Doctor's Thesis

Task-relevant Model-based Reinforcement Learning for Contact-rich Robotic Tasks

Cheng-Yu Kuo

September 14, 2023

Program of Information Science and Engineering
Graduate School of Science and Technology
Nara Institute of Science and Technology

A Doctor's Thesis
submitted to Graduate School of Science and Technology,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Cheng-Yu Kuo

Thesis Committee:

Professor Takamitsu MATSUBARA	(Supervisor)
Professor Takahiro WADA	(Co-supervisor)
Assistant Professor Hikaru SASAKI	(Co-supervisor)

Task-relevant Model-based Reinforcement Learning for Contact-rich Robotic Tasks *

Cheng-Yu Kuo

Abstract

This thesis presents "task-relevant model-based reinforcement learning" to learn complex robot dynamics and perform tasks in contact-rich environments. Model-based Reinforcement Learning (MBRL) offers a promising solution for capturing complex robot dynamics that pose challenges for analytic solutions attempting to capture them accurately. However, controlling the robot with an MBRL-learned dynamics model is limited to the available entries of the robot state, which may not be sufficient for completing the intended task. To enhance standard MBRL's ability, we present the task-relevant MBRL, that reformulates the robot state to 1) satisfy the requirements of the intended task, 2) be adequate for dynamics learning with MBRL, and 3) be able to adjust the motion behavior while performing the task. Our method is verified through two contact-rich applications, including improving contact-safety during the process of learning common kitchen tasks and achieving walking acquisition with a compliant bipedal robot. Our method's success in both setups demonstrates its effectiveness and applicability to various robotics applications that require flexibility, robustness, and safety in uncertain and contact-rich environments.

Keywords:

Model-based Reinforcement Learning, Dynamics Learning for Control, Contact-rich Applications, Bipedal Locomotion, Compliant Robot

*Doctor's Thesis, Department of Information Science, Graduate School of Information Science and Technology, Nara Institute of Science and Technology, September 14, 2023.

Contents

List of Figures

1. Introduction

1.1. Modern Robots and Their Challenges

Recent years have seen remarkable advancements in the field of robotics. The applications of robotics have expanded beyond highly controlled settings, such as manufacturing factories, to more uncertain and dynamic environments, such as open or contact-rich environments. This expansion is due to the integration of modern robotic systems with cutting-edge technologies, especially allowing for the introduction of elastic components that make these robots function similarly to animal anatomy and physiology [1–4]. These elastic components have significantly enhanced the impact tolerance of robots, allowing them to operate more flexible and efficiently, while also contributing to their agility and operating safety.

However, introducing these elastic components significantly increases the complexity of the robot's dynamics. This makes it challenging to accurately capture an analytical dynamics model of the robot. These modeling errors can cause imprecise control, such as unexpected oscillations and positioning errors.

1.2. Compliant Robots for Contact-rich Tasks

Compliant robots, with their high tolerance for impacts, are best suited for contact-rich environments that require robots to interact with dynamic and unpredictable surroundings. Such environments pose challenges due to uncertain sensory feedback, unpredictable contacts, and variable environmental conditions. When performing contact-rich tasks, the robot's dynamics are largely affected, as in object manipulation tasks [5–7], or even fully reliant, as in legged locomotion [8, 9], on physical interactions with the environment. Despite the challenges in analytically capturing the dynamics of a compliant robot, these effects make it

even more difficult. Consequently, researchers are shifting their focus to learning-based approaches to tackle these complex robotic problems.

1.3. Learning-based Methods for Robot Control

Learning-based methods for robot control have shown significant promise in addressing the challenges posed by modern robots and contact-rich tasks [10, 11]. These challenges include handling external uncertainties such as environmental changes [12] , as well as internal uncertainties like sensing errors [13, 14]. These methods employ machine learning algorithms to learn control policies (Model-free Reinforcement Learning [15, 16], MFRL), controller parameters (Bayesian Optimization [17, 18], BO), or dynamics learning for control (Model-based Reinforcement Learning [19, 20], MBRL). Studies with these methods have allows modern robots to operate effectively in uncertain and dynamic environments.

Although MFRL and BO have yielded impressive results in robotic applications, they are considered black-box approaches that are task-specific and training-sample-intensive. They require not only enormous training samples but also re-training from scratch for different robot-tasks. Therefore, many studies rely on virtual scaling approaches such as Sim-to-Real for efficient learning of the intended task [13, 21–23]. However, these approaches lengthen the learning process, making it difficult to expand to different robot systems or tasks, limit the ability of onsite learning, and have difficulty in handling unexpected situations that were not covered in the training process.

1.4. Model-based Reinforcement Learning

1.4.1. The Strength of Model-based Reinforcement Learning

On the other hand, Model-based Reinforcement Learning (MBRL) [24, 25] offers an attractive solution for robots to autonomously learn their dynamics and perform tasks through control planning methods that use the learned dynamics model, such as Model Predictive Control (MPC) [26–28]. Since MBRL learns

the robot's state-transition dynamics instead of the robot-task, it is more generalized over tasks and much more robust to environmental changes when using online planning with newly observed states during operation. Moreover, MBRL has demonstrated impressive sample efficiency with a probabilistic model (probabilistic MBRL) [29], which is suitable for learning robot dynamics onsite, where hardware sample collection is both demanding and time-consuming.

1.4.2. The Limitation of Model-based Reinforcement Learning

Although MBRL is sample-efficient, more generalizable, and more robust to environmental changes, it has significant limitations. The optimization objective of MPC has to be formulated based on the available entries of the robot state. In other words, MPC cannot perform the intended task if the corresponding entries are not available in the pre-defined robot state. One solution is to accommodate all possible entries in a robot state. However, the computation cost of MPC largely depends on the model's dimensionality, requiring a compact dynamics model when high control frequency is anticipated.

With the above in mind, a robot state with the following features is considered necessary for MBRL:

1. The robot state should be compact enough to achieve the desired control frequency.
2. The entries in the robot state should be sufficient for performing the intended task.
3. The entries in the robot state should be adequate for MBRL to learn the state-transition dynamics of the robot system

1.5. The Task-relevant Model-based Reinforcement Learning

This thesis presents "task-relevant model-based reinforcement learning," which uses a reformulated task-relevant robot state that fulfills the above-mentioned

features. Specifically, we reformulate a task-relevant robot state that contains relevant entries for the intended robotic task. The task-relevant robot state is transformed from the standard robot state to ensure that the dynamics learned through MBRL are sufficient to express the underlying dynamics of the robotic system. This is essential because MPC relies on the prediction of the learned dynamics, and deficient state can lead to severe modeling and prediction errors.

By utilizing the task-relevant state, MPC can solve the optimization problem and find the control relative for achieving the intended task based on a state-formulated objective function. Additionally, we establish a task-relevant control space for MPC exploration, allowing us to adjust the characteristics of the robot's motion for secondary objectives.

In summary, task-relevant MBRL consists of three parts (see Fig. 1.1):

1. *Task-relevant Robot State* that adequately expresses the underlying dynamics of the robotic system.

2. *Task-relevant Task Objective* that is formulated based on the task-relevant robot state, which is used by MPC to solve the optimization problem.

3. *Task-relevant control space* that enables the user to adjust the characteristics of the robot's motion.

1.6. Performance Verification

The effectiveness of task-relevant MBRL is verified through two distinct applications with different robot systems:

1. Extending the robot state to improve contact-safety during the process of learning kitchen tasks with compliant robot arms.

2. Condensing the robot state to achieve compliant bipedal robot walking with real-time planning and onsite learning capability.

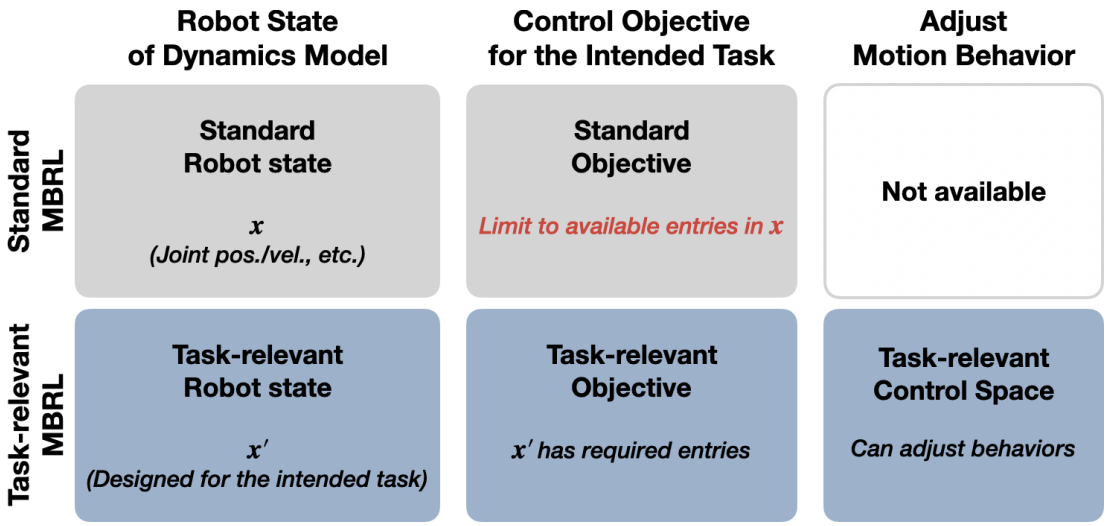

Figure 1.1. This figure explains the difference between standard MBRL and the presented task-relevant MBRL.

1.6.1. Application 1:Extending Robot-state for Improving Contact-safety During Learning Process

In our first application, we utilize the task-relevant MBRL scheme to develop a contact-safe MBRL for robots operating in contact-rich environments. Our method enables safe contact behavior during the learning process, which is typically challenging in MBRL due to sample scarcity in the early stages of learning.

During the early stages of learning, we cannot rely on the inaccurate data-driven dynamics model to generate reliable controls for the intended task. Although operating with these unreliable controls is considered necessary exploration, they can cause damage to the robot and its surroundings, especially in contact-rich environments.

To mitigate this risk, we utilize the task-relevant MBRL as follows:

1. *Task-relevant Robot State*: We reformulated an extended robot state that accommodates model-uncertainty such that the accuracy of the learned model becomes accessible for MPC planning.

2. *Task-relevant Task Objective*: A standard objective that encourage the intended contact-rich task completion.

3. *Task-relevant control space*: We utilize the uncertainty information in the robot state to establish an uncertainty-aware control space that encourage cautious behavior when the learned model is inaccurate.

This reformulation allows us to correlate the robot's behavior with the learning process to improve operating safety.

We verified the effectiveness of our method by conducting kitchen tasks that exemplify contact-rich environments. These tasks included a particle mixing task with both simulated and hardware robots, as well as a particle scooping task with a hardware robot. Our results showed that our method required a less intensive learning process with lower measured contact forces, providing evidence of its effectiveness.

1.6.2. Application 2: Condensing Robot-state to Achieve Compliant Bipedal Robot Walking

In our second application, we utilize the task-relevant MBRL scheme to achieve walking with a compliant bipedal robot. The deployment of compliant bipedal robots in human-centric environments holds great potential. However, their complex dynamics pose challenges to analytical approaches. Therefore, MBRL is a promising solution to learn a fully data-driven dynamics model to capture such high complexity dynamics.

While MBRL is capable of learning a data-driven dynamics model to express high complexity dynamics, bipedal walking tasks require a high control frequency to secure stability. However, the computation expense of MBRL is highly dependent on dynamics dimensionality. A higher dimension dynamics model is needed to accommodate the compliance dynamics of a compliant bipedal robot.

To address this issue, we utilize the task-relevant MBRL scheme with the law of conservation of energy to reformulate a condensed robot state for a spring-loaded bipedal robot. Specifically, we the task-relevant MBRL for compliant bipedal robot walking is formulated as follows:

1. *Task-relevant Robot State*: We reformulated an energy-state that expresses the robot's Center-of-Mass (CoM) dynamics with potential and kinetic en-

ergies. We view the installed springs as temporary energy containers and treat all actuators as energy sources.

2. *Task-relevant Task Objective*: We designed a reference walking trajectory for MPC planning based on the energy entries of the energy-state.

3. *Task-relevant control space*: We utilize both energy-state and the reference walking trajectory for establish an energy-state-aware control space that enhance MPC planning reliability.

This reformulation allows us to significantly reduce the dynamics dimensionality and achieve the high control frequency required for the bipedal walking task.

We verified the effectiveness of our method by learning a walking task with simulated and hardware planar spring-loaded bipedal robot. In simulation, we performed walking on uneven terrains and walking at various walking speeds, including fixed and alternating speeds. For the hardware experiment, we performed walking at a fixed speed to demonstrate the real-world capability of our method. All results showed successful on-site walking acquisition with a compact nine-dimension dynamics model, 40Hz real-time planning, and on-site learning within a few minutes.

1.7. Summary

This thesis presents "task-relevant model-based reinforcement learning" for learning robot dynamics and performing tasks through MPC. The method reformulates a task-relevant robot state that contains sufficient entries for the intended task, and is suitable for MBRL to learn the state-transition dynamics of the robot system. By utilizing the task-relevant state, MPC can solve the optimization problem and find the control relative for achieving the intended task based on a state-formulated objective function. The effectiveness of task-relevant MBRL is verified through two distinct contact-rich applications (see Fig. 1.2):

1. Extending robot state to enhance operating safety during the learning process.

2. Condensing robot state to achieve learning and performing walking skills with compliant bipedal robot.

The success in both applications show the effectiveness and applicability of our method. The presented "task-relevant model-based reinforcement learning" can potentially be applied to various robotics applications that require high flexibility, robustness, and safety in uncertain and contact-rich environments.

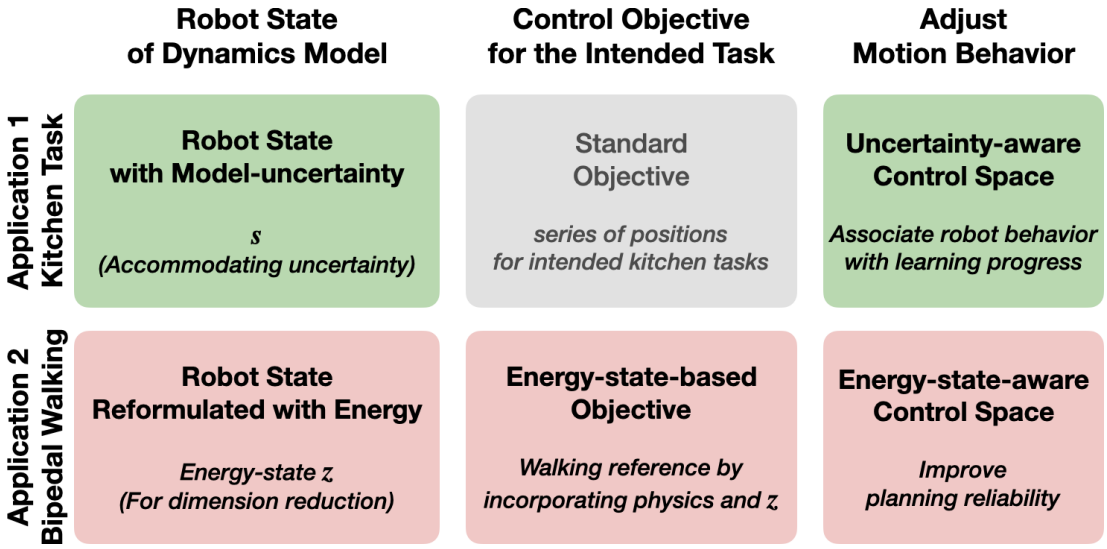

Figure 1.2. This figure illustrates how the task-relevant MBRL is employed in both applications presented in this thesis.

1.8. Thesis Structure

The contents of this thesis are structured as follows:

- Chapter 2 provides the preliminary information.
- Chapter 3 details the first application, which explains how task-relevant MBRL was utilized to improve contact-safety during the learning process of kitchen tasks.
- Chapter 4 details the second application, which explains how task-relevant MBRL was utilized to achieve walking with a spring-loaded bipedal robot.

- Chapter 5 raises some open issues that require further study and investigation.
- Finally, Chapter 6 presents the conclusion to this thesis.

2. Preliminaries

The Model-based Reinforcement Learning (MBRL) with Gaussian Process (GPs) dynamics has demonstrated high sample-efficient [29, 30]. However, the control frequency of probabilistic Model Predictive Control (pMPC) [31] with a GPs dynamics model is inversely proportional to the squared training sample size, leading to a trade-off between having a higher control frequency or a more accurate dynamics model trained with larger amount of samples. This trade-off is insignificant if offline planning is applied, such as PILCO [29], or low online control frequency is acceptable [30]. Nevertheless, rapid responses are anticipated for handling the frequently-changing environmental uncertainties during contact-rich applications. Therefore, we implement the Fourier-featured Linear Gaussian Model (LGM-FF) [32] for dynamics modeling to approximate a standard GPs dynamics and alleviate the trade-off between sample size and control frequency.

This section introduced the basic knowledge of MBRL we implemented for our applications. Including how we model the LGM-FF dynamics; how we exploit probabilistic state predictions with the LGM-FF dynamics; how we utilized pMPC for online planning; and how the MBRL learning process is formulated.

2.1. Probabilistic Dynamics Acquisition

2.1.1. Nominal Dynamics Model for Robotic System

Consider a data-driven dynamics model $f(\cdot) : \mathbb{X} \times \mathbb{U} \to \mathbb{X}$ that represents a robotic system's true state-transition dynamics under a fixed time interval:

$$\boldsymbol{x}_{t+1} = f(\tilde{\boldsymbol{x}}_t) + \epsilon, \tag{2.1}$$

where $\tilde{\boldsymbol{x}}_t := \left[\boldsymbol{x}_t^\top, \boldsymbol{u}_t^\top\right]^\top \in \mathbb{R}^{D+U}$ is the state-control vector with the robot's state $\boldsymbol{x} \in \mathbb{X} \subset \mathbb{R}^D$ and control signal $\boldsymbol{u} \in \mathbb{U} \subset \mathbb{R}^U$. Additionally, $\epsilon \sim \mathcal{N}(0, \boldsymbol{\Sigma}_n)$ is the system noise with $\boldsymbol{\Sigma}_n = \text{diag}\left[\sigma_{n-1}, ..., \sigma_{n-D}\right]$ corresponding to each dimension of the state. Given N collected state-control tuples from the robotic system, latent dynamics models $f := \{f_i\}_{\forall i=1,...,D}$ are trained independently for each state dimension with input $\tilde{\boldsymbol{X}} := [\tilde{\boldsymbol{x}}_1, ..., \tilde{\boldsymbol{x}}_N]$ and target $\boldsymbol{y} := \{\boldsymbol{y}_1, ..., \boldsymbol{y}_D\}$ with $\boldsymbol{y}_i := [\boldsymbol{x}_{1,i}, \boldsymbol{x}_{N,i}]$, where $\boldsymbol{x}_{N,i}$ denotes the i-th dimension of the N-th collected state $\boldsymbol{x}_N$.

2.1.2. Extracting Fourier Features from Covariance Kernel Expansions

Under the Bochner's theorem [33], Fourier transforming a GPs covariance kernel expansions of two input vectors $k(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}')$ will return a proper distribution. Therefore, we can approximate the kernel expansion by drawing all rows of $\boldsymbol{V} \in \mathbb{R}^{m \times D}$ i.i.d from the Fourier transformed covariance kernel $\rho(\omega)$ [34, 35]:

$$\begin{aligned} k(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}') &= \int \rho(\omega) \exp\left(i\omega^\top (\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}')\right) \mathrm{d}\omega \\ &\approx n^{-1} \exp\left(i\boldsymbol{V}(\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}')\right) \\ &= n^{-1} \cos\left(\boldsymbol{V}(\tilde{\boldsymbol{x}} - \boldsymbol{x}')\right) \\ &= n^{-1} \left(\cos(\boldsymbol{V}\tilde{\boldsymbol{x}}) \cos(\boldsymbol{V}\tilde{\boldsymbol{x}}') + \sin(\boldsymbol{V}\tilde{\boldsymbol{x}}) \sin(\boldsymbol{V}\tilde{\boldsymbol{x}}')\right) \\ &= \phi(\tilde{\boldsymbol{x}})^\top \phi(\tilde{\boldsymbol{x}}'), \end{aligned} \tag{2.2}$$

and we obtain the feature map $\phi(\tilde{\boldsymbol{x}}) : \mathbb{X} \times \mathbb{U} \to \mathbb{R}^{2m}$ as

$$\phi(\tilde{\boldsymbol{x}}) = n^{-\frac{1}{2}} \left[\cos(\boldsymbol{V}\tilde{\boldsymbol{x}})^\top, \sin(\boldsymbol{V}\tilde{\boldsymbol{x}})^\top\right]^\top. \tag{2.3}$$

In our following applications, a bias term is added to the feature map. Thus, our use of feature map is defined as

$$\Phi(\boldsymbol{x}) := \left[1, \phi(\tilde{\boldsymbol{x}})^\top\right]^\top : \mathbb{X} \times \mathbb{U} \to \mathbb{R}^M, \tag{2.4}$$

where $M = 2n + 1$ is the user-defined feature size.

2.1.3. Approximating Gaussian Process Dynamics via Fourier-featured Linear Gaussian Model

With the Fourier feature map from Eq. (2.3), we utilize the Fourier-featured Linear Gaussian Model (LGM-FF) [32] to approximate the standard GPs dynamics for modeling the state-transition dynamics model $f(\cdot)$. The for each state dimension $i = 1, ..., D$, the predictive distribution's mean μ_i and variance σ_i^2 given input $\tilde{\boldsymbol{x}}^\star$ is obtained via:

$$p\left(f_i\left(\tilde{\boldsymbol{x}}^\star\right)|\tilde{\boldsymbol{X}}, \boldsymbol{y}_i\right) \sim \mathcal{GP}\left(\mu_i\left(\tilde{\boldsymbol{x}}^\star\right), \sigma_i\left(\tilde{\boldsymbol{x}}^\star\right)\right) \tag{2.5}$$

$$\mu_i\left(\tilde{\boldsymbol{x}}^\star\right) = \boldsymbol{w}_i^\top \Phi_i\left(\tilde{\boldsymbol{x}}^\star\right) \in \mathbb{R} \tag{2.6}$$

$$\sigma_i^2\left(\tilde{\boldsymbol{x}}^\star\right) = \Phi_i\left(\tilde{\boldsymbol{x}}^\star\right)^\top \boldsymbol{A}_i^{-1} \Phi_i\left(\tilde{\boldsymbol{x}}^\star\right) \in \mathbb{R}, \tag{2.7}$$

where $\boldsymbol{w}$ is the corresponding weight, obtained through *Maximum a posteriori* [36, 37] estimation as:

$$\boldsymbol{w}_i = \sigma_{n,i}^{-2} \boldsymbol{A}_i^{-1} \Phi_i\left(\tilde{\boldsymbol{x}}^\star\right) \boldsymbol{y}_i \in \mathbb{R}^{M\times 1}, \tag{2.8}$$

$$\boldsymbol{A}_i = \sigma_{n,i}^{-2} \Phi_i\left(\tilde{\boldsymbol{x}}^\star\right) \Phi_i\left(\tilde{\boldsymbol{x}}^\star\right)^\top + \lambda_i \boldsymbol{I}_M \in \mathbb{R}^{M\times M}, \tag{2.9}$$

where the optimization variables $\sigma_{n,i}$ and λ_i are obtained via *Maximum Marginal Likelihood Estimation* [36, 38]. The feature size M is depends on the complexity of the system dynamics, where where a higher complexity state transition will require larger amount of features to fit. For finding the proper feature size, simply increase the value of M gradually until the LGM-FF accurately fits the sample.

By having N training samples, the computation cost of training an LGM-FF model is $\mathcal{O}(DNM^3)$. In comparison, training a standard GPs model is $\mathcal{O}(DN^3)$, in which the training time increases significantly as the sample size grows.

2.2. Finite-horizon Model Predictive Control with Uncertainty Propagation

Given all latent dynamics models predict and train independently, the subscription $i = 1, ..., D$ that indicates each prediction dimension is omitted in this section.

In addition, a probabilistic state is denoted as $p(\boldsymbol{x}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, with corresponding mean vector $\boldsymbol{\mu} = [\mu_1, ..., \mu_D]^\top$ and covariance matrix $\boldsymbol{\Sigma} = \text{diag}[\sigma_1^2, ..., \sigma_D^2]$.

2.2.1. State Predictions with Uncertainty Propagation by Exploiting Moment-matching Techniques

As the LGM-FF outputs a single-step predictive state as a probabilistic distribution with mean and variance, Eq. (2.5), multi-step state predictions requires uncertainty propagation of obtain. Following the *Fubini's Theorem* [39], the moment-matching technique propagates state uncertainties by integrating the dynamics model over the inputted state distribution and approximate the corresponding future states as a Gaussian distribution, as illustrated in Fig. 2.1 [29,40].

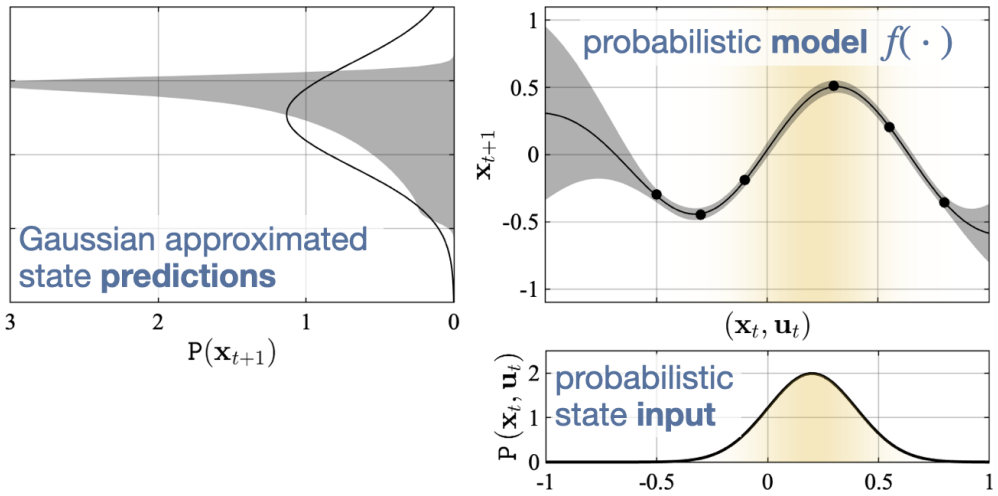

Figure 2.1. This figure illustrates the process of approximating a Gaussian distributed predictive state via exploiting moment-matching technique.

Specifically, given a probabilistic state $p(\boldsymbol{x}_t) \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ and a deterministic control signal $p(\boldsymbol{u}_t) \sim \mathcal{N}(\boldsymbol{u}_t, \text{diag}[\mathbf{0}_U])$ where $\mathbf{0}_U$ denotes a U-dimensional column vector of zeros, each latent dimension of the predictive state $p(\boldsymbol{x}_{t+1}) \sim \mathcal{N}(\boldsymbol{\mu}_{t+1}, \boldsymbol{\Sigma}_{t+1})$ is obtained by integrating the dimension-specific LGM-FF model

over the state distribution $p(\boldsymbol{x}_t)$:

$$\begin{aligned}
\mu_{t+1} &= \mu_{MM}\left(p\left(\boldsymbol{x}_t\right), \boldsymbol{u}_t\right) \\
&= \mathbb{E}_{\boldsymbol{x}_t}\left[f\left(\tilde{\boldsymbol{x}}_t\right)|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\right] = \int \mu\left(\tilde{\boldsymbol{x}}_t\right) p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t \\
&= \int \boldsymbol{w}^\top \Phi_t p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t = \boldsymbol{w}^\top \boldsymbol{q}_t \in \mathbb{R}, \\
\sigma_{t+1}^2 &= \sigma_{MM}^2\left(p\left(\boldsymbol{x}_t\right), \boldsymbol{u}_t\right) \\
&= \mathrm{var}_{\boldsymbol{x}_t, f}\left[f\left(\tilde{\boldsymbol{x}}_t\right)|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\right] \\
&= \mathbb{E}_{\boldsymbol{x}_t}\left[\sigma\left(\tilde{\boldsymbol{x}}_t\right)^2|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\right] + \mathbb{E}_{\boldsymbol{x}_t}\left[\mu\left(\tilde{\boldsymbol{x}}_t\right)^2|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\right] - \mathbb{E}_{\boldsymbol{x}_t}\left[\mu\left(\tilde{\boldsymbol{x}}_t\right)|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\right]^2 \\
&= \int \Phi\left(\tilde{\boldsymbol{x}}_t\right)^\top \boldsymbol{A}^{-1} \Phi\left(\tilde{\boldsymbol{x}}_t\right) p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t \\
&= \mathrm{tr}\left(\boldsymbol{A}^{-1}\boldsymbol{Q}_t\right) + \boldsymbol{w}^\top \boldsymbol{Q}_t \boldsymbol{w} - \left(\boldsymbol{w}^\top \boldsymbol{q}_t\right)^2 \in \mathbb{R}.
\end{aligned} \tag{2.10}$$

$$\tag{2.11}$$

where $\Phi_t := \Phi\left(\tilde{\boldsymbol{x}}_t\right)$, and analytic solutions for

$$\boldsymbol{q}_t := \int \Phi_t p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t, \tag{2.12}$$

$$\boldsymbol{Q}_t := \int \Phi_t \Phi_t^\top p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t \tag{2.13}$$

are provided in Appendix A.1.

With the above in mind, we summarize the state prediction with uncertainty propagation via exploiting moment-matching on LGM-FF as follows:

$$\left.\begin{array}{lll}
 & p\left(\boldsymbol{x}_{t+1}\right) = f_{MM}\left(p\left(\boldsymbol{x}_t\right), \boldsymbol{u}_t\right) & \\
\text{subject to} & p\left(\boldsymbol{x}_t\right) \sim \mathcal{N}\left(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t\right), \ \forall t \in \mathbb{N} & \\
\text{with latent predictions} & \mu_{t+1} = \mu_{MM}\left(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t, \boldsymbol{u}_t\right) & \\
 & \sigma_{t+1}^2 = \sigma_{MM}^2\left(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t, \boldsymbol{u}_t\right) &
\end{array}\right\}. \tag{2.14}$$

The computation cost of exploiting moment-matching with standard GPs dynamics is $\mathcal{O}\left(DN^2\right)$, where D is the state dimension and N is the training sample size that increases as the learning process progresses. In comparison, exploiting moment-matching with LGM-FF has a computation cost of $\mathcal{O}\left(DM^2\right)$, where M is the number of Fourier features, resulting a fixed time consumption for prediction. In addition, previous works has demonstrated that $M \ll N$ in robotics

tasks [12, 32].

2.2.2. Finite-horizon Probabilistic Model Predictive Control

Model Predictive Control (MPC) is a process that finds the optimal control by relying on future predictions via the system's dynamics model [41]. Several pieces of research has proofed the mathematics convergence and stability of a probabilistic MPC (pMPC) [26, 31, 42, 43], an MPC with data-driven probabilistic dynamics, and its effectiveness in robotics applications are also demonstrated [12, 29, 30].

Specifically for each time-step t, pMPC finds the optimal H-step control sequence $\boldsymbol{u}_\star = [\hat{\boldsymbol{u}}_1, ..., \hat{\boldsymbol{u}}_H]$ that minimizes a user-defined finite-horizon loss $\mathcal{L}(\cdot)$ by recursively exploit probabilistic state predictions:

$$\left.\begin{array}{lll} \text{minimize}_{\boldsymbol{u}_\star} & \mathcal{L}\left(p\left(\boldsymbol{x}_t\right)\right) & = \sum_{k=2}^{H+1} \mathbb{E}\left[\ell\left(\hat{\boldsymbol{x}}_k\right) | p\left(\hat{\boldsymbol{x}}_k\right)\right] \in \mathbb{R} \\ \text{subject to} & p\left(\hat{\boldsymbol{x}}_{k+1}\right) & = f_{MM}\left(p\left(\hat{\boldsymbol{x}}_t\right), \boldsymbol{u}_t\right),\ \hat{\boldsymbol{x}}_1 = \boldsymbol{x}_t \\ & p\left(\hat{\boldsymbol{x}}_k\right) \sim \mathcal{N}\left(\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\Sigma}}_t\right), & k = 1, ..., H+1 \\ & \hat{\boldsymbol{u}}_k \in \mathbb{U}, & k = 1, ..., H \end{array}\right\}, \quad (2.15)$$

where $\ell(\cdot)$ is the immediate loss. The computation cost of pMPC planning with LGM-FF dynamics model is $\mathcal{O}(HDM^2)$, depending on prediction horizon H, the state dimension D and the Fourier feature size M. By truncating the planning horizon H short, a lower computation cost enables pMPC to recursively solve the problem with newly observed state at a fixed and high control frequency, thus making it more robust to environmental changes than long-horizon pMPC planning [29].

2.3. Model-based Reinforcement Leaning

Model-based Reinforcement Learning (MBRL) [24, 44] is a process that learns the systems dynamics model from training samples that collected from actual trial-and-error. During the trial-and-error (or the planning phase), the system

will operate the optimal control via pMPC with the model trained with previously collected samples. Meanwhile, new samples are also collected during the planning phase and the model will get updated in the upcoming training phase. By recurring the planning and training phase, we can obtain a dynamics model with improved accuracy by collecting more training samples, while we can also expect a better performance by planning with a higher accuracy dynamics model.

2.3.1. Model-based Reinforcement Learning Process

In MBRL learning process, trials will repeat until reaching the desired performance, where a trial is defined as completing one planning and one training phase. In each trial's planning phase, finite steps are repeated at a fixed frequency until any user-defined termination condition is satisfied. During each time-step t, the pMPC determine an H-step optimal control sequence $\boldsymbol{u}_\star = [\hat{\boldsymbol{u}}_1, ..., \hat{\boldsymbol{u}}_H]$ that minimizes the finite-horizon loss $\mathcal{L}(\boldsymbol{x}_t)$ based on the observed state $\boldsymbol{x}_t$. The first control sequence is selected, $\boldsymbol{u}_t := \hat{\boldsymbol{u}}_1$ and applied to the system. After the planning phase is terminated, the MBRL enters training phase and refines its model using samples collected from all previous trials.

2.3.2. Ahead pMPC Planning for Delay Compensation

As the pMPC finds the optimal control based on the observed state, its planning time will cause a delay between state observation and applying the optimal control. Due to the delay, the model accuracy will suffers from inconsistent control between each training sample. Similar to the parallel MPC workflow [28] and the asynchronous control [45], we implement the ahead pMPC planning to alleviate the delay and the inconsistency control issue. In specific, the pMPC plans for an n step ahead control sequence $\boldsymbol{u}_{t+n}$ based on a predicted state $\hat{\boldsymbol{x}}_{t+n}$ obtained by recursively exploiting state prediction via Eq. (2.14). As the control signal $\boldsymbol{u}_t$ at time-step t was previously planned and applied right after the state observation, the ahead pMPC compensates for the control delay and improves the control consistency of training samples, Fig. 2.2. Furthermore, the ahead pMPC can increase control frequency as the pausing is no longer necessary.

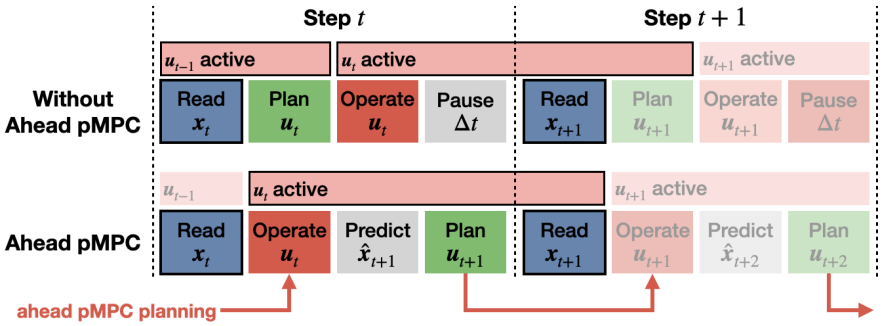

Figure 2.2. Example of control delay compensation with one-step ($n = 1$) ahead pMPC planning.

Algorithm 1: MBRL with Ahead pMPC

Initial inputs:

Empty sample set: $\tilde{\boldsymbol{X}}$, $\boldsymbol{y}$

Trial-end termination condition: $\mathbb{X}_{end}$

Finite-horizon loss function: $\mathcal{L}(\cdot)$; Immediate loss function: $\ell(\cdot)$

Ahead pMPC steps: n

MBRL process:

while (**not** reaching the desired performance) **do**

`ResetSystem()`

$t = 0$

 # Get initial moment-matching prediction model

$f_{MM} \leftarrow$ `TrainModel(`$\tilde{\boldsymbol{X}}, \boldsymbol{Y}$`)`

 # Get Initial state

$\boldsymbol{x}_0 \leftarrow$ `GetState()`

 # Initial ahead pMPC planning

$p(\hat{\boldsymbol{x}}_1) \sim \mathcal{N}(\boldsymbol{x}_0, \boldsymbol{0}_D)$

for $i = 1, ..., n$ **do**

$\boldsymbol{u}^\star = \arg\min_{\boldsymbol{u}} \mathcal{L}(p(\hat{\boldsymbol{x}}_i))$

$\boldsymbol{u}_i = \boldsymbol{u}^\star(1)$

$p(\hat{\boldsymbol{x}}_{i+1}) = f_{MM}(p(\hat{\boldsymbol{x}}_i), \boldsymbol{u}_i)$

while $\boldsymbol{x}_t \notin \mathbb{X}_{end}$ **do**

$t \ += 1$

 # Get current state

$\boldsymbol{x}_t \leftarrow$ `GetState()`

 # Operate the optimal control

`Operate(`$\boldsymbol{u}_t$`)`

 # Future state predictions via Eq. (2.14)

$p(\hat{\boldsymbol{x}}_t) \sim \mathcal{N}(\boldsymbol{x}_0, \boldsymbol{0}_D)$

for $i = 1, ..., n$ **do**

$p(\hat{\boldsymbol{x}}_{t+i}) = f_{MM}(p(\hat{\boldsymbol{x}}_{t+i-1}), \boldsymbol{u}_{t+i-1})$

 # n-step ahead pMPC planning via Eq. (2.15)

$\boldsymbol{u}^\star = \arg\min_{\boldsymbol{u}} \mathcal{L}(p(\hat{\boldsymbol{x}}_{t+n}))$

$\boldsymbol{u}_{t+n} = \boldsymbol{u}^\star(1)$

 # Save sample to corresponding sets

$\tilde{\boldsymbol{x}}_t = \left[\boldsymbol{x}_t^\top, \boldsymbol{u}_t^\top\right]^\top$

$\boldsymbol{y}_t \leftarrow$ `GetState()`

$\tilde{\boldsymbol{X}} = \left\{\tilde{\boldsymbol{X}}, \tilde{\boldsymbol{x}}_t\right\}, \boldsymbol{Y} = \{\boldsymbol{Y}, \boldsymbol{y}_t\}$

3. Extended Robotic State by Incorporating Model-uncertainty to Enhance Contact Safety During the Model-based Reinforcement Learning Process

3.1. Contact Safety and Model-uncertainty

3.1.1. Model-uncertainty of Probabilistic Model-based Reinforcement Learning

Probabilistic Model-based Reinforcement Learning (MBRL) is an attractive option for robotics scenarios due to its effectiveness and high sample efficiency [44, 46]. This allows for on-site learning of robotic tasks. However, the fully data-driven dynamics model cannot accurately represent the system dynamics during the early stages of the learning process when there is data scarcity, causing high model-uncertainty, as illustrated in Fig. 3.1. While model-uncertainty is high, we cannot expect the probabilistic Model Predictive Control (pMPC) to generate reliable control for the intended robotic task. Although applying these unreliable controls is required as *exploration*, it could damage the robot itself and its surroundings in a physical environment. This is particularly important when the robot is performing a contact-rich task in a contact-rich environment, such as

performing common kitchen tasks (Fig. 3.2). As safety is always the primary consideration and the learning process occurs on-site, a contact-safe learning process in MBRL must be considered.

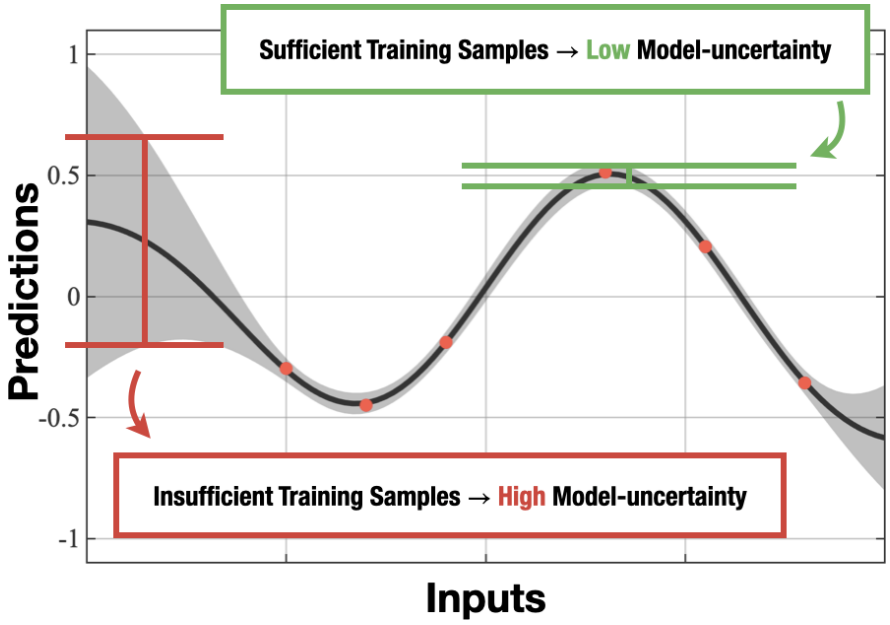

Figure 3.1. This figure illustrates the relationship between model-uncertainty and sample scarcity.

3.1.2. Enhancing Contact-safety During the Learning Process

Tactile exploration, through touching an unknown object, exemplifies the contact-safe behavior of the learning process. When humans lack sufficient knowledge about an object, their fundamental fear and intolerance of uncertainty elicits cautious behaviors when approaching it to maximize survival [47, 48]. Inspired by this concept, a robot can also exhibit contact-safe exploration behaviors by moving cautiously during high uncertainty. Therefore, it may be reasonable to associate model uncertainty with the robot's control space to achieve this behavior.

We present a contact-safe MBRL that reduces the intensity of unexpected intensive contacts during the learning process by promoting a safety characteristic:

Figure 3.2. This figure provides examples of contact-rich kitchen tasks, where unsafe and intensive contact behaviors can result in damage to the robot or its environment.

The robot takes small and gentle actions while the model-uncertainty is high and performs the intended task confidently when the model-uncertainty is reduced as exploration becomes sufficient. Such safety is achieved by an uncertainty-aware approach that associates the learning progress with the control space of the probabilistic Model Predictive Control (pMPC) optimization problem. Previous research handled pMPC control space using a differentiable squashing function [29], but this approach may result in unreliable predictions near constraint boundaries [49]. To associate the implicit model-uncertainty with the pMPC optimization problem to improve contact safety during exploration, a task-relevant state that explicitly accommodates the model-uncertainty is necessary for reformulating the robot state

A deterministic-reformulated probabilistic model predictive control (pMPC) [31] was proposed to support dynamic and state-associated control space by following Pontryagin's Maximum Principle [50]. This was achieved by prolonging the state with the prediction uncertainty from the standard Gaussian Process (GPs)-based dynamics model. By using a similar concept [31], we deterministi-

cally reformulated a task-relevant robotic state for the LGM-FF dynamics. This allows us to explicitly accommodate model uncertainty into the robot state while maintaining high sample efficiency and low computation cost when performing robotic tasks. By making model uncertainty an explicit part of the robotic state, we can use this information to constrain pMPC exploration within an adjustable uncertainty-aware control space to achieve the intended safety characteristics.

3.1.3. Existing Methods for Enhancing Contact-safety

Safe Contact-rich Manipulations using Reinforcement Learning

To enhance contact safety in contact-rich robotic scenarios, two common Reinforcement Learning (RL) approaches have been developed.

The first approach involves learning to generate "safe" policies to avoid collisions. For example, Levine et al. [51] created a unified control policy by learning from a set of desired trajectories that considered safety, and Yamada et al. [52] switched to a collision-avoidance model-based policy if the model-free RL policy was likely to cause a collision during an object-manipulation task in an obstructed environment.

The second approach involves learning a controller to adjust robot stiffness with RL to avoid intensive contacts. For example, Martín-Martín et al. [53] safely performed surface-wiping and door-opening tasks with a model-free RL-learned variable impedance controller that penalized forces exceeding a payload. Beltran et al. [54] learned the gain of the paralleled position/force controller to perform fail-safe ring/peg insertion tasks, and Wirnshofer et al. [55] achieved a safe peg insertion task with an RL-learned controller that combined a compliance controller with a set of goal-directed policies.

While the approaches mentioned above focus on contact safety during execution, they do not consider safety during exploration. Furthermore, their desired behaviors are expressed in terms of reward/loss functions. However, multi-objective optimizations require careful setups to prevent convergence issues [56].

In contrast, we prioritize the contact-safety during the learning process (or the exploration) by implementing an uncertainty-aware control space, and optimizing for a single objective loss function to achieve the intended robotic task.

Utilizing Uncertainties Information with Probabilistic Methods

Uncertainty in probabilistic approaches provides rich information, which has been utilized for various purposes in prior works. Some used uncertainty to encourage agents to expand their exploration coverage [57, 58], while others planned for a policy that minimizes uncertainty [29, 30].

Lee et al. [59] proposed a guided uncertainty-aware approach that directs the robot to an uncertain area and switches to a reinforcement learning (RL) policy to perform a peg-in-hole task. LaGrassa et al. [60] performed a door-opening task by patching the model-based reinforcement learning (MBRL) with an imitation-learned model-free local policy when the MBRL's dynamics model contains high uncertainty.

In contrast to the above studies, our method explores a novel application that associates model uncertainty with the predictive model control (pMPC) limits to adjust the agent's MBRL learning behavior.

3.2. Methodology

3.2.1. Extended Robot State for Accommodating Model-uncertainty

Our objective is to achieve a contact-safe learning process by associating model-uncertainty with MBRL exploration such that the risk of causing damage to the environment or the robot itself can be reduced. Specifically, this is achieved with a uncertainty-aware pMPC control space that is associated with model-uncertainty. Therefore, such model-uncertainty has to be explicitly accessible from the robotic state.

Given a robotic state $\boldsymbol{x}_t \in \mathbb{X} \subset \mathbb{R}^D$ and control $\boldsymbol{u}_t \in \mathbb{U} \subset \mathbb{R}^U$, a Gaussian distributed predictive state $p(\boldsymbol{x}_{t+1}) \sim \mathcal{N}(\boldsymbol{\mu}_{t+1}, \boldsymbol{\Sigma}_{t+1})$ is obtained via exploiting moment-matching with LGM-FF dynamics model, as shown in Eq. (2.14):

$$p(\boldsymbol{x}_{t+1}) = f_{MM}(p(\boldsymbol{x}_t), \boldsymbol{u}_t), \tag{3.1}$$

where the predictive covariance matrix $\boldsymbol{\Sigma}_t$ indicates the model-uncertainty.

In order to allow pMPC to access such implicit model-uncertainty while solving optimization problems to achieve contact safety, we reformulated a deterministic robotic state $\boldsymbol{s}_t := \left[\boldsymbol{\mu}_t^\top, \mathbf{1}^\top \boldsymbol{\Sigma}_t\right]^\top \in \mathbb{S} \subset \mathbb{R}^{2D}$ that accommodates the predictive mean and the diagonal elements of the covariance as deterministic entries, where $\mathbf{1}$ denotes a column vector of ones. Meanwhile, a deterministic formulated dynamics model $f_u(\cdot) : 2D + U \to 2D$ is presented to adopt the state changes:

$$\boldsymbol{s}_{t+1} = \begin{bmatrix} \boldsymbol{\mu}_{t+1} \\ \boldsymbol{\Sigma}_{t+1}\mathbf{1} \end{bmatrix} = f_u(\boldsymbol{s}_t, \boldsymbol{u}_t) = \begin{bmatrix} \mu_{MM,1}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t, \boldsymbol{u}_t) \\ \vdots \\ \mu_{MM,D}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t, \boldsymbol{u}_t) \\ \sigma^2_{MM,1}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t, \boldsymbol{u}_t) \\ \vdots \\ \sigma^2_{MM,D}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t, \boldsymbol{u}_t) \end{bmatrix}, \tag{3.2}$$

where $\mu_{MM,i}(\cdot), \sigma^2_{MM,i}(\cdot)$ is the moment-matching equation from Eq. (2.14), with $i = 1, ..., D$ indicates the corresponding latent dimension.

In the following applications, robots are velocity-controlled with discrete acceleration control signals. Therefore, the entries of the deterministic reformulated state $\boldsymbol{s}$ are defined as follows:

$$\boldsymbol{s} := \begin{bmatrix} \boldsymbol{\mu}_{\boldsymbol{p}} \\ \boldsymbol{\mu}_{\dot{\boldsymbol{p}}} \\ \boldsymbol{\Sigma}_{\boldsymbol{p}}\mathbf{1} \\ \boldsymbol{\Sigma}_{\dot{\boldsymbol{p}}}\mathbf{1} \end{bmatrix}, \tag{3.3}$$

where $\boldsymbol{p} \in \mathbb{R}^k$ and $\dot{\boldsymbol{p}} \in \mathbb{R}^k$ are the actuators' position velocity of a k-Degree-of-Freedom (DoF) robot with $\boldsymbol{\mu}_{\boldsymbol{p}}$, $\boldsymbol{\mu}_{\dot{\boldsymbol{p}}}$, $\boldsymbol{\Sigma}_{\boldsymbol{p}}$ and $\boldsymbol{\Sigma}_{\dot{\boldsymbol{p}}}$ are their corresponding mean vector and covariance matrices.

3.2.2. Contact-safe Exploration with Model Predictive Control

Contact-safe Characteristics During Exploration

In the following text, the term "action" refers to the robot's behavior, while "control" denotes the control command sent to the agent.

Using the deterministically reformulated robot dynamics model and state, we present a contact-safe MBRL with uncertainty-aware pMPC control space to reduce the intensity of unexpected intensive contacts during the learning process. Specifically, our method promotes two safety characteristics:

1. **Safety-Measure-A:** We encourage the agent to take smaller actions by limiting control during high uncertainty.

2. **Safety-Measure-B:** We encourage the agent to choose controls that reduce high velocities during high uncertainty.

We include "Safety-Measure-B" because limiting control with "Safety-Measure-A" also restricts the agent's ability to reduce its velocity.

Probabilistic Model Predictive Control with Uncertainty-aware Control Space

We establish an uncertainty-aware control space to constrain pMPC exploration during periods of high model-uncertainty. This is based on the ability to measure uncertainty from the predictive state of the deterministically reformulated dynamics model (Eq. 3.2).

Given a state $\boldsymbol{s}_t$, for all receding horizon H, we established an uncertainty-aware control space for pMPC to solve the optimal control problem that minimizes the

finite horizon loss $\mathcal{L}(\cdot)$:

$$\left.\begin{array}{lll} \text{minimize} & \mathcal{L}(\boldsymbol{s}_t) & = \sum_{k=2}^{H+1} \ell(\hat{\boldsymbol{s}}_k) \\ \text{subject to} & \hat{\boldsymbol{s}}_{k+1} & = f_u(\hat{\boldsymbol{s}}_k, \hat{\boldsymbol{u}}_k) \\ & \hat{\boldsymbol{s}}_1 & = \boldsymbol{s}_t \\ & \hat{\boldsymbol{s}}_k & \in \mathbb{S},\ k = 1, ..., H+1 \\ & \hat{\boldsymbol{u}}_k & \in \mathbb{U}',\ k = 1, ..., H \\ & \mathbb{U}' & = g_u(\mathbb{U}, \hat{\boldsymbol{s}}_k) \subset \mathbb{U} \end{array}\right\}, \tag{3.4}$$

where $\ell(\cdot) : \mathbb{S} \to \mathbb{R}$ is the immediate loss, $\hat{\boldsymbol{s}}_1, ..., \hat{\boldsymbol{s}}_{H+1}$ are the predictive state horizon within the pMPC optimization problem, and $g_u(\cdot) : \mathbb{S} \times \mathbb{U} \to \mathbb{U}$ is the uncertainty-aware control space the is designed to meet the Safety-Measure-A and Safety-Measure-B.

The Uncertainty-aware Control Space

In the following, we assume a standard control space that has an upper bound and a lower bound denoted by $\mathbb{U} \in [\boldsymbol{u}_{min}, \boldsymbol{u}_{max}]$, where $\boldsymbol{u}_{min}, \boldsymbol{u}_{max} \in \mathbb{U}$ are the boundaries.

We present linear control space that use scaling and translating mechanisms to achieve two desired safety characteristics. During high uncertainty, the scaling mechanism scales down the control space to satisfy Safety-Measure-A, which prevents intensive actions. However, scaling down the control space also limits the robot's ability to reduce momentum. Therefore, the translating mechanism translates the control space in the negative direction of the robot's velocity to prevent high momentum and ensure full Safety-Measure-B. Therefore, the uncertainty-aware control space is obtained via:

$$\mathbb{U}' = g_u(\mathbb{U}, \hat{\boldsymbol{s}}_k) = [K_s(\hat{\boldsymbol{s}}_k)\,\boldsymbol{u}_{min} - K_t(\hat{\boldsymbol{s}}_k), K_s(\hat{\boldsymbol{s}}_k)\,\boldsymbol{u}_{max} - K_t(\hat{\boldsymbol{s}}_k)] \in \mathbb{U}, \tag{3.5}$$

where $K_s(\cdot) : \mathbb{S} \to \mathbb{R}$ and $K_t(\cdot) : \mathbb{S} \to \mathbb{R}$ are the scaling and translating factor that modifies the control space (see Fig. 3.3).

To fulfill Safety-Measure-A, we have identified the following requirements for

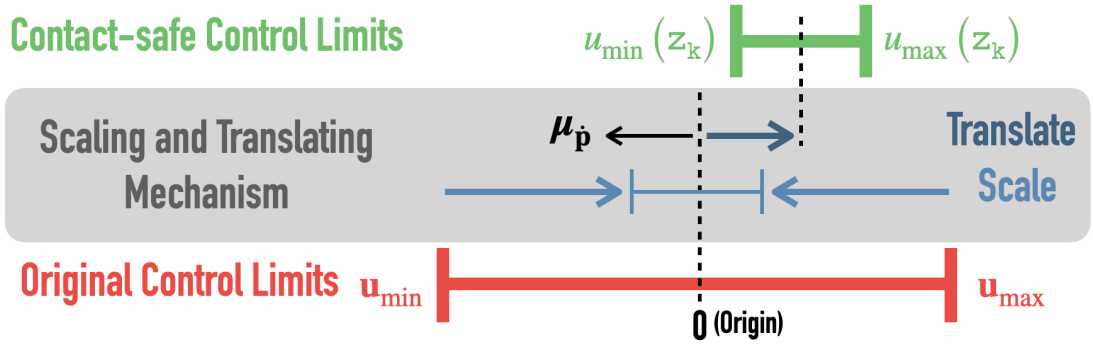

Figure 3.3. This figure illustrates the scaling and translating mechanisms of the uncertainty-aware control space. The scaling contributes to encouraging gentle behaviors, and the translating prevent high momentum (or velocity $\mu_{\dot{p}}$) during high model-uncertainty.

the function $K_s(\cdot)$:

(A1) The value should decrease during periods of high uncertainty to limit intensive action during these times.

(A2) The value must be constrained within the range of $(0, 1]$ to prevent a control space with zero range, negative range, or a range exceeding the standard control space.

(A3) The value should return to 1.0 during low uncertainty to enable the robot to perform its intended task confidently within the standard control space $\mathbb{U}$.

(A4) The function must have a parameter for adjusting the uncertainty sensitivity to achieve adjustable awareness.

Similarly, to achieve Safety-Measure-B, we have identified the following requirements for the function $K_s(\cdot)$:

(B1) The value should increase with velocity during high uncertainty to provide greater stopping power when both momentum and uncertainty are high.

(B2) The value should have a minimum of 0 during low uncertainty to enable the robot to achieve high momentum when needed within the standard control space $\mathbb{U}$.

(B3) The function must have a parameter for tuning the uncertainty sensitivity to achieve adjustable awareness.

(B4) The function must have a parameter for tuning the velocity sensitivity to achieve adjustable stopping power.

We present factoring functions with exponential decay mappings for scaling ($K_s(\cdot)$) and translating ($K_t(\cdot)$) that satisfy the two desired safety characteristics. These functions are simple mappings of non-linear decay or growth (by flipping the decay function) within the range of [0,1] under non-negative real inputs.

$$K_s(\hat{\boldsymbol{s}}) = \underbrace{(1-\beta_s)}_{\text{(A3)}} \overbrace{\exp(-\underbrace{\alpha_s}_{\text{(A4)}} \|\boldsymbol{\Sigma}_{\mathbf{p}}\|_2)}^{\text{(A1)}} + \underbrace{\beta_s}_{\text{(A2)}}, \tag{3.6}$$

$$K_t(\hat{\boldsymbol{s}}) = \underbrace{\overbrace{(1-\exp(-\underbrace{\alpha_t}_{\text{(B3)}} \|\boldsymbol{\Sigma}_{\mathbf{p}}\|_2))}^{\text{(B1)}} \underbrace{\gamma_t}_{\text{(B4)}} \boldsymbol{\mu}_{\dot{\mathbf{p}}}}_{\text{(B2)}}, \tag{3.7}$$

where $\alpha_s, \alpha_t \in \mathbb{R}\geq 0$ are the adjustable uncertainty-awareness parameters. The velocity sensitivity is denoted by $\gamma_t \in \mathbb{R}\geq 0$ and is typically set as the ratio of the control's feasible upper bound to the agent's maximum velocity. This prevents over-translation. β_s represents the minimum scaling value. To ensure a consistent scale of uncertainty, the uncertainty at the current state is estimated by measuring only the position variance $\boldsymbol{\Sigma}_{\boldsymbol{p}}$.

We can adjust the agent's learning behavior by changing its uncertainty-awareness parameters α_s and α_t. For example, we can use the standard contact-unsafe MBRL with α_s and α_t equal to 0, or a contact-safe MBRL with excessive uncertainty-awareness, where $\ln \alpha_s$ and $\ln \alpha_t$ equal 10. As an example where we assume a single dimensional velocity mean $\boldsymbol{\mu}_{\dot{\boldsymbol{p}}}$, Fig. 3.4 exemplifies the relationship between states $\boldsymbol{s}$ and control space modification factors $K_s(\cdot), K_t(\cdot)$ under under $\beta_s, \gamma_t = 0.2$. Note that the translating is only active while $\boldsymbol{\mu}_{\dot{\mathbf{p}}} \neq 0$ under a certain level of uncertainty.

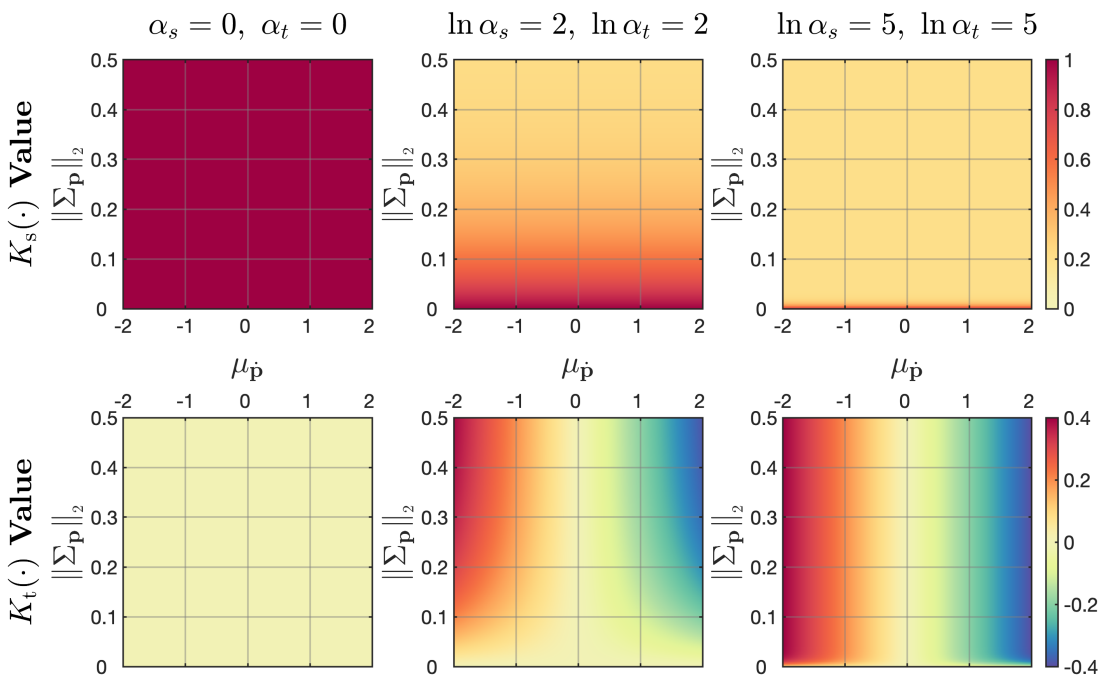

Figure 3.4. This figure shows the relationship between the scaling factor $K_s(\cdot)$, translating factor $K_t(\cdot)$ and the state $\boldsymbol{s}$ under different uncertainty-awareness α_s, α_t settings. Color indicates corresponding values.

3.3. Experimental Evaluation

3.3.1. Contact-rich Kitchen Tasks

Mixing and scooping operations are common tasks in cooking [61]. However, completing these tasks involves extensive contact between different objects, making it challenging to learn them from scratch using robotic hardware without causing damage to the robot or its environment. Therefore, they are suitable for demonstrating the effectiveness of our method.

We conducted both simulated and hardware experiments with HEBI robotic hardware, as shown in Fig. 3.5. The simulated robot emulates a hardware mixing setup to evaluate our method's effectiveness under various uncertainty-awareness settings. Hardware experiments verified the real-world potential of our method.

The mixing task's goal is to perform a circulating movement with the attached tool and mix the particles inside the container. The scooping task's goal is to scoop particles from one container and pour them into another.

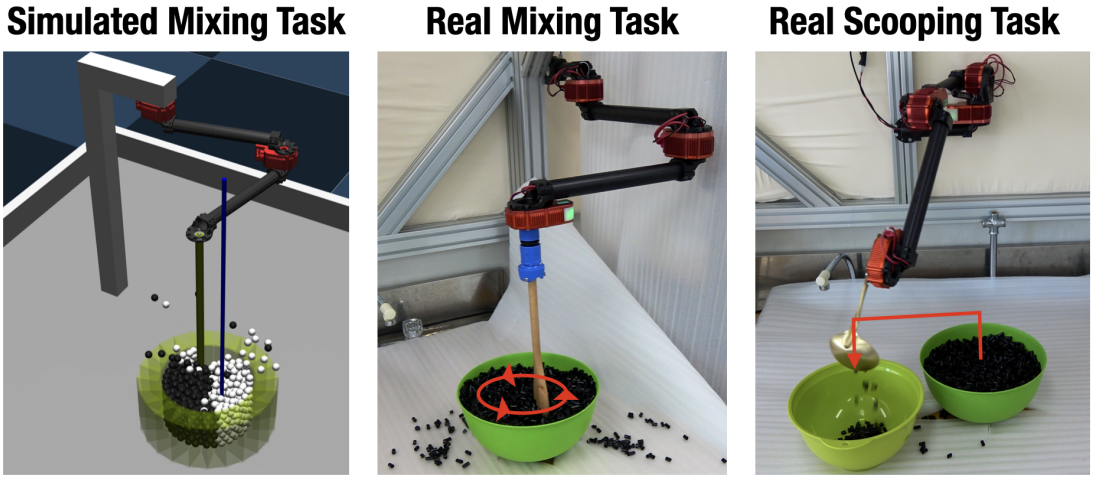

Figure 3.5. This figure shows the robot setup for the following experiments. Left: Mixing task using a simulated 2-DoF robotic arm built with HEBI parts. (The blue stick represents the reference target position for each step). Middle: Mixing task using a hardware 2-DoF robotic arm built with HEBI parts. Right: Scooping task using a hardware 4-DoF robotic arm built with HEBI parts.

3.3.2. Learning Implementation and Control Objectives

State Definition for Model-based Reinforcement Learning

Model-Based Reinforcement Learning (MBRL) learns the fully data-driven state-transition dynamics of a robot, as shown in Eq. 3.2. Therefore, learning requires a carefully defined robotic state. Since the actuators' behavior is periodic, we defined the position and velocity of a k-degree-of-freedom (DoF) robotic arm in Eq. 3.3 as follows:

$$\boldsymbol{p} := \begin{bmatrix} \sin \boldsymbol{\Theta} \\ \cos \boldsymbol{\Theta} \end{bmatrix}, \; \dot{\boldsymbol{p}} := \dot{\boldsymbol{\Theta}}, \; \boldsymbol{\Theta} = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_k \end{bmatrix}, \tag{3.8}$$

where $\boldsymbol{\Theta}$ denotes the vector that contains rotational positions of all k actuators. To achieve smooth movements, both simulated and hardware robots are velocity-controlled with a discrete acceleration control signal of $\boldsymbol{u} = \Delta\dot{\boldsymbol{p}}/\Delta t$. This control signal is generated via pMPC.

Task Objective of Probabilistic Model Predictive Control

The objective of our experiments is to learn the robot's dynamics from scratch while it performs the intended task. We achieve this by tracking provided hints: the world-space reference end-effector trajectories for each task. To emphasize the impact of unexpected contacts during the learning process, force/torque sensors are used, and the reference trajectories of the tasks do not contain stiff contacts.

For the simulated mixing task, the reference trajectory follows a circulating pattern with a diameter of 0.1 meters and a period of five seconds. The hardware mixing task has the same diameter as the simulated mixing task but with a period of three seconds. The reference trajectory for the scooping task consists of a series of positions and orientations for the spoon, which were recorded from human demonstrations of scooping and transporting particles from one container to another in world-space.

For better demonstration of our method, the immediate loss $\ell(\cdot)$ in Eq. 3.4 does not contain a regularization term for control sigma $\boldsymbol{u}$. Our method is responsible for all the behavior changes during the learning process of MBRL. With the above in mind, the immediate loss $\ell(\cdot)$ is designed to fulfill the trajectory tracking objective:

$$\ell(\boldsymbol{s}_t) = \underbrace{\mathrm{k}_p \|\boldsymbol{p}_t^{ee\star} - \boldsymbol{p}_t^{ee}\|_2}_{\text{position loss}} + \underbrace{\mathrm{k}_o \|\boldsymbol{o}_t^{ee\star} - \boldsymbol{o}_t^{ee}\|_2}_{\text{orientation loss}}, \tag{3.9}$$

where $\mathrm{k}_p, \mathrm{k}_o \in \mathbb{R}_{\geq 0}$ are corresponding weights and $\boldsymbol{p}_t^{ee}, \boldsymbol{o}_t^{ee}$ are the robot's end-effector position and orientation in the world-space at time step t that track references $\boldsymbol{p}_t^{ee\star}, \boldsymbol{o}_t^{ee\star}$. The end-effector position in world-space $\boldsymbol{p}_t^{ee}, \boldsymbol{o}_t^{ee}$ is obtained by forward kinematics, where joint positions are calculated via an Euler equation from the state $\boldsymbol{s}$:

$$\hat{\boldsymbol{\Theta}} = \mathrm{Re}\left[-i\left(\boldsymbol{\mu}_{\cos\boldsymbol{\Theta}} + i\boldsymbol{\mu}_{\sin\boldsymbol{\Theta}}\right)\right]. \tag{3.10}$$

In the followings, the tracking error is defined as $\|\boldsymbol{p}_t^{ee\star} - \boldsymbol{p}_t^{ee}\|_2$. All the experiments share these parameters: number of features $M = 65$; pMPC receding horizon $H = 3$; all entries of $\boldsymbol{u}_{min}$ and $\boldsymbol{u}_{max}$ are -0.2 and 0.2 rad/s; $\beta_s = 0.3$ and $\gamma_t = 0.2$ in Eq. 3.6 and Eq. 3.7, respectively; k_p and $\mathrm{k}_o = 1.0$ in Eq. 3.9.

3.3.3. Simulation Experiments and Results

Simulated Environment Setup

Mixing Task: We simulate a real particle-mixing task environment using the Mujoco simulator (see Fig. 3.5-left). The two-Degree-of-Freedom (DoF) arm is 0.65 m in total length, with a 0.02 m-diameter cylinder-shaped stick attached to its end-effector. The stick is 0.5 m long and does not touch the bottom of the bowl during mixing. A force sensor is attached to the stick to measure contact forces. The bowl has a diameter of 0.24 m and is fixed beneath the arm, filled with 900 particles that are 0.03 m in diameter and have a total mass of 9 kg. The learning process consists of 12 trials, each with 100 steps taken at 0.1-second intervals.

Comprehensive Comparisons

This section demonstrates the effectiveness of our contact-safe MBRL by comparing the following three setups

- Uncertainty-aware: our uncertainty-aware MBRL where the pMPC control space is associated with model-uncertainty.
- Fixed: The standard MBRL with uncertainty-awareness parameters set to $\alpha_s, \alpha_t = 0$ that has a fixed (unmodified) control space.
- Linear Changing: The contact-safe MBRL with globally evaluated uncertainty via sample size that increase linearly during the learning process. (Replacing $\|\mathbf{\Sigma_p}\|_2$ in Eq. (3.6) and Eq. (3.7) with N^{-1}).

We conducted 20 experiments to test our method under different uncertainty-awareness settings and compared it to the above two baselines. We measured the learning efficiency by averaging the tracking error across all trials and evaluated the contact intensity reduction by averaging the top 3%, 5%, and 10% measured stick forces.

(1) Contact-safe MBRL vs. standard MBRL (contact-unsafe): This comparison aims to demonstrate the effect of changing the uncertainty-awareness parameters, α_s and α_t. Figure 3.6 shows the results of 20 learn-from-scratch trials with "fixed" and our method, using different uncertainty-awareness settings:

$[\alpha_s = 0, \ln \alpha_s = 6, \ln \alpha_s = 9]$ and $[\alpha_t = 0, \ln \alpha_t = 6, \ln \alpha_t = 9]$. Here, $\alpha_s = 0$ or $\alpha_t = 0$ disables the corresponding control space modifying mechanism. All results with $\ln \alpha_s = 9$ are highly sensitive to model uncertainty and failed to perform the task due to overly conservative behavior. In comparison, other results show similar learning efficiency with "fixed", where tasks are learned within eight trials. By comparing the average "max stick forces," our method significantly reduces contact intensity during the learning process, especially with the setting $\{\ln \alpha_s, \ln \alpha_t = 6\}$, which has high learning efficiency while being contact-safe.

(2) Uncertainties for contact-safe MBRL: Fig. 3.7 demonstrates the benefits of utilizing the model uncertainty of our method over "linear changing" when operating in the best uncertainty-awareness setting we experienced, which was $\{\ln \alpha_s, \ln \alpha_t = 6\}$. Results for "fixed" are also provided. Although the learning efficiencies of the two baselines and our method are similar, "linear changing" and our method show significantly lower averaged contact forces throughout the learning process. However, in the 8th, 9th, and 11th trials, the contact forces observed in "linear changing" have a higher standard deviation than those in our method, indicating that "linear changing" has a higher possibility of encountering intensive contacts. This is because "linear changing" globally evaluates uncertainty by sample size, which cannot handle novel situations after collecting a certain amount of samples. In contrast, our method adaptively evaluates uncertainty by model uncertainty, which is based on exploration coverage. As a result, the $K_s(\cdot)$ value of our method is adaptive and holds higher variances throughout 15 trials, resulting in the lowest overall contact intensity.

Contact-safe MBRL Behavior Analysis

We selected the individual results of the following settings to demonstrate the scaling and translating mechanisms' roles (Fig. 3.8):

(1): The standard contact-unsafe MBRL $\{\alpha_s, \alpha_t = 0\}$.

(2): Contact-safe MBRL with uncertainty-awareness settings $\{\alpha_s = 0, \alpha_t = 6\}$ (scaling disabled).

(3): Contact-safe MBRL with uncertainty-awareness settings $\{\alpha_s = 0, \alpha_t = 9\}$ (scaling disabled).

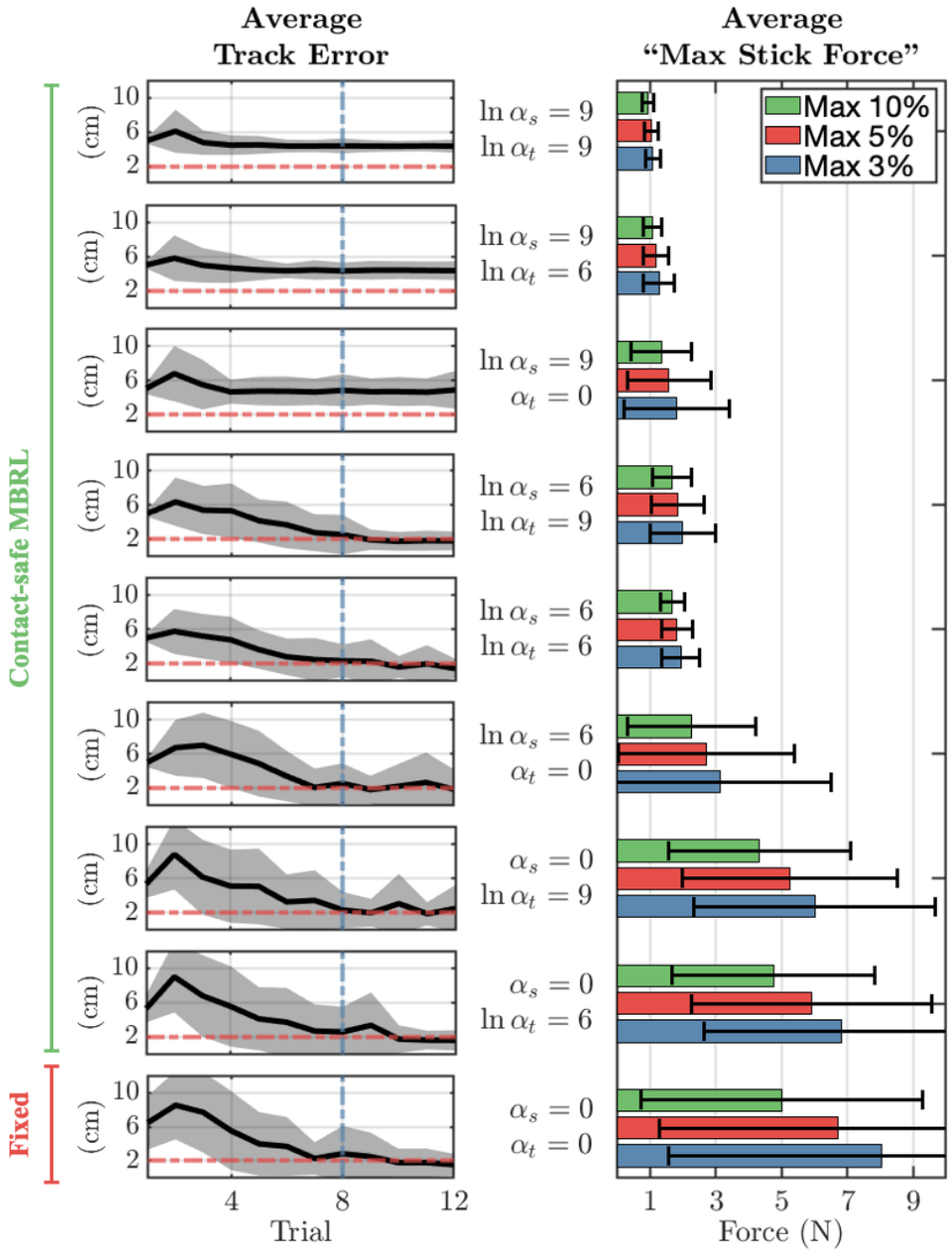

Figure 3.6. This figure shows the relationship between uncertainty-awareness and robot's exploration behavior. The left half of the display shows the average tracking error, while the right half shows the measured "max stick force" over 20 experiments for each uncertainty-awareness setting. The red dashed line indicates 2 cm to distinguish task acquisition status, while the blue dashed line indicates the eighth trial and measures task completion and learning efficiency.

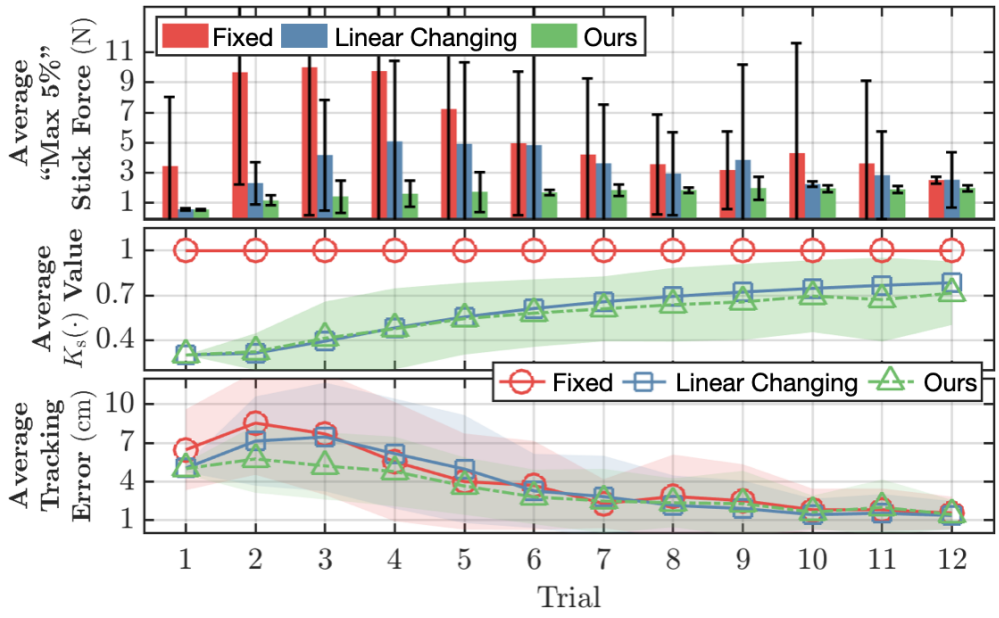

Figure 3.7. The upper figure shows the average "max 5% stick force." The middle figure shows the average "$K_x(\cdot)$" value, which is inversely proportional to the uncertainty level. The bottom figure shows the average tracking error, illustrating learning efficiency. All values are averaged over 20 experiments, and color fills illustrate the 95% confidence interval. "fixed" refers to contact-unsafe MBRL, while "linear changing" refers to $\{\ln \alpha_s, \ln \alpha_t = 6\}$ contact-safe MBRL with global uncertainty, which replaces position uncertainty $\|\mathbf{\Sigma}_p\|_2$ with inverted sample size N^{-1}. "Ours" refers to our contact-safe MBRL with settings $\{\ln \alpha_s, \ln \alpha_t = 6\}$.

(4): Contact-safe MBRL with uncertainty-awareness settings $\{\alpha_s = 6, \alpha_t = 0\}$ (translating disabled).

(5): Contact-safe MBRL with uncertainty-awareness settings $\{\alpha_s = 9, \alpha_t = 0\}$ (translating disabled).

(6): Contact-safe MBRL with uncertainty-awareness settings $\{\alpha_s = 6, \alpha_t = 6\}$.

Fig. 3.8 illustrates the stick force and velocity observations during the MBRL learning process under various uncertainty-awareness settings throughout twelve trials.

Fig. 3.8-(A) shows that increasing α_t effectively reduces momentum while exploring novel area. However, without scaling, intensive contacts occur despite the lower velocities (Fig. 3.8-(B)). This is because contacting the bowl's edge can significantly reduce the stick's velocity, limiting the effect of the translating

mechanism (Fig. 3.9: upper), and disabling the scaling mechanism allows intensive control signals. Consequently, safety measures are disabled when the robot contacts with objects, which is considered contact-unsafe acts.

Fig. 3.8-(C) emphasizes the essence of safety-measure-B, or the translating mechanism. While the translating mechanism is disabled, intensive contacts occurred because the robot's capability to reduce its velocity when exploring novel areas is limited by the scaling mechanism, as illustrated in Fig. 3.9: lower. In other words, if the robot approaches novel areas with high momentum, the ability to slow down is gradually limited as model-uncertainty increases. Exploration with high momentum in novel areas can increase the risk of intensive contacts, which is also considered contact-unsafe acts.

On the other extreme, setting (5) shows overly conservative behavior, resulting in task acquisition failure. This is because its excessively high uncertainty-awareness limits the robot's ability to perform the task despite it being considered safe to do so.

Lastly, Fig. 3.8-(D) shows that enabling both scaling and translating mechanisms with proper uncertainty-awareness allows the robot to acquire the task at lower contact forces and explore areas that frequently have contacts safely.

3.3.4. Hardware Experiment and Results

Hardware Environment Setup

Mixing Task: The configuration for the mixing task on the hardware robot is similar to that used in the simulation setup. A stick is attached to the end-effector without making contact with the bottom of the bowl during mixing. The metal bowl used has a diameter of 0.24 m and is filled with cut drinking straws to simulate a cooking ingredient. The learning process consists of 15 trials with each trial containing 100 steps that last for 0.1 seconds each.

Scooping Task: The scooping task involves a four-degree-of-freedom arm with a spoon attached to the end-effector, and two bowls: one filled with pieces of straw and one empty. During the learning process, the arm is gravity-compensated and velocity-controlled. This experiment demonstrates the effectiveness of our method in changing learning behavior in a large workspace. The learning process consists

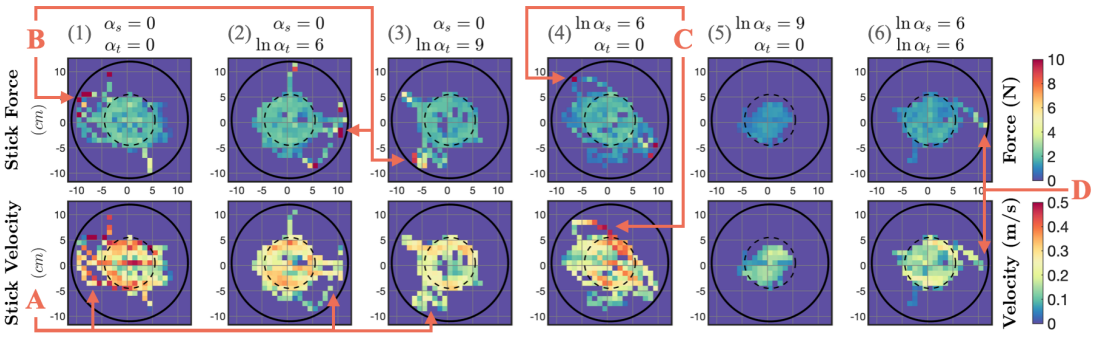

Figure 3.8. This figure illustrates the behavior analysis under various uncertainty-awareness settings throughout twelve trials. The upper half shows the stick force, and the bottom half shows corresponding velocity during the learning process. Each figure shows the top-view with bowl boundary (black solid lines) and the reference mixing trajectory (black dashed lines). Color differences indicate the maximum value observed at each location in the bowl. The labels (1)-(6) correspond to different uncertainty-awareness settings: (1) standard contact-unsafe MBRL, (2)-(3) scaling disabled, (4)-(5) translating disabled, and (6) the best combination setting. Panel (A) shows that translating reduced the velocity of entering a novel area. Panel (B) shows that intensive contacts are disabled during scaling. Panel (C) shows that intensive contacts occur when exploring a novel area if translating is disabled. Panel (D) shows that contacts are explored safely when both scaling and translating are enabled.

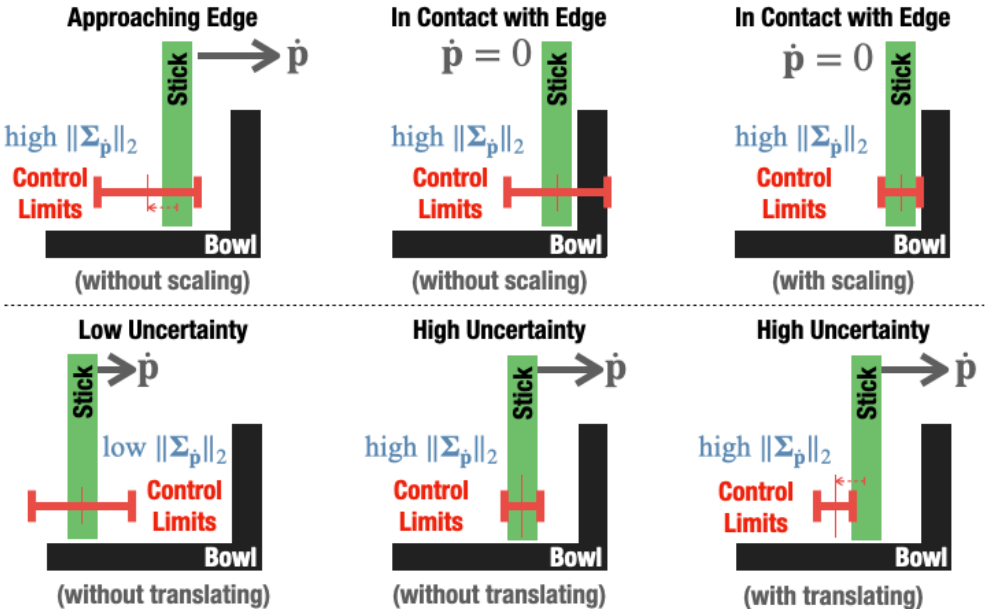

Figure 3.9. This figure illustrates the changes in control limits with/without scaling (upper), as well as with/without translating (lower).

of 40 trials and 65 steps, each of which has 0.2-second intervals. For hardware experiment, we compare our uncertainty-aware MBRL (with uncertainty-awareness parameter $\{\alpha_s, \alpha_t = 5\}$) with "linear changing" and "fixed" as in simulation experiment.

Mixing Task

As the hardware robot was not installed with load cells, we measure joint torques from actuator sensors to estimate contact forces. We expect that the measured torque will increase as more intensive contact occurs.

Fig. 3.10 shows that all three methods has a similar learning efficiency where the mixing task were learned at around 11th trial. While having a similar learning efficiency, Fig. 3.11 shows that our method significantly reduced the measured torque during the learning process, resulting a contact-safe learning process that we anticipated.

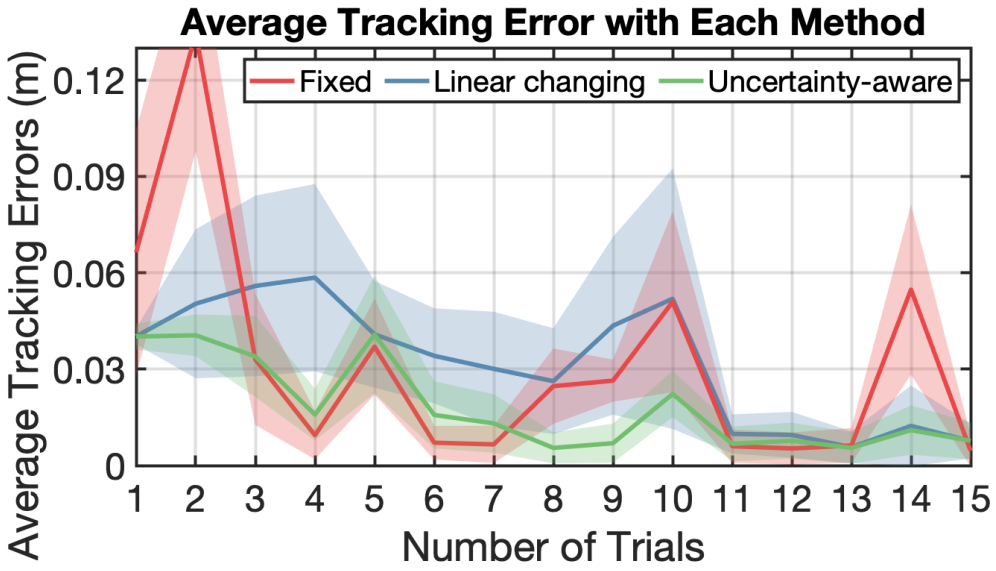

Figure 3.10. This figure shows the tracking error of each method during the mixing task learning process.

By comparing the environmental differences between each method (see Fig. 3.12), we find that our method produces a much clearer environment after the learning process, with a lower density of scattered straw cuts. These results demonstrate

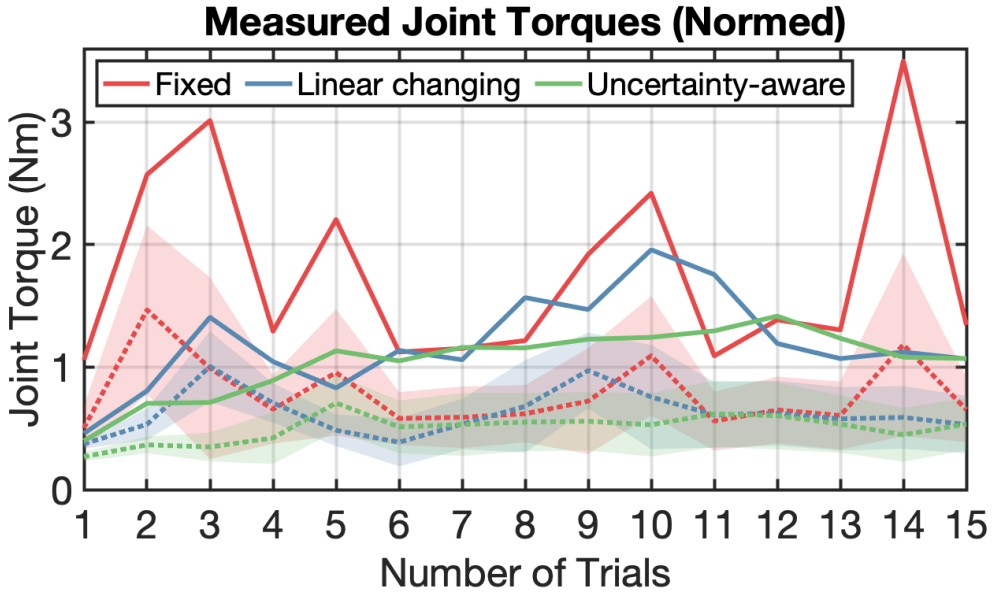

Figure 3.11. This figure shows the measured torque for each method during the mixing task learning process. The solid lines indicate the maximum measured torque, while the dashed lines represent the average measured torque within each trial. Standard deviation is shown as color shades.

the real-world feasibility of our method, producing similar results to those found in the above simulation experiments.

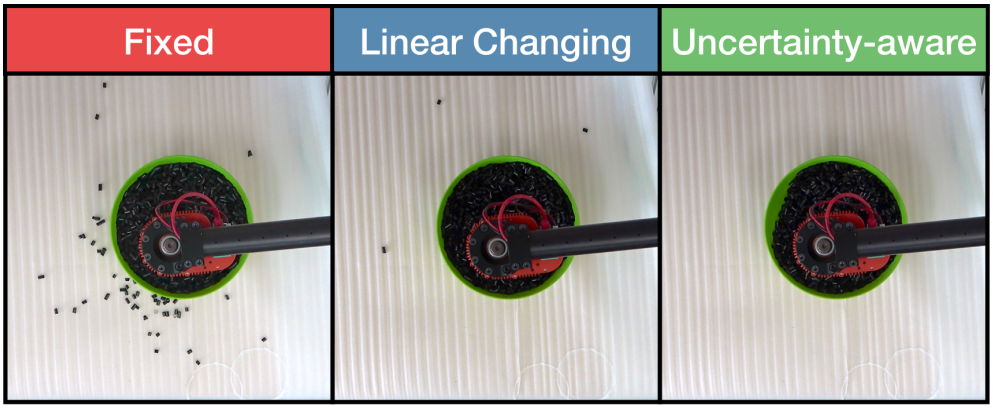

Figure 3.12. This figure illustrates the environmental differences between each method after learning the mixing task. The intensity of the learning process can be evaluated by the density of scattered straw cuts in the environment.

Scooping Task

We conducted 40 trials to learn the scooping task with a 4-DoF robotic arm, demonstrating the scalability of our contact-safe MBRL scheme to more complex tasks. The objective of the scooping task is to follow a human-demonstrated trajectory to scoop and transport straw cuts from one container to another, as shown in Figure 3.13. All three methods ("contact-safe," "fixed," and "linear changing") successfully completed over six scoops within the 40 trial learning process.

Figure 3.13. This figure illustrates the scooping task trajectory. The orange line showcases the scooping trajectory from the front view.

Similar to what we observed in the simulation results shown in Fig. 3.8, our method results in a much more condensed exploration trajectory throughout the 40-trial learning process (see Fig. 3.14). In comparison, both the "fixed" and "linear changing" approaches waved the spoon across the scene during the learning process, which is considered dangerous in such a contact-rich environment. As in the mixing task, we compare the density of scattered straw cuts in the environment after six success scoops after the learning process. Fig. 3.15 demonstrates that our method results in a significantly lower density of scattered straw cuts during the learning process. This demonstrates the method's effectiveness in improving contact safety during the learning process.

3.4. Discussion

3.4.1. Possibilities of Utilizing Model-uncertainty for Various Purposes

In our application, we use model uncertainty to encourage the robot to behave cautiously during the early stages of the learning process. This helps prevent damage during sample scarcity. Although our experimental results show that our method has similar learning efficiency to standard MBRL, it is important to note that our application's workspace and robot system are relatively small and simple. Tasks with larger scales may still suffer from reduced learning efficiency due to cautious exploration.

There are several possible applications for utilizing model-uncertainty. One example is using greedy exploration to encourage the robot to explore more aggressively during periods of high model-uncertainty. This can effectively increase learning efficiency when trying to learn the robot system's dynamics. This approach is particularly effective for learning various tasks by quickly gaining a comprehensive understanding of the robot's dynamics, which can later be enhanced and refined through various assigned tasks.

Another example is the ability to switch between analytic and learned dynamics according to the level of model-uncertainty. This approach allows for a higher success rate in performing the task. During periods of high model uncertainty,

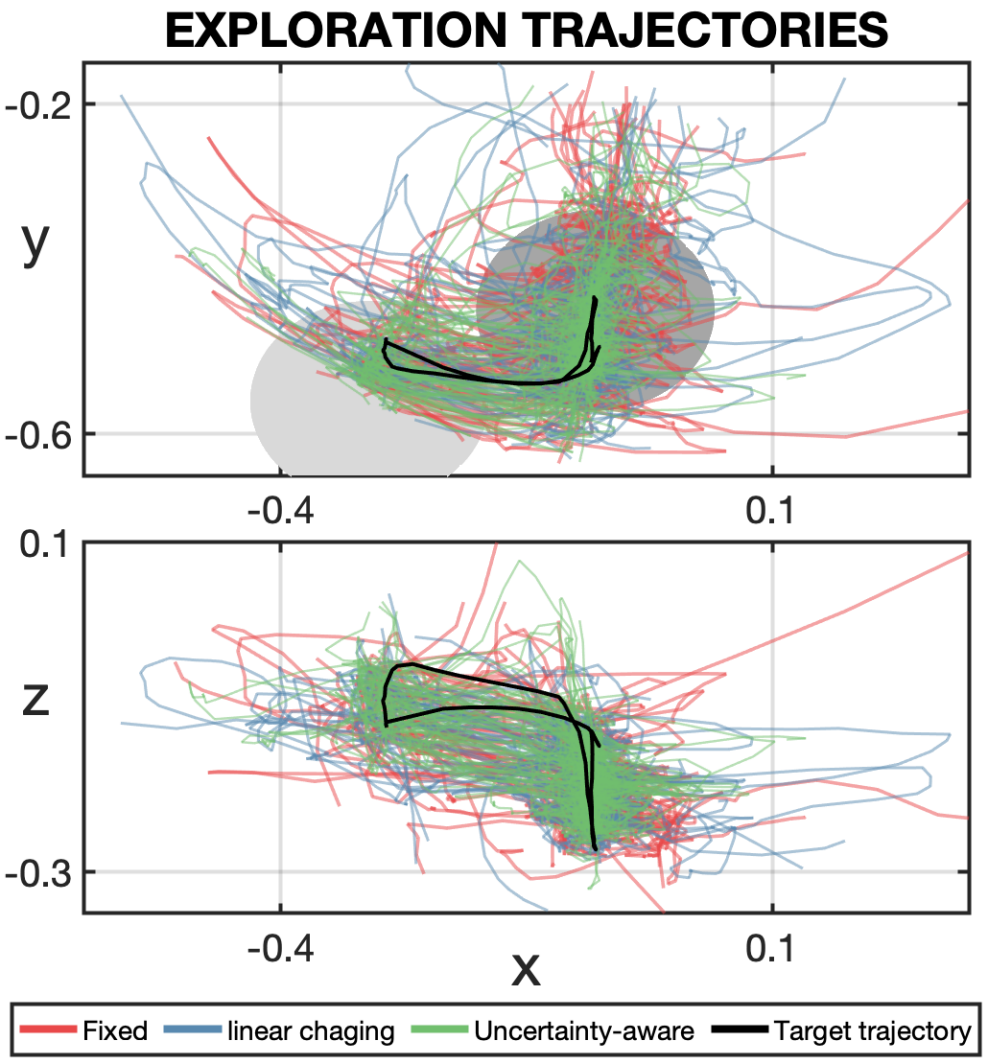

Figure 3.14. This figure illustrates the exploration trajectory of the scooping task from the top view. Each line indicates one of 40th trajectories of each method. The black line is the human-demonstrated trajectory that transport the straw cuts from the darker grey container to the light grey container. The result shows that our method has a much condensed exploration coverage around the target trajectory, resulting a less intensive learning process.

the task can be performed using an inaccurate analytic dynamics model. As learning progresses, the approach can gradually switch to a learned data-driven dynamics model to better adapt to environmental uncertainties that are difficult to capture analytically.

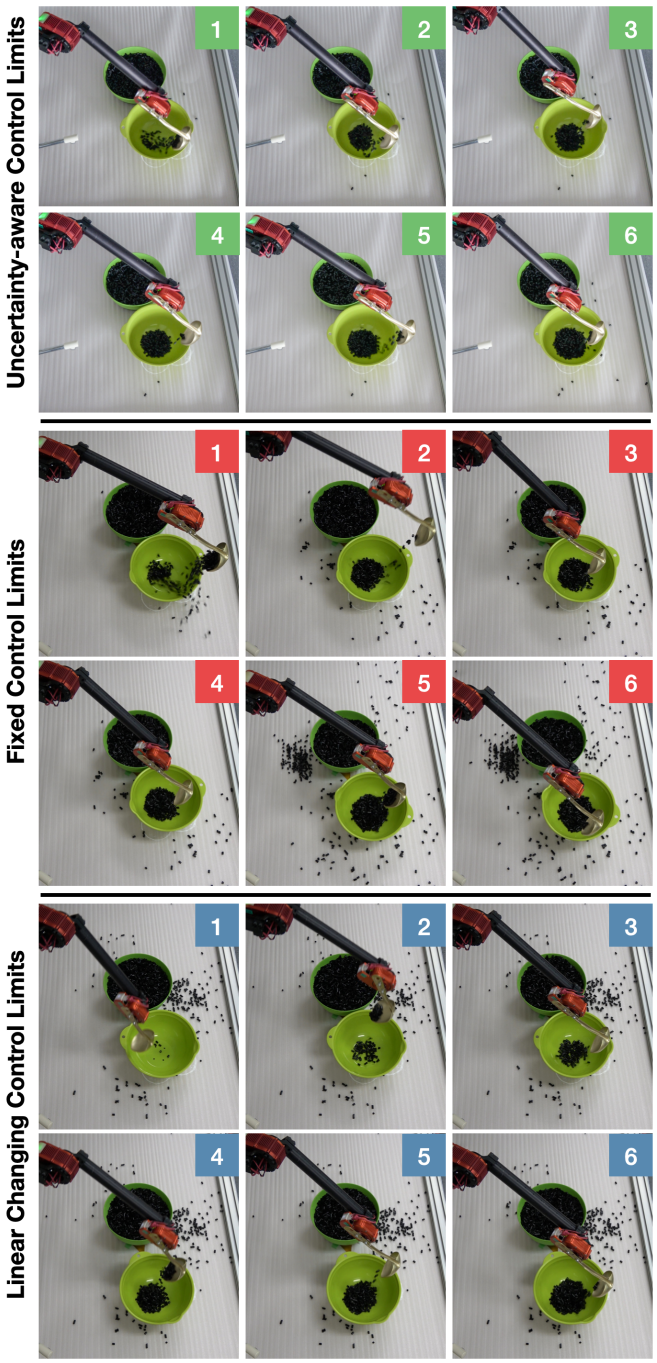

Figure 3.15. This figure illustrates the environment outcome after six success scoops with the presented uncertainty-aware control space, the fixed control space and the linearly changing control space.

3.4.2. The Definition and Attainment of Cautious Behaviors

The cautious behavior we defined is characterized by small actions and low momentum. These characteristics are achieved by adjusting the control space (discrete accelerations) through scaling and translating mechanisms. However, the definition of caution in robotics is vague, and our heuristic method for achieving this objective is specific to velocity-controlled robots.

Therefore, to achieve the objective of having a robot act cautiously during high uncertainty, our method reformulates the robotic state and accommodates such uncertainty. This allows control planning to access the model's uncertainty, but the user must adjust the control space based on their own definition of caution and how they control the robot.

3.5. Conclusion

This application utilizes task-relevant MBRL to improve contact safety during the learning process by incorporating model uncertainty.

To achieve this, we extend the robot state to accommodate model uncertainty and establish an uncertainty-aware control space to associate learning progress with robot behavior. The uncertainty-aware control space encourages the robot to explore cautiously when model uncertainty is high and predictions are inaccurate, and perform the intended task confidently after the model uncertainty reduces.

The effectiveness of improving contact safety during the learning process is validated through particle mixing tasks with simulated and hardware robots, and particle scooping tasks with a hardware robot. The results show that accommodating model uncertainty with the presented task-relevant MBRL can effectively reduce contact intensiveness during the learning process, while achieving a similar learning efficiency compared with standard MBRL.

4. Condensed Robotic State by Incorporating Energy-exchange Dynamics For Learning Spring-loaded Bipedal Robot Walking with Model-based Reinforcement Learning

4.1. Spring-loaded Bipedal Robot and Energy-exchange Dynamics

4.1.1. High-complexity Dynamics of Spring-loaded Bipedal Robot

Bipedal robots are gaining attention from researchers due to their potential for deployment in human-centric environments. Impressive results in bipedal robot locomotion have been achieved with rigid bipedal robots [62–64]. However, highly rigid bipedal robots are not suitable for mimicking human behavior, especially when handling impacts during running or hopping, where human tissues and muscles act elastically [65].

The Spring-loaded Inverted Pendulum (SLIP) model is widely used as a basis

for modern bipedal robots to add compliance [2,66–68]. These compliant bipedal robots utilize springs in their multiple leg linkages to achieve prismatic compliance and potentially resemble humans in anatomy and behavior [69, 70]. This type of robot can handle impacts and traverse different terrains [67, 71], providing the captivating prospect of a non-organic machine performing human-associated tasks. However, the increased compliance of these robots results in a complex dynamic system, making analytical approaches challenging due to the following reasons:

1. These robots utilize a multi-body-with-springs mechanism, which has a different mass distribution than the standard SLIP model. Additionally, their springs exhibit non-linear and non-centripetal behavior under impacts.

2. As a floating-based robot, the dynamics of a bipedal robot largely depend on its contact conditions that provide support during locomotion.

3. Capturing an accurate analytical dynamics model of a spring-loaded bipedal robot is challenging.

4.1.2. Existing Studies of Learning Compliant Locomotion Skills

Several learning-based studies tackled the problem of compliant bipedal locomotion and achieve different bipedal maneuvers. These methods include Model-free Reinforcement Learning (MFRL) [22, 72] and Bayesian Optimization (BO) [23, 73]. However, both MFRL and BO are black-box approaches that are task-specific and data-intensive, requiring virtual scaling like Sim-to-Real to learn each gait. To enhance generalizability, parameterized policies are learned with MFRL and BO to achieve different walking speeds [74, 75], walking heights [76], or both [77]. Nevertheless, these policies are limited to the predefined walking parameters involved in the training, and adding new parameter requires long re-learning. This not only lengthens the learning process, but also limits on-site learning capability.

Probabilistic Model-based Reinforcement Learning (MBRL) is a promising solution for achieving high sample efficiency and generalizability in robotics. MBRL

involves learning a data-driven dynamics model of a robot and can perform different skills using probabilistic Model Predictive Control (pMPC) with different objectives [31,78]. However, MBRL's computational expense [79] has made previous studies on bipedal locomotion mainly conducted in simulation [80,81] or with hardware robots using offline planning [82], which cannot handle environmental changes.

Furthermore, the compliance dynamics of compliant bipedal robots require more training samples and higher dimensional dynamics, making the implementation of MBRL for compliant bipedal locomotion challenging. The increase in sample size and model dimension largely impacts the control frequency because the computation load of pMPC depends on either the training sample size [30] or model dimension [32].

In conclusion, approaches based on MFRL and BO require large amounts of data and have limited generalizability, making on-site learning difficult and require enormous training cycle to learn new skills. Conversely, model-based reinforcement learning (MBRL) approaches are sample-efficient, but require high computational loads that challenge real-time planning ability.Therefore, the key to increasing the feasibility of MBRL implementation on compliant bipedal robots is to explore a simplified and compact dynamics model.

4.1.3. Condensing Robot State Following the Law of Conservation of Energy

The interaction of any robot's mechanical system with its environment can be characterized by energy exchange [83]. In addition, Hutter et al. demonstrated the energy flows between the robot's kinetic, gravitational, and spring's elastic energy [84] during running with a SLIP robotic leg. Therefore, we can express a spring-loaded bipedal robot's CoM dynamics as gravitational and kinetic energy, and view its springs as a energy container. Furthermore, by viewing actuators as the energy source of the robot, we can obtain their provided energy by taking the integral of motion of their Equation-of-Motion (EoM) [85].

With the above in mind, we present an energy-based method that uses MBRL to learn a data-driven state transition dynamics of the robot in a task-relevant

formulation of energy-state, as shown in Fig. 4.1. The robot's energy-state interaction is characterized as Energy-exchange Dynamics (EED). By formulating the robot's state as an energy-state, we effectively reduce the dimensionality of the dynamics model. This relaxes the computation burden of MBRL and improves real-time planning capability.

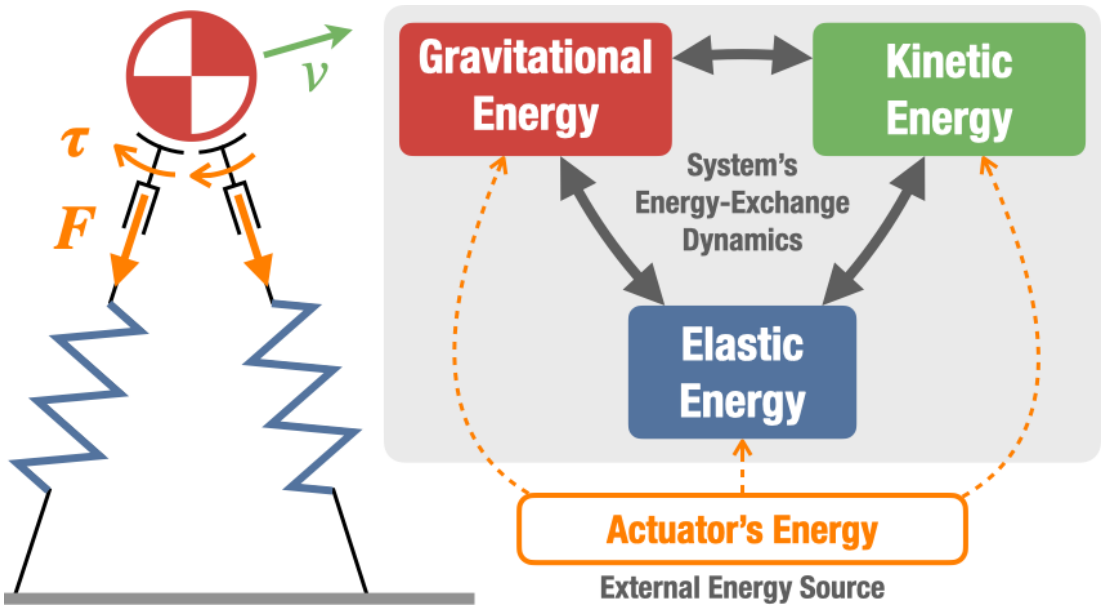

Figure 4.1. This figure illustrates the interaction of Energy-Exchange Dynamics (EED) of a spring-loaded bipedal robot. Energy flows between gravitational, kinetic, and elastic energy, with actuators serving as energy sources that supply energy to the system. This study employs the concept of EED to formulate a condensed energy-state for learning and performing bipedal walking with MBRL.

4.1.4. Dynamics Complexity Reduction for Bipedal Robot Dynamics

As a floating-based robot, a bipedal robot's dynamics are highly dependent on the changing contact conditions while performing locomotion skills. For instance, the robot behaves differently between single support and double support phases. To handle such dynamic dissimilarity, a common approach in bipedal locomotion is the Finite State-Machine (FSM) [86]. This method decomposes the locomotion task into pre-defined state phases for switching to the proper controller of that

phase.

A similar concept is used for MBRL applications. The dynamics are decomposed over phases and modeled with dedicated models to cope with dynamic dissimilarity between phases. This can potentially reduce model complexity.

4.2. Methodology

4.2.1. Spring-loaded Inverted Pendulum Bipedal Robot Model

In this section, we use a simple Spring-loaded Inverted Pendulum (SLIP) formulated planer bipedal system to explain our method.

Consider an m-kg point-mass planer bipedal robot system with k spring-loaded actuators, characterized by a spring constant matrix $\boldsymbol{K} \in \mathbb{R}^{k \times k}$. We assume the following variables are available from sensor readouts: The system's Center-of-Mass (CoM) height h, CoM velocity $\boldsymbol{v} \in \mathbb{R}^V$, spring deflections $\boldsymbol{\Theta}_S \in \mathbb{R}^k$. and all actuated joints' position and velocity $\boldsymbol{\Theta}, \dot{\boldsymbol{\Theta}} \in \mathbb{R}^k$ that includes spring deflections.

4.2.2. Condensed Robot State for Dimensionality Reduction

MBRL with LGM-FF dynamics, as shown in Eq. 2.5, can be computationally expensive when dealing with high-dimensional dynamics. However, high-dimensional dynamics are necessary to accommodate the additional compliance dynamics added to the bipedal robot. Therefore, to make MBRL implementation on compliant bipedal robots more feasible, a reduced-dimensional robotic state that is sufficient to express robot dynamics and effective for locomotion tasks is needed.

Inspired by the law of conservation of energy [87], which states that the energy within a closed system cannot be generated or destroyed, we reformulated an energy-state for MBRL that expresses the dynamics of the robot's CoM as energy, views its elastic components as temporary energy containers, and characterizes their interaction as energy exchange. By viewing elastic components

as temporary energy containers, this reformulation allows us to treat all springs as a single container and avoid modeling them individually [88]. Moreover, by treating the robot's actuators as energy sources to the system, we can achieve further reduction in dimensionality by concealing state entries that can be implicitly expressed.

Center-of-Mass's Dynamics Reformulation with Its Energy in Physics

Consider a standard robotic state $\boldsymbol{x} \in \mathbb{X}$ that includes entries for both robot configuration and motion:

$$\boldsymbol{x} = [\underbrace{\boldsymbol{\Theta}^\top,\ \dot{\boldsymbol{\Theta}}^\top,\ \boldsymbol{\Theta}_S^\top}_{\text{configuration}},\ \overbrace{h,\ \boldsymbol{v}^\top}^{\text{motion}}]^\top \in \mathbb{X} \subset \mathbb{R}^{3k+V+1}, \tag{4.1}$$

where configuration entries describe the robot's configuration, while motion entries are necessary to express the dynamics of the robot's center of mass. By reformulating the dynamics of the robot's center of mass (CoM) in terms of potential and kinetic energy, we can treat the robot's elastic components as temporary energy storage in the system's energy exchange dynamics. Therefore, we reformulate a robot's energy-state, $\tilde{\boldsymbol{z}}$, as follows:

$$\tilde{\boldsymbol{z}} = [\underbrace{\boldsymbol{\Theta}^\top,\ \dot{\boldsymbol{\Theta}}^\top}_{\text{configuration}},\ \overbrace{\boldsymbol{E}^\top}^{\text{energy-exchange}}\ ]^\top \in \mathbb{R}^{2k+V+2}, \tag{4.2}$$

where $\boldsymbol{E} \in \mathbb{R}^{V+2}$ contains potential, kinetic and elastic energies:

$$\boldsymbol{E} = \begin{bmatrix} E_g \\ \boldsymbol{E}_k \\ E_S \end{bmatrix}, \tag{4.3}$$

where $E_g \in \mathbb{R}$ is potential energy, $\boldsymbol{E_k} \in \mathbb{R}^V$ is kinetic energy and $E_S \in \mathbb{R}$ is elastic energy. These energies are obtained via:

$$E_g = \mathrm{m}|\boldsymbol{g}|h, \tag{4.4}$$

$$\boldsymbol{E_k} = 0.5\mathrm{m}\boldsymbol{v}^{\circ 2}, \tag{4.5}$$

$$E_S = 0.5\boldsymbol{\Theta}_S^\top \boldsymbol{K}\boldsymbol{\Theta}_S, \tag{4.6}$$

where $\boldsymbol{g}$ is the gravity vector and $\circ$ denotes element-wise operation. Without an external energy source, the system's total energy remain constant; that is,

$$\mathbf{1}^\top \boldsymbol{E} = \text{const.}, \tag{4.7}$$

where $\mathbf{1}$ is a vector of ones.

By comparing Eq. 4.1 and Eq. 4.2, we can see that reformulating the equation in terms of energy-state achieved a $k-1$ dimension reduction to the robotic state.

Actuators' Dynamics Reformulation with the Energy Conservation Equation

Consider an Equation-of-Motion (EoM) of the robot's actuators:

$$\boldsymbol{\tau} = \boldsymbol{J}\ddot{\boldsymbol{\Theta}} + \boldsymbol{K}\boldsymbol{\Theta} + \boldsymbol{D}\dot{\boldsymbol{\Theta}}, \tag{4.8}$$

where $\boldsymbol{\tau} := [\tau_1, ..., \tau_k]^\top \in \mathbb{R}^k$ represents the torque of all actuators, $\boldsymbol{J} \in \mathbb{R}^{k\times k}$ is the inertia matrix, and $\boldsymbol{D} \in \mathbb{R}^{k\times k}$ is the damping matrix. By integrating the equation of motion of the actuators, we can obtain the energy provided by the actuators, denoted as $E_{act} \in \mathbb{R}$:

$$E_{act} = \int \boldsymbol{\tau}^\top \dot{\boldsymbol{\Theta}} dt = \frac{1}{2}(\boldsymbol{J}\dot{\boldsymbol{\Theta}})^\top \dot{\boldsymbol{\Theta}} + \frac{1}{2}(\boldsymbol{K}\boldsymbol{\Theta})^\top \boldsymbol{\Theta} + \int (\boldsymbol{D}\dot{\boldsymbol{\Theta}})^\top \dot{\boldsymbol{\Theta}} dt,. \tag{4.9}$$

Assuming that the actuators are the only energy source of the robotic system, the system's energy conservation equation is as follows:

$$\mathbf{1}^\top \boldsymbol{E} = E_{act} + \text{Const.} = \frac{1}{2}(\boldsymbol{J}\dot{\boldsymbol{\Theta}})^\top \dot{\boldsymbol{\Theta}} + \frac{1}{2}(\boldsymbol{K}\boldsymbol{\Theta})^\top \boldsymbol{\Theta} + \text{Const.},. \tag{4.10}$$

This equation shows that the robot's system energy $\boldsymbol{E}$ is a function of the actuator's position $\boldsymbol{\Theta}$ and velocity $\dot{\boldsymbol{\Theta}}$. Since all entries of the system energy are obtained from the accessible CoM's state (h, $\boldsymbol{v}$) and the spring's deflection ($\boldsymbol{\Theta}_S$), the actuator joints' position $\boldsymbol{\Theta}$ and velocity $\dot{\boldsymbol{\Theta}}$ are interdependent. As a result, we can implicitly express and conceal one another. In this work, our design of the energy-state $\boldsymbol{z} \in \mathbb{Z}$ conceals velocity entries and is formulated as follows:

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{\Theta} \\ \boldsymbol{E} \end{bmatrix} \in \mathbb{R}^{k+V+2}, \tag{4.11}$$

which reduces k dimensions from the state $\tilde{\boldsymbol{z}}$, and reduces $2k-1$ dimensions from the standard robot state in Eq. 4.1.

Learning Energy-exchange Dynamics with Model-based Reinforcement Learning

With the reformulated energy-state $\boldsymbol{z}$, MBRL learns an Energy-exchange Dynamics (EED) f_e for robot control via LGM-FF from Eq. 2.5:

$$p\left(\boldsymbol{z}_{t+1}\right) = f_e\left(p\left(\boldsymbol{z}_{t+1}\right), \boldsymbol{u}_t\right) + \epsilon, \tag{4.12}$$

where $\boldsymbol{u}_t \in \mathbb{U}$ is the user-defined control signal based on how actuators are controlled and ϵ is the system noise that follows Gaussian distribution.

4.2.3. Task-decomposed Dynamics for Complexity Reduction

Although modeling the EED significantly reduces the model dimension for a compliant bipedal robot, a high complexity model is required to handle the dynamic differences caused by changing contact conditions on a floating-based robot.

The Finite State Machine (FSM) [86] breaks down the locomotion task into distinct phases based on different contact conditions and switches control objectives between these phases. Drawing inspiration from the FSM, we introduce the task-decomposed dynamics model which models a lower complexity dynamics model for each task phase and has distinct control objectives between phases. In addi-

tion, the probabilistic Model Predictive Control (pMPC) planning in each phase uses the corresponding task-decomposed dynamics model to leverage prediction and solve the optimization problem.

Specifically, each of N_p predefined task phases $\mathcal{P} := \{\mathcal{P}_p\}_{\forall p=1...N_p}$, we model a distinct EED dynamics model:

$$f_e := \{f_e^p(\cdot)\}_{\forall p=1,...,N_p}, \tag{4.13}$$

where the task phase during operation is determined by the energy-state $\boldsymbol{z}_t$ at that moment.

4.2.4. Walking with Probabilistic Model Predictive Control

Walking Gait Parameter

The leg length of the swing and support legs, as well as their respective orientation in world-space, are denoted by L_{swg}, L_{sup}, ψ^w_{swg}, and ψ^w_{sup}, as shown in Fig. 4.2. The values of L and ψ^w are assumed to be obtainable from the actuator $\boldsymbol{\Theta}$ and spring's positions $\boldsymbol{\Theta}_S$.

The walking gait is then described by four parameters: the stance leg length L_{stan}, the lifted leg length L_{lift}, the desired stride angle θ_{stride}, and the desired velocity of the center of mass $\boldsymbol{v}_p \in \mathbb{R}^V$ at the peak.

Walking Gait Phase Definition and Their Motions

The walking gait can be divided into two phases, each with a distinct objective: Double-support (DS) and Single-support (SS), $\mathcal{P} = \{DS, SS\}$. During DS, the rear leg serves as the support leg. The transition between the DS and SS phases is determined by the ground reaction forces of the feet, F_{swg} and F_{sup}, which have hard thresholds. Variable definitions for our application are detailed in Section 4.3.3.

The target gait motion is shown in Fig. 4.2, with the following requirements:

- DS phase: The angle between the two legs matches θ_{stride}, and the swing leg's

length matches the stance leg's length. We denote these requirements as

$$\boldsymbol{R}^{DS} = \left\{\psi^{w}_{swg} = -0.5\theta_{stride}, \psi^{w}_{sup} = 0.5\theta_{stride}, L_{swg} = L_{stan}\right\}. \tag{4.14}$$

- SS phase: The support leg's length matches the stance leg's length, such that the system acts as an inverted pendulum. This requirement is denoted as:

$$\boldsymbol{R}^{SS} = \{L_{sup} = L_{stan}\}. \tag{4.15}$$

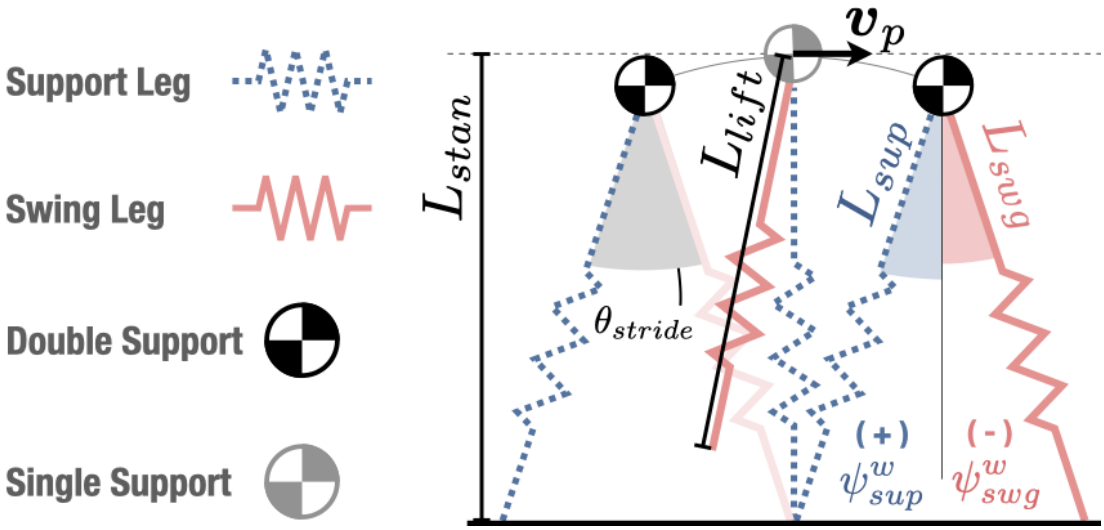

Figure 4.2. This figure illustrates the walking gait with the swing/support leg definition and gait parameters.

Walking Trajectory for Double-support Phase

To achieve the desired gait, the system must possess the target energy $E^{\star} \in \mathbb{R}$, in accordance with the law of conservation of energy and the provided gait parameters:

$$E^{\star} = E^{\star}_{g} + \mathbf{1}^{\top}\boldsymbol{E}^{\star}_{k} = \mathrm{m}|\boldsymbol{g}|L_{stan} + 0.5\mathrm{m}\mathbf{1}^{\top}\boldsymbol{v}_{p}^{\circ 2}, \tag{4.16}$$

where $E^{\star}_{g} \in \mathbb{R}$ and $\boldsymbol{E}^{\star}_{k} \in \mathbb{R}^{V}$ are the target gravitational and kinetic energy, respectively. $\boldsymbol{v}p$ is the desired CoM velocity at the peak. Therefore, the objective

is to provide a trajectory that pMPC can use to find optimal controls that supply the system with the necessary energy to progress forward.

As the target energy remains constant, the reference trajectory at time-step t is:

$$E^{\star}_{g,t} = E^{\star}_{g} = \mathrm{m}|\boldsymbol{g}|L_{stan}, \tag{4.17}$$

$$\boldsymbol{E}^{\star}_{k,t} = \boldsymbol{E}^{\star}_{k} = 0.5\mathrm{m}\boldsymbol{v}^{\circ 2}_{p}. \tag{4.18}$$

and the corresponding reference trajectory of the DS phase is

$$\boldsymbol{T}^{DS}_{t} = \left\{E^{\star}_{g,t}, \boldsymbol{E}^{\star}_{k,t}\right\}. \tag{4.19}$$

Walking Trajectory for Single-support Phase

Assuming that the system has gained energy after the DS phase, the objective of the SS phase is to execute a stable leg swing with the swing leg to progress to the DS phase.

To synchronize the inverted pendulum's swing motion, the swing leg's references, $L^{\star}_{swg}$ and $\psi^{\star}_{swg}$, must be a function of the support leg orientation ψ^{w}_{sup}. As long as the swing leg's terminal configuration matches the defined stance configuration of the DS phase, the swing leg's intermediate trajectory can be chosen according to the desired gait pattern.

We denote the swing leg's trajectory as follows:

$$L^{\star}_{swg,t} = S_{L}\left(\psi^{w}_{sup,t}\right), \tag{4.20}$$

$$\psi^{\star}_{swg,t} = S_{\psi}\left(\psi^{w}_{sup,t}\right), \tag{4.21}$$

where $S_{L}, S_{\psi} : \mathbb{R} \to \mathbb{R}$ are mappings between the support leg orientation ψ^{w}_{sup} and the reference swing leg's trajectories. Our design of swing leg trajectory during the SS phase is detailed in Section 4.3.3. Therefore, the reference trajectory of the SS phase is

$$\boldsymbol{T}^{SS}_{t} = \left\{L^{\star}_{swg,t}, \psi^{\star}_{swg,t}\right\}. \tag{4.22}$$

4.2.5. Enhance Reliability with Energy-state-aware Control Space

Expanding the control space can potentially increase the exploration time needed to find the optimal control. However, performing bipedal locomotion skills requires a high control frequency (short pMPC planning time) to ensure stability. Therefore, it is necessary to have a control space that filters out unreliable controls but does not limit the robot's capabilities. These unreliable controls can be identified by comparing the current robot state and its corresponding reference trajectory. Thus, we present an energy-state-aware control space that constrains the exploration control space $\mathbb{U}$ of pMPC to reduce the chance of pMPC producing unreliable controls.

At each time-step t, the modification is based on system's state $\boldsymbol{x}_t$, the reference trajectory $\boldsymbol{x}_t^\star$, the user's prior knowledge of which control signals are more reliable:

$$\mathbb{U}_t' = g_e\left(\boldsymbol{z}_t, \boldsymbol{T}_t^p, \mathbb{U}\right), \tag{4.23}$$

where $g : \mathbb{Z} \times \mathbb{U} \to \mathbb{U}' \subset \mathbb{U}$ modifies the control space.

We summarize an H-horizon pMPC with energy-state-aware control space as follows:

$$\left.\begin{aligned}
&\text{minimize}_{\boldsymbol{u}^\star} \ \mathcal{L}\left(p\left(\boldsymbol{z}\right)\right) = \sum_{k=2}^{H+1} \mathbb{E}\left[\ell\left(\hat{\boldsymbol{z}}_k\right) | p\left(\hat{\boldsymbol{z}}_k\right)\right] \\
&\text{subject to} \quad p\left(\hat{\boldsymbol{z}}_{k+1}\right) = f_e\left(p\left(\hat{\boldsymbol{z}}_k\right), \hat{\boldsymbol{u}}_k\right) \\
&\qquad\qquad p\left(\hat{\boldsymbol{z}}_k\right) \sim \mathcal{N}\left(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right), \qquad k = 1, ..., H+1 \\
&\qquad\qquad \hat{\boldsymbol{u}}_k \in \mathbb{U}_k', \qquad k = 1, ..., H \\
&\qquad\qquad \mathbb{U}_k' = g_e\left(\boldsymbol{z}_k, \boldsymbol{T}_t^p, \mathbb{U}\right)
\end{aligned}\right\}, \tag{4.24}$$

where $\hat{\boldsymbol{z}}_1, ..., \hat{\boldsymbol{z}}_1$ are the predictive state rollouts with initial state $p\left(\hat{\boldsymbol{z}}_1\right) = p\left(\hat{\boldsymbol{z}}\right)$, $\mathcal{L}\left(\cdot\right)$ is the finite-horizon loss, and $\ell\left(\cdot\right)$ is the immediate loss.

4.3. Experimental Evaluation with Spring-loaded Bipedal Robot Walking

4.3.1. Robot Configuration

Hardware Robot Configuration

In our study, we utilized a spring-loaded bipedal robot [2] with lightweight legs that have parallel-linkage structures, as shown in Fig. 4.3(a). The robot features four spring-loaded actuators that drive its lever and thigh links for prismatic compliance. Its stiffness-to-weight ratio is approximately ten times less than that of a widely used compliant bipedal robot, Cassie [89]. The legs' lightness allowed us to approximate the robot's Center-of-Mass (CoM) with its hip joint. We controlled the actuators' velocity at 40 Hz with a limit of ± 6 rad/s. The robot was attached to a rotational boom and constrained in the sagittal plane. For conversion between local, task, and world-space, please refer to Appendix A.2. We computed the hardware results on the on-robot computer with an i7-4700EQ CPU.

Simulated Robot Configuration

To mimic the real robot, a simulated replica was built in Mujoco [90], as shown in Fig. 4.3. Friction and damping were added to all joints. The simulated robot was constrained in the sagittal plane (xz-plane) and controlled at a frequency of 40Hz, which matched the real robot. Simulated experiments were conducted using an Apple M1 Max-equipped computer.

4.3.2. Energy-state Definition

Following Eq. (4.11), we define the robot's energy-state as:

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{\Theta} \\ \theta_T \\ \boldsymbol{E} \end{bmatrix} \in \mathbb{R}^9, \tag{4.25}$$

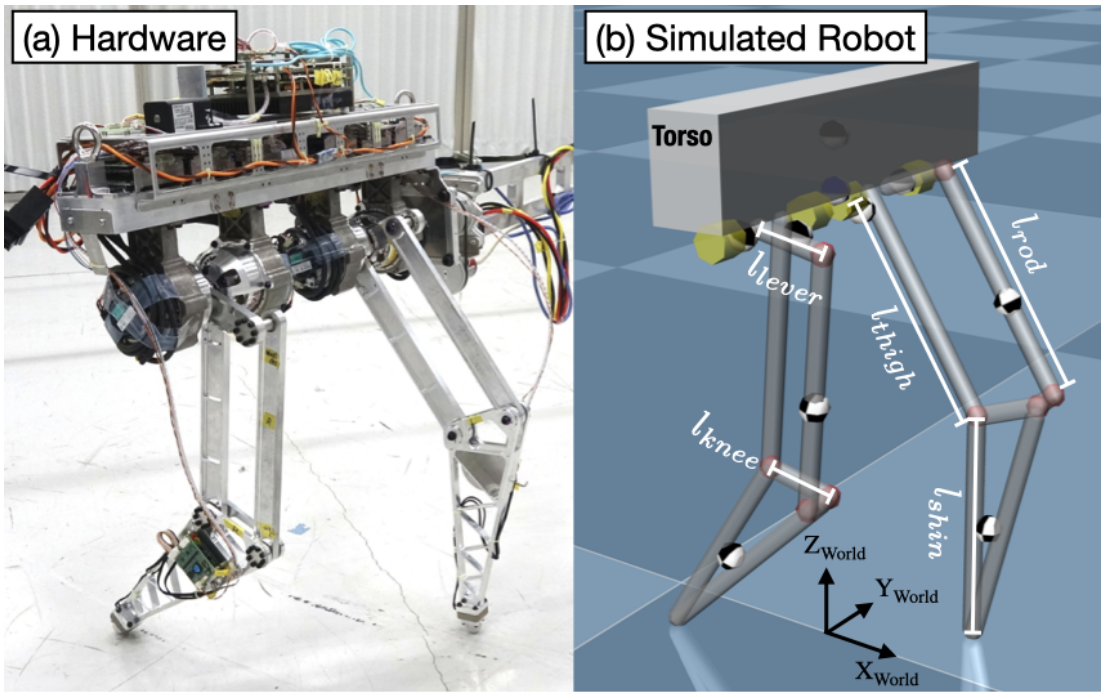

Figure 4.3. This figure shows the robot used in the study: (a) the hardware spring-loaded planar bipedal robot, and (b) its replica in Mujoco. Corresponding parameters are provided in Table 4.1.

where $\theta_T \in \mathbb{R}$ is added to capture world-space leg orientation with definitions of $\boldsymbol{\Theta}$ and $\boldsymbol{E}$ are as follows:

$$\boldsymbol{\Theta} = [\phi_{swg}, \psi_{swg}, \phi_{sup}, \psi_{sup}]^\top \in \mathbb{R}^4, \tag{4.26}$$

$$\boldsymbol{E} = \left[E_g, E_R, \frac{\dot{P}_T}{|\dot{P}_T|} E_T, E_S\right]^\top \in \mathbb{R}^4, \tag{4.27}$$

where ϕ being the inner angle of each leg corresponding to the leg length L; E_g is the gravitational energy; E_S is the total elastic energy of all springs; ER and E_T are the kinetic energies obtained from radial and tangential CoM velocities $(\dot{P}_R, \dot{P}_T)$, referring to the support leg shown in Fig. 4.4. Directions are added to the tangential kinetic energies for additional information (positive when walking forward). Both ϕ and ψ include spring deflections and are configured in task-space to simplify the walking control problem. They are obtained from actuator position $\boldsymbol{\Theta}$ and spring deflections $\boldsymbol{\Theta}_s$, as detailed in Appendix A.2.

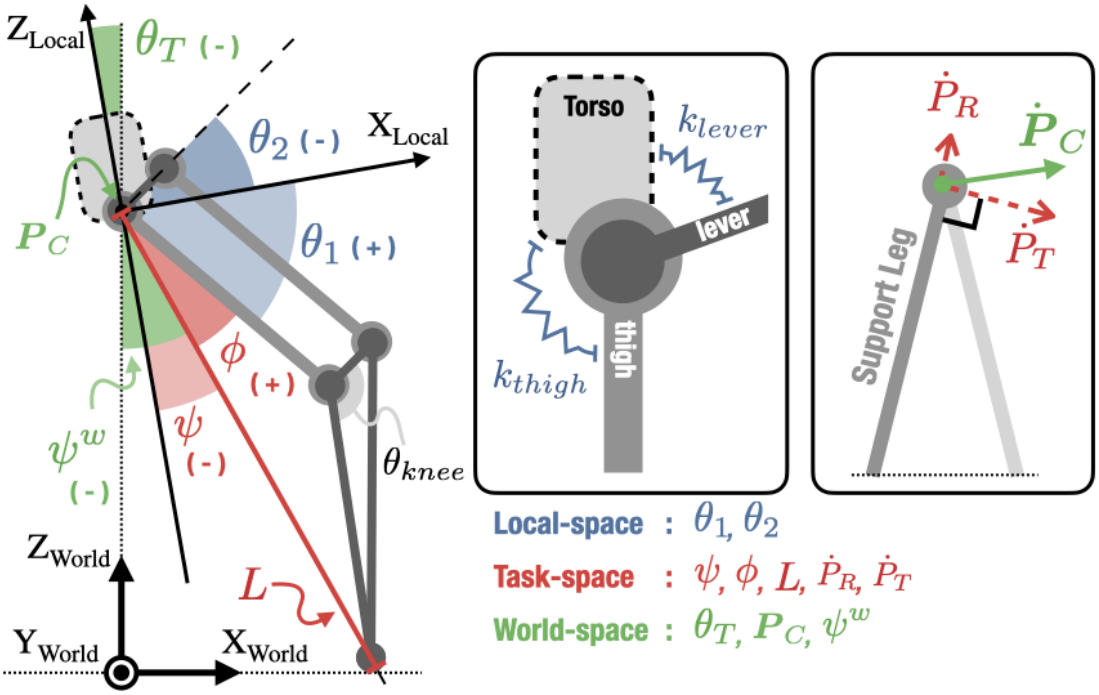

Figure 4.4. This figure shows the robot's configuration with corresponding parameters and variables definition provided in Table 4.1 and Table 4.2, respectively.

Parameter	Value	Unit
Gravity ($\boldsymbol{g}$)	9.807	m/s^2
Mass (m)	11.25	kg
l_{thigh}, l_{shin}, l_{rod}	0.25	m
l_{lever}, l_{knee}	0.0705	m
θ_{knee}	110	Deg
k_{thigh}, k_{roc}	150	Nm/rad

Table 4.1. Parameters of the robot used in this research.

4.3.3. Control Objectives

Parameters for the Desired Walking Gait

Stance leg length L_{stan} and lifted leg length L_{lift} were measured at joint angles $\phi_{min} = 14°$ and $\phi_{max} = 37°$, with a target stride angle of $\theta_s = 28°$ and a target velocity of $\boldsymbol{v}_p = 0.4$ m/s in the x-direction at peak. Therefore, the target energy

Space	Variables	Description
Local	θ_1	Joint 1 position. Actuator 1 position with spring deflection.
	θ_2	Joint 2 position. Actuator 2 position with spring deflection.
Task	ψ	Leg's orientation. Obtained from θ_1, θ_2.
	ϕ	Leg's inner angle. Obtained from θ_1, θ_2.
	L	Leg's length. Obtained from ϕ.
World	θ_T	Torso orientation. Obtained from IMU.
	$\mathbf{P}_C$	Center-of-mass position. Approximated with the hip joint. Obtained from support leg.
	ψ^w	Leg's orientation in world-space. Obtained from ψ and θ_T

Table 4.2. System variables in local-, task-, and world-space.

was obtained as

$$E^\star = \mathrm{m}|\boldsymbol{g}|L_{stan} + 0.5\mathrm{m}\mathbf{1}^\top \boldsymbol{v}_p^{\circ 2} \approx 54.426 \text{ J}. \tag{4.28}$$

The transition between phases is determined by the ground reaction forces of the feet, $\boldsymbol{F}_{swg}$ and $\boldsymbol{F}_{sup}$, with the following conditions: a) DS→SS if $|\boldsymbol{F}_{sup}| \leq 5$N, and b) SS→DS if $|\boldsymbol{F}_{swg}| \geq 20$N. $\boldsymbol{F}_{swg}$ and $\boldsymbol{F}_{sup}$ are only used to identify phase transitions and are not used in the pMPC planning. The difference in phase transition thresholds prevents recursive phase switching caused by sensing errors.

Reference Swing Leg Trajectory for the Single-support Phase

We designed a human-inspired reference trajectory for our application by incorporating a recent human gait analysis [91] to execute a stable leg swing. To achieve a smooth motion, the reference swing leg trajectory was heuristically fitted to the human gait [91] using multiple sigmoid functions, as shown by the black dashed lines in Fig. 4.5. The settling angle, θ_{set}, marks the point at which the swing leg is prepared to touch down by maintaining the target configuration ($L^\star_{swg} = L_{stan}$ and $\psi_{sup} - \psi^\star_{swg} = \theta_{stride}$).

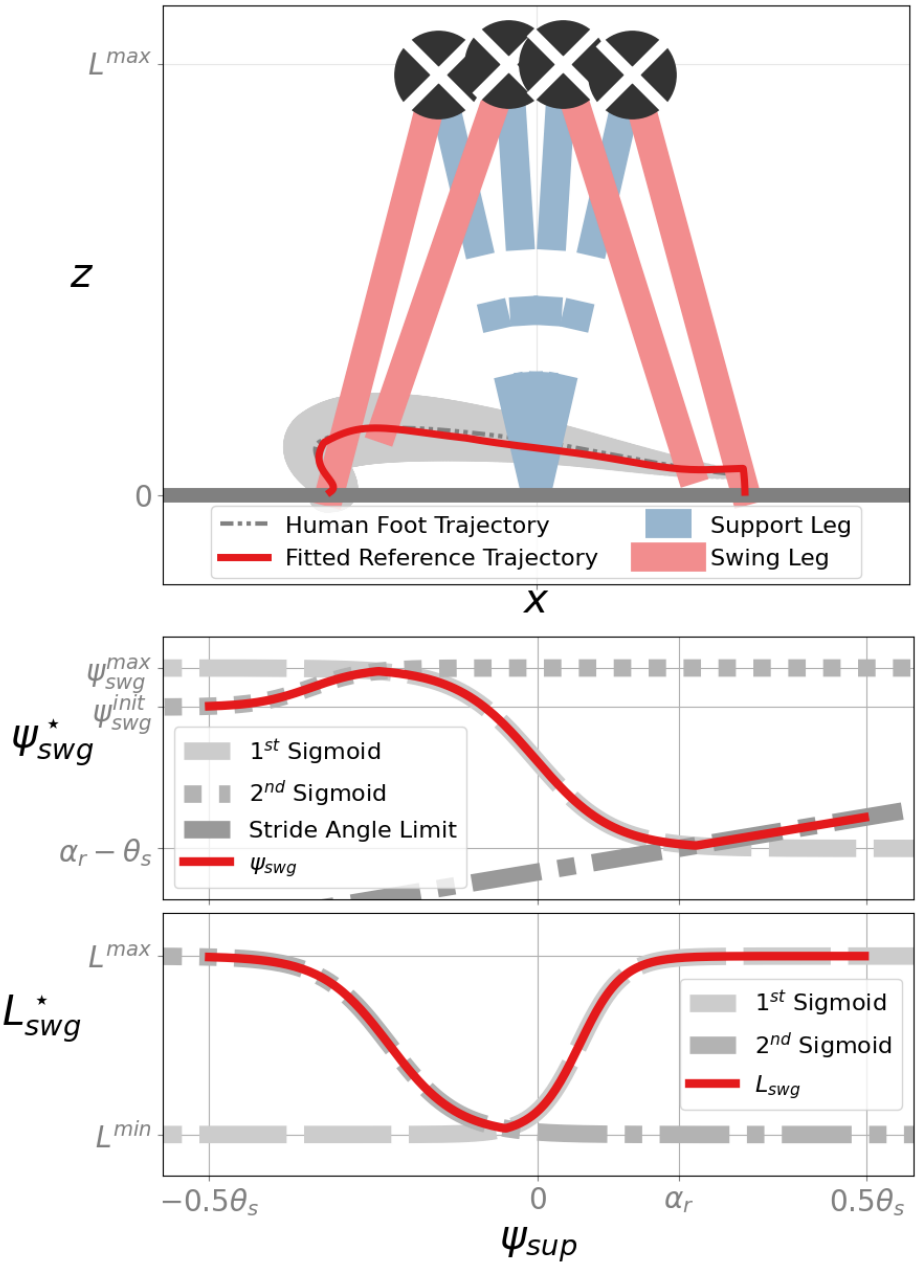

Figure 4.5. This figure illustrates the fitting of swing leg trajectory using multiple sigmoid functions. The top figure shows the swing motion in the xz-plane, with the foot position's mean indicated by a grey dashed line (provided in the analysis [91]), and the ± 2 standard deviation area shown in grey shade. The red line indicates the fitted trajectory. The middle and bottom figures show the trajectory fitted by the sigmoid functions for leg length and swing angle, respectively.

Control Strategy and Task Objective of Probabilistic Model Predictive Control

The following designs are associated with the reference trajectories $\boldsymbol{T}_t^{DS}$ and $\boldsymbol{T}_t^{SS}$ from Eq. (4.19) and Eq. (4.22), as well as the requirements $\boldsymbol{R}^{DS}$ and $\boldsymbol{R}^{SS}$ from Section 4.2.4.

1. **Control Strategy for Double-support Phase:** A kicking motion is expected to provide the system with the necessary energy. Therefore, the control is defined as:

$$\boldsymbol{u}^{DS} := \left[\dot{\phi}_{sup}\right] \tag{4.29}$$

   The immediate loss for the pMPC is set accordingly:

$$\ell_{DS}\left(\boldsymbol{x}_k\right) := -\exp\left(-|E^\star - \mathbf{1}^\top \boldsymbol{E}_k|\right), \tag{4.30}$$

   where the directional kinetic energy in Eq. (4.27) encourages gaining energy by kicking forward. Following $\boldsymbol{R}^{DS}$, positions of ϕ_{sup}, ψ_{sup}, and ψ_{swg} are fixed via PD controllers.

2. **Control Strategy for Single-support Phase:** The goal is to perform a leg swing while keeping the body orientation at $-5°$ to avoid exceeding the actuator limits due to hardware constraints. PD controllers hold the support leg length using $\boldsymbol{R}^{SS}$, with an additional $\psi_{sup} = (-5° - \theta_T)$ to counteract the torso tilt. Meanwhile, using pMPC, the optimal swing leg control, $\boldsymbol{u}^{SS} := \left[\dot{\phi}_{swg}, \dot{\psi}_{swg}\right]^\top$, is obtained given the reference state. An immediate loss is set to:

$$\ell_{SS}\left(\boldsymbol{x}_k\right) := -0.5\exp\left(-|\phi_k^\star - \phi_{swg,k}|\right) - 0.5\exp\left(-|\psi_k^\star - \psi_{swg,k}^w|\right), \tag{4.31}$$

   where the conversion from $L_{swg,k}^\star$ to $\phi_{swg,k}^\star$ and obtaining the world-space leg orientation ψ^w is provided in Appendix A.2.

Energy-state-aware Control Space

The purpose is to provide a reliable control space to constrain pMPC exploration. Since the reference state of pMPC $(\phi^{\star}_{swg}, \psi^{\star}_{swg}, E^{\star})$ is already known, we can determine a theoretical control relative to these references and establish a control range for pMPC. Specifically, we modify the control space by adding a range of velocities $[-\delta, \delta]$ above the theoretical velocities $u^{\star}_i$:

$$\mathbb{U}'_t := \{u_i \in [u^{\star}_i - \delta_i, u^{\star}_i + \delta_i]\}_{\forall u_i \in \boldsymbol{u}_t} \subset \mathbb{U}. \tag{4.32}$$

1. **Double-support Phase:** For all steps, the theoretical kicking velocity $\dot{L}sup^{\star}$ is obtained from the required radial kinetic energy $E^{\star}_R$:

$$E^{\star}_R = \max\left(E^{\star} - E_g - E_T, 0\right), \tag{4.33}$$
$$\dot{L}^{\star}_{sup} = \sqrt{2\mathrm{m}^{-1}E^{\star}_R}, \tag{4.34}$$

   where Eq. (4.33) calculates the energy that the radial kinetic energy can provide, and Eq. 4.34 converts $E^{\star}_R$ into the theoretical velocity, which is then converted to $\dot{\phi}^{\star}_{sup}$ via Eq. (A.12).

2. **Single-support Phase:** For all steps, the theoretical velocities $\dot{\psi}^{\star}_{swg}$ and $\dot{\psi}^{\star}_{sup}$ are obtained by scaling the distance between the current state and the reference trajectory. Specifically, we use the following equations:

$$\dot{\phi}^{\star}_{swg,t} = \alpha\left(\phi^{\star}_{swg,t} - \phi_{swg,t}\right), \tag{4.35}$$
$$\dot{\psi}^{\star}_{swg,t} = \alpha\left(\psi^{\star}_{swg,t} - \psi_{swg,t}\right), \tag{4.36}$$

   here, α is a scaling factor, and we set the control frequency to $\alpha = 40$ for an exact conversion from position to velocity.

Based on the theoretical velocities at each time-step, we define the energy-

state-aware control space for the DS and SS phase as:

$$\mathbb{U}'_{DS} = \left\{ \dot{\phi}_{sup} \in \left[\dot{\phi}^{\star}_{sup} - \delta, \dot{\phi}^{\star}_{sup} + \delta \right] \right\}, \tag{4.37}$$

$$\mathbb{U}'_{SS} = \begin{Bmatrix} \dot{\phi}_{swg} \in \left[\dot{\phi}^{\star}_{swg} - \delta, \dot{\phi}^{\star}_{swg} + \delta \right] \\ \dot{\psi}_{swg} \in \left[\dot{\psi}^{\star}_{swg} - \delta, \dot{\psi}^{\star}_{swg} + \delta \right] \end{Bmatrix}, \tag{4.38}$$

where δ is defined as 1.2 rad/s for simulated robots and 0.6 rad/s for hardware. This allows pMPC to explore approximately 20% and 10% of the actuator's full capability, respectively.

4.3.4. Model-based Reinforcement Learning Process

In our experiments, MBRL repeated trials until reaching the target sample duration. In each trial, steps were repeated at a rate of 40 Hz until any termination condition was satisfied. During each step, the ahead pMPC scheme was applied to find the optimal control while alleviating control delay, visit Section 2.3.2 for details. Specifically, at time step t, the pMPC finds the optimal control sequence $\boldsymbol{u}_{\star} = [\hat{\boldsymbol{u}}_1, ..., \hat{\boldsymbol{u}}_H]$ of length $H = 3$ based on the dynamics model of that phase and a predictive state $\hat{\boldsymbol{x}}_{t+1}$, as obtained via Eq. (4.12). The first control signal $\hat{\boldsymbol{u}}_1$ is then assigned as the one-step-ahead control that will be applied to the system at time step $t + 1$. After each trial is completed, the MBRL model is updated using samples collected from all previous trials. The learning process is summarized in Fig. 4.6.

Termination conditions for simulation trials include reaching a target distance of 20 m, falling over ($P_z \leq 0.35$ m), or falling back ($\dot{P}_x \leq -0.75$ m/s), where P_z represents the z-component of P_C and $\dot{P}_x$ represents the x-component of $\dot{P}_C$. For hardware trials, the conditions include reaching joint limits ($\theta_1 \notin [36.5°, 143.5°]$ and $\theta_2 \notin [-40.5°, 83.5°]$) or the operator engaging the emergency stop when the robot becomes unstable or reaches a target distance of approximately 8.23 m.

4.3.5. Simulation Experiments and Results

The target sample duration is set to 480 seconds. We evaluate our method on its ability in three aspects:

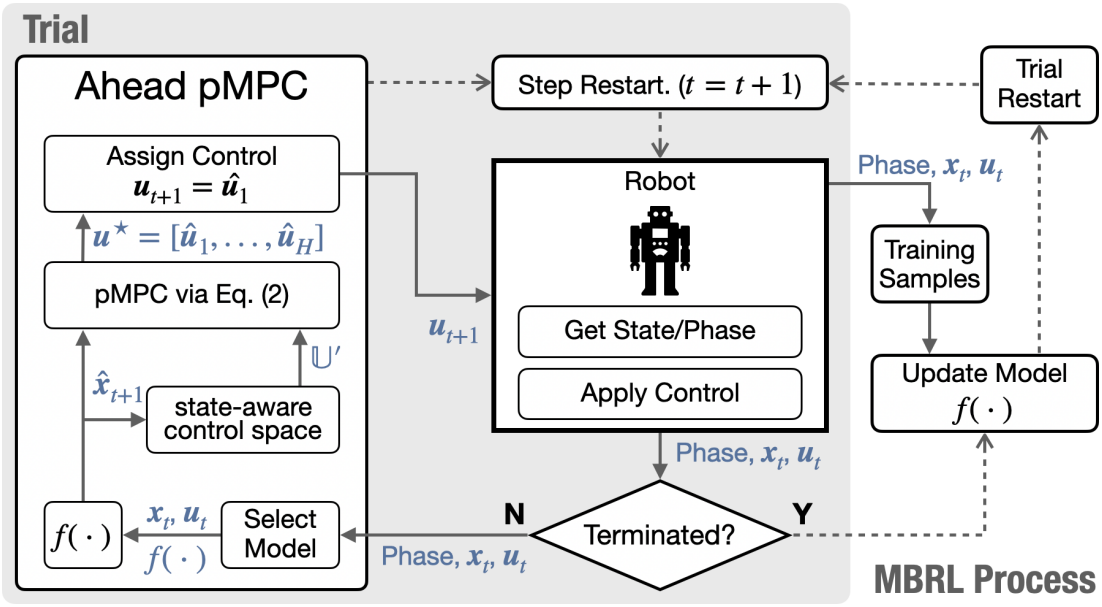

Figure 4.6. The model-based reinforcement learning process with ahead probabilistic model predictive control.

1. The effectiveness of leveraging Energy-exchange Dynamics (EED) for learning bipedal walking.

2. The effectiveness of enhancing planning reliability by constraining pMPC exploration with energy-state-aware control space.

3. The generalizability of our method across walking conditions, including uneven terrains and different walking speeds.

Leveraging Energy-exchange Dynamics for Learning Bipedal Walking

This section demonstrates the effectiveness of utilizing Energy-exchange Dynamics (EED) by comparing the following three instances:

(a) **Standard MBRL**, which learns the standard dynamics without energy terms. The state of standard MBRL is given by:

$$\boldsymbol{x} = \left[\boldsymbol{\Theta}^\top, \theta_T, \dot{\boldsymbol{\Theta}}^\top, P_z, \dot{P}_R, \dot{P}_T, \boldsymbol{\Theta}_S\right]^\top \in \mathbb{R}^{16}, \tag{4.39}$$

where P_z is the z-element of robot's CoM position $\boldsymbol{P}_C$; and $\boldsymbol{\Theta}_S \in \mathbb{R}^4$ contains all four spring's deflection.

(b) **TDE2-MBRL** [88], which only considers the spring's elastic energy as an energy-state. The state of TDE2-MBRL is given by:

$$\boldsymbol{x} = \left[\boldsymbol{\Theta}^\top, \theta_T, \dot{\boldsymbol{\Theta}}^\top, \boldsymbol{E}^\top\right]^\top \in \mathbb{R}^{13}. \tag{4.40}$$

(c) **Ours**, which considers both the spring and actuator's motion as energy-state. The energy-state of our method is given by:

$$\boldsymbol{z} = \left[\boldsymbol{\Theta}^\top, \theta_T, \boldsymbol{E}^\top\right]^\top \in \mathbb{R}^9 \tag{4.41}$$

All three methods are applied with the proposed state-aware control space to ensure fair comparison.

Fig. 4.7 and Table 4.3 demonstrate that learning EED not only does not impede learning performance, but also reduces the time cost per pMPC optimization iteration. This leads to higher success rates and highlights the importance of increasing optimization iterations. With more iterations within a fixed time interval, pMPC can possibly find better solutions. In our application, each pMPC optimization process terminates at 25 ms, as the robot is controlled at 40Hz.

Additionally, Fig. 4.8 shows an example of an energy-state trajectory and energy-exchange during successful walking with our method. The dashed line shows the model-predictive trajectory with the learned model. Our method can capture energy loss during walking and effectively compensate for such energy loss, leading to successful walking.

Instance	State Dimension	Average Time per pMPC Iteration	Speed Improvement
Ours	**9**	**0.58 ms ±4.61%**	**1.71×**
TDE2-MBRL [88]	13	0.83 ms ±2.73%	1.19×
Standard MBRL	16	0.99 ms ±2.84%	1×

Table 4.3. State dimensions and respective average time per pMPC optimization iteration for each instances, with standard deviations provided in percentages. One pMPC optimization iteration includes one H-horizon state prediction for optimization in Eq. (4.24).

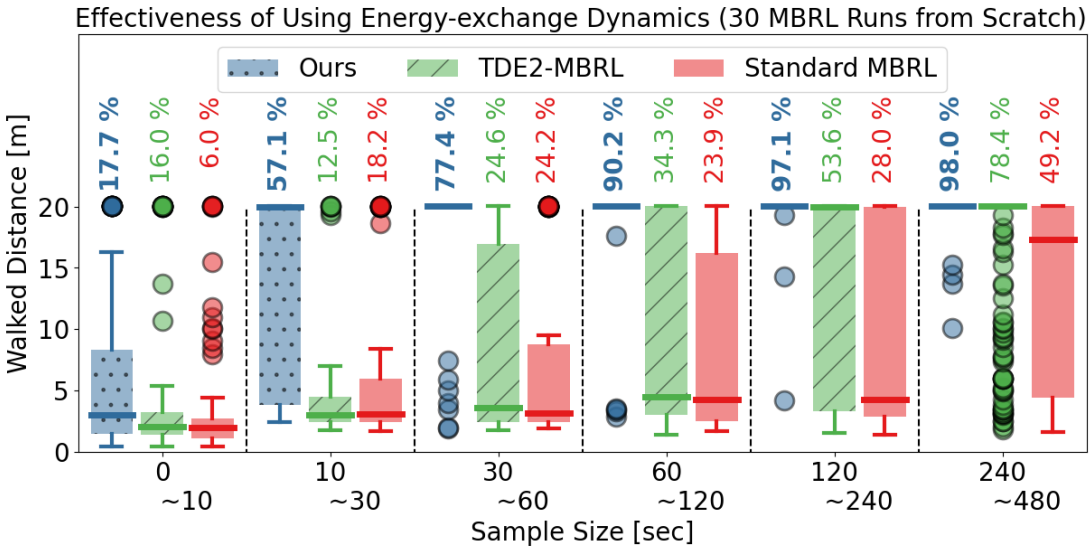

Figure 4.7. This figure displays the results of using energy to learn the walking task via simulation. All three methods were tested using the proposed state-aware control space to ensure a fair comparison. The success rate of achieving a 20-meter walk for each method is shown on top of its respective bar. The results demonstrate that MBRL with EED is effective in learning and performing the walking task, while using a compact robot state.

Enhancing Planning Reliability with Energy-state-aware Control Space

This section presents the effectiveness of constraining pMPC exploration for enhancing the planning reliability by comparing the following three instances:

(a) **No pMPC** refers to a control method where theoretical velocities, as obtained through Eq. (4.34) to Eq. (4.36), are directly taken as control input without pMPC planning. This highlights the limitation of using analytical methods for controlling compliant bipedal robots.

(b) **Standard pMPC** sets the pMPC control space to the actuator's performance limit, enabling pMPC to search from all possible controls.

(c) **Ours**, the proposed method that pMPC explores an established control space around the theoretical velocities.

Fig. 4.9 illustrates that the "No pMPC" was unable to execute the walking task due to its inability to counteract joint elasticity. The "Standard pMPC"

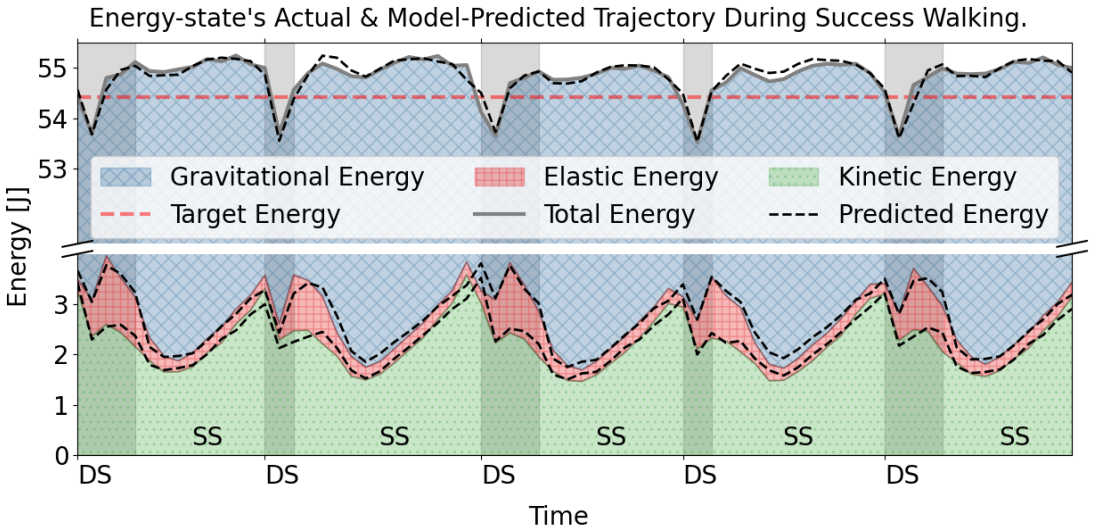

Figure 4.8. This figure shows the energy-state's actual and model-predicted (predicted mean) trajectory during successful walking in simulation, providing an example of the exchange between gravitational, elastic, and kinetic energy, as well as the energy loss during the DS phase that is attribute to joints' friction and damping. Our method effectively captures this energy loss, as shown in the model-predicted trajectory.

had a slow learning speed, which resulted from a wide pMPC exploration range. In contrast, our method exhibited a remarkable improvement in both learning efficiency and walking reliability. We achieved a 97.1% success rate after only two minutes of training samples.

In addition, Fig. 4.10 illustrates the swing leg motion of the proposed approach. The later-stage results ($\geq$240s) show improved tracking capability, with a much more concentrated trajectory than the early-stage ($\leq$30s) results. This improved tracking capability contributes to walking stability and a higher success rate, as shown in Fig. 4.9.

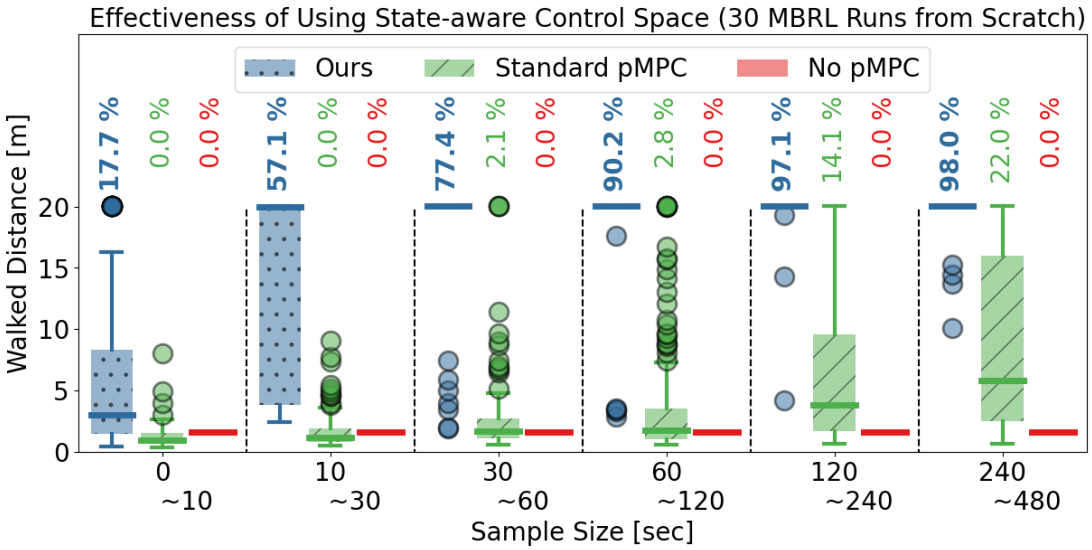

Figure 4.9. This figure shows the simulation results of constraining the pMPC control space to enhance planning reliability for performing the walking task. The success rate of achieving a 20-meter walk for each method is displayed above its respective bar. The increase in learning efficiency and success rate confirms the effectiveness of our method in enhancing planning reliability.

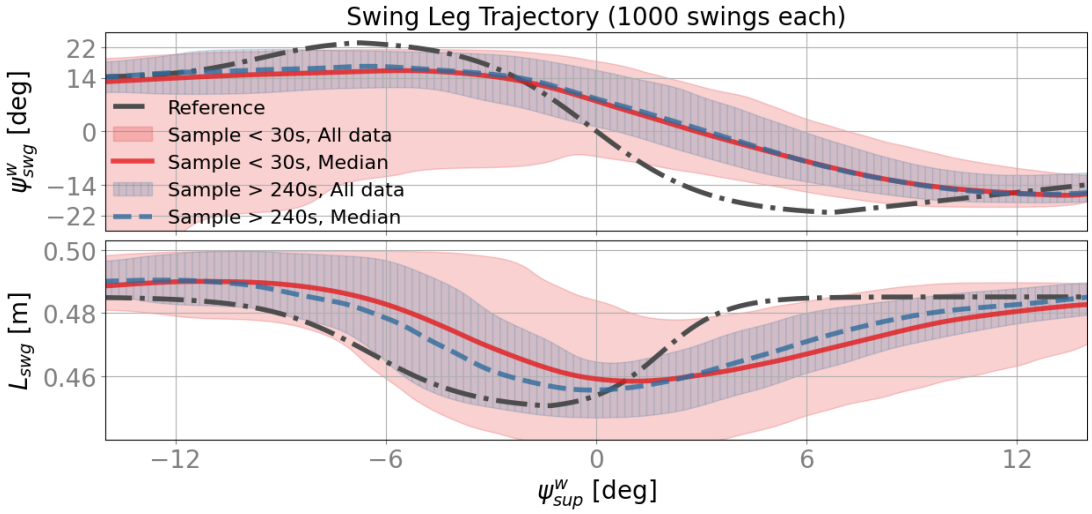

Figure 4.10. This figure displays the swing leg trajectory, obtained through our method in simulation (including failed attempts). The results indicate that increasing the number of training samples collected improves tracking performance, leading to successful completion of the walking task.

Generalizability Across Walking Conditions

The same learning framework and settings used in previous experiments were applied in the following experiments to demonstrate our method's ability to generalize across different walking conditions.

First, we evaluate the ability of our method to learn robot dynamics under unknown terrains that have random ground heights ranging from 0-10mm and 0-15mm (equivalent to 2% and 3% leg length), as shown in Fig. 4.11. The ground condition is unknown to the robot. The results shown in Fig. 4.12 demonstrate that our method achieved over 95% and 80% success rates on the 10mm and 15mm terrains, respectively. This shows its ability to handle ground uncertainties without prior knowledge of the ground's status.

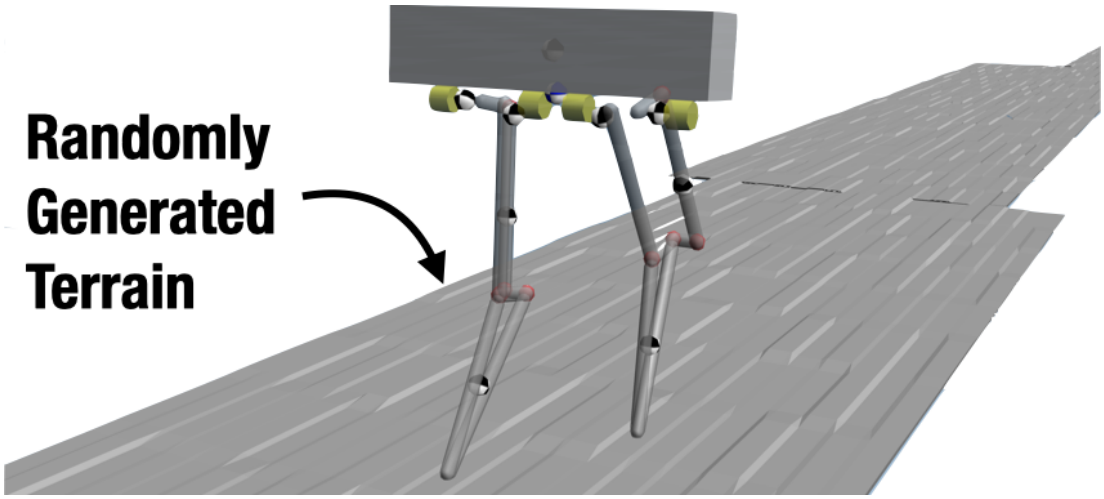

Figure 4.11. This figure shows the randomly generated uneven terrain that the robot walked on. The terrain's unevenness can be observed through the changing shading, as the ground height is randomly generated between 0 and 15 mm.

Secondly, we tested our method's ability to walk at different speeds by changing the velocity of the CoM at peak $\boldsymbol{v_p}$ to 0.3m/s, 0.7m/s, and alternating between these two velocities every five meters to handle continuous speed changes. All other settings were kept unchanged. The results (refer to Fig. 4.13) showed over 90% success across all three tests, demonstrating our method can generalize over different and changing walking speeds.

The "cross-performing" test results in Fig. 4.13 show that over 75% success was achieved when using all 30 models learned with 0.3m/s walking speed to perform

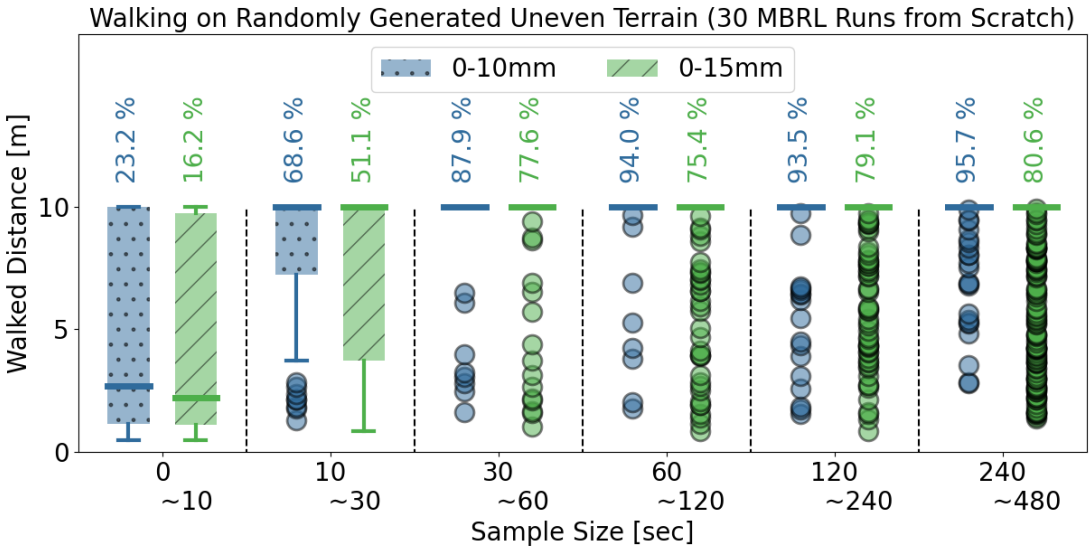

Figure 4.12. This figure displays the simulation results of 30 learning attempts using our method to walk on uneven terrain, demonstrating the robustness of our approach in traversing unknown terrain. The uneven terrain is randomly generated, with ground height ranging from 0-10mm and 0-15mm. The success rate of achieving a 10-meter walk for each method is shown above its respective bar.

0.7m/s walking, and vice versa. This emphasizes the advantages of learning robot dynamics over learning robot tasks.

4.3.6. Hardware Experiment and Results

For hardware validation with our method, we compared our method with "No pMPC," which directly applies theoretical control input to attempt walking. Our goal is to learn the robot's dynamics from scratch and perform walking with 40 Hz online planning until the boom reaches a 270° rotation, which is approximately an 8.23-meter walking distance (as shown in Fig. 4.14). The target sample duration of the learning process is set to 180 seconds.

As shown in Fig. 4.15, our method achieved a 73.3% success rate in this walking task with only approximately 180 seconds of samples, while "No pMPC" failed. This result demonstrates the real-world capability of our approach, even with a relatively low-spec CPU (i7-4700EQ) used for this experiment.

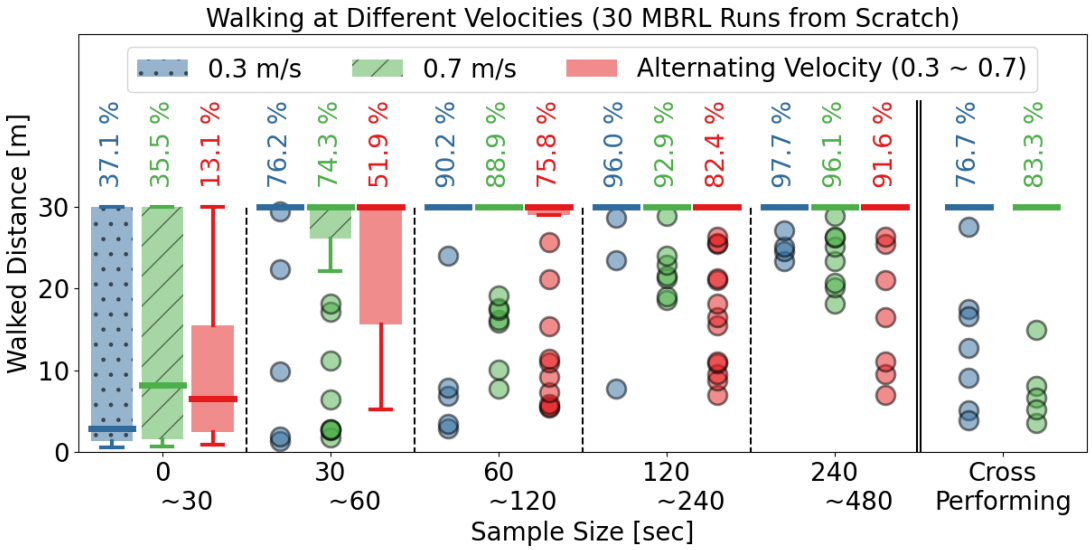

Figure 4.13. This figure displays the simulation results of 30 learning attempts using our method to walk at different CoM's peak velocities. The "alternating velocity" alternates between 0.3 m/s and 0.7 m/s every five meters of walking. The "cross-performing" demonstrates the average walking distance over five trials of using all 30 models learned with 0.3 m/s to perform 0.7 m/s walking, and vice versa. The success rate of achieving a 30-meter walk for each method is shown above its corresponding bar.

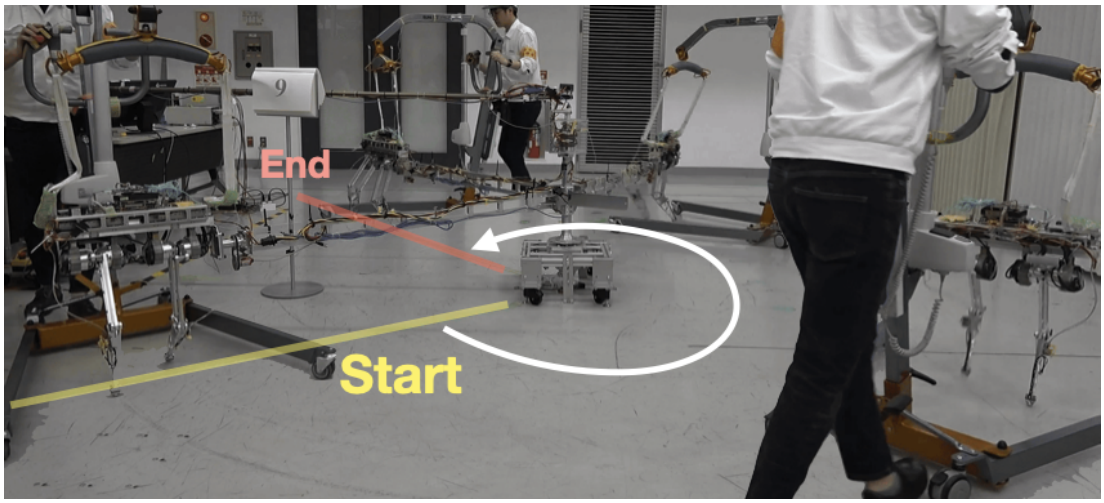

Figure 4.14. This figure illustrates the walking task with the hardware robot. The goal is to perform walking until the boom reaches 270° rotation.

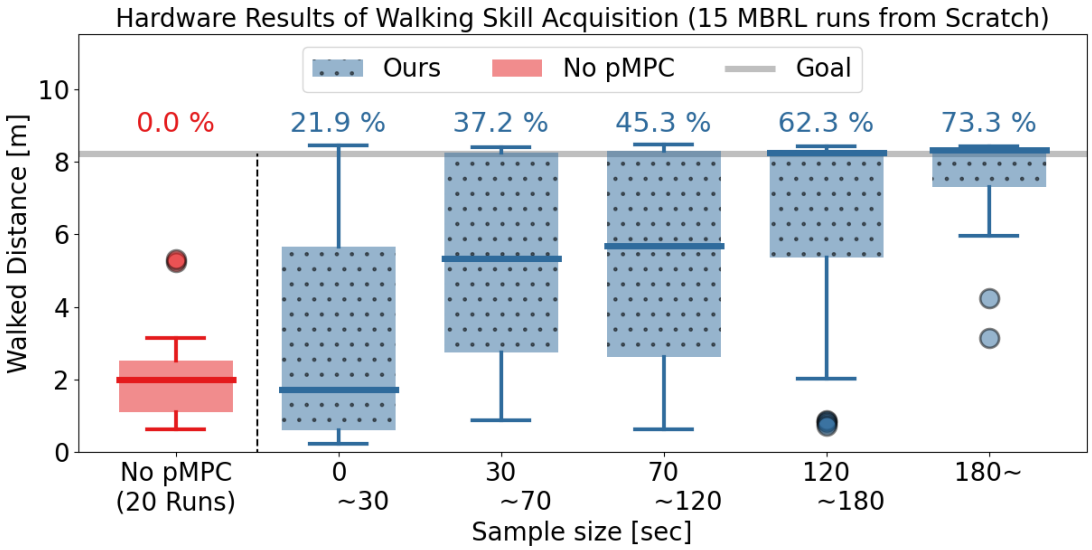

Figure 4.15. This figure displays the hardware results of our method learning the walking task. The walked distance was determined by combining the z-rotation readouts of the IMU and the boom length. The success rate for achieving the goal distance of 7.5m (accounting for sensor errors) is shown as a percentage above each corresponding bar. Due to the low-spec on-robot CPU, the pMPC averaged a time consumption of 2.44 ms per iteration, which is significantly longer than the simulation experiments. Nonetheless, our method achieved over 70% success with just three minutes of collected training samples.

4.4. Conclusion

This application uses task-relevant MBRL to alleviate the computational burden and achieve real-time 40-Hz control for walking with a spring-loaded bipedal robot.

To learn a compact dynamics model capable of online planning, we condensed the robot's state by applying the law of conservation of energy. Specifically, we reformulated an energy-state where the dynamics of the robot's CoM is expressed as potential and kinetic energy, its springs are viewed as a single energy container, and its actuators are treated as an energy source. The interactions between these components are characterized as energy-exchange and learned with MBRL.

Based on the energy-state, we designed an energy-state-based reference trajectory that allows MPC to find the optimal control enabling the robot to walk. We also established an energy-state aware control space to improve the planning reliability. Additionally, we decomposed the robot's dynamics model into task-phases

to cope with the dynamics dissimilarity between different contact conditions.

The effectiveness of using task-relevant MBRL to perform bipedal walking tasks is demonstrated through walking on uneven terrain, walking at different speeds, and walking at changing speeds with a simulated spring-loaded bipedal robot. The real-world feasibility is validated through fixed-speed walking with a hardware spring-loaded bipedal robot. All results showed successful on-site walking acquisition with (1) a compact nine-dimensional dynamics model, (2) 40Hz real-time planning capability, and (3) on-site learning within a few minutes.

5. Discussions

5.1. Open Issues

In this section, we discuss some open issues and limitations of the presented "task-relevant model-based reinforcement learning" approach that require further study and investigation.

5.1.1. Computation Burden for Higher Complexity Robots

In the bipedal walking application, we condensed the robot's state to a lower-dimensional energy-state. This allowed us to achieve real-time control planning at 40 Hz on a four Degree-of-Freedom (DoF) planar bipedal robot. However, implementing Model-based Reinforcement Learning (MBRL) on a more complex robot (such as a six or eight DoF compliant bipedal robot) may be difficult due to the high computational expense of MBRL and the adequacy of robot state for MBRL to learn dynamics, making it challenging to achieve the desired control frequency. Therefore, we need to overcome the computational burden of MBRL to enhance the overall applicability of our method.

5.1.2. Stability and Quality of the Long-horizon Reference Trajectory for the Intended Task

Unlike model-free learning-based approaches, where the user can define a terminal state and let the learning process find the intermediate trajectory, control planning with our MBRL has a shorter horizon that requires a long-horizon trajectory to recursively solve the control problem. Therefore, the quality of the reference trajectory is crucial for achieving optimal performance.

In both applications discussed in this thesis, reference trajectories are heuristically defined as a time series of robotic state entries. Although sufficient for mimicking the motion of the intended task, these trajectories do not guarantee the quality of task completion. For example, the quality of mixing cannot be evaluated as such information is not accessible from the defined robotic state. Similarly, the swing leg trajectory during the single-support phase of the walking task only reproduces human motion. While the success of these trajectories in both simulation and hardware environments suggests feasibility, they do not guarantee stability.

Therefore, it is crucial for our MBRL scheme to have state entries that are sufficient for evaluating task completion, and a method that enables us to obtain a well-thought-out reference trajectory for control planning.

5.1.3. Precise Contact Dynamics Capturing

Our method learns the combined dynamics of the robot and its contacts from collected data, which makes it difficult to distinguish between the robot dynamics and the contact dynamics. While this is not a problem when performing tasks that have similar contact behaviors, the robot's behavior may vary if the intended task requires a wide variety of contact behaviors, such as gripping a hard object immediately after gripping a soft one. Thus, further investigation is needed to address this limitation.

5.2. Implementation of Task-relevant Model-based Reinforcement Learning

This section offers a step-by-step guide on how to implement Task-relevant MBRL for generalizing to different robotic tasks. The main idea behind task-relevant MBRL is to adapt MBRL to the intended task. As MBRL learns the state-transition dynamics of the robot system, it is necessary to access the available robotic state entries of the learned dynamics for task performance. Therefore, it is crucial to reformulate a sufficiently informative task-relevant state to generalize to different tasks.

We concluded five steps for implementing task-relevant MBRL onto new robotic systems and robotic tasks:

1. **Construct a standard robotic state for dynamics learning**: To learn the state-transition dynamics, MBRL requires a sufficient state entry for predicting the robot's future state. Therefore, constructing a standard robotic state is essential for the robot's dynamics learning.

2. **Define the task-relevant state entries of the intended task**: Based on the intended task, define the requirements (or state entries) that are essential for performing the intended robotic task.

3. **Reformulate the standard robotic state to accommodate the task-relevant state entries**: It is important to note that the MBRL learned dynamics model predicts future states based on state entries. Therefore, incorporating task-relevant state entries requires a proper reformulation of the standard robotic state. For example, the energy-state reformulation for a bipedal robot follows the law of conservation of energy to ensure that the reformulated state sufficiently describes the dynamics of the robot system.

4. **Design a task-relevant reference trajectory for control planning**: Once sufficient state entries for the intended robotic task have been obtained, a task-relevant reference trajectory can be formulated using the reformulated robotic state. This allows for control planning when performing the intended robotic tasks.

5. **(Optional) Design a task-relevant control space to adjust the robot's motion behavior**: In our applications, we use the reformulated state to improve learning safety for everyday kitchen tasks and planning reliability for bipedal walking. Similarly, users can define a task-relevant control space to adjust the robot's motion behavior.

6. Conclusion

In recent years, robotics has made significant advancements and is now being used in a wider range of environments, including unpredictable and dynamic scenarios. New technology has allowed for the integration of elastic components, which make robots more flexible and efficient, similar to animal anatomy and physiology. However, these components also increase the complexity of the robot's dynamics, making it difficult to accurately model and control analytically.

Compliant robots are best suited for contact-rich environments where physical interactions with the environment are necessary, but their dynamics are even challenging with analytical approaches. Therefore, modern learning-based approaches are needed to handle such complexity.

This thesis presents "task-relevant model-based reinforcement learning," which reformulates standard model-based reinforcement learning (MBRL) so that robots can learn their dynamics and perform intended tasks with desired motion characteristics. The method reformulates a task-relevant robot state that contains sufficient entries for the intended task and is suitable for model-based reinforcement learning to learn the state-transition dynamics of the robot system. By utilizing the task-relevant state, the optimization problem can be solved, and the control can be found relative to achieving the intended task based on a state-formulated objective function. In addition, we utilize the reformulated state to adjust the robot's motion characteristic, such as improving contact-safety and planning reliability.

The effectiveness of task-relevant model-based reinforcement learning has been verified through two distinct contact-rich applications with different robot systems. The success of our method in both experiments demonstrates its effectiveness and applicability to various robotics applications that require flexibility, robustness, and safety in uncertain and contact-rich environments.

A. Appendix: Mathematical Details for LGM-FF

A.1. Analytic Solution for Integrating Feature Maps

The followings provides the analytic solution to Eq. (2.12):

$$\begin{aligned}\boldsymbol{q}_t = \int \Phi_t p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t &= \begin{bmatrix} 1 \\ \int \cos\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t \\ \int \sin\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t \end{bmatrix} = \begin{bmatrix} 1 \\ \tilde{\boldsymbol{q}}_t \end{bmatrix} \\ &= \begin{bmatrix} 1 \\ n^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\left(\boldsymbol{V}\tilde{\boldsymbol{\Sigma}}_t\right)^{\circ 2}\right) \circ \cos\left(\boldsymbol{V}\tilde{\boldsymbol{\mu}}_t\right) \\ n^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\left(\boldsymbol{V}\tilde{\boldsymbol{\Sigma}}_t\right)^{\circ 2}\right) \circ \sin\left(\boldsymbol{V}\tilde{\boldsymbol{\mu}}_t\right) \end{bmatrix} \in \mathbb{R}^M,\end{aligned} \tag{A.1}$$

where $p\left(\boldsymbol{x}_t\right) \sim \mathcal{N}\left(\boldsymbol{\mu}_t \in \mathbb{R}^D, \boldsymbol{\Sigma}_t \in \mathbb{R}^{D\times D}\right)$; $\tilde{\boldsymbol{\mu}}_t := \left[\boldsymbol{\mu}_t^\top, \boldsymbol{u}_t^\top\right]^\top$ is the state-control mean; $\tilde{\boldsymbol{\Sigma}}_t := \mathrm{diag}\left[\mathrm{diag}\left(\boldsymbol{\Sigma}_t\right), \boldsymbol{0}_U\right]$ is the state-control covariance matrix with padded U-dimensional zeros column vector, $\boldsymbol{0}_U$, for control's dimensions; Φ_t is the LGM-FF feature map from Eq. (2.4); $\boldsymbol{V} \in \mathbb{R}^{m\times D}$ is the sampled feature matrix from Eq. (2.2), and $\circ$ denotes element-wise operation.

The analytic solution to Eq. (2.13) is obtained by

$$\begin{aligned}\boldsymbol{Q}_t &= \int \Phi_t \Phi_t^\top p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t \\ &= \int \begin{bmatrix} 1 & \cos\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) & \sin\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) \\ \cos\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) & \cos\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right)\cos\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) & \cos\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right)\sin\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) \\ \sin\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) & \cos\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right)\sin\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) & \sin\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right)\sin\left(\boldsymbol{V}\tilde{\boldsymbol{x}}_t\right) \end{bmatrix} p\left(\boldsymbol{x}_t\right) \mathrm{d}\boldsymbol{x}_t \\ &= \begin{bmatrix} 1 & \tilde{\boldsymbol{q}}_t^\top \\ \tilde{\boldsymbol{q}}_t & \begin{matrix} \boldsymbol{CC} & \boldsymbol{CS} \\ \boldsymbol{CS} & \boldsymbol{SS} \end{matrix} \end{bmatrix} \in \mathbb{R}^{M\times M},\end{aligned} \tag{A.2}$$

where $\boldsymbol{CC} \in \mathbb{R}^{m\times m}$, $\boldsymbol{CS} \in \mathbb{R}^{m\times m}$, and $\boldsymbol{SS} \in \mathbb{R}^{m\times m}$ are calculated as follow:

$$\boldsymbol{CC} = \boldsymbol{G} \circ \cos\left(\boldsymbol{V}\tilde{\boldsymbol{\mu}}_t \mathbf{1}_n^\top - \mathbf{1}_n \tilde{\boldsymbol{\mu}}_t^\top \boldsymbol{V}^\top\right) + \boldsymbol{H} \circ \cos\left(\boldsymbol{V}\tilde{\boldsymbol{\mu}}_t \mathbf{1}_n^\top + \mathbf{1}_n \tilde{\boldsymbol{\mu}}_t^\top \boldsymbol{V}^\top\right), \tag{A.3}$$

$$\boldsymbol{CS} = \boldsymbol{G} \circ \sin\left(\boldsymbol{V}\tilde{\boldsymbol{\mu}}_t \mathbf{1}_n^\top - \mathbf{1}_n \tilde{\boldsymbol{\mu}}_t^\top \boldsymbol{V}^\top\right) + \boldsymbol{H} \circ \sin\left(\boldsymbol{V}\tilde{\boldsymbol{\mu}}_t \mathbf{1}_n^\top + \mathbf{1}_n \tilde{\boldsymbol{\mu}}_t^\top \boldsymbol{V}^\top\right), \tag{A.4}$$

$$\boldsymbol{SS} = \boldsymbol{G} \circ \cos\left(\boldsymbol{V}\tilde{\boldsymbol{\mu}}_t \mathbf{1}_n^\top - \mathbf{1}_n \tilde{\boldsymbol{\mu}}_t^\top \boldsymbol{V}^\top\right) - \boldsymbol{H} \circ \cos\left(\boldsymbol{V}\tilde{\boldsymbol{\mu}}_t \mathbf{1}_n^\top + \mathbf{1}_n \tilde{\boldsymbol{\mu}}_t^\top \boldsymbol{V}^\top\right), \tag{A.5}$$

with $\boldsymbol{G} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{H} \in \mathbb{R}^{m\times m}$ defined as

$$\boldsymbol{G} = n^{-1} \exp\left(-0.5\left(\boldsymbol{V}\tilde{\boldsymbol{\Sigma}}_t \mathbf{1}_n^\top - \mathbf{1}_n \tilde{\boldsymbol{\Sigma}}_t^\top \boldsymbol{V}^\top\right)^{\circ 2}\right), \tag{A.6}$$

$$\boldsymbol{H} = n^{-1} \exp\left(-0.5\left(\boldsymbol{V}\tilde{\boldsymbol{\Sigma}}_t \mathbf{1}_n^\top + \mathbf{1}_n \tilde{\boldsymbol{\Sigma}}_t^\top \boldsymbol{V}^\top\right)^{\circ 2}\right), \tag{A.7}$$

where $\mathbf{1}_m$ is an m-dimensional column vector of ones. Check the originated article for more details: [32].

A.2. Joint/Task/World Space Conversion of the Spring-loaded Bipedal Robot

The conversion between local- (θ_1, θ_2), task- (ϕ, ψ, L) and world-space (ψ^w, θ_T) are derived by the following equations:

$$\psi^w = \psi + \theta_T, \tag{A.8}$$

$$\phi = 0.5\,(\theta_{knee} - \theta_1 + \theta_2), \tag{A.9}$$

$$\psi = 0.5\,(\pi - \theta_{knee} - \theta_1 - \theta_2), \tag{A.10}$$

$$L = (l_{thigh} + l_{shin})\cos\phi, \tag{A.11}$$

$$\dot{L} = (l_{thigh} + l_{shin})\,\dot{\phi}\sin\phi, \tag{A.12}$$

$$\theta_i = \theta_i^{act} + \theta_i^{spr},\ \forall i = 1, 2, \tag{A.13}$$

where θ^{act} and θ^{spr} are the readouts of actuator and spring.

Acknowledgements

I would like to take this opportunity to express my heartfelt gratitude to all those who have supported me throughout my PhD journey. This journey would not have been possible without the encouragement, guidance, and support of those around me.

First and foremost, I would like to thank my supervisor, Prof. Takamitsu Matsubara. His expertise, guidance, and patience have been invaluable to me throughout my research. His unwavering support and encouragement have been instrumental in my success.

I would also like to express my sincere appreciation to my colleagues and friends who have supported and motivated me throughout this journey. It has been a pleasure to work alongside such talented and dedicated individuals. Their valuable insights and suggestions have helped me to shape my research and have contributed significantly to the success of my PhD.

I am also grateful for the financial support provided by Japan Society for the Promotion of Science, which enabled me to carry out my research to the best of my abilities. Without their support, my research would not have been possible.

Finally, I would like to thank my family for their unwavering support and encouragement. Their love, belief in me, and sacrifices have been the driving force behind my success. Their unwavering support and understanding during the most challenging times of my PhD journey have been a source of strength and motivation.

In conclusion, I would like to express my gratitude to all those who have supported and encouraged me throughout my PhD journey. Your support has been invaluable, and I am forever grateful for it. Thank you all for your unwavering support, guidance, and encouragement.

Bibliography

[1] V. L. Nguyen, C.-Y. Lin, and C.-H. Kuo, "Gravity Compensation Design of Planar Articulated Robotic Arms Using the Gear-Spring Modules," *Journal of Mechanisms and Robotics*, vol. 12, p. 031014, 02 2020.

[2] T. Kamioka, H. Shin, R. Yamaguchi, and M. Muromachi, "Development and analysis of a biped robot with prismatic compliance," in *IEEE International Conference on Robotics and Automation*, pp. 10398–10404, 2022.

[3] M. Plooij and M. Wisse, "A novel spring mechanism to reduce energy consumption of robotic arms," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2901–2908, 2012.

[4] M. Bamdad, M. Feyzollahzadeh, and S. R. Safavi, "Modeling and dynamic analysis of a robotic arm with pneumatic artificial muscle by transfer matrix method," *Mechanics Based Design of Structures and Machines*, vol. 0, no. 0, pp. 1–23, 2023.

[5] T. Z. Zhao, J. Luo, O. Sushkov, R. Pevceviciute, N. Heess, J. Scholz, S. Schaal, and S. Levine, "Offline meta-reinforcement learning for industrial insertion," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 6386–6393, 2022.

[6] X. Zhang, S. Jin, C. Wang, X. Zhu, and M. Tomizuka, "Learning insertion primitives with discrete-continuous hybrid action space for robotic assembly tasks," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 9881–9887, 2022.

[7] S. Nasiriany, H. Liu, and Y. Zhu, "Augmenting reinforcement learning with behavior primitives for diverse manipulation tasks," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 7477–7484, 2022.

[8] Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, and V. Sindhwani, "Data efficient reinforcement learning for legged robots," in *3rd Conference on Robot Learning*, vol. 100, pp. 1–10, 2019.

[9] P. Wu, A. Escontrela, D. Hafner, P. Abbeel, and K. Goldberg, "Daydreamer: World models for physical robot learning," in *6th Conference on Robot Learning*, vol. 205, pp. 2226–2240, 2022.

[10] M. A. Lee, Y. Zhu, P. Zachares, M. Tan, K. Srinivasan, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Learning multimodal representations for contact-rich tasks," *IEEE Transactions on Robotics*, vol. 36, no. 3, pp. 582–596, 2020.

[11] Íñigo Elguea-Aguinaco, A. Serrano-Muñoz, D. Chrysostomou, I. Inziarte-Hidalgo, S. Bøgh, and N. Arana-Arexolaleiba, "A review on reinforcement learning for contact-rich robotic manipulation tasks," *Robotics and Computer-Integrated Manufacturing*, vol. 81, p. 102517, 2023.

[12] C.-Y. Kuo, A. Schaarschmidt, Y. Cui, T. Asfour, and T. Matsubara, "Uncertainty-aware contact-safe model-based reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3918–3925, 2021.

[13] Y.-G. Kim, M. Na, and J.-B. Song, "Reinforcement learning-based sim-to-real impedance parameter tuning for robotic assembly," in *2021 21st International Conference on Control, Automation and Systems (ICCAS)*, pp. 833–836, 2021.

[14] Y. Fan, J. Luo, and M. Tomizuka, "A learning framework for high precision industrial assembly," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 811–817, 2019.

[15] T. Davchev, K. S. Luck, M. Burke, F. Meier, S. Schaal, and S. Ramamoorthy, "Residual learning from demonstration: Adapting DMPs for contact-rich manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4488–4495, 2022.

[16] N. Vulin, S. Christen, S. Stevšić, and O. Hilliges, "Improved learning of robot manipulation tasks via tactile intrinsic motivation," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2194–2201, 2021.

[17] L. Roveda, M. Magni, M. Cantoni, D. Piga, and G. Bucca, "Human–robot collaboration in sensorless assembly task learning enhanced by uncertainties adaptation via bayesian optimization," *Robotics and Autonomous Systems*, vol. 136, p. 103711, 2021.

[18] V. Gabler and D. Wollherr, "Bayesian optimization with unknown constraints in graphical skill models for compliant manipulation tasks using an industrial robot," *Frontiers in Robotics and AI*, vol. 9, p. 993359, 2022.

[19] A. S. Anand, M. Hagen Myrestrand, and J. T. Gravdahl, "Evaluation of variable impedance- and hybrid force/motioncontrollers for learning force tracking skills," in *2022 IEEE/SICE International Symposium on System Integration (SII)*, pp. 83–89, 2022.

[20] W. Guo, C. Wang, Y. Fu, and F. Zha, "Deep reinforcement learning algorithm for object placement tasks with manipulator," in *2018 IEEE International Conference on Intelligence and Safety for Robotics (ISR)*, pp. 608–613, 2018.

[21] J. Matas, S. James, and A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation," in *2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings*, vol. 87 of *Proceedings of Machine Learning Research*, pp. 734–743, PMLR, 2018.

[22] F. Yu, R. Batke, J. Dao, J. Hurst, K. Green, and A. Fern, "Dynamic bipedal turning through sim-to-real reinforcement learning," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots*, pp. 903–910, 2022.

[23] N. Csomay-Shanklin, M. Tucker, M. Dai, J. Reher, and A. D. Ames, "Learning controller gains on bipedal walking robots via user preferences," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 10405–10411, 2022.

[24] A. S. Polydoros and L. Nalpantidis, "Survey of model-based reinforcement learning: Applications on robotics," *Journal of Intelligent & Robotic Systems*, vol. 86, no. 2, pp. 153–173, 2017.

[25] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1714–1721, 2017.

[26] G. Cao, E. M. Lai, and F. Alam, "Gaussian process model predictive control of an unmanned quadrotor," *Journal of Intelligent and Robotic Systems*, vol. 88, no. 1, pp. 147–162, 2017.

[27] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious model predictive control using gaussian process regression," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2736–2743, 2020.

[28] Y. Cui, S. Osaki, and T. Matsubara, "Reinforcement learning ship autopilot: Sample-efficient and model predictive control-based approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2868–2875, 2019.

[29] M. P. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pp. 465–472, 2011.

[30] Y. Cui, S. Osaki, and T. Matsubara, "Autonomous boat driving system using sample-efficient model predictive control-based reinforcement learning approach," *Journal of Field Robotics*, vol. 38, no. 3, pp. 331–354, 2020.

[31] S. Kamthe and M. P. Deisenroth, "Data-efficient reinforcement learning with probabilistic model predictive control," in *International Conference on Artificial Intelligence and Statistics*, vol. 84, pp. 1701–1710, 2018.

[32] C.-Y. Kuo, Y. Cui, and T. Matsubara, "Sample-and-computation-efficient probabilistic model predictive control with random features," in *IEEE International Conference on Robotics and Automation*, pp. 307–313, 2020.

[33] W. Feller, *An Introduction to Probability Theory and Its Applications*. Wiley, 1968.

[34] A. Rahimi and B. Recht, "Random Features for Large-Scale Kernel Machines," in *Proceedings of the Advances in Neural Information Processing Systems 20*, pp. 1177–1184, 2008.

[35] Q. Le, T. Sarlós, and A. Smola, "Fastfood: Approximating kernel expansions in loglinear time," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, pp. 244–252, 2013.

[36] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2006.

[37] C. K. I. Williams, "Prediction with gaussian processes: From linear regression to linear prediction and beyond," in *Learning in Graphical Models*, pp. 599–621, Springer Netherlands, 1998.

[38] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[39] G. Fubini, *Sugli integrali multipli*, pp. 243–249. Cremonese, 1958.

[40] M. P. Deisenroth, *Efficient reinforcement learning using Gaussian processes*. KIT Scientific Publishing, 2010.

[41] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[42] E. D. Klenske, M. N. Zeilinger, B. Schölkopf, and P. Hennig, "Gaussian process-based predictive control for periodic error correction," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 1, pp. 110–121, 2016.

[43] Grancharova, Alexandra, Kocijan, Juš, and Johansen, Tor A, "Explicit stochastic predictive control of combustion plants based on Gaussian process models," *Automatica*, vol. 44, no. 6, pp. 1621–1631, 2008.

[44] M. Deisenroth, C. Rasmussen, and D. Fox, "Learning to control a low-cost manipulator using data-efficient reinforcement learning," in *Robotics: Science and Systems*, 2011.

[45] Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, and V. Sindhwani, "Data efficient reinforcement learning for legged robots," in *3th Conference on Robot Learning*, 2019.

[46] M. P. Deisenroth, G. Neumann, and J. Peters, "A survey on policy search for robotics," *Foundations and Trends in Robotics*, vol. 2, pp. 1–142, 2013.

[47] R. N. Carleton, "Into the unknown: A review and synthesis of contemporary models involving uncertainty," *Journal of Anxiety Disorders*, vol. 39, pp. 30 – 43, 2016.

[48] R. N. Carleton, "Fear of the unknown: One fear to rule them all?," *Journal of Anxiety Disorders*, vol. 41, pp. 5 – 21, 2016.

[49] H. Schättler and U. Ledzewicz, *Geometric Optimal Control: Theory, Methods and Examples*, vol. 53. Springer New York, 2012.

[50] L. S. Pontryagin, E. F. Mishchenko, V. G. Boltyan-skii, and R. V. Gamkrelidze, *The Mathematical Theory of Optimal Processes*. Wiley, 1962.

[51] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search," in *IEEE International Conference on Robotics and Automation*, pp. 156–163, 2015.

[52] J. Yamada, G. Salhotra, Y. Lee, M. Pflueger, K. Pertsch, P. Englert, G. S. Sukhatme, and J. J. Lim, "Motion planner augmented reinforcement learning for robot manipulation in obstructed environments," in *4th Conference on Robot Learning*, 2020.

[53] R. Martín-Martín, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg, "Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1010–1017, 2019.

[54] C. Beltran, D. Petit, I. G. Ramirez-Alpizar, T. Nishi, S. Kikuchi, T. Matsubara, and K. Harada, "Learning force control for contact-rich manipulation tasks with rigid position-controlled robots," *IEEE Robotics and Automation Letters*, vol. 5, pp. 5709 – 5716, 2020.

[55] F. Wirnshofer, P. Schmitt, G. von Wichert, and W. Burgard, "Controlling contact-rich manipulation under partial observability," in *Robotics: Science and Systems*, 2020.

[56] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Inc., 2001.

[57] T. Miyamoto, H. Sasaki, and T. Matsubara, "Exploiting visual-outer shape for tactile-inner shape estimation of objects covered with soft materials," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6278–6285, 2020.

[58] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of the 35th International Conference on Machine Learning*, pp. 1856–1865, 2018.

[59] M. A. Lee, C. Florensa, J. Tremblay, N. Ratliff, A. Garg, F. Ramos, and D. Fox, "Guided uncertainty-aware policy optimization: Combining learning and model-based strategies for sample-efficient policy learning," in *IEEE International Conference on Robotics and Automation*, pp. 7505–7512, 2020.

[60] A. LaGrassa, S. Lee, and O. Kroemer, "Learning skills to patch plans based on inaccurate models," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 9441–9448, 2020.

[61] M. Bollini, S. Tellex, T. Thompson, N. Roy, and D. Rus, *Interpreting and Executing Recipes with a Cooking Robot*, pp. 481–495. Springer International Publishing, 2013.

[62] K. Ishihara, T. D. Itoh, and J. Morimoto, "Full-body optimal control toward versatile and agile behaviors in a humanoid robot," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 119–126, 2020.

[63] S. Caron, A. Kheddar, and O. Tempier, "Stair climbing stabilization of the hrp-4 humanoid robot using whole-body admittance control," in *2019 International Conference on Robotics and Automation*, pp. 277–283, 2019.

[64] P. Seiwald, S.-C. Wu, F. Sygulla, T. F. C. Berninger, N.-S. Staufenberg, M. F. Sattler, N. Neuburger, D. Rixen, and F. Tombari, "Lola v1.1 – an upgrade in hardware and software design for dynamic multi-contact locomotion," in *IEEE-RAS 20th International Conference on Humanoid Robots*, pp. 9–16, 2020.

[65] J. Mizrahi and Z. Susak, "In-vivo elastic and damping response of the human leg to impact forces.," *Journal of Biomechanical Engineering*, vol. 104, no. 1, pp. 63–6, 1982.

[66] M. M. Plecnik, D. W. Haldane, J. K. Yim, and R. S. Fearing, "Design exploration and kinematic tuning of a power modulating jumping monopod," *Journal of Mechanisms and Robotics*, vol. 9, no. 1, p. 011009, 2017.

[67] C. Hubicki, J. Grimes, M. Jones, D. Renjewski, A. Spröwitz, A. Abate, and J. Hurst, "Atrias: Design and validation of a tether-free 3d-capable spring-mass bipedal robot," *The International Journal of Robotics Research*, vol. 35, no. 12, pp. 1497–1521, 2016.

[68] X. Xiong and A. D. Ames, "Bipedal hopping: Reduced-order model embedding via optimization-based control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3821–3828, IEEE, 2018.

[69] H. Geyer, A. Seyfarth, and R. Blickhan, "Compliant leg behaviour explains basic dynamics of walking and running," *Proceedings of the Royal Society B: Biological Sciences*, vol. 273, no. 1603, pp. 2861–2867, 2006.

[70] H. Geyer, U. Saranli, A. Goswami, and P. Vadakkepat, "Gait based on the spring-loaded inverted pendulum," in *Humanoid Robotics: A Reference*, pp. 1–25, Springer Netherlands, 2018.

[71] J.-K. Huang and J. W. Grizzle, "Efficient anytime CLF reactive planning system for a bipedal robot on undulating terrain," *IEEE Transactions on Robotics*, pp. 1–18, 2023.

[72] R. Batke, F. Yu, J. Dao, J. Hurst, R. L. Hatton, A. Fern, and K. Green, "Optimizing bipedal maneuvers of single rigid-body models for reinforcement learning," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots*, pp. 714–721, 2022.

[73] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, "Bayesian optimization for learning gaits under uncertainty," *Annals of Mathematics and Artificial Intelligence*, vol. 76, no. 1, pp. 5–23, 2016.

[74] K. Green, Y. Godse, J. Dao, R. L. Hatton, A. Fern, and J. Hurst, "Learning spring mass locomotion: Guiding policies with a reduced-order model," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3926–3932, 2021.

[75] J. Siekmann, Y. Godse, A. Fern, and J. Hurst, "Sim-to-real learning of all common bipedal gaits via periodic reward composition," in *2021 IEEE International Conference on Robotics and Automation*, pp. 7309–7315, 2021.

[76] L. Yang, Z. Li, J. Zeng, and K. Sreenath, "Bayesian optimization meets hybrid zero dynamics: Safe parameter learning for bipedal locomotion control," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 10456–10462, 2022.

[77] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for robust parameterized locomotion control of bipedal robots," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2811–2817, 2021.

[78] M. P. Deisenroth, M. F. Huber, and U. D. Hanebeck, "Analytic moment-based gaussian process filtering," in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 225–232, 2009.

[79] T. M. Moerland, J. Broekens, A. Plaat, and C. M. Jonker, "Model-based reinforcement learning: A survey," *Foundations and Trends® in Machine Learning*, vol. 16, no. 1, pp. 1–118, 2023.

[80] S. Levine and V. Koltun, "Learning complex neural network policies with trajectory optimization," in *Proceedings of the 31st International Confer-*

ence on International Conference on Machine Learning, vol. 32 of *ICML'14*, p. II–829–II–837, 2014.

[81] J. Morimoto and C. Atkeson, "Minimax differential dynamic programming: An application to robust biped walking," in *Advances in Neural Information Processing Systems*, pp. 1539–1546, 2002.

[82] M. P. Deisenroth, R. Calandra, A. Seyfarth, and J. Peters, "Toward fast policy search for learning legged locomotion," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1787–1792, 2012.

[83] G. Folkertsma and S. Stramigioli, "Energy in robotics," *Foundations and Trends® in Robotics*, vol. 6, no. 3, pp. 140–210, 2017.

[84] M. Hutter, C. D. Remy, M. A. Höpflinger, and R. Siegwart, "SLIP running with an articulated robotic leg," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4934–4939, 2010.

[85] T. Yoshikawa, *Foundations of Robotics: Analysis and Control*. The MIT Press, 01 2003.

[86] H.-W. Park, A. Ramezani, and J. W. Grizzle, "A finite-state machine for accommodating unexpected large ground-height variations in bipedal robot walking," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 331–345, 2013.

[87] R. Hagengruber, *Emilie du Châtelet between Leibniz and Newton*. Springer, 2012.

[88] C.-Y. Kuo, H. Shin, T. Kamioka, and T. Matsubara, "TDE2-MBRL: Energy-exchange dynamics learning with task decomposition for spring-loaded bipedal robot locomotion," in *IEEE-RAS 22th International Conference on Humanoid Robots*, pp. 550–557, IEEE, 2022.

[89] Z. Xie, G. Berseth, P. Clary, J. Hurst, and M. van de Panne, "Feedback control for cassie with deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1241–1246, 2018.

[90] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.

[91] G. Zhao, M. Grimmer, and A. Seyfarth, "The mechanisms and mechanical energy of human gait initiation from the lower-limb joint level perspective," *Scientific Reports*, vol. 11, p. 22473, Nov 2021.

Publication List

Journal

1. Cheng-Yu Kuo, Andreas Schaarschmidt, Yunduan Cui, Tamim Asfour, Takamitsu Matsubara, "Uncertainty-aware Contact-safe Model-based Reinforcement Learning," IEEE Robotics and Automation Letters (RA-L), 6(2), pp.3918-3925, 2021 (with IEEE ICRA 2021 option)

2. Cheng-Yu Kuo, Hirofumi Shin, Takamitsu Matsubara, "Reinforcement Learning with Energy-Exchange Dynamics for Spring-loaded Biped Walking," IEEE Robotics and Automation Letters (RA-L), 8(10), pp.6243-6250, 2023

International Conference

1. Cheng-Yu Kuo, Yunduan Cui, Takamitsu Matsubara, "Sample-and-computational-efficient Probabilistic Model Predictive Control with Random Features," IEEE International Conference on Robotics and Automation (ICRA), pp.307-313, 2020

2. Cheng-Yu Kuo, Andreas Schaarschmidt, Yunduan Cui, Tamim Asfour, Takamitsu Matsubara, "Uncertainty-aware Contact-safe Model-based Reinforcement Learning," IEEE International Conference on Robotics and Automation (ICRA), 2021 (as a RA-L presentation option)

3. Cheng-Yu Kuo, Hirofumi Shin, Takumi Kamioka, Takamitsu Matsubara, "TDE2-MBRL: Energy-exchange Dynamics Learning with Task Decomposition for Spring-loaded Bipedal Robot Locomotion," IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp.550-557, 2022