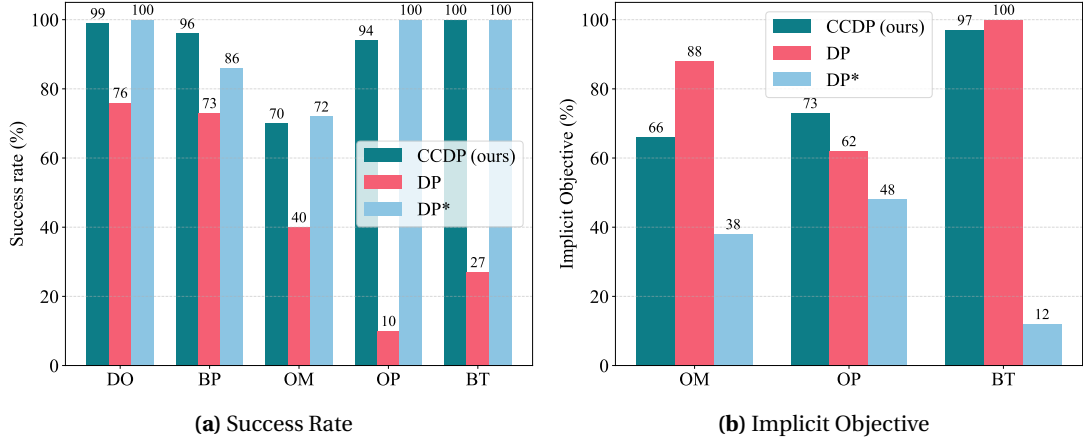


Figure 6.5: Results from 100 randomly sampled test scenarios. (a) Success rates across the five tasks. (b) Implicit objective-fulfillment metrics across the corresponding experiments. DO: Door Opening; BP: Button Pressing; OM: Object Manipulation; OP: Object Packing; BT: Bartender.

Table 6.1: Hyperparameter configurations used in the experiments.

	BP	DO	OP	OM	BT
δ_z	0.3	0.3	0.3	0.5	0.3
# history (L)	0	2	0	2	2
# pred (H)	1	8	8	8	8
# applied (b)	1	8	8	4	4
Batch size	1024	32	64	64	64
# hidden layer	[256,256,256]	[256,512,1024]			
# epoches	100				
noise scheduler	Cosine				
denoising step	100				

objective (e.g., optimality) is achieved. Since one of the aims of imitation learning is to perform tasks without explicitly modeling task or reward functions, some experiments define an implicit objective that is not directly provided to the robot. We then assess how well different methods meet this objective. The following sections detail the experimental setup and results.

In this work, we adopt the pipeline proposed in (Chi et al., 2024), where at each timestep, the robot observes a history of L past actions and states, predicts a trajectory of length H (typically the full horizon), and executes only the first b steps of that trajectory. This process is repeated until the task is completed. The specific values of H , L , and b for each experiment are provided in TABLE 6.1.

6.4.1 Door Opening (DO)

We consider a door-opening task in which the robot must infer the unknown opening direction (e.g., upward, sliding, or pulling) without explicit instructions. Demonstrations are unlabeled (except for the DP* baseline), so the robot relies solely on the actions as key features (i.e., $\mathbf{z}(\mathbf{x}^f, \mathbf{u}^f) = \mathbf{u}^f$). As shown in Fig. 6.5a, while the standard DP policy achieves a lower success rate, our method attains performance comparable to DP* without requiring labeling or explicit access to the door opening direction. An interaction is deemed successful if the robot opens the door within five trials.

6.4.2 Button Pressing (BP)

In this task, the robot must press a button within a predefined area without knowing its exact location. It samples various target points (used as key features, i.e., $\mathbf{z}(\mathbf{x}^f, \mathbf{u}^f) = \mathbf{x}_T$), and the task is deemed unsuccessful if the button is not found within 30 attempts. Our method outperforms the baselines without requiring additional discretization. In contrast, the DP* baseline discretizes the target area into bins, sampling from these regions and removing a bin after a failure; this approach improves over DP but is less efficient than CCDP.

6.4.3 Object Manipulation (OM)

In this task, the robot must move an object from one point to another without knowing its mass. As an implicit objective, the optimal manipulation strategy varies with object weight: light objects are best handled with single-arm pick-and-place, moderately heavy objects require bimanual manipulation, and very heavy objects are pushed. The demonstrations reflect these strategies by using single-hand pick-and-place 70% of the time, bimanual pick-and-place 20% of the time, and pushing 10% of the time. Although the demonstrations do not explicitly specify when each modality should be applied.

During testing, the robot is randomly assigned to one of the mass categories and must select the appropriate manipulation modality in the correct order. Here, the type of primitive serves as the key feature for failure recovery (i.e., $\mathbf{z}(\mathbf{x}^f, \mathbf{u}^f) = m$). A standard DP policy, being biased toward the dominant single-hand primitive, yields lower success rates. In contrast, both our method (CCDP) and the DP* baseline achieve higher success rates, with DP* slightly outperforming our method. When evaluating how frequently each method selects the appropriate action based on the implicit objective, our method adheres to the predefined order 66% of the time, compared to 88% for DP and 38% for DP*. Here, "respecting the order" means that if an object can be picked up with one hand (i.e., it is light), it should not be pushed or picked up bimanually.

6.4.4 Objects Packing (OP)

This task demonstrates that the proposed method extends beyond failure recovery. In Object Packing, the robot must place objects into designated baskets; however, as baskets fill up, it must choose the next closest available one. The demonstrations implicitly encode basket preferences (the closer, the more preferred) through the frequency of occurrence, without an explicit cost function. Here, key avoidance features are derived from the final states of object trajectories (i.e., $\mathbf{z}(\mathbf{x}^f, \mathbf{u}^f) = \mathbf{x}_T$). During testing, 12 objects are randomly placed on a table and must be assigned to baskets with a capacity of 4 each. The task is successful if each basket contains exactly 4 objects, and the implicit objective is achieved if the robot consistently selects the nearest available basket.

In this task, the standard DP policy performs poorly because its reliance on random sampling and history conditioning leads to prolonged attempts to use already-filled baskets. In contrast, both the DP* baseline and our method achieve over 90% success, with our method notably outperforming DP* in selecting the closest available basket.

6.4.5 Bartender (BT)

In this final task, we again demonstrate the method’s applicability beyond failure recovery. The robot is taught to fill multiple cups, with demonstrations implicitly prioritizing the nearest cup as the preferred option, followed by the second nearest, and so on. During testing, the robot must fill all cups while we record how often it selects the closest unfilled cup. Once a cup is filled, the sampler is guided to avoid targeting that cup again. The task is considered successful if all cups are filled, and for the implicit objective, we check if the closest available cup at each step is selected.

6.4.6 Ablation Study

We further evaluated the door-opening task with an ablation study to understand the role of the different components in (6.4). As shown in Fig. 6.6, when the task is solved on the first attempt, all model variants behave similarly. The differences appear when the robot requires multiple attempts. The history-conditioned denoiser baseline (which is a usual DP model) typically needs more trials to reach a successful solution. In contrast, our full model, which includes the conditional denoiser, solves almost all task variations within four attempts. This performance aligns with the ideal behavior for this task: if the first strategy fails, the robot should systematically explore alternative directions instead of repeatedly sampling similar actions.

We additionally evaluated a variant that removes the unconditional denoiser and defines the denoiser as

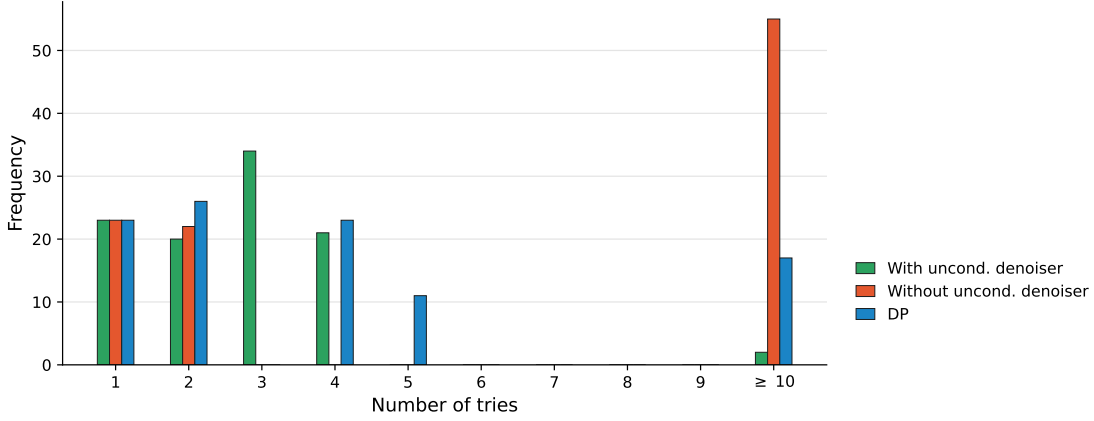


Figure 6.6: Frequency of the required number of trials to open the door using different denoisers. Data collected across 100 distinct door locations.

$$\epsilon_{\theta}(\mathbf{u}_t^k, \mathbf{x}_t, \mathbf{h}_t^L, \mathbf{z}^f 1 : N, k) = w_x \epsilon_x(\mathbf{u}_t^k, \mathbf{x}_t, k) + w_h \epsilon_h(\mathbf{u}_t^k, \mathbf{h}_t^L, k) + \sum_{i=1}^N w_z^i \epsilon_z(\mathbf{u}_t^k, \mathbf{z}_i^f, k). \quad (6.8)$$

Removing the unconditional denoiser leads to a clear degradation in performance. As illustrated in Fig. 6.6, once this variant fails twice, it tends to diverge during the denoising steps and is unable to recover. Its overall success rate becomes even lower than the DP baseline. This confirms that the unconditional denoiser plays a critical role in the CCDP framework: it stabilizes the composition of multiple experts and preserves the ability to explore new actions after repeated failures.

6.5 Discussion

Our results show that our approach significantly boosts task success compared to DP while preserving the implicit objectives from the demonstration data (unlike DP*). The performance gap between DP and CCDP was expected since DP does not consider past failures—a limitation our method explicitly addresses.

In this discussion, we elucidate the factors contributing to CCDP’s superior performance over DP* and explore potential avenues for further improvement. Additionally, we identify the current limitations of the approach and propose directions for future work to refine the method.

6.5.1 CCDP vs DP*

Apart from the major difference that DP* requires the system to be classified before application, whereas CCDP can be directly applied without any classification, another key distinction

lies in the nature of the guidance provided by each method. Specifically, DP* employs a *positive forcing* approach, where the policy is constrained to sample from a specific region. This restriction can limit the system's ability to fully leverage additional sensory and state information. In contrast, CCDP utilizes a *negative forcing* strategy, which restricts the policy from sampling in only one particular area, thereby allowing exploration across multiple regions. This flexibility enables the system to integrate broader contextual cues and converge toward areas that are guided by the available sensory and state information.

6.5.2 NOT Operation

Research such as (N. Liu et al., 2022) has demonstrated that, rather than combining multiple distributions via a product as in (6.2)—an operation analogous to a logical AND—it is possible to adopt an alternative approach based on a NOT operation. Hence, instead of learning from a dataset consisting of points that are well separated, the focus shifts to learning from a dataset of points that are closely clustered, after which the NOT operation is applied. This inversion could potentially simplify dataset creation and streamline the learning process.

However, both our experimental experience and the observations reported in (N. Liu et al., 2022) indicate that the NOT operation is inherently unstable. Our attempts to implement this strategy did not yield robust results. Consequently, we defer further exploration of this approach to future work. Nonetheless, the proposed method, which employs the composition of diffusion models to refine samples interactively, remains a viable and effective strategy.

6.5.3 Limitations and Future Work

There are two main limitations in the current version of the proposed model. First, failure key features have been defined manually. It would be advantageous to develop an autonomous approach for extracting these features, potentially by leveraging existing work in latent space extraction methods. The second limitation is that we have not yet investigated how to optimally adjust the combination weights in (6.4) to achieve improved performance. Our experience suggests that although larger weights may enhance exploration, they can also induce system instability.

As a further avenue for future work, we propose incorporating an offline exploratory mechanism to extract more effective recovery data and policies. This aspect was not considered in the current study, as our primary goal was to develop a purely data-driven approach without reliance on a simulation or a model.

6.6 Conclusion

This chapter aimed to address a common challenge in constrained robotic control, namely failure. Specifically, we explored the problem of failure recovery in imitation learning when the

demonstration dataset contains only successful demonstrations and lacks explicit examples of failure. Our goal was to investigate how the prior distribution learned from these demonstrations can be refined to account for potential failures, using failure signals as guidance to adapt and improve the robot’s behavior.

We proposed CCDP, a diffusion policy framework enhanced with failure avoidance, which adapts the action sampling process based on past unsuccessful attempts. Because CCDP only changes the prior sampling distribution, it can be combined with other approaches that refine final solutions, as shown in the previous chapters.

CCDP guides the system toward success by steering sampling away from regions similar to previously failed actions. This is achieved by decomposing the diffusion policy into multiple submodules, including a failure avoidance component trained on a synthesized dataset of actions dissimilar to known failures. During inference, this module guides sampling away from failure-prone regions, leading to more efficient exploration.

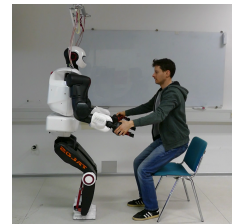
Experimental results across tasks such as object manipulation, packaging, and door opening show that CCDP improves both success rates and adherence to the implicit objectives encoded in demonstrations, outperforming baseline methods. A current limitation is that CCDP only handles failures with static causes. Future work could address this by integrating CCDP with more advanced failure detection and reasoning systems. Nevertheless, even in its present form, CCDP provides a valuable tool for improving the prior distribution of the action space, thereby enhancing the sampling efficiency of other methods.

7 Conclusion

7.1 Summary of contributions

In this thesis, we focused on the problem of constrained motion planning, which is ubiquitous in robotics applications such as assistive robotics and contact-rich manipulation. Our goal was to improve the efficiency and robustness of planning by addressing feasibility at different levels:

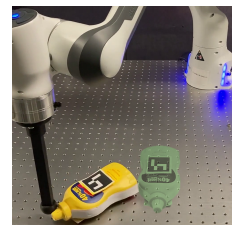
Chapter 3: We introduced better representations of the robot state that allow efficient handling of constraints such as humanoid balancing and joint limits. This facilitates robot control and results in natural motions that remain intuitive for a human user to influence.



Chapter 4: We leveraged expert demonstrations to guide the planner toward desired goals more efficiently. We showed that demonstrations are not only useful for initializing the planner (warm-starting) but also for shaping the optimization landscape by augmenting the cost function with additional gradient terms that bias the planner toward promising regions.



Chapter 5: We proposed a method to improve the performance of sampling-based motion planners in the presence of constraints by explicitly modeling the feasibility distribution. By combining feasibility and optimality distributions through a product-of-experts formulation, we ensured that sampled actions were more likely to be feasible and near-optimal. We further showed how tensor decomposition methods can be used to efficiently combine multiple experts and enable straightforward sampling.



Chapter 6: We addressed the challenge that in constrained settings, failures are often unavoidable. We proposed a method to recover from failures without requiring additional data or models, using only successful demonstrations. The key idea was to relax recovery from “choosing the correct action” to “choosing a sufficiently dissimilar action.” We demonstrated that such behavior can be achieved through the composition of conditional diffusion models.



We validated these approaches in a diverse set of experiments, including physical human–robot interaction, obstacle avoidance, object manipulation, and manifold-constrained planning. Experiments were conducted both in simulation and on real robotic platforms such as the Franka Emika Panda, Kinova arm, and the Talos humanoid robot

7.2 Key Insights

Key Insights

Beyond the specific technical contributions, this thesis provides several key insights into the problem of feasibility guided exploration and adaptation for constrained robotic manipulation:

- **Feasibility as a guiding principle.** Feasibility should not be treated as an afterthought, checked only after sampling. Instead, it must be explicitly incorporated in the planning and learning process. Encoding feasibility directly, through manifolds, feasibility distributions, or decomposition into experts, leads to more efficient, robust, and natural robot behaviors.
- **Demonstrations as heuristic priors.** Demonstrations are most useful when interpreted as heuristic priors rather than fixed solutions. They can guide optimization, provide warm-starts, and shape cost functions, while still allowing the system to adapt to new embodiments, task parameters, and environments.
- **Products of Experts as a unifying mechanism.** The products of experts (PoE) framework enables modular integration of different knowledge sources, feasibility, optimality, demonstrations, or failure avoidance, in a principled way. This paradigm provides a systematic path from “sample-then-project” approaches toward “project-then-sample” strategies, significantly improving sampling efficiency.
- **Adaptation and recovery without explicit models.** Real-world contact-rich manipulation often precludes precise modeling or exact parameter estimation. This thesis shows that adaptation can be achieved implicitly by shaping and composing distributions (e.g., feasibility-aware sampling, diffusion policy composition, domain contraction (T. Xue

et al., 2024a, 2025)), enabling recovery and robustness without heavy reliance on exact system identification.

- **Scalability through representation choices.** Selecting the right representation makes intractable problems manageable. Examples include manifold embeddings for humanoid balance, distance fields for whole-body geometry (Y. Li, Chi, et al., 2024; Y. Li, Zhang, et al., 2024), and tensor-train decompositions for high-dimensional distributions. Representations are therefore as important as the solver itself.
- **A general principle for constrained robotic manipulation.** Across the different approaches, a unifying message emerges: efficiency and robustness come from integrating feasibility guidance at every stage of the pipeline. The thesis advocates for a shift of perspective—“plan and adapt within the feasible set, not around it.”

7.3 Limitations

7.3.1 Chapter-specific limitations and future work

Each method proposed in each chapter carries its own specific limitations:

- **Chapter 3:** The proposed manifold representation for humanoid assistance relied on simplified models and quasi-static assumptions, where human-applied forces dominated in one direction and the robot primarily maintained balance. This restricted the ability to capture richer dynamic interactions between the human and the robot. *Future work:* Extend the manifold framework to handle fully dynamic interactions, including multi-directional forces and joint dynamics, and validate the approach in scenarios where both human and robot contribute actively to motion. Another important aspect that can highly modify the proposed reactive controller is to add anticipatory behavior where the robot considers solving an optimal control problem on the manifold to predict future states and adapt its actions accordingly.
- **Chapter 4:** Demonstrations were used for warm-starting and shaping, under the assumption that new tasks are close to previously observed experiences. This assumption may not hold in more diverse or unseen scenarios. *Future work:* Investigate methods to generalize beyond the demonstration distribution, for example, by combining demonstrations with reinforcement learning or by integrating uncertainty-aware mechanisms that can guide exploration when demonstrations provide limited coverage.
- **Chapter 5:** The feasibility-aware sampling with products of experts required explicit knowledge of constraint functions. Without such knowledge, the method cannot be applied directly. *Future work:* Explore ways to learn or approximate feasibility constraints from data, such as through differentiable constraint learning or probabilistic feasibility models, which would broaden the applicability of the approach to environments where constraints are not analytically available.

- **Chapter 6:** The proposed failure recovery strategy assumed that alternative successful actions already exist in the demonstration set, and that after a failure, the robot's observation space remains in-domain (i.e., it has seen similar states before). These assumptions can be restrictive in real-world scenarios with sparse data or out-of-distribution failures. *Future work:* Develop recovery strategies that can extrapolate beyond the demonstration set, for example by leveraging generative models or exploration mechanisms, and investigate how to handle out-of-distribution failure states where no direct recovery trajectory is available.

Overall, these limitations highlight the gap between controlled experimental validation and deployment in complex real-world scenarios, motivating the future directions discussed in the next section.

7.3.2 General limitations

Other than chapter specific limitations, there were some other limitations which were mostly across all chapters, several general assumptions and limitations are worth highlighting:

- **System infrastructure and observability.** We assumed that the only missing component is the constrained motion planning part, while all other elements of the robotic system work perfectly. This includes torque-level controllers, accurate system identification, and full-state observability. These assumptions allowed us to focus on the specific planning challenges, but in practice they are not always available. For example, if only raw high-dimensional images were accessible without a reliable state estimator, directly applying tensor-based methods would not be feasible, and dimensionality reduction would be required first.
- **Dependence on structured priors.** Many of the proposed methods rely on structured assumptions about feasibility or demonstrations. This reliance may limit applicability to highly unstructured tasks, robots with very different embodiments, or environments where constraints change rapidly.
- **Scalability and efficiency.** Although the proposed approaches improved efficiency compared to standard baselines, their scalability to very high-dimensional systems or deployment in real-time scenarios with tight latency requirements remains challenging.
- **Scope of constraints and validation.** The thesis primarily focused on feasibility expressed through kinematic, quasi-static, or geometric constraints. Other important forms of constraints, such as safety requirements, uncertainty handling, or coordination among multiple agents, were only touched upon indirectly. In addition, most methods assumed access to high-quality demonstration or state data, an assumption that may not hold in unstructured or noisy environments. Finally, while several experiments on physical robots were conducted, the validation was limited in scope. Broader testing

across different robotic platforms, tasks, and real-world conditions is still necessary to fully assess the robustness and generality of the proposed approaches.

7.4 Future Directions

In this section, we outline possible future directions in which the proposed methods can be extended, and discuss how they may also contribute to advancing other research areas.

7.4.1 Toward Other Algorithms

This thesis focused mainly on optimal control approaches (both gradient-based and sampling-based) and imitation learning. Other powerful paradigms, such as reinforcement learning (RL) (Sutton & Barto, 2018) and foundation models, were not explicitly considered. Nevertheless, the proposed methods are not isolated and can be integrated with these approaches in several ways.

For example, the manifold representation introduced in Chapter 3 could provide a structured state representation for RL, while the feasibility-aware sampling strategy from Chapter 5 could serve as an exploration mechanism for RL. Similarly, the failure recovery method from Chapter 6 could be used as a fallback policy when RL agents encounter unexpected failures. Conversely, RL can support our framework by filtering out low-quality demonstrations or augmenting datasets with synthetic samples, while foundation models could assist with failure detection or high-level reasoning. Exploring such bidirectional integration is a promising avenue for future work.

7.4.2 Toward Fully Integrated Systems

The methods proposed in this thesis have been studied mostly in isolation, but they can be combined into a single, unified framework. Fundamentally, our overall approach assumes that robot behaviors are generated by sampling from a distribution, modifying these samples through optimization, and applying the resulting actions to the robot. Within this context, each method developed in this thesis improves a different stage of this general pipeline:

1. Demonstrations provide compact, **low-dimensional task representations** that simplify and accelerate the optimization process.
2. **Feasibility models** guide the sampling process toward regions that satisfy task and environmental constraints.
3. **Imitation terms** shape the sampled distributions toward promising behaviors, improving both convergence and smoothness in optimization.
4. **Failure recovery mechanisms** handle unexpected events during execution, allowing

the system to adapt online and maintain task success.

An integrated system combining these components would unify the contributions presented across different chapters, leading to a more comprehensive solution for constrained motion planning and adaptive robotic manipulation. Such a framework would enable efficient exploration, robust optimization, and adaptive execution within a single, consistent architecture.

7.4.3 Common Sense Beyond Feasibility

In this thesis, guidance was mainly achieved by modeling feasibility distributions, which can be seen as a restricted form of common sense. Humans, however, use broader priors when interacting with objects, such as knowing that one must be close to an object in order to manipulate it. Extending feasibility guidance toward more general common-sense reasoning is an exciting research direction.

Foundation models (Barreiros et al., 2025) offer one possible avenue: although they are not yet reliable at physical reasoning, they can provide expressive priors or initializations for motion planners. Combining them with feasibility-aware models in a product-of-experts framework may yield more robust and physically grounded solutions. Potential applications include tool-use tasks, where reasoning about which tool to use is as important as how to use it, or whole-body manipulation tasks, which require reasoning about complex body-object interactions. New task representations, such as those proposed in (Y. Li, Chi, et al., 2024), may be particularly beneficial in this context.

7.4.4 Multi-agent, Multi-task, and Multi-scene Settings

Ultimately, robots will be deployed in environments where they must handle multiple tasks, across diverse scenes, and in collaboration with multiple agents. Extending feasibility-guided exploration and adaptation to such settings is a natural and ambitious next step.

One promising perspective is to treat each robot or task policy as an individual expert, and their combined behavior as a product of experts. This could provide a principled way to coordinate multiple agents or tasks. The challenge lies in the complexity of interactions, which are often nonlinear and high-dimensional, making feasibility modeling significantly harder. Nevertheless, exploring scalable composition mechanisms for multi-agent, multi-task, and multi-scene settings appears to be a compelling direction for future research.

7.5 Closing Remarks

This thesis has contributed new perspectives on feasibility-guided exploration and adaptation for constrained robotic manipulation. While the proposed methods were validated in controlled scenarios and partially on physical robots, the ultimate goal is to enable robots

to act reliably, safely, and adaptively in the unstructured environments of daily life. Bridging the gap between theoretical developments, efficient algorithms, and large-scale real-world deployment remains a challenging but promising direction. We hope that the ideas presented here—in particular, the use of feasibility distributions, demonstration guidance, and recovery strategies—can serve as building blocks toward more autonomous, versatile, and trustworthy robotic systems.

A Appendix

A.1 Appendix of chapter 5

A.1.1 Proj-MPPI

For this baseline, we aimed to project each sample into a feasible step before accepting it. Specifically, after sampling an action $\hat{\mathbf{u}}_t^i$ for sample i at state $\hat{\mathbf{x}}_t^i$, we project the action into $\bar{\mathbf{u}}_t^i = \alpha^* \hat{\mathbf{u}}_t^i$, where

$$\begin{aligned} \alpha^* &= \underset{\alpha \in [0,1]}{\operatorname{argmax}} \quad \alpha, \\ \text{s.t.} \quad &\mathbf{g}(\hat{\mathbf{x}}_{t+1}^i, \alpha \hat{\mathbf{u}}_t^i) \leq \mathbf{0}. \end{aligned} \tag{A.1}$$

This formulation performs a line search to determine the optimal scaling factor α^* , which identifies the best feasible action in the same direction as the sampled action while ensuring that constraints are satisfied.

A.1.2 NFs-PoE-MPPI

In this study, we compared our proposed method against normalizing flows (NFs) as a baseline for learning the static distribution. As discussed in the main text, NFs offer an advantage over other generative models by learning an invertible mapping function from a latent space to the primary space. This property enables the mapping of nominal solutions from the primary space to the latent space for sampling and subsequent remapping back to the primary space. However, it was noted in the main text that defining the nominal distribution in the latent space poses challenges.

To address this issue, we adopted an approach similar to what was outlined in (Power & Berenson, 2023) for FlowMPPI. Specifically, for half of the samples, we performed standard sampling in the primary space (similar to the MPPI method). For the remaining samples, we initially transformed the nominal mean $\boldsymbol{\mu}_{t+h}^x$ from the task space to the latent space $\boldsymbol{\mu}_{t+h}^z$, assumed a Gaussian distribution around the nominal latent mean, i.e., $\mathbf{z} \sim \mathcal{N}(\mathbf{z} | \boldsymbol{\mu}_{t+h}^z, \boldsymbol{\Sigma}_{t+h}^z)$,

Table A.1: Hyperparameters employed for learning the feasibility distribution using both tensor train and normalizing flow methods across various tasks. Note that the obstacle avoidance task refers to both PNGRID and the online obstacle avoidance task.

	Param.	Obstacle Avoidance	Push. Cyl.	Push. Box	Push. Mus.	Sphere man.	Sin. man.	Whole-body
TT	# State Discretization	100	10	10	10	25	25	50
	# Action Discretization	20	10	10	10	10	10	50
	State Refinement Scale	1	20	10	10	10	10	1
	Action Refinement Scale	10	20	10	10	10	10	1
NFs	# Coupling Layers	6	8	8	8	8	6	-
	Size of layers	256	256	256	256	256	256	-
	Batch size	4096	4096	8192	8192	8192	8192	-
	learning rate	0.0004	0.0002	0.0001	0.0001	0.0002	0.0004	-

combined it with the distribution learned using NFs, and then sampled from it. Finally, the samples were mapped back to the primary space. We predefined Σ_{t+h}^z to prioritize either the optimality or feasibility distribution based on its value. Given that the latent feasible space distribution has an identity covariance (with zero mean), a value of Σ_{t+h}^z greater than 1 in a specific dimension indicates a higher priority for the feasibility distribution, while a value less than 1 prioritizes the optimality distribution. The optimal value for Σ_{t+h}^z can vary depending on the task.

NFs were trained using the method described by Dinh et al., 2016. To learn the conditional distribution using NFs, we trained the mapping function for all state and action variables simultaneously, applying an identity transformation to the state variables. We employed the same amount of data as used for TT-SVD, with the parameters detailed in Table A.1.

A.1.3 MPPI on the Sphere Manifold

For the intrinsic method used in Section 5.4.2, we adopt the exponential map to enable sampling in the tangent space of the sphere, followed by mapping the samples back to the manifold. At each timestep, the tangent space is defined at a point $\mathbf{x}_t \in \mathcal{M} = \{\mathbf{x} \in \mathbb{R}^3 \mid \|\mathbf{x}\| = r\}$ on the sphere. Any valid action $\mathbf{u}_t \in \mathbb{R}^3$ must lie in this tangent space, and thus satisfy the orthogonality condition $\mathbf{x}_t^\top \mathbf{u}_t = 0$.

This tangent space is a 2D subspace of $\mathbb{R}^3$, and we generate actions $\mathbf{u}_t$ by first sampling in this lower-dimensional space: $\mathbf{u}_t^{2D} \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$. The mapping from the 2D tangent coordinates to the 3D ambient space is defined as

$$\mathbf{u}_t = \mathbf{T}(\mathbf{x}_t) \mathbf{u}_t^{2D}, \quad (\text{A.2})$$

where the transformation matrix $\mathbf{T}(\mathbf{x}_t) = [\mathbf{e}_1^\top(\mathbf{x}_t), \mathbf{e}_2^\top(\mathbf{x}_t)]^\top$ is constructed from an orthonormal basis of the tangent space at $\mathbf{x}_t$

$$\mathbf{e}_1 = \frac{\mathbf{l} \times \mathbf{x}_t}{\|\mathbf{l} \times \mathbf{x}_t\|}, \quad \mathbf{e}_2 = \frac{\mathbf{e}_1 \times \mathbf{x}_t}{\|\mathbf{e}_1 \times \mathbf{x}_t\|}. \quad (\text{A.3})$$

Here, $\mathbf{l}$ is an arbitrary vector chosen to avoid alignment with $\mathbf{x}_t$, ensuring a well-defined cross

product

$$\mathbf{l} = \begin{cases} [1, 0, 0]^\top & \text{if } |\mathbf{x}_t^\top [1, 0, 0]^\top| \leq 0.9, \\ [0, 1, 0]^\top & \text{otherwise.} \end{cases} \quad (\text{A.4})$$

After sampling actions and mapping them to the task space, the next state $\mathbf{x}_{t+1}$ is computed using the exponential map on the sphere

$$\mathbf{x}_{t+1} = \exp_{\mathbf{x}_t}(\mathbf{v}) = \cos\left(\frac{\|\mathbf{v}\|}{r}\right) \mathbf{x}_t + \sin\left(\frac{\|\mathbf{v}\|}{r}\right) \frac{r \mathbf{v}}{\|\mathbf{v}\|}, \quad (\text{A.5})$$

where $\mathbf{v} = \mathbf{u}_t \cdot dt$ is the scaled action vector, and r is the radius of the sphere. This formulation ensures that the resulting state $\mathbf{x}_{t+1}$ remains on the sphere manifold.

A.1.4 DIAL-mpc and TT-PoE-DIAL

Inspired by the success of diffusion-based methods across various domains, Pan et al., 2024 proposed a novel MPPI variant for trajectory optimization. In their approach, each MPPI iteration is repeated N_{diffuse} times using different noise realizations. Their results (which we also observe in our own experiments) demonstrate that this method significantly outperforms vanilla MPPI and several other baselines. Building on this idea, H. Xue et al., 2024 further extended the method to a full MPC framework.

In their formulation, the noise covariance at each diffusion step is defined as

$$\Sigma_{t+h}^n = \exp\left(-\frac{N_{\text{diffuse}} - n}{\beta_1 N_{\text{diffuse}}} - \frac{H - h}{\beta_2 H}\right) \mathbf{I}, \quad (\text{A.6})$$

where β_1 and β_2 are temperature parameters controlling the diffusion process and the time horizon, respectively. For simplicity, we define two decay factors: $f_{\text{denoise}} = \exp\left(-\frac{1}{\beta_1 N_{\text{diffuse}}}\right)$ and $f_{\text{horizon}} = \exp\left(-\frac{1}{\beta_2 H}\right)$.

Algorithm 2 outlines how this diffusion-based MPPI method can be integrated into the TT-PoE pipeline. The only modifications required are the addition of lines 5 and 6, and replacing Gaussian sampling with the TT-based sampling approach described in Equation (5.12).

A.1.5 Experiments

Different settings used for different methods in various tasks are listed in Table A.2. These variables include parameters like the length of the planning horizon H , the covariance matrices in the task space Σ_{task} , the covariance matrix Σ_z for the latent space, the temperature variable β used in MPPI, the simulation time step dt , the maximum velocity u_{max} , and the state limits x_{max} . Task success is defined by meeting predefined criteria: completion within a specified number of steps T_{succ} and achieving a cost value below a predetermined threshold c_{succ} . It is important to note that for the MPPI method, action commands are manually clipped to stay

Algorithm 2 TT-PoE-DIAL

Require: $\mathbf{x}_t, \boldsymbol{\mu}_{t,\dots,t+H-1}, \boldsymbol{\sigma}_{t,\dots,t+H-1}$
Return: $\mathbf{u}_t^*$

```

1:  $\hat{\mathbf{x}}_t \leftarrow \mathbf{x}_t$ 
2: for  $n$  from 0 to  $N_{\text{diffuse}}$  do
3:   for  $h$  from 0 to  $H-1$  do
4:     Update the covariance matrix ▷ Eq. (A.6)
5:     Calculate  $\mathcal{P}_{\mathcal{N}_m}$  ▷ Eq. (5.13)
6:      $\mathcal{P}_h \leftarrow$ Product of the distributions ▷ Eq. (5.12)
7:      $\hat{\mathbf{u}}_{t+h}^0, \dots, \hat{\mathbf{u}}_{t+h}^N \leftarrow$ Sample  $N$  actions
8:      $\hat{\mathbf{x}}_{t+h+1}^0, \dots, \hat{\mathbf{x}}_{t+h+1}^N \leftarrow$ Forward Dynamics
9:   end for
10:   $c^0, \dots, c^N \leftarrow$ Calculate cost values
11:   $\mu_t \leftarrow$  update the mean value ▷ Eq. (2.14)
12: end for
13:  $\mathbf{u}_t^* \leftarrow \mu_t$ 

```

Table A.2: Hyperparameters employed with different controllers. The obstacle avoidance task refers to both PNRID and the online obstacle avoidance task.

	Obstacle Avoidance	Push. Cyl.	Push. Box	Push. Must.	Sphere man.	Sinus man.	Whole-body
H	15	32	20	20	50	30	15
Σ_{task}	$0.125\mathbf{I}$	$0.3\mathbf{I}$	$0.05\mathbf{I}$	$0.05\mathbf{I}$	$0.1\mathbf{I}$	$0.05\mathbf{I}$	$0.05\mathbf{I}$
Σ_z	$10\mathbf{I}$	$0.5\mathbf{I}$	$0.01\mathbf{I}$	$0.05\mathbf{I}$	$0.2\mathbf{I}$	$10\mathbf{I}$	-
β	0.05	0.05	0.05	0.05	0.05	0.05	0.01
$dt[s]$	0.1	0.05	0.05	0.05	0.1	0.1	0.1
$u_{\max}[\frac{m}{s}]$	1	1	1	1	1	1	Real Limits
$x_{\max}[m]$	1.25	1	1	0.4	0.25	0.5	Real Limits
$T_{\text{succ.}}$	100	500	500	400	400	200	25
$c_{\text{succ.}}$	10^{30}	10^4	10^4	10^3	3×10^5	10^{30}	10^6
N_{diffuse}	-	-	-	-	-	-	3
f_{denoise}	-	-	-	-	-	-	0.22
f_{horizon}	-	-	-	-	-	-	0.95

within feasible ranges while other methods handle this internally.

Upon computing the cost value for sample i , we proceed to normalize them using the minimum observed cost value among the samples, expressed as

$$c^i = \frac{\hat{c}^i}{c_{\min}}, \quad \text{s.t.} \quad c_{\min} = \min(\hat{c}^0, \dots, \hat{c}^N), \quad (\text{A.7})$$

where the notation $\hat{\cdot}$ denotes the values of the cost functions before normalization. Additionally, we incorporate an extra step by assuming one of the samples to have zero action. This facilitates the system to halt in case all samples become infeasible. Further explanation regarding the employed cost functions in these experiments is provided in the following.

Obstacle avoidance:

In these tasks (i.e., PNGRID and the online obstacle avoidance), our objective is to navigate towards a designated target point $\mathbf{p}_{\text{target}}$, while navigating around obstacles and staying within the environmental boundary. For addressing this task, we employed a single integrator point mass as our agent, resulting in a 4-dimensional state-action space. The TT-distribution for this function is depicted in Fig. 5.6-c for zero velocity. Notably, the region without obstacles exhibits a higher probability value compared to other areas. The occurrence of negative probabilities in some points is attributed to the imperfections in the TT-SVD approximation. While this could be mitigated by increasing the maximum rank value, which is set to 300 for this task, we observed satisfactory results with this configuration. The distribution is determined by discretizing the state space into 100 partitions and the action space into 20 partitions. However, the partition in the TT-cores corresponding to the action commands undergoes an additional refinement by interpolating between different nodes resulting in 10 times more partition. The samples generated by the model with zero velocity are illustrated in Fig. 5.6-d and can be contrasted with the samples obtained from a Gaussian mixture model (GMM) employing $K = 400$ Gaussian components, as shown in Fig. 5.6-b. The GMM model is trained with an EM algorithm using the Scikit-learn library (Pedregosa et al., 2011) in Python.

To assess the efficacy of a sample, we employ a performance evaluation as

$$\hat{c}_{\text{Obst. Avoid.}}(\hat{\mathbf{U}}_t) = \sum_{h=0}^{H-1} r_{t+h} \left(10 \|\hat{\mathbf{x}}_{t+h} - \mathbf{p}_{\text{target}}\|^2 + 10^{30} \text{coll}(\hat{\mathbf{x}}_{t+h}) + 10^{-3} \|\hat{\mathbf{u}}_{t+h}\|^2 \right) + 10^3 r_{t+H} \|\hat{\mathbf{x}}_{t+H} - \mathbf{p}_{\text{target}}\|^2 \quad (\text{A.8})$$

where $\text{coll}(\cdot)$ denotes a function that returns 1 if the point collides with an obstacle or exceeds the feasible area, and 0 otherwise. During the planning and learning phases of the feasibility distribution for the PNGRID task, we consider a 5-centimeter margin around obstacles to define collisions. In (A.8), r_{t+h} serves as an indicator to determine whether the agent has previously reached the target by time step $t+h$ or not. This mechanism prevents the controller from solely aiming for the target at the final time step. We consider the target as reached if the distance between the agent and the target point falls below 5 centimeters.

For Expert 2, we have utilized the following feasibility distribution

$$p^f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) = \mathbf{1}\{\hat{\mathbf{x}}_t \in \mathcal{X}, \hat{\mathbf{x}}_{t+1} \in \mathcal{X}, (\hat{\mathbf{x}}_t, \hat{\mathbf{x}}_{t+1}) \cap \mathcal{O} = \emptyset\},$$

where, $\mathcal{X}$ denotes the allowable state-space (inside the boundaries), $\mathcal{O}$ the obstacle set, and $\mathbf{1}\{\cdot\}$ is the indicator function.

For the online obstacle avoidance task, we utilized the same values without accounting for any collision margins. The only additional modification was that the obstacle became detectable to the solver when the robot was within 40 centimeters horizontally from the center of obstacles, which had radii of up to 40 centimeters, selected randomly.

Planar-pushing:

We have modeled the robot end-effector as a 2D point mass under the control of a velocity controller. For each object, we have utilized distinct parameters, detailed in Table A.2, and is subjected to unique cost functions, elaborated upon subsequently. For Expert 2, however, all objects share the same feasibility distribution

$$p^f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) = \mathbf{1} \left\{ \hat{\mathbf{x}}^{\text{obj}} \in \mathcal{X} \wedge (\Delta_o > 0 \vee \Delta_d > 0) \right\},$$

where

$$\begin{aligned} \Delta_o &= \|\hat{\mathbf{x}}_{t+1}^{\text{obj}} - \hat{\mathbf{x}}_t^{\text{obj}}\|, \\ \Delta_d &= \|\hat{\mathbf{x}}_t^{\text{rob}} - \hat{\mathbf{x}}_t^{\text{obj}}\| - \|\hat{\mathbf{x}}_{t+1}^{\text{rob}} - \hat{\mathbf{x}}_{t+1}^{\text{obj}}\|. \end{aligned}$$

Here, $\hat{\mathbf{x}}^{\text{obj}}$ and $\hat{\mathbf{x}}^{\text{rob}}$ denote the object and robot states, respectively. This distribution assigns high probability only to samples in which the object remains within the feasible region and either (i) the object moves ($\Delta_o > 0$) or (ii) the robot-object distance decreases ($\Delta_d > 0$).

Cylinder: The cost function designed for achieving a target point $\mathbf{p}_{\text{target}}$ in this task is expressed as

$$\begin{aligned} \hat{c}_{\text{cyl}}(\hat{\mathbf{U}}_t) &= \sum_{h=0}^{H-1} r_{t+h+1} \left(10 \|\hat{\mathbf{x}}_{t+h+1}^{\text{obj}} - \mathbf{p}_{\text{target}}\|^2 + 10^{-2} \text{coll_o}(\hat{\mathbf{x}}_{t+h+1}^{\text{obj}}) + 10 \text{coll_b}(\hat{\mathbf{x}}_{t+h+1}^{\text{obj}}) \right. \\ &\quad \left. + \text{coll_b}(\hat{\mathbf{x}}_{t+h+1}^{\text{rob}}) + 10^{-1} \text{SDF}^2(\hat{\mathbf{x}}_{t+h+1}^{\text{obj}}, \hat{\mathbf{x}}_{t+h+1}^{\text{rob}}) \right), \quad (\text{A.9}) \end{aligned}$$

where functions $\text{coll_o}(\cdot)$ and $\text{coll_b}(\cdot)$ determine collision with the obstacles and the boundary, respectively, returning 1 in case of collision and 0 otherwise. During the planning phase, a 5- and 10-centimeter margin around obstacles is considered for collision detection in the mustard bottle and box task, respectively. Additionally, $\text{SDF}(\cdot)$ calculates the signed distance value between the robot and the agent. Similarly to the Obstacle avoidance experiment, r_{t+h} serves as an indicator to verify whether the agent reached the target before the time step $t+h$. We define the target as reached when the distance between the agent and the target point is less than 5 centimeters.

Pushing a Mustard Bottle and a Box We use the same cost function for these two tasks which aimed at reaching a target position $\mathbf{p}_{\text{target}}$ and desired yaw angle o^d as

$$\begin{aligned} \hat{c}_{\text{mus./Box}}(\hat{\mathbf{U}}_t) &= \sum_{h=0}^{H-1} r_{t+h+1} \left(10 \|\hat{\mathbf{x}}_{t+h+1}^{\text{obj}} - \mathbf{p}_{\text{target}}\|^2 + (\hat{o}_{t+h} - o^d)^2 + 10^5 e^{-\frac{h}{4}} \text{coll_b}(\hat{\mathbf{x}}_{t+h+1}^{\text{obj}}) \right. \\ &\quad \left. + 10^5 e^{-\frac{h}{4}} \text{coll_b}(\hat{\mathbf{x}}_{t+h+1}^{\text{rob}}) + 10 \|\hat{\mathbf{x}}_{t+h+1}^{\text{rob}} - \hat{\mathbf{x}}_{t+h+1}^{\text{obj}}\|^2 \right) + 10^3 r_{t+H} \left(\|\hat{\mathbf{x}}_{t+H}^{\text{obj}} - \mathbf{p}_{\text{target}}\|^2 + (\hat{o}_{t+H} - o^d)^2 \right) \\ &\quad + s_{\text{stuck}} \text{Ang_rel}(\hat{\mathbf{x}}_t^{\text{obj}}, \hat{\mathbf{x}}_t^{\text{rob}}, \hat{\mathbf{x}}_{t+H}^{\text{rob}}, \theta^d), \quad (\text{A.10}) \end{aligned}$$

where $\hat{o}$ denotes the orientation of the object, and the remaining notation is the same as in the previous task. Additionally, for boundary cost considerations, a margin of 10 and

16 centimeters is assumed during the training and planning phases, respectively. The key distinction between (A.10) and (A.9) lies in the final term, which evaluates the degree of rotation of the robot around the object compared to its initial position. Due to potential issues arising from the object geometry causing the robot to become stuck on one side (i.e., $s = 1$), this term is introduced to encourage the robot to rotate around the object by approximately $\theta^d = \pm \frac{\pi}{2}$ when such circumstances occur. The value of s is determined by

$$s_{\text{stuck}} = \begin{cases} 1, & \text{if } \exp(-|\text{dist}_{t+H}^{\min} - \text{dist}_t|) > 0.96 \\ 0, & \text{otherwise} \end{cases},$$

where dist_{t+h} represents the distance of the object to the target (disregarding orientation) at time step $t + h$, and the superscript min indicates the minimum distance among the samples. It is important to note that this cost is applied to all samples, penalizing instances where the system remains on one side of the object.

In this experiment, the task is deemed complete if the object is within a distance of less than 10 centimeters from the target, and the orientation difference from the desired angle is less than 0.2 radians.

Tube Tracking:

In this task, our objective is to control the robot end-effector's position within a predefined area. In the sphere manifold scenario, the area is bounded by two spheres: the inner sphere has a radius of 15 cm, while the outer sphere has a radius of 20 cm. For the sinusoidal wave, the area is defined as the region inside $\{z \mid z \in 0.1 \sin(4y\pi) \pm 0.03, \quad y \in [-1, 1]\}$. During both the learning and planning phases, collision detection incorporates a 1-cm margin to ensure a safety buffer.

The cost function designed for this task is expressed as

$$\hat{c}_{\text{Manifold}}(\hat{\mathbf{U}}_t) = \sum_{h=0}^{H-1} r_{t+h} \left(10^2 \|\hat{\mathbf{x}}_{t+h} - \mathbf{p}_{\text{target}}\|^2 + 10^4 \text{coll}(\hat{\mathbf{x}}_{t+h}) + 10^{-3} \|\hat{\mathbf{u}}_{t+h}\|^2 \right) + 10^2 r_{t+H} \|\hat{\mathbf{x}}_{t+H} - \mathbf{p}_{\text{target}}\|^2, \quad (\text{A.11})$$

where the variables and terms retain the same definitions as previously described.

For Expert 2, we employ the following feasibility distribution

$$p^f(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) = \mathbf{1}\{\hat{\mathbf{x}}_t \in \mathcal{X}, \hat{\mathbf{x}}_{t+1} \in \mathcal{X}, \hat{\mathbf{x}}_t, \hat{\mathbf{x}}_{t+1} \in \mathcal{M}_\epsilon\},$$

where

$$\mathcal{M}_\epsilon = \{\mathbf{x} \mid \text{dist}(\mathbf{x}, \mathcal{M}) \leq \epsilon\}$$

is an ϵ -margin around the manifold $\mathcal{M}$. This ensures that both the current and next states lie

within the admissible state space $\mathcal{X}$ and remain close to $\mathcal{M}$ within the specified tolerance.

Whole-body Obstacle Avoidance:

In this experiment, we control a Franka Emika robot in the presence of randomly placed spherical obstacles. The optimality distribution cost is

$$\hat{c}_{\text{wb}}(\hat{\mathbf{U}}_t) = \sum_{h=0}^{H-1} r_{t+h} \left(10 \|\hat{\mathbf{x}}_{t+h} - \mathbf{p}_{\text{target}}\|^2 + 10^6 \text{coll_b}(\hat{\mathbf{x}}_{t+h}) + 10^6 \exp\left(-\frac{\text{dist}(\hat{\mathbf{x}}_{t+h})}{0.01}\right) + 10 \|\hat{\mathbf{u}}_{t+h}\|^2 \right) + 10^3 r_{t+H} \|\hat{\mathbf{x}}_{t+H} - \mathbf{p}_{\text{target}}\|^2, \quad (\text{A.12})$$

where all quantities are in joint space, $\text{dist}(\cdot)$ is computed via the Robot Distance Field (Y. Li, Zhang, et al., 2024), and obstacles are five spheres of radius 0.2 m.

The feasibility expert uses a soft-penalty distribution:

$$p^f(\mathbf{x}_{1:N_o}^o, \mathbf{x}_t, \mathbf{u}_t) = \prod_{\tau \in \{t, t+1\}} \left(1 - \exp\left(-\frac{\text{dist}(\mathbf{x}_\tau)}{\lambda}\right) \right),$$

where $\lambda = 0.5$. This definition assigns low probability to states near obstacles, ensuring both the current and next robot configurations remain safely distant from all obstacles.

A.1.6 Real-robot Experiment

For the real robot experiment, we adapted the pipeline initially developed for simulating the pushing of a mustard bottle. However, we made an adjustment by reducing the number of horizons to 10. As a safety precaution, we constrained the robot's speed to one-twentieth of its simulated counterpart, with $u_{\text{max}} = 0.05$ and $\mathbf{u}^{\text{real}} = \frac{\mathbf{u}^{\text{sim}}}{20}$. Additionally, each command is executed 20 times, pushing the object to the same target point as planned. After each set of executions, the robot refreshed its action commands and iterated the process until reaching the target.

In this experiment, we employed a 7-axis Franka robot, which we controlled using a Cartesian velocity controller operating at 1000 Hz. For the visual feedback, we utilized a RealSense D435 camera, operating at 30 Hz. Detailed results and observations from these experiments are available in the accompanying videos.

Bibliography

- Abbeel, P., & Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. *Proceedings of the twenty-first international conference on Machine learning*, 1.
- Aceituno, B., & Rodriguez, A. (2022). A Hierarchical Framework for Long Horizon Planning of Object-Contact Trajectories. *IROS*, 189–196.
- Aceituno-Cabezas, B., & Rodriguez, A. (2020). A global quasi-dynamic model for contact-trajectory optimization in manipulation. *Proceedings of Robotics: Science and Systems (RSS)*.
- Adetola, V., DeHaan, D., & Guay, M. (2009). Adaptive model predictive control for constrained nonlinear systems. *Systems & Control Letters*, 58(5), 320–326.
- Ahmadi, A. A., & Zhang, J. N. (2018). On the complexity of testing attainment of the optimal value in nonlinear optimization. *Mathematical Programming*, 171(1), 105–128.
- Alizadeh, F., & Goldfarb, D. (2003). Second-order cone programming. *Mathematical Programming*, 95(1), 3–51.
- Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., & Zaremba, W. (2017). Hindsight experience replay. *Advances in Neural Information Processing Systems (NeurIPS)*, 30.
- Anitescu, M. (2006). Optimization-based simulation of nonsmooth rigid multibody dynamics. *Mathematical Programming*, 105, 113–143.
- Atkeson, C. G., & Stephens, B. (2007). Multiple balance strategies from one optimization criterion. *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, 57–64.
- Barreiros, J., Beaulieu, A., Bhat, A., Cory, R., Cousineau, E., Dai, H., Fang, C.-H., Hashimoto, K., Irshad, M. Z., Itkina, M., Kuppaswamy, N., Lee, K.-H., Liu, K., McConachie, D., McMahon, I., Nishimura, H., Phillips-Grafflin, C., Richter, C., Shah, P., . . . Team, T. L. (2025). A careful examination of large behavior models for multitask dexterous manipulation. <https://arxiv.org/abs/2507.05331>

- Batsheva, A., Chertkov, A., Ryzhakov, G., & Oseledets, I. (2023). PROTES: Probabilistic Optimization with Tensor Sampling. *arXiv preprint arXiv:2301.12162*.
- Berenson, D. (2011). *Constrained manipulation planning*. Carnegie Mellon University.
- Bertsekas, D. P. (1999). *Nonlinear programming* (2nd). Athena Scientific.
- Bhardwaj, M., Sundaralingam, B., Mousavian, A., Ratliff, N., Fox, D., Ramos, F., & Boots, B. (2021). Further experimental results: manipulation-mpc [Accessed: 2025-07-30]. <https://sites.google.com/view/manipulation-mpc/further-experimental-results#h.ud1wkx4676yl>
- Bhardwaj, M., Sundaralingam, B., Mousavian, A., Ratliff, N. D., Fox, D., Ramos, F., & Boots, B. (2022). Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation. *Conference on Robot Learning*, 750–759.
- Boggs, P. T., & Tolle, J. W. (1995). Sequential quadratic programming. *Acta Numerica*, 4, 1–51.
- Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.
- Calinon, S. (2016). A tutorial on task-parameterized movement learning and retrieval. *Intelligent Service Robotics*, 9(1), 1–29. <https://doi.org/10.1007/s11370-015-0187-9>
- Calinon, S. (2019). Mixture models for the analysis, edition, and synthesis of continuous time series. In N. Bouguila & W. Fan (Eds.), *Mixture models and applications* (pp. 39–57). Springer, Cham. https://doi.org/10.1007/978-3-030-23876-6_3
- Calinon, S., Guenter, F., & Billard, A. (2007). On learning, representing and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation*, 37(2), 286–298.
- Calinon, S., & Lee, D. (2019). Learning control. In P. Vadakkepat & A. Goswami (Eds.), *Humanoid robotics: a reference* (pp. 1261–1312). Springer. https://doi.org/10.1007/978-94-007-6046-2_68
- Calli, B., Singh, A., Bruce, J., Walsman, A., Konolige, K., Srinivasa, S., Abbeel, P., & Dollar, A. M. (2017). Yale-CMU-Berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 36(3), 261–268.
- Carius, J., Ranftl, R., Farshidian, F., & Hutter, M. (2021). Constrained stochastic optimal control with learned importance sampling: A path integral approach. *The International Journal of Robotics Research*, 41(2), 189–209.
- Chavan-Dafle, N., Holladay, R., & Rodriguez, A. (2018). In-hand manipulation via motion cones. *arXiv preprint arXiv:1810.00219*.

- Chavan-Dafle, N., Holladay, R., & Rodriguez, A. (2020). Planar in-hand manipulation via motion cones. *The International Journal of Robotics Research*, 39(2-3), 163–182.
- Chi, C., Xu, Z., Feng, S., Cousineau, E., Du, Y., Burchfiel, B., Tedrake, R., & Song, S. (2024). Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. *The International Journal of Robotics Research*.
- Cornelio, C., & Diab, M. (2024). Recover: a neuro-symbolic framework for failure detection and recovery. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 12435–12442.
- Dafle, N., Rodriguez, A., Paolini, R., Tang, B., Srinivasa, S., Erdmann, M., Mason, M., Lundberg, I., Staab, H., & Fuhlbrigge, T. (2014). Extrinsic dexterity: In-hand manipulation with external forces. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 1578–1585.
- Dai, H., Valenzuela, A., & Tedrake, R. (2014). Whole-body motion planning with centroidal dynamics and full kinematics. *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, 295–302.
- Deisenroth, M. P., & Rasmussen, C. E. (2011). PILCO: a model-based and data-efficient approach to policy search. *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 465–472.
- Diehl, M., Bock, H. G., Diedam, H., & Wieber, P.-B. (2005). Fast direct multiple shooting algorithms for optimal robot control. In *Fast Motions in Biomechanics and Robotics*.
- Dinh, L., Sohl-Dickstein, J., & Bengio, S. (2016). Density estimation using Real NVP. *International Conference on Learning Representations (ICLR)*.
- Dolgov, S., Anaya-Izquierdo, K., Fox, C., & Scheichl, R. (2020). Approximation and sampling of multivariate probability distributions in the tensor train decomposition. *Statistics and Computing*, 30, 603–625.
- Drillis, R., Contini, R., & Bluestein, M. (1964). Body Segment Parameters; a Survey of Measurement Techniques. *Artificial limbs*, (2), 44–66.
- Duan, J., Pumacay, W., Kumar, N., Wang, Y. R., Tian, S., Yuan, W., Krishna, R., Fox, D., Mandlekar, A., & Guo, Y. (2024). Aha: a vision-language-model for detecting and reasoning over failures in robotic manipulation. *arXiv preprint arXiv:2410.00371*.
- El-Hussieny, H., Asker, A., & Salah, O. (2017). Learning the sit-to-stand human behavior: An inverse optimal control approach. *13th International Computer Engineering Conference (ICENCO)*, 112–117.
- Farouki, R. T. (2012). The bernstein polynomial basis: a centennial retrospective. *Computer Aided Geometric Design*, 29(6), 379–419.

- Florence, P., Lynch, C., Zeng, A., Ramirez, O. A., Wahid, A., Downs, L., Wong, A., Lee, J., Mordatch, I., & Tompson, J. (2022). Implicit behavioral cloning. *Conference on robot learning*, 158–168.
- Forte, D., Gams, A., Morimoto, J., & Ude, A. (2012). On-line motion synthesis and adaptation using a trajectory database. *Robotics and Autonomous Systems*, 60(10), 1327–1339.
- Geravand, M., Korondi, P. Z., Werner, C., Hauer, K., & Peer, A. (2017). Human sit-to-stand transfer modeling towards intuitive and biologically-inspired robot assistance. *Autonomous Robots*, 41(3), 575–592.
- Gkanatsios, N., Jain, A., Xian, Z., Zhang, Y., Atkeson, C., & Fragkiadaki, K. (2023). Energy-based Models are Zero-Shot Planners for Compositional Scene Rearrangement. *Robotics: Science and Systems*.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.
- Graesdal, B. P., Chia, S. Y. C., Marcucci, T., Morozov, S., Amice, A., Parrilo, P. A., & Tedrake, R. (2024). Towards tight convex relaxations for contact-rich manipulation. *Proceedings of Robotics: Science and Systems (RSS)*.
- Han, S.-P., & Mangasarian, O. L. (1979). Exact penalty functions in nonlinear programming. *Mathematical Programming*, 17(1), 251–269.
- Hartley, R., Mangelson, J., Gan, L., Ghaffari-Jadidi, A., Walls, J. M., Eustice, R. M., & Grizzle, J. W. (2018). Legged robot state-estimation through combined forward kinematic and preintegrated contact factors. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 4422–4429.
- Hinton, G. E. (1999). Product of experts. *Proc. Intl Conf. on Artificial Neural Networks. (ICANN)*.
- Hirukawa, H., Hattori, S., Harada, K., Kajita, S., Kaneko, K., Kanehiro, F., Fujiwara, K., & Morisawa, M. (2006). A universal stability criterion of the foot contact of legged robots - adios zmp. *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*, 1976–1983.
- Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. *arXiv preprint arxiv:2006.11239*.
- Holladay, R., Lozano-Pérez, T., & Rodriguez, A. (2023). Robust planning for multi-stage forceful manipulation. *The International Journal of Robotics Research*.
- Howell, T., Gileadi, N., Tunyasuvunakool, S., Zakka, K., Erez, T., & Tassa, Y. (2022). Predictive sampling: Real-time behaviour synthesis with mujoco. *arXiv preprint arXiv:2212.00541*.

- Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., et al. (2022). Inner monologue: embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*.
- Ibanez, A., Bidaud, P., & Padois, V. (2012). Unified preview control for humanoid postural stability and upper-limb interaction adaptation. *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 1801–1808.
- Ibanez, A., Bidaud, P., & Padois, V. (2017). Optimization-based control approaches to humanoid balancing. In *Humanoid robotics: a reference* (pp. 1–27). Springer Netherlands.
- Ijspeert, A. J., Nakanishi, J., Hoffmann, H., Pastor, P., & Schaal, S. (2013). Dynamical movement primitives: Learning attractor models formotor behaviors. *Neural Computation*, 25(2), 328–373.
- Jankowski, J., Bruder Müller, L., Hawes, N., & Calinon, S. (2023). VP-STO: Via-point-based Stochastic Trajectory Optimization for Reactive Robot Behavior. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 10125–10131.
- Jankowski, J., Bruder Müller, L., Hawes, N., & Calinon, S. (2025). Robust pushing: exploiting quasi-static belief dynamics and contact-informed optimization. *The International Journal of Robotics Research*, 0(0).
- Jinkyu Kim, I. K., & Park, F. C. (2016). Randomized path planning for tasks requiring the release and regrasp of objects. *Advanced Robotics*, 30(4), 270–283.
- Jordan, M. I., & Jacobs, R. A. (1993). Hierarchical Mixtures of Experts and the EM Algorithm. *Neural Computation*, 6, 181–214.
- Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., & Hirukawa, H. (2003). Biped walking pattern generation by using preview control of zero-moment point. *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, 2, 1620–1626 vol.2. <https://doi.org/10.1109/ROBOT.2003.1241826>
- Kajita, S., & Tani, K. (1991). Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 1405–1411.
- Kalman, R. E. (1960). Contributions to the theory of optimal control. *Boletin de la Sociedad Matemática Mexicana*, 5, 102–119.
- Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4), 373–395.
- Kavraki, L. E., Svestka, P., Latombe, J.-C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, 12(4), 566–580.

- Kingston, Z., Moll, M., & Kavraki, L. E. (2018). Sampling-based methods for motion planning with constraints. *Annual review of control, robotics, and autonomous systems*, 1(1), 159–185.
- Kobilarov, M. (2012). Cross-entropy motion planning. *The International Journal of Robotics Research*, 31(7), 855–871.
- Kuffner, J. J., & LaValle, S. M. (2000). RRT-connect: An efficient approach to single-query path planning. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2, 995–1001.
- Kumar, A., Fu, Z., Pathak, D., & Malik, J. (2021). Rma: rapid motor adaptation for legged robots. *Robotics: Science and Systems*.
- LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., & Huang, F. (2006). A tutorial on energy-based learning. *Predicting structured data*, 1(0).
- Lee, J., Hwangbo, J., & Hutter, M. (2019). Robust recovery controller for a quadrupedal robot using deep reinforcement learning. *arXiv preprint arXiv:1901.07517*.
- Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., & Hutter, M. (2020). Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47).
- Lembono, T. S., Mastalli, C., Fernbach, P., Mansard, N., & Calinon, S. (2020). Learning how to walk: warm-starting optimal control solver with memory of motion. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 1357–1363.
- Lembono, T. S., Paolillo, A., Pignat, E., & Calinon, S. (2020). Memory of Motion for Warm-Starting Trajectory Optimization. *IEEE Robotics and Automation Letters (RA-L)*, 5(2), 2594–2601.
- Li, J., Lu, L., Zhao, L., Wang, C., & Li, J. (2021). An integrated approach for robotic Sit-To-Stand assistance: Control framework design and human intention recognition. *Control Engineering Practice*, 107, 104680.
- Li, W., & Todorov, E. (2004). Iterative linear quadratic regulator design for nonlinear biological movement systems. *Proceedings of the 1st International Conference on Informatics in Control, Automation and Robotics*, 222–229.
- Li, Y., Chi, X., Razmjoo, A., & Calinon, S. (2024). Configuration space distance fields for manipulation planning. *Proc. Robotics: Science and Systems (RSS)*.
- Li, Y., Zhang, Y., Razmjoo, A., & Calinon, S. (2024). Representing Robot Geometry as Distance Fields: Applications to Whole-body Manipulation. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 15351–15357.

- Liang, J., Makoviychuk, V., Handa, A., Chentanez, N., Macklin, M., & Fox, D. (2018). Gpu-accelerated robotic simulation for distributed reinforcement learning. *Conference on Robot Learning*, 270–282.
- Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., & Le, M. (2023). Flow Matching for Generative Modeling. *The Eleventh International Conference on Learning Representations*.
- Liu, C., & Atkeson, C. G. (2009). Standing balance control using a trajectory library. *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 3031–3036.
- Liu, N., Li, S., Du, Y., Torralba, A., & Tenenbaum, J. B. (2022). Compositional visual generation with composable diffusion models. *European Conference on Computer Vision*, 423–439.
- Liu, Z., Bahety, A., & Song, S. (2023). Reflect: summarizing robot experiences for failure explanation and correction. *arXiv preprint arXiv:2306.15724*.
- Losey, D. P., Jeon, H. J., Li, M., Srinivasan, K., Mandlekar, A., Garg, A., Bohg, J., & Sadigh, D. (2022). Learning latent actions to control assistive robots. *Autonomous robots*, 46(1), 115–147.
- Ma, X., Yan, X., Smith, E., Zhu, Y., Tenenbaum, J. B., Wu, J., & Zhao, T. (2021). Learning latent graph dynamics for visual manipulation of deformable objects. *Conference on Robot Learning (CoRL)*, 1348–1358.
- Manchester, Z., & Kuindersma, S. (2020). Variational contact-implicit trajectory optimization. *Robotics Research: The 18th International Symposium ISRR*, 985–1000.
- Mansard, N., Delprete, A., Geisert, M., Tonneau, S., & Stasse, O. (2018). Using a Memory of Motion to Efficiently Warm-Start a Nonlinear Predictive Controller. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2986–2993.
- Mason, M. T. (1986). Mechanics and Planning of Manipulator Pushing Operations. *Intl Journal of Robotics Research*, 5(3), 53–71.
- Mirabel, J., Tonneau, S., Fernbach, P., Seppälä, A.-K., Campana, M., Mansard, N., & Lamiriaux, F. (2016). Density estimation using Real NVP. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.
- Mombaur, K., Truong, A., & Laumond, J.-P. (2010). From human to humanoid locomotion: an inverse optimal control approach. *Autonomous Robots*, 28(3), 369–383.
- Mühlbauer, M., Hulin, T., Weber, B., Calinon, S., Stulp, F., Albu-Schäffer, A., & Silvério, J. (2024). A Probabilistic Approach to Multi-Modal Adaptive Virtual Fixtures. *IEEE Robotics and Automation Letters (RA-L)*, 9(6), 5298–5305.

- Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., & Abbeel, P. (2018). Overcoming exploration in reinforcement learning with demonstrations. *International Conference on Robotics and Automation (ICRA)*, 6292–6299.
- Nesterov, Y., & Nemirovskii, A. (1994). *Interior-point polynomial algorithms in convex programming*. SIAM.
- Niekum, S., Chitta, S., Barto, A. G., Marthi, B., & Osentoski, S. (2013). Incremental semantically grounded learning from demonstration. *Robotics: Science and Systems*, 9, 10–15607.
- Nocedal, J., & Wright, S. J. (2006). *Numerical optimization*. Springer.
- Orin, D. E., Goswami, A., & Lee, S.-H. (2013). Centroidal dynamics of a humanoid robot. *Autonomous robots*, 35, 161–176.
- Orthey, A., Chamzas, C., & Kavraki, L. E. (2023). Sampling-Based Motion Planning: A Comparative Review. *Annual Review of Control, Robotics, and Autonomous Systems*, 7.
- Osa, T., Pajarinen, J., Neumann, G., Bagnell, J. A., Abbeel, P., & Peters, J. (2018). An algorithmic perspective on imitation learning. *Found. Trends Robotics*, 7, 1–179.
- Oseledets, I. V. (2011). Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5), 2295–2317.
- Pan, C., Yi, Z., Shi, G., & Qu, G. (2024). Model-based Diffusion for Trajectory Optimization. *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- Pang, T., Suh, H. T., Yang, L., & Tedrake, R. (2023). Global planning for contact-rich manipulation via local smoothing of quasi-dynamic contact models. *IEEE Transactions on Robotics*.
- Papadimitriou, C. H. (1981). On the complexity of integer programming. *Journal of the ACM*, 28(4), 765–768.
- Paraschos, A., Daniel, C., Peters, J. R., & Neumann, G. (2013). Probabilistic movement primitives. In C. J. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 26). Curran Associates, Inc.
- Pardo, D., Neunert, M., Winkler, A. W., & Buchli, J. (2015). Projection based whole body motion planning for legged robots. *ArXiv, abs/1510.01625*.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

- Pezzato, C., Salmi, C., Spahn, M., Trevisan, E., Alonso-Mora, J., & Corbato, C. H. (2023). Sampling-based model predictive control leveraging parallelizable physics simulations. *arXiv preprint arXiv:2307.09105*.
- Phiquepal, C., & Toussaint, M. (2023). Control-tree optimization: an approach to MPC under discrete partial observability. *Proceedings of the 20th Robotics: Science and Systems Conference (RSS)*. <https://doi.org/10.15607/RSS.2023.XIX.089>
- Pignat, E., Silvério, J., & Calinon, S. (2022). Learning from demonstration using products of experts: applications to manipulation and task prioritization. *International Journal of Robotics Research (IJRR)*, 41(2), 163–188. <https://doi.org/10.1177/02783649211040561>
- Posa, M., Cantu, C., & Tedrake, R. (2014). A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1), 69–81.
- Posa, M., Kuindersma, S., & Tedrake, R. (2016). Optimization and stabilization of trajectories for constrained dynamical systems. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 1366–1373.
- Power, T., & Berenson, D. (2023). Variational Inference MPC using Normalizing Flows and Out-of-Distribution Projection. *Robotics science and systems*.
- Pradalier, C., Colas, F., & Bessiere, P. (2003). Expressing bayesian fusion as a product of distributions: Applications in robotics. *IROS*, 2, 1851–1856.
- Qin, M., Brawer, J., & Scassellati, B. (2022). Task-oriented robot-to-human handovers in collaborative tool-use tasks. *31st IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 1327–1333.
- Rasouli, S., Gupta, G., Nilsen, E., & Dautenhahn, K. (2022). Potential applications of social robots in robot-assisted interventions for social anxiety. *International Journal of Social Robotics*, 14(5), 1–32.
- Ratliff, N. D., Issac, J., Kappler, D., Birchfield, S., & Fox, D. (2018). Riemannian motion policies. *arXiv preprint arXiv:1801.02854*.
- Razmjoo, A., Lombono, T. S., & Calinon, S. (2021). Optimal control combining emulation and imitation to acquire physical assistance skills. *20th International Conference on Advanced Robotics (ICAR)*, 338–343.
- Rodriguez, A. (2021). The unstable queen: uncertainty, mechanics, and tactile feedback. *Science Robotics*, 6(54), eabi4667.
- Rozo, L., Calinon, S., Caldwell, D. G., Jimenez, P., & Torras, C. (2016). Learning physical collaborative robot behaviors from human demonstrations. *Trans. on Robotot*, 32(3), 513–527.

- Sacks, J., & Boots, B. (2023). Learning Sampling Distributions for Model Predictive Control. *Conference on Robot Learning*, 1733–1742.
- Savostyanov, D., & Oseledets, I. (2011). Fast adaptive interpolation of multi-dimensional arrays in tensor train format. *The 2011 International Workshop on Multidimensional (nD) Systems*, 1–8.
- Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61, 85–117.
- Scipy 1.0: fundamental algorithms for scientific computing in python. (2020). *Nature Methods*, 17, 261–272.
- Shetty, S., Lembono, T., Löw, T., & Calinon, S. (2024). Tensor Trains for Global Optimization Problems in Robotics. *International Journal of Robotics Research (IJRR)*.
- Shetty, S., Xue, T., & Calinon, S. (2024). Generalized policy iteration using tensor approximation for hybrid control. *Proc. Intl Conf. on Learning Representations (ICLR)*.
- Shomin, M., Forlizzi, J., & Hollis, R. (2015). Sit-to-stand assistance with a balancing mobile robot. *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 3795–3800.
- Sirintuna, D., Giammarino, A., & Ajoudani, A. (2022). Human-robot collaborative carrying of objects with unknown deformation characteristics. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 10681–10687.
- Song, J., Meng, C., & Ermon, S. (2020). Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.
- Stilman, M. (2010). Global manipulation planning in robot joint space with task constraints. *IEEE Transactions on Robotics*, 26(3), 576–584.
- Stolle, M., & Atkeson, C. G. (2006). Policies based on trajectory libraries. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 3344–3349.
- Suh, C., Um, T. T., Kim, B., Noh, H., Kim, M., & Park, F. C. (2011). Tangent space RRT: A randomized planning algorithm on constraint manifolds. *IEEE International Conference on Robotics and Automation*, 4968–4973.
- Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.
- Talbot, J., Nubert, J., Tuna, E., Cadena, C., Dümbgen, F., Tordesillas, J., Barfoot, T. D., & Hutter, M. (2024). Continuous-time state estimation methods in robotics: a survey. *arXiv preprint arXiv:2411.03951*.

- Tassa, Y., Erez, T., & Todorov, E. (2012). Synthesis and stabilization of complex behaviors through online trajectory optimization. *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 4906–4913.
- Tillmann, A. M., & Pfetsch, M. E. (2014). The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing. *Transactions on Information Theory*, 60(2), 1248–1259.
- Todorov, E., & Jordan, M. I. (2002). Optimal feedback control as a theory of motor coordination. *Nature neuroscience*, 5(11), 1226–1235.
- Tözeren, A. (1999). *Human body dynamics: classical mechanics and human movement*. Springer Science & Business Media.
- Triantafyllidis, E., Acero, F., Liu, Z., et al. (2023). Hybrid hierarchical learning for solving complex sequential tasks using the robotic manipulation network roman. *Nature Machine Intelligence*, 5, 991–1005. <https://doi.org/10.1038/s42256-023-00709-2>
- Usvyatsov, M., Ballester-Ripoll, R., & Schindler, K. (2022). tntorch: Tensor Network Learning with PyTorch.
- Vulin, N., Christen, S., Stevšić, S., & Hilliges, O. (2021). Improved learning of robot manipulation tasks via tactile intrinsic motivation. *Robotics and Automation Letters*, 6(2), 2194–2201.
- Wagener, N., Cheng, C., Sacks, J., & Boots, B. (2019). An Online Learning Approach to Model Predictive Control. *Proceedings of Robotics: Science and Systems (RSS)*.
- Wang, L., Zhao, J., Du, Y., Adelson, E. H., & Tedrake, R. (2024). Poco: Policy composition from and for heterogeneous robot learning. *Proc. Robotics: Science and Systems (RSS)*.
- Whiten, A., McGuigan, N., Marshall-Pescini, S., & Hopper, L. M. (2009). Emulation, imitation, over-imitation and the scope of culture for child and chimpanzee. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1528), 2417–2428.
- Wieber, P.-B., Tedrake, R., & Kuindersma, S. (2016). Modeling and control of legged robots. In *Springer handbook of robotics* (pp. 1203–1234). Springer.
- Williams, G., Aldrich, A., & Theodorou, E. A. (2017). Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40(2), 344–357.
- Williams, G., Wagener, N., Goldfain, B., Drews, P., Rehg, J. M., Boots, B., & Theodorou, E. A. (2017). Information theoretic MPC for model-based reinforcement learning. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 1714–1721.
- Xie, W., Valentini, M., Lavering, J., & Correll, N. (2024). Deligrasp: inferring object properties with LLMs for adaptive grasp policies. *8th Annual Conference on Robot Learning*.

- Xue, H., Pan, C., Yi, Z., Qu, G., & Shi, G. (2024). Full-Order Sampling-Based MPC for Torque-Level Locomotion Control via Diffusion-Style Annealing.
- Xue, T., Girgin, H., Lembono, T., & Calinon, S. (2023). Demonstration-guided Optimal Control for Long-term Non-prehensile Planar Manipulation. *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 4999–5005.
- Xue, T., Razmjoo, A., Shetty, S., & Calinon, S. (2024a). Robust Manipulation Primitive Learning via Domain Contraction. *Proc. Conference on Robot Learning (CoRL)*.
- Xue, T., Razmjoo, A., Shetty, S., & Calinon, S. (2025). Robust contact-rich manipulation through implicit motor adaptation. *International Journal of Robotics Research (IJRR)*.
- Xue, T., Razmjoo, A., Shetty, S., & Calinon, S. (2024b). Logic-Skill Programming: An Optimization-based Approach to Sequential Skill Planning. *Proc. Robotics: Science and Systems (RSS)*.
- Yakey, J. H., LaValle, S. M., & Kavraki, L. E. (2001). Randomized path planning for linkages with closed kinematic chains. *IEEE Transactions on Robotics and Automation*, 17(6), 951–958.
- Yan, X., Zhu, Y., Kosaian, J., Tenenbaum, J. B., & Wu, J. (2020). Learning predictive representations for deformable objects using contrastive estimation. *Conference on Robot Learning (CoRL)*, 564–574.
- Yang, Z., Mao, J., Du, Y., Wu, J., Tenenbaum, J. B., Lozano-Pérez, T., & Kaelbling, L. P. (2023). Compositional Diffusion-Based Continuous Constraint Solvers. *Conference on Robot Learning*.
- Yu, W., Liu, C. K., & Turk, G. (2017). Preparing for the unknown: learning a universal policy with online system identification. *Proceedings of Robotics: Science and Systems*.
- Yunt, K., & Glocker, C. (2006). Trajectory optimization of mechanical hybrid systems using SUMT. *IEEE International Workshop on Advanced Motion Control*, 665–671.
- Zhang, F., & Gienger, M. (2024). Affordance-based robot manipulation with flow matching. *arXiv preprint arXiv:2409.01083*.
- Zhu, Y., Stone, P., & Zhu, Y. (2022). Bottom-up skill discovery from unsegmented demonstrations for long-horizon robot manipulation. *IEEE Robotics and Automation Letters*, 7(2), 4126–4133.

Amirreza Razmjoo Fard

Research Assistant, École Polytechnique Fédérale de Lausanne (EPFL)/Idiap Research Institute, Martigny, Switzerland
amirreza.razmjoo@gmail.com — +41 (76) 467-7413 — www.linkedin.com/in/amir-razmjoo — amirrazmjoo.github.io

RESEARCH INTERESTS

Contact-rich Manipulation, Generative Models for Motion Planning, Sampling-based motion planning, Task and Motion Planning,

EDUCATION

École Polytechnique Fédérale de Lausanne (EPFL), Jan. 2021 — Nov. 2025
PhD, EDEE program Lausanne, Switzerland

Thesis Topic: Feasibility Guided Exploration and Adaptation for Constrained Robotic Manipulation

Keywords: Contact-rich Manipulation, Sampling-based MPC, Products of Experts, Generative Models

Supervisor: Dr. Sylvain Calinon, Prof. Auke Ijspeert

Sharif University of Technology, Sep. 2018 — Sep. 2020
MSc in Mechanical Engineering Tehran, Iran

GPA: 18.85/20 (4.0/4.0)

Thesis Title: Teaching to Point at Different Objects as an Interactive Gesture to a Robot by Learning from Demonstration

Supervisors: Prof. Ali Meghdari, Dr. Alireza Taheri

University of Tehran (with honors), Sep. 2014 — Sep. 2018
BSc in Mechanical Engineering Tehran, Iran

GPA: 18.83/20 (3.95/4.0)

Thesis Title: Development of a Health-Monitoring Device for Activity Recognition and Fall Detection

Supervisors: Dr. Sadighi, Dr. Zakerzadeh

EXPERIENCE

Honda Research Institute GmbH, Offenbach, Germany
Research Intern Oct. 2024 – Mar. 2025

- Developed **Composition of Conditional Diffusion Policies (CCDP)**, a diffusion-policy based algorithm that guides sampling during the denoising process. This approach enabled robots to avoid previously failed samples and increased success rates by an average of **44%** (range: 23–84% depending on the task), without requiring additional classification, skill segmentation, or high-level planning. Applied and validated on tasks such as *door opening*, *object manipulation*, and *object packing*, etc.

Idiap Research Institute, Martigny, Switzerland
Research Assistant Jan. 2021 – Present

- Developed **TT-PoE-MPC**, a sampling-based model predictive control algorithm that combines feasibility distributions with MPPI/DIAL-MPC (optimality-focused approaches). This reduced the learning burden (only feasibility needs to be learned, optimality is computed online) and improved success rates by **20% on average** (range: 5–35%), while also reducing cost and planning steps. Applied to tasks such as *non-prehensile manipulation*, *obstacle avoidance*, and *manifold navigation*.
- Designed novel **task representations** to enhance optimal control methods, including: – *Configuration Space Distance Fields*, which linearize planning in configuration space and accelerate convergence. – *Robust Manifolds*, enabling robots to withstand external disturbances (e.g., humanoid resisted ~200N user-applied force for 2–8s without losing stability). – Kinematic-aware signed distance fields (SDFs) and stiffness manifolds for constraint-aware control.
- Collaborated on **robust policy learning**, leveraging Tensor-Train decomposition for direct policy retrieval from weighted advantage functions. This reduced training time by ~**80%** and enabled faster, safer deployment.
- Contributed to **task and motion planning** research, combining high-level symbolic reasoning with low-level control, including: – Dynamic Logic-Geometric Programming (D-LGP). – Graphs of tensor networks for long-horizon planning. – Learning symbolic and continuous planning from demonstrations.

CEDRA (Center of Excellence in Design, Robotics, and Automation)

Research Assistant

Tehran, Iran

Sep. 2018 – Sep. 2020

- Demonstrated how to combine features with non-linear relationships across coordinate systems using **Products of Experts**.
- Developed a **ROS-based control system** for two social robots (Arash2 & Armin), including a learning-from-demonstration module, GUI integration, and navigation system support in collaboration with a front-end developer.

AWARDS**Outstanding Paper Award at TARo Workshop at IROS**

2025

For the paper *CCDP: Model-Free Failure Recovery with Guided Diffusion Sampling***Outstanding Paper Finalist at RSS**

2024

For the paper *Configuration Space Distance Fields for Manipulation Planning***Summa cum laude**

2018

Achieved the top rank among 121 students upon graduating from the University of Tehran.

Admission Offer

2018

Excellent student, M.Sc. admission offer to Sharif University of Technology.

Dean's list

2015-2018

Ranked among the top 10 individuals of the year in that particular program.

F.O.E prize

2015, 2017

Recipient of the award given to the top three students of the year.

Ranked 256 (top 0.2%)

2014

Among more than 220,000 people in the nationwide university entrance exam.

SKILLS

- **Robot Control:** Diffusion-Policy, Flow-Matching Policy, Imitation Learning, MPPI, DIAL-MPC, iLQR
- **Research Topics:** Product of Experts, Generative Models, Optimal Control, Learning from Demonstrations, (physical-) Human-Robot Interaction, Contact-rich Manipulation, Impedance/Admittance Control,
- **Programming:** Python, MATLAB, ROS, L^AT_EX, Linux, C++ (Basics)
- **Softwares (libraries):** Mujoco, Pytorch, IsaacGym, Pybullet, Pinocchio, Crocoddyl, CasADi, Genesis
- **Languages:** English (Professional Working Proficiency), Azerbaijani (Native), Persian (Native)

REFERENCES**Dr. Sylvain Calinon**

Senior Researcher at Idiap Research Institute, Head of RLI Group

E-mail: sylvain.calinon@idiap.ch

Telephone: (office) +41 27 721 77 61

Dr. Michael Gienger

Chief Scientist at Honda Research Institute

E-mail: Michael.Gienger@honda-ri.de

Dr. T. S. Lembono

Applied Scientist at Amazon

E-mail: tlembono@amazon.de

Teng Xue

Research Assistant at the Idiap Research Institute

E-mail: teng.xue@idiap.ch

SELECTED PUBLICATIONS

- Y. Li, N. Darwiche, A. Razmjoo, S. Liu, Y. Du, A. Ijspeert, and S. Calinon, **Geometry-aware Policy Imitation**, under review, 2025
- T. Xue, Y. Zhang, A. Razmjoo, and S. Calinon, **Monte Carlo Tree Search with Tensor Factorization for Robot Optimization**, under review, 2025

- A. Razmjoo, T.Xue, S. Shetty and S. Calinon, **Sampling-Based Constrained Motion Planning with Products of Experts**, IJRR, 2025
- Y. Zhang, T. Xue, A. Razmjoo, and S. Calinon, **Learning Problem Decomposition for Efficient Sequential Multi-Object Manipulation Planning**, RA-L, 2025.
- A. Razmjoo, S. Calinon, M. Gienger, and F. Zhang, **CCDP: Composition of Conditional Diffusion Policies with Guided Sampling**, IROS, 2025.
- T.Xue, A. Razmjoo, S. Shetty and S. Calinon, **Robust Contact-rich Manipulation through Implicit Motor Adaptation**, IJRR, 2025
- O. Beker, N. Gürtler, J. Shi, A. Geist, A. Razmjoo, G. Martius, and S. Calinon, **A Smooth Analytical Formulation of Collision Detection and Rigid Body Dynamics With Contact**, IROS, 2025.
- T. Xue, A. Razmjoo, and S. Calinon, **Robust Manipulation Primitive Learning via Domain Contraction**, CoRL, 2024.
- Y. Zhang, A. Razmjoo, and S. Calinon, **Learn2Decompose: Learning Problem Decomposition for Efficient Task and Motion Planning**, RA-L, 2024.
- T. Xue, A. Razmjoo, S. Shetty, and S. Calinon, **Logic-Geometric learning and Control Using Graph of Tensor Networks**, *RSS-Workshop*, 2024.
- Y. Zhang, T. Xue, A. Razmjoo, **Logic Dynamic Movement Primitives for Long-horizon Manipulation Tasks in Dynamic Environments**, *RA-L*, 2024
- T.Xue, A. Razmjoo, S. Shetty and S. Calinon, **Logic-Skill Programming: An Optimization-based Approach to Sequential Skill Planning**, *RSS*, 2024
- Y.Li, X. Chi, A. Razmjoo, S. Calinon, **Configuration Space Distance Fields for Manipulation Planning**, *RSS*, 2024, (**Best Paper Finalist**)
- T. Xue, A. Razmjoo, and S. Calinon, **D-LGP: Dynamic Logic-Geometric Program for Combined Task and Motion Planning**, *ICRA*, 2024.
- Y. Li, Y. Zhang, A. Razmjoo, and S. Calinon. **Learning Robot Geometry as Distance Fields: Applications to Whole-body Manipulation.**, *ICRA*, 2024.
- A. Razmjoo, T. Brecelj, K. Savevska, A. Ude, T. Petrič, and S. Calinon, **Optimal Latent Manifold for Reliable Physical Interaction: A Sit-to-Stand Assistance Application**, 6th Ergonomic Physical Human-Robot Collaboration: Harnessing Advancements in Robot Learning workshop, *IROS*, 2023.
- A. Razmjoo, T. Brecelj, K. Savevska, A. Ude, T. Petric, and S. Calinon, **Learning Joint Space Reference Manifold for Reliable Physical Assistance**, *IROS*, 2023.
- B. Ti, A. Razmjoo, Y. Gao, J. Zhao, S. Calinon, **A Geometric Optimal Control Approach for Imitation and Generalization of Manipulation Skills**, *Robotics and Autonomous Systems (RAS)*, 2023.
- A. Razmjoo, T. S. Lembono, S. Calinon, **Optimal Control Combining Emulation and Imitation to Acquire Physical Assistance Skills**, *ICAR*, 2021.

Ce document a été imprimé au Centre d'impression de l'EPFL, certifié myClimate depuis 2017, sur papier Nautilus® SuperWhite, 100% recyclé et issu de déchet de consommation, certification FSC™ et Blue Angel.



repro.epfl.ch