Doctoral Thesis

Leveraging Human Risk Sensitivity for Enriching Imitation Learning

Hanbit Oh

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

Supervisor: Takamitsu Matsubara
Robot Learning Lab. (Division of Information Science)

Submitted on March 4, 2024

A Doctoral 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

Hanbit Oh

Thesis Committee:

Supervisor	Professor Takamitsu MATSUBARA (Division of Information Science)
Co-supervisor	Professor Takahiro WADA (Division of Information Science)
Co-supervisor	Assistant Professor Hikaru SASAKI (Division of Information Science)

Leveraging Human Risk Sensitivity for Enriching Imitation Learning[*]

Hanbit Oh

Abstract

Humans demonstrate interesting behaviors, such as moderating actions to perceived risks and enhancing task performance. However, Imitation Learning (IL), where robots learn to perform these same tasks by observing human demonstrations, often fails to capture such behavior. Since it relies on a black-box algorithm that maps state to action while overlooking the underlying human characteristic (*i.e.,* risk sensitivity), limiting the applicability of IL and the robot's generalizability. Therefore, this thesis proposes a novel mechanism to design IL algorithms based on behavioral psychology findings, embodying principles to capture and leverage risk sensitivity to enhance IL. We focus on the risk sensitivity revealed from the *speed-accuracy trade-off*, in which humans in risky areas slow down their movement speed to enhance accuracy. We use such data to capture the environmental risk. Our mechanism is verified through two IL applications: 1) the risk estimator to improve the safety of on-policy IL, as a robot iteratively optimized through human corrections in risky areas while executing its unmatured policy, and 2) the risk-sensitive disturbance model to ensure demonstration feasibility of disturbance-injected IL, as disturbances are injected into human demonstrations to train robust policy while regulating its level small in risky areas. Verification shows that, through enhanced risk sensitivity, significantly improved training safety, and our best performance compared favorably with other IL methods.

Keywords:

Imitation Learning, Human Risk Sensitivity, On-policy Imitation learning, Disturbance-injected Imitation Learning

[*]Doctoral Thesis, Department of Information Science, Graduate School of Information Science and Technology, Nara Institute of Science and Technology, March 4, 2024.

人間のリスク鋭敏性の活用による模倣学習の強化[*]

Hanbit Oh

内容梗概

人間は、危険を察知し適切な行動を選択することで高度な作業スキルを発揮している。一方でそのような人間の行動特性は、人間の教示でロボットを学習する模倣学習（IL）で考慮することができない。標準的なILは、人間の作業実演を用いてロボットにこれらと同じ作業を教えるアプローチである。人間の実演から状態と行動の対応関係をブラックボックス化したアルゴリズムに依存しており、危険に鋭敏な行動を駆動する根本的な特性（リスク鋭敏性）を見落としてしまう。これにより、模倣学習の適用性とロボットの汎化性能が制限される。そこで本論文では、人間の行動心理学に基づいてILアルゴリズムを設計する新しいメカニズムを提案し、ILを強化するためにリスク鋭敏性を捉え、その活用に着目する。具体的に注目する人間のリスク鋭敏性として、人間の速度-精度トレードオフを考慮する。人間は環境の理解を基づいてリスクを感知しており状態のリスクに応じて速度を調整する。そのため、人間がみせる速度教示データから環境のリスク情報を推定することができる。提案メカニズムを使用した 2 つの適用事例を示す。1) 人間のリスク鋭敏性から派生した危険予測器を導入し、オンポリシーIL枠組みの安全を向上させる。ロボットが危険な状態に遭遇した際に人間の介入・教示を能動的に要請することにより、ロボットは反復的に最適化され、安全なオンポリシーILが実現される。2) 人間のリスク鋭敏性から派生した外乱モデルを導入し、外乱注入IL枠組みで安全性を確保する。人間の実演に外乱が注入することと同時に外乱レベルが危険な領域では小さく調整され、人間実演の安全性の確保と外乱への頑健なポリシーの学習性を両立したILが実現される。これらの事例では、提案手法がリスク鋭敏性を取り入れることで、模倣学習の訓練段階の安全性かつロボットの自律性能が向上することを確認した。

キーワード

模倣学習・人間リスク鋭敏性・オンポリシー模倣学習・外乱注入模倣学習

[*]奈良先端科学技術大学院大学 先端科学技術研究科 情報科学領域 博士論文, 2024年 3月 4日.

Contents

List of Figures

List of Tables

1. Introduction

1.1. Robotic Manipulation and Its Challenge

In recent years, remarkable advancements in robot technology have revolutionized the manufacturing industries by increasing productivity beyond that of humans [1]. However, implementing robotic manipulation skills is still challenging. This is because programming robots for a specific domain is time-consuming, but they need to be re-programmed if the environment changes even slightly. For example, in a housework domain where there are often new objects added or the position of objects changes, it is expensive to measure and re-program each time the environment changes. Instead of programming the robot for such domains, robotic systems are typically allowed to operate by a human expert, who can control a robot optimally for various scenarios. Nevertheless, it is also a difficult situation due to a shortage of skilled human resources. This situation strongly motivates an approach to generating robot manipulation with human-like performance instead of high-cost programming.

1.2. Machine Learning for Robot Controllers

Creating robot controllers via machine learning has found widespread usage in both research [2–4] and commercial [5–10] applications. Controller learning often utilizes large-scale exploration [3], reward mechanisms (*e.g.,* an optimal control or reinforcement learning), and with highly accurate dynamics models [2] to learn autonomous control. However, in scenarios with exploration or dynamics model sparsity, Imitation Learning (IL) [11–13] is an intuitive method for learning skills via observations of an expert demonstrator, avoiding explicit programming, reward design, or large-scale exploration. Moreover, the fundamental idea behind

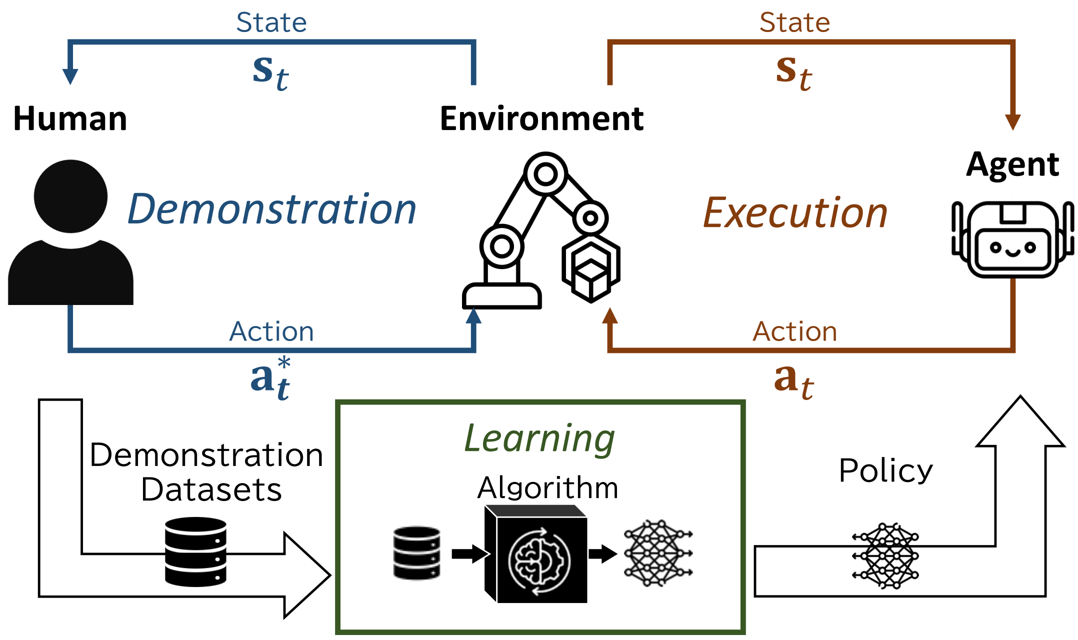

Figure 1.1. Overview of the imitation learning.

IL is that a human can demonstrate a desired behavior by teleoperating a robot or a machine. As a result, IL can be implemented in any system that necessitates autonomous robot behaviors comparable to those of an expert's operation.

1.3. Imitation Learning for Robot Manipulation

The key objective of Imitation Learning (IL) [11–13] is to teach robotic agents how to perform a task by mimicking an expert using human demonstrations as a guide. The standard form of this approach is training an agent to learn a policy that outputs expert actions from given states of the environment as shown in Figure 1.1. This approach involves collecting a sequence of state-action pairs from human demonstrations and providing it to an agent to train for learning a policy that maps its current state to the corresponding expert action. These learned policies can be applied to execute various robot manipulation tasks [5–9].

Although many conventional imitation learning methods have been proposed, they are often inherently flawed when learning realistic robot manipulation tasks from humans. Specifically, actual human demonstrations have intrinsic behavioral characteristics that enhance task performance, but existing methods rely solely on learning state-to-action mappings with a black-box algorithm, and they neglect the analysis of these behavioral characteristics underlying demonstrations. For example, humans exhibit risk-sensitive behaviors such as regulating their actions cautiously depending on perceived environmental risks (*e.g.,* collisions) to perform a task safely. However, in standard IL, robotic agents overlook such underlying risk sensitivity and only learn fragmentary state-to-action behaviors from demonstrations [5, 9, 10, 14], resulting in task execution of policy that can be risky. In other words, these gaps in IL induced by the overlooking of human behavioral characteristics limit robot generalizability and applicability.

1.4. Human Behavioral Characteristics Inherent in Demonstrations

This thesis explores the idea of identifying human behavioral characteristics from human demonstrations based on findings in behavioral psychology. Humans demonstrate a variety of interesting behaviors that enhance their task performance based on understanding the domain and environment. As a specific human behavioral characteristic, we focus on human risk sensitivity. For example, in a task that reaches a shaft between obstacles, humans slow down their hand speed to increase their accuracy as the gap between obstacles narrowers (Figure 1.2). This behavioral characteristic has been actively studied in neuroscience and called the *speed-accuracy trade-off* [15, 16]. Based on these findings, we can identify human behavior characteristics (*e.g.,* risk sensitivity) revealed from human demonstrations (*e.g.,* speed-accuracy trade-off).

1.5. Leveraging Human Risk Sensitivity for Enriching Imitation Learning

This thesis presents a novel mechanism, "Leveraging Human Risk Sensitivity for Enriching Imitation Learning," that designs imitation learning algorithms focusing on utilizing human risk sensitivity, thereby embodying principles for capturing

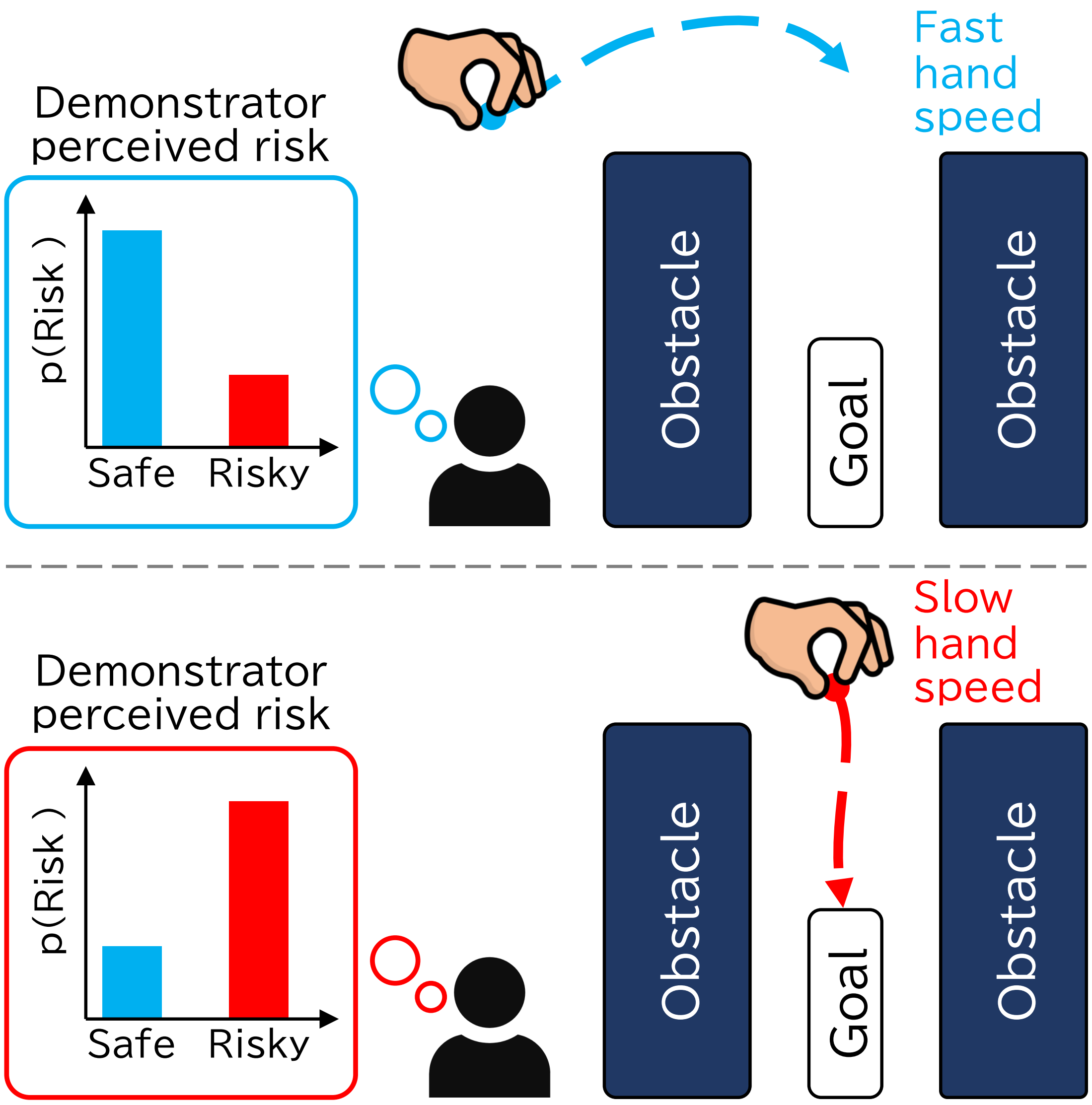

Figure 1.2. Human risk sensitivity in a clearance-limited reaching task. In clearance-limited tasks, demonstrator-perceived risk is in the mind of humans. (top): Humans tend to perceive open areas as safe; thus, they use fast hand speed, even though accuracy may decrease. On the other hand, (bottom): humans tend to perceive narrow areas as risky, and in order to enhance accuracy, they slow down their hand speed.

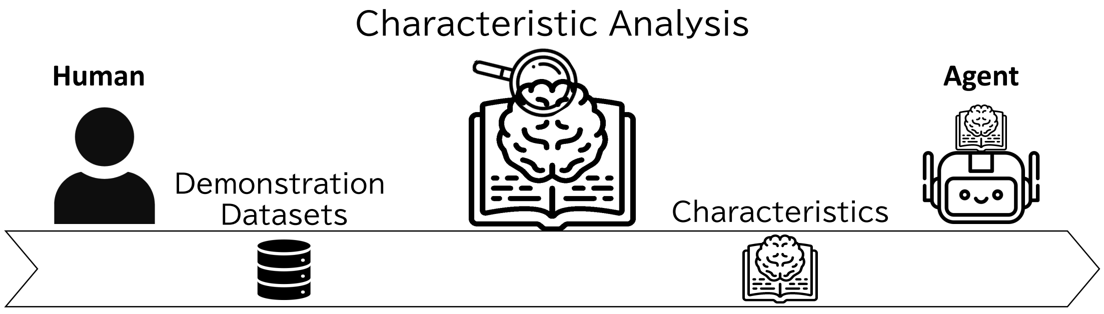

Figure 1.3. Overview of the proposed mechanism.

and exploiting actual demonstrator risk sensitivity (Figure 1.3). Specifically, we develop methods to capture risk sensitivity inherent in human demonstrations based on behavioral psychology. By typifying human risk sensitivity, we establish an enriched imitation learning framework that captures and leverages human risk sensitivity within the imitation learning paradigm. Through this integration, robots can deepen their understanding of human behavior and more effectively imitate expert behaviors.

In summary, leveraging human risk sensitivity for enriching imitation learning consist of two major components:

(i) Developing a method to capture risk sensitivity from human demonstrations based on findings in behavioral psychology;

(ii) Establishing a framework that captures and leverages human risk sensitivity to enrich imitation learning.

1.6. Verification of Our Mechanism's Effectiveness

As the specific human risk sensitivity, we focus on the speed-accuracy trade-off exhibited from human demonstrations, as described in Chapter 1.4. Therefore, the effectiveness of our mechanism is verified through two distinct imitation learning frameworks that leverage the speed-accuracy trade-off to enrich risk sensitivity as follows:

(i) Leveraging human risk sensitivity to improve safety of on-policy imitation learning;

(ii) Leveraging human risk sensitivity to ensure demonstration feasibility of disturbance-injected imitation learning.

1.6.1. Application 1: Leveraging Human Risk Sensitivity to Improve Safety of On-policy Imitation Learning

Our first application exploits the speed-accuracy trade-off to engage risk sensitivity into on-policy imitation learning setup for improving training safety. On-policy IL is a branch of imitation learning where a robot performs a task using its current policy, and then refines its policy by learning from human feedback on its actions in states a robot encounters. This approach allows a human to observe the learning process, making it possible to train the robot more efficiently until its performance is guaranteed. However, deploying unmatured policies often poses significant collision risks in clearance-limited tasks, such as industrial insertion. Therefore, we present a novel on-policy imitation learning algorithm that incorporates a risk estimator based on human risk sensitivity. Our algorithm enables a robot to cede control to a human demonstrator in states where high risk is detected during on-policy training, thereby reducing the likelihood of collisions. Specifically, our method consist of two features:

(i) *Risk-sensitive Model*: We introduce the risk estimator that learns to capture the speed-accuracy trade-off from human demonstrations and approximates the demonstrator-perceived risks for given states;

(ii) *Risk-aware On-policy IL Framework*: We incorporate the risk estimator into on-policy IL so that a robot can cede control to a human demonstrator in high-risk areas while executing its policy in low-precision areas.

This incorporation of human risk sensitivity into on-policy IL can improve the safety of on-policy learning and ensure its efficiency.

Our method's effectiveness is assessed through simulations and real-robot experiments that trained a UR5e 6-DOF robotic arm to perform assembly tasks.

Our results significantly improved training safety, and our best performance compared favorably with other learning methods.

1.6.2. Application 2: Leveraging Human Risk Sensitivity to Ensure Demonstration Feasibility of Disturbance-injected Imitation Learning

Our second application exploits the speed-accuracy trade-off to engage risk sensitivity in disturbance-injected imitation learning setup for ensuring demonstration feasibility. Disturbance-injected IL is a branch of imitation learning that involves adding artificial disturbances to human action commands during demonstrations to generate richer datasets. In this, injecting disturbances simulates errors that can occur during task execution, inducing human recovery actions of demonstration. This scheme enables robots to learn policies robust to diverse sources of errors. However, in prior works, two naive setups limit its applicability. First, scenarios requiring humans to perform cautious actions are common, but the fixed disturbance level parameter used by prior methods cannot regulate the level small in a state-dependent manner, causing unintended collisions during human demonstrations (*i.e.,* lack of risk sensitivity), thereby significantly hindering demonstration feasibility. In addition, scenarios requiring humans to choose between multiple optimal actions are common, but the deterministic policy models used by existing methods fail to capture these actions (*i.e.,* lack of flexibility), significantly hindering the robot policy's generalizability.

Therefore, we present the first Bayesian imitation learning framework that unifies learning of flexibility, robustification, and risk sensitivity. Specifically, our method consist of two features:

(i) *Risk-sensitive Model*: Risk-sensitive disturbance model is introduced inspired by the speed-accuracy trade-off exhibited by humans;

(ii) *Risk-sensitive Disturbance-injected IL Framework*: Bayesian inference is used to learn flexible non-parametric multi-action policies, while simultaneously robustifying policies by injecting risk-sensitive disturbances to induce human recovery action and ensuring demonstration feasibility.

Our proposed method of combining flexibility, robustification, and risk sensitivity allows a robot to learn robust multi-action policies while ensuring demonstration feasibility.

Our method is evaluated through risk-sensitive simulations and real-robot experiments (*e.g.,* table-sweep task, shaft-reach task and shaft-insertion task) using the UR5e 6-DOF robotic arm, to demonstrate the improved characterisation of behavior. Results show significant improvement in task performance, through improved flexibility, robustness as well as demonstration feasibility.

1.7. Thesis Structure

The contents of this thesis are structured as follows:

- Chapter 2 provides the preliminary information.
- Chapter 3 details the first implementation, which describes how human speed-accuracy trade-off was utilized to improve safety of on-policy imitation learning.
- Chapter 4 details the second implementation, which describes how human speed-accuracy trade-off was utilized to ensure demonstration feasibility of disturbance-injected imitation learning for robust and flexible policies.
- Chapter 5 raises several possible future extensions and open issues that require further study.
- Finally, Chapter 6 presents the conclusion to this thesis.

2. Preliminaries

2.1. Imitation Learning from Expert's Demonstration

The objective of imitation learning is to learn a control policy by imitating the action from the expert's demonstration data. The classical approach of imitation learning is Behavioral Cloning (BC) [17]. In BC, first, a human expert creates training data by demonstration. Subsequently, a supervised learning algorithm is executed directly regressing a policy from observed states to actions. Prior applications of BC have been impressive, including the learning of neural network policies for autonomous car driving based on recorded human driver's demonstrations [9].

A dynamics model of this setting is denoted as Markovian with a state $\mathbf{s}_t \in \mathbb{R}^Q$, an action $a_t \in \mathbb{R}$ and a state transition distribution $p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, a_t)$. A policy $\pi(a_t \mid \mathbf{s}_t)$ decides an action from a state. A trajectory $\tau = (\mathbf{s}_0, a_0, \mathbf{s}_1, a_1 \dots a_{T-1}, \mathbf{s}_T)$ which is a sequence of state-action pairs of T steps. The trajectory distribution is indicated as:

$$p(\tau \mid \pi) = p(\mathbf{s}_0) \prod_{t=0}^{T-1} \pi(a_t \mid \mathbf{s}_t) p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, a_t). \tag{2.1}$$

A key aspect of imitation learning is to replicate the expert's behavior, and as such the function which computes the similarity of two policies using trajectories is defined as:

$$J(\pi, \pi' \mid \tau) = -\sum_{t=0}^{T-1} \mathbb{E}_{\pi(a|\mathbf{s}_t), \pi'(a'|\mathbf{s}_t)} \left[||a - a'||_2^2 \right]. \tag{2.2}$$

A learned policy π^R is obtained by solving the following optimization problem

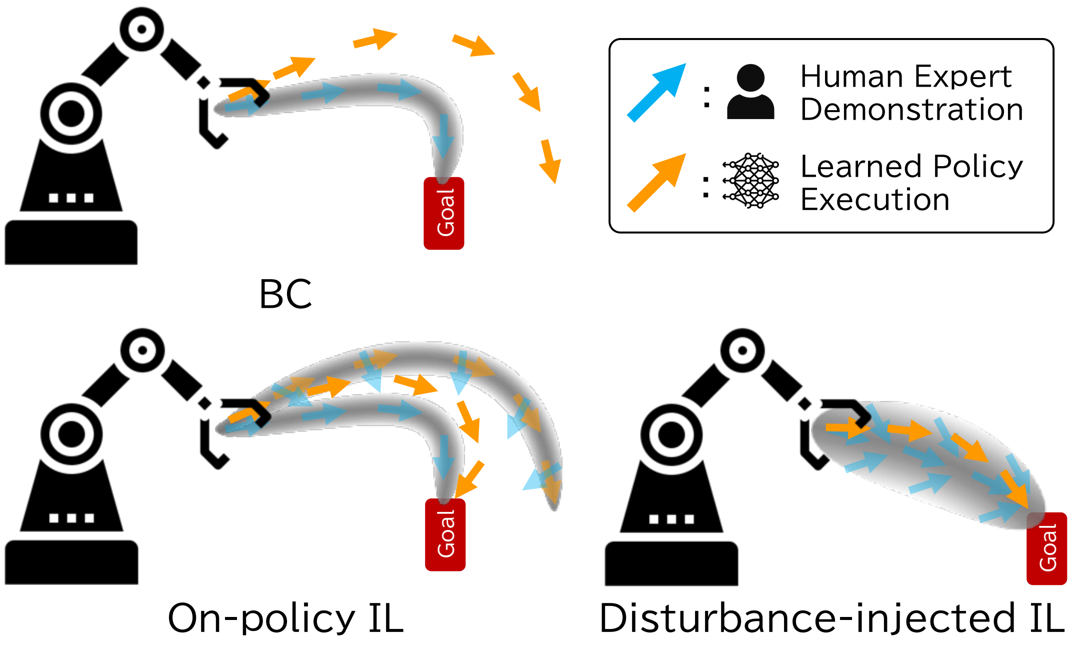

Figure 2.1. Comparing imitation learning methods on robot learning to reach a goal object. Policies learned by expert' demonstrations (blue arrow) inevitably make at least occasional mistakes in operation (orange arrow). BC: A small mistake at an early stage will lead the learner to deviate from the distribution of expert's demonstration (grey). On-policy IL: allow the robot to learn recovery behaviors in the state when it deviates from the expert's demonstration. Disturbance-injected IL: inject disturbance assuming learner's mistakes into expert behavior to expanded expert's demonstration coverage, learning robust policies from mistakes.

using a trajectory collected by an expert's policy π^*:

$$\pi^R = \arg\max_{\pi} \mathbb{E}_{p(\tau|\pi^*)} \left[J(\pi, \pi^* \mid \tau) \right]. \tag{2.3}$$

The benefits of BC are straightforward application and the capability to learn from demonstrations.

In imitation learning of this setup, however, a learned policy may suffer from the problem of error compounding, caused by covariate shift as shown in Figure 2.1-BC. This is defined as the distributional difference between the trajectories in the training data as observed during the data collection stage, and those in testing:

$$\left| \mathbb{E}_{p(\tau|\pi^*)}[J(\pi^R, \pi^* \mid \tau)] - \mathbb{E}_{p(\tau|\pi^R)}[J(\pi^R, \pi^* \mid \tau)] \right|. \tag{2.4}$$

2.2. On-policy Imitation Learning

A more general approach to minimizing the covariate shift in imitation learning is on-policy IL. This scheme involves iteratively collecting additional demonstrations of the expert's actions in the states where the robot encounters during learned policy deployment (Figure 2.1-On-policy IL).

Dataset Aggregation (DAgger) [18], one famous on-policy IL, learns a sequence of policies during the iterative on-policy demonstrations. Initially, the policy π_1^R is learned in BC using only a recorded expert's demonstration trajectory τ. Subsequently, at iteration k, DAgger collects on-policy demonstrations by employing a meta-policy $\pi_k^M(\mathbf{a}_t \mid \mathbf{s}_t)$ that outputs actions from either a learned policy π_k^R or a human demonstrator π^* based on the robot's control mode:

$$\pi_k^M(\mathbf{a}_t \mid \mathbf{s}_t) = g(\mathbf{s}_t)\pi^*(\mathbf{a}_t^* \mid \mathbf{s}_t) + (1 - g(\mathbf{s}_t))\pi_k^R(\mathbf{a}_t^R \mid \mathbf{s}_t), \tag{2.5}$$

where $g(\mathbf{s}_t) = \mathbb{1}$ is a binary decision function, which determines whether a robot operates in an *auto mode* ($g(\mathbf{s}_t) = 0$, controlled by learned policy π_k^R) or an *expert mode* ($g(\mathbf{s}_t) = 1$, controlled by an expert's policy π^*). The specific implementation of $g(\mathbf{s}_t)$ in DAgger is using $\delta_k = (\delta_0)^k$, where $\delta_0 \in [0, 1]$ is the initial probability of allocating expert mode:

$$g(\mathbf{s}_t; \delta_k) = \begin{cases} 1, & \text{if } o_t \leq \delta_k \\ 0, & \text{otherwise} \end{cases}, \tag{2.6}$$

where, random variable o_t is sampled from the uniform distribution at each time step t as $o_t \sim U(0, 1)$, in turn, a meta policy π_k^M randomly selects robot control mode. At the end of each iteration k, additional data, where states visited by a meta policy π_k^M and actions given by expert policy π^*, added to the datasets $\mathcal{D}$ used to learn the policy π_{k+1}^R. A detail of the method is shown in Algorithm 1. In this, policy optimization convergence is theoretically guaranteed [18]. Accordingly, DAgger-like on-policy approaches can alleviate the covariate shift problem and achieve better performance with smaller training datasets size [19].

However, this approach has limited applicability in practice due to the risk of deploying an unmatured learned robot. This situation is especially problematic in clearance-limited tasks (Figure 1.2), because the risk of collisions can greatly hinder task performance and significantly damage robots. In Chapter 3, we present

Algorithm 1 Dataset Aggregation (DAgger) [18]

```
Input: initial data set of expert's demonstrations $\mathcal{D}$,
    initial probability of expert mode allocation $\delta_0$,
    the maximum number of iteration $K$
Output: $\pi_K^R$
1: for $k = 1$ to $K$ do
2:     Get dataset through the stochastic mixed policy $\pi_k^M$: $\{a_t^*, \mathbf{s}_t\}_{t=1}^N \sim p(\tau \mid \pi_k^M)$.
3:     Aggregate datasets :$\mathcal{D} \leftarrow \mathcal{D} \cup \{a_t^*, \mathbf{s}_t\}_{t=1}^N$.
4:     Train the policy $\pi_{k+1}^R$ on $\mathcal{D}$ by policy optimization of BC as (2.3).
5: end for
```

a novel on-policy IL to estimate the environmental precision as a risk of policy deployment and prompt human intervention control in states where high risk is estimated for collision risk mitigation, instead of randomly switching control mode such as DAgger.

2.3. Disturbance-injected Imitation Learning

Another valuable IL branch for reducing covariate shift is disturbance-injected IL, which adds artificial disturbances to the human behavioral commands during the demonstration to generate a richer dataset (Figure 2.1-Disturbance-injected IL). In this context, disturbance injection simulates errors that may occur during task execution to induce human recovery actions in the demonstration. This scheme allows the robot to learn policies robust to diverse sources of errors. Disturbances for Augmenting Robot Trajectories (DART) is a representative method of disturbance-injected IL, where the level of disturbance injection is optimized to reduce the covariate shift between the collected demonstration data and the predicted trajectory [20]. The disturbance distribution is optimized iteratively during the data collection process. Finally, the robust policy is learned using the collected data.

This injection disturbance is assumed that sampled from a Gaussian distribution as $\epsilon_t \sim \mathcal{N}(0, \sigma_k^2)$, where k is the number of optimization iterations. The injection disturbance ϵ_t is added into the expert's action a_t^*. The distribution of trajectories from a disturbance injected expert, is denoted as $p(\boldsymbol{\tau} \mid \pi^*, \sigma_k^2)$ and the distribution of trajectories from a learned policy is $p(\boldsymbol{\tau} \mid \pi_k^R)$. To reduce the covariate shift, DART proposes to use the upper bound of covariate shift by Pinsker's inequality as:

$$\begin{aligned}&\left|\mathbb{E}_{p(\boldsymbol{\tau}|\pi^*,\sigma_k^2)}\left[J(\pi^R,\pi^* \mid \boldsymbol{\tau})\right] - \mathbb{E}_{p(\boldsymbol{\tau}|\pi_k^R)}\left[J(\pi^R,\pi^* \mid \boldsymbol{\tau})\right]\right| \\ &\leq T\sqrt{\frac{1}{2}\mathrm{KL}\left(p(\boldsymbol{\tau} \mid \pi_k^R) \mid\mid p(\boldsymbol{\tau} \mid \pi^*, \sigma_k^2)\right)},\end{aligned} \tag{2.7}$$

where, KL$(\cdot \mid\mid \cdot)$ is Kullback-Leibler divergence. However, the upper bound (2.7) is analytically intractable to compute since the trajectory distribution of learned policy $p(\boldsymbol{\tau} \mid \pi_k^R)$ is unknown. Therefore, DART solves the upper bound by replacing the trajectory distribution of the learned policy with the trajectory distribution of the disturbance-injected expert. As such, data are collected over several iterations and a disturbance distribution is optimized at each iteration as:

$$\begin{aligned}\sigma_{k+1}^2 = \arg\max_{\sigma^2} &\, \mathbb{E}_{p(\boldsymbol{\tau}|\pi^*,\sigma_k^2)} \\ &\left[\sum_{t=0}^{T-1} \mathbb{E}_{\pi_k^R(a_t'|\mathbf{s}_t)}\left[\log \mathcal{N}\left(a_t' \,\middle|\, a_t, \sigma^2\right)\right]\right],\end{aligned} \tag{2.8}$$

where, a learned policy at k th iteration π_k^R is obtained in the similar form as (2.3) by following:

$$\pi_k^R = \arg\max_{\pi} \sum_{i=1}^{k-1} \mathbb{E}_{p(\boldsymbol{\tau}|\pi^*,\sigma_i^2)}[J(\pi,\pi^* \mid \boldsymbol{\tau})]. \tag{2.9}$$

Although DART can reduce the covariate shift by injecting disturbances into expert demonstrations, its applicability still suffers from the following issues. The applied policy model is deterministic, which means that it cannot recognize complex human behavior (*e.g.,* multiple optimal actions) from the training data. Additionally, as shown in Figure 2.2-(a), the disturbance is injected uniformly regardless of the current state of the robot, which may induce dangerous situations (*e.g.,* physical contacts as in Figure 1.2-bottom). Furthermore, the disturbance level optimization (2.8) corresponds to the maximum likelihood estimation based on the assumption of a deterministic policy model and a fixed disturbance level parameter; thus, non-parametric policy learning (*e.g.,* [21]) in which effectively captures multiple optimal actions without requiring the specified number of optimal actions in each state, cannot be directly integrated into

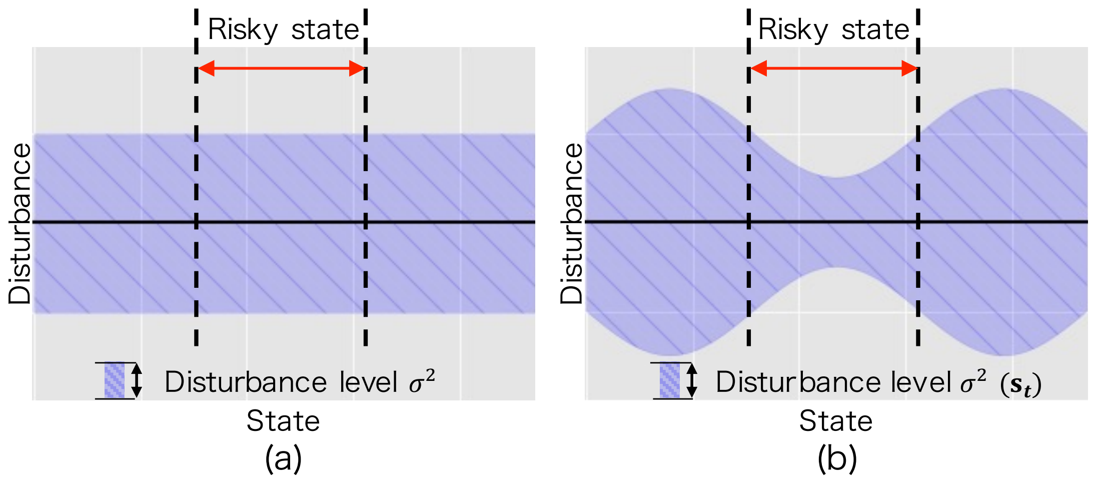
Risky state
Disturbance
Disturbance level σ^2
State
(a)
Risky state
Disturbance
Disturbance level σ^2 $(\boldsymbol{s_t})$
State
(b)

Figure 2.2. Illustration of the comparison between (a) state-independent and (b) state-dependent disturbance models. (a): a constant disturbance level regardless of the risk of the state, may be dangerous in risky states. (b): the disturbance level can be modified according to the state; *e.g.*, in risky states the disturbance level is reduced.

the DART framework. Therefore, a scheme to resolve these issues simultaneously via non-parametric Bayesian inference is derived in Chapter 4.

3. Leveraging Human Risk Sensitivity to Improve Safety of On-policy Imitation Learning

3.1. Safety Issue of On-policy Imitation Learning

As described in the Chapter 2.2, on-policy imitation learning allows a robot to learn covariate shift minimized policy efficiently [22]. This is achieved by repeatedly optimizing a robot while it performs a task with its unmatured policy a human provides corrective demonstrations while observing its execution.

However, deploying unmatured policies poses significant risks. In particular, in clearance-limited tasks, such as aperture-passing and ring-threading, a robot's small mistakes may lead to collisions and break objects in the environment. In order to ensure safety, a robot must recognize the risk of a collision and take appropriate steps to avoid it by stopping performing a task and allowing human intervention. In robotics, detecting collision risks requires precision information, such as the narrowness of the environment, which provides the spatial context of collisions while a robot executes a task. Although precision can be obtained with a model of the environment, on-policy IL is the model-free scheme, and this contradictive situation limits its applicability significantly.

3.2. Existing Methods to Improve Safety of On-policy Imitation Learning

Several approaches have been investigated to improve the safety of on-policy IL by measuring the risk of executions in given states and requesting intervention by ceding control to a human demonstrator when encountering risky states. One approach to guarantee on-policy learning safety is using the risk awareness of humans based on their understanding of the environment or tasks. Humans continuously monitor a robot's execution and intervene when a robot encounters risky states and provide corrective action [23–25] or reset the task execution [26]. However, these approaches have the constraint of forcing users to constantly monitor the robot.

Instead of continuous human monitoring, other approaches have been investigated that leverage robotic risk awareness based on their policy analysis [27–30]. These approaches allow a robot to quantify the execution risks and actively ask a human to intervene when the risk exceeds a threshold. Previous research defined risk indicators as the uncertainty of a robot's decision about the visited state [27] or the discrepancy between the actions proposed by a robot's policy and a human expert [28]. However, neither metric can detect collision risks since a robot still lacks precision information of its environment. Although Hoque et al. introduced a precision estimation metric [29], it requires a robot to experience hundreds of collisions by itself to optimize the precision estimator; thus, this metric is limited in practical application. Alternatively, this paper explores implicitly estimating environmental precision from human demonstrations without requiring collision experiences.

3.3. Speed-accuracy Trade-off in Clearance-limited Tasks

During human demonstrations of clearance-limited robotic tasks, humans can perceive environmental precision based on their understanding of a domain and its environment and carefully regulate a robot's speed through constrained spaces (*e.g.,* obstacles) to avoid collisions [15]. This phenomenon has been extensively examined in neuroscience to identify the human balance between speed and accuracy, commonly called the speed-accuracy trade-off [16]. This idea has also been studied in robotics to efficiently complete tasks while ensuring collision avoidance. In a path-planning context, speed-accuracy cost-maps have been achieved by providing explicit environmental dynamics models [31] and incorporating a heuristic search algorithm [32].

On the other hand, this thesis explores the idea of identifying environmental precision from human demonstrations in a model-free manner to improve the safety of on-policy IL. Instead of assuming an environmental dynamics model, we use the human speed-accuracy trade-off to capture the *demonstrator-perceived precision* and use this precision as an indirect indicator of environmental precision. To that end, we employ a leader-following teleoperation system where a robot directly follows human hand movements, and the human's speed-accuracy trade-off is directly reflected in demonstrations. This allows the speed of a robot controlled by a human to reflect the demonstrator-perceived precision.

3.4. Proposed Method

In this section, we present a novel on-policy learning approach that incorporates demonstrator-perceived precision as intervention criteria to improve safety during on-policy learning (Figure 3.1): Demonstrator-perceived Precision-aware On-policy Imitation Learning (DP-OnIL). We introduce a precision estimator that learns to capture such speed distribution from demonstrations and approximates the precision for given states. Since DP-OnIL solicits human intervention in states where the estimated precision must be extremely high (*i.e.,* risk of collisions is excessive), DP-OnIL enhances the safety of on-policy IL.

In the following, Chapter 3.4.1 describes how the demonstrator-perceived precision is estimated based on the speed-accuracy trade-off exhibited by humans, Chapter 3.4.2 introduces the collision-risk-estimation metric from both the precision and the uncertainty analysis of a learned policy, Chapter 3.4.3 introduces an intervention design for mitigating collision risks, and Chapter 3.4.4 describes DP-OnIL's overall algorithmic procedure.

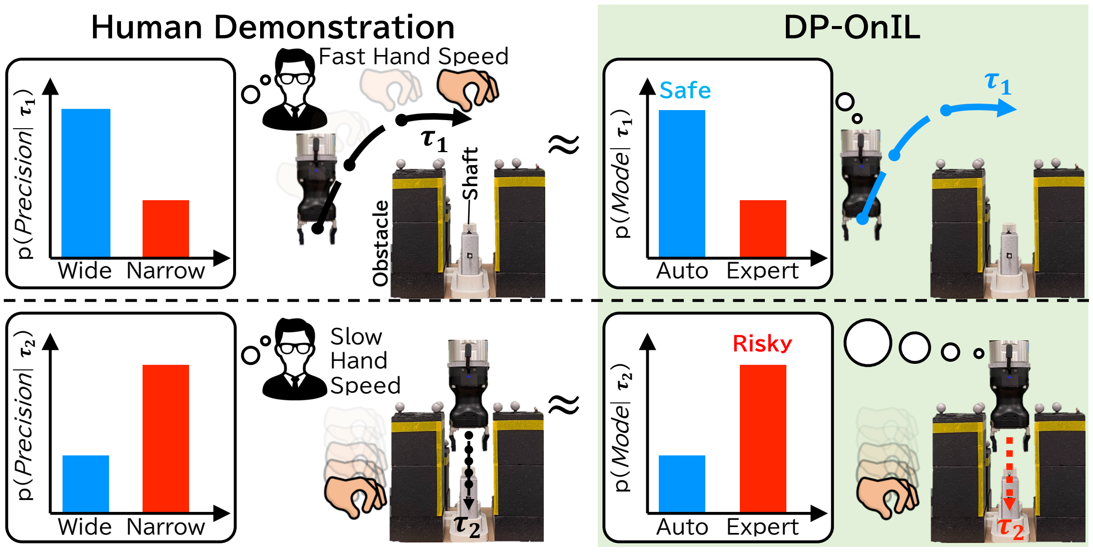

Figure 3.1. Overview of Demonstrator-perceived Precision-aware On-policy Imitation Learning (DP-OnIL). In clearance-limited tasks, demonstrator-perceived precision is in the mind of humans. By capturing this precision level from demonstration data and incorporating it into on-policy IL, a robot can cede control to a human (expert mode, bottom) in high-precision areas while executing its policy (auto mode, top) in low-precision areas, thus enhancing safety.

3.4.1. Demonstrator-perceived Precision Estimation

First, we defined speed transformation function f_v, which computes speed v_t from a pair of states along 1-step transitions: $f_v(\mathbf{s}_t, \mathbf{s}_{t+1}) = v_t$. For the following formulation, v_t is given by f_v. Under this speed definition, human speeds are corrupted by state-dependent noise [33], whose variance increases with the size of the input actions during demonstrations. Such variations in demonstrations are called *aleatoric uncertainty*, and a natural way to capture this uncertainty is to use a probabilistic neural network regression model [34] that consists of two neural networks predicting the mean and variance (*i.e.,* aleatoric uncertainty), respectively. Specifically, the speed estimator is defined as $V_\lambda(v_t|\mathbf{s}_t)$, which outputs the Gaussian distribution with mean network $\mu_\lambda(\mathbf{s}_t)$ and variance network $\sigma^2_\lambda(\mathbf{s}_t)$ for a given state $\mathbf{s}_t$ with parameter λ:

$$V_\lambda(v_t|\mathbf{s}_t) = \mathcal{N}(v_t|\mu_\lambda(\mathbf{s}_t), \sigma^2_\lambda(\mathbf{s}_t)). \tag{3.1}$$

In practice, training dataset $\mathcal{D}$ for involving human speeds v^*_t can be calculated by f_v using transition $(\mathbf{s}_t, \mathbf{a}^*_t, \mathbf{s}_{t+1})$ of a human expert's trajectory: $\mathcal{D} = \{\mathbf{a}^*_t, \mathbf{s}_t, v^*_t\}^T_{t=1}$. For learning probabilistic speed estimator $V_\lambda(v_t|\mathbf{s}_t)$ in an imitation learning context, negative log-likelihood loss L of the estimator is defined:

$$L(V_\lambda|\mathcal{D}) = \sum_{t=1}^{T} -\log \mathcal{N}(v^*_t|\mu_\lambda(\mathbf{s}_t), \sigma^2_\lambda(\mathbf{s}_t)). \tag{3.2}$$

Therefore, the speed estimator's parameter λ is optimized by minimizing the expected loss along the training dataset:

$$\lambda' = \arg\min_\lambda \mathbb{E}_{\mathcal{D}\sim p(\boldsymbol{\tau}|\pi_{\theta^*})}[L(V_\lambda|\mathcal{D})]. \tag{3.3}$$

Due to the speed-accuracy trade-off of humans [16], in narrow areas, the human speed mean and variance are decreased. For this human behavior, there are two types of modeling possibilities for precision estimator $\mathrm{Pre}_{\lambda'}(\mathbf{s}_t)$:

- $\mathrm{Pre}^{\mu}_{\lambda'}(\mathbf{s}_t) = \{\mu_{\lambda'}(\mathbf{s}_t)\}^{-1}$, where the precision is inversely proportional to the estimated speed's mean;
- $\mathrm{Pre}^{\mathrm{UCB}}_{\lambda'}(\mathbf{s}_t) = \{\mu_{\lambda'}(\mathbf{s}_t) + \sigma_{\lambda'}(\mathbf{s}_t)\}^{-1}$, where the precision is inversely proportional to the estimated speed's Upper Confidence Bound (UCB), which is the sum of the mean and the standard deviation.

Implementing the former type is simpler, although it is expected to be less sensitive for capturing demonstrator-perceived precision than the latter type, which consider speed variance simultaneously. The DP-OnILs used for each precision model are defined as DP-OnIL$_\mu$ and DP-OnIL$_{\mathrm{UCB}}$.

3.4.2. Collision Risk Estimation

To estimate the collision risk, the robot must analyze not only the environment's precision but also the uncertainty of the learned policy for performing the task. Such policy uncertainty, called *epistemic uncertainty*, stems from a lack of demonstration data and increases the risk that the robot will make unmatured decisions, which may induce collisions.

To account for epistemic uncertainty in a learned policy, its decisions must be analyzed probabilistically based on a given state. Accordingly, we employ an ensemble neural network as a policy model similar to the prior study [27]. As such, each component of the ensemble policies is learned by (2.3). Then the ensemble of learned policies outputs actions for any given state $\mathbf{s}_t$, and variances $\sigma^2_{\theta^L}(\mathbf{s}_t)$ of these actions can be interpreted as the level of epistemic uncertainty in the decision. Finally, to quantify the collision risk by comprehensively evaluating state $\mathbf{s}_t$ regarding both estimated precision $\mathrm{Pre}_{\chi'}(\mathbf{s}_t)$ and the epistemic uncertainty of learned policy $\sigma^2_{\theta^L}(\mathbf{s}_t)$, collision risk $\mathrm{Risk}(\mathbf{s}_t)$ is defined as the product of both factors:

$$\mathrm{Risk}(\mathbf{s}_t) = \mathrm{Pre}_{\chi'}(\mathbf{s}_t) \cdot \sigma^2_{\theta^L}(\mathbf{s}_t). \tag{3.4}$$

3.4.3. Intervention Design

Deciding intervention using the collision risk estimation of Chapter 3.4.2 is introduced to improve the safety of the on-policy learning. To prompt human intervention triggered by collision risk, decision function $g(\mathbf{s}_t; \chi)$ is defined that is activated when $\mathrm{Risk}(\mathbf{s}_t)$ exceeds threshold χ:

$$g(\mathbf{s}_t; \chi) = \begin{cases} 1, & \text{if } \mathrm{Risk}(\mathbf{s}_t) > \chi \\ 0, & \text{otherwise} \end{cases}, \tag{3.5}$$

Algorithm 2 DP-OnIL

```
Input: Number of iterations $K$, threshold $\chi$
Output: Parameter of learned policy $\theta^L_K$, parameter of precision estimator $\lambda_K$
1: Get the initial dataset through a human expert:
   $\mathcal{D} = \{\mathbf{a}^*_t, v^*_t, \mathbf{s}_t\}^T_{t=1} \sim p(\boldsymbol{\tau} \mid \pi_{\theta^*})$
2: Initialize $\theta^L_0$ and $\lambda_0$ by (2.3) and (3.3) on $\mathcal{D}$
3: for $k = 1$ to $K$ do
4:     Get the dataset through a meta-policy:
       $\{\mathbf{a}^*_t, v^*_t, \mathbf{s}_t \mid g(\mathbf{s}_t, \chi) = 1\}^T_{t=1} \sim p(\boldsymbol{\tau} \mid \pi_{\theta^M_k})$
5:     Aggregate datasets:
       $\mathcal{D} \leftarrow \mathcal{D} \cup \{\mathbf{a}^*_t, v^*_t, \mathbf{s}_t \mid g(\mathbf{s}_t, \chi) = 1\}^T_{t=1}$
6:     Learn $\theta^L_k$ and $\lambda_k$ by (2.3) and (3.3) on $\mathcal{D}$
7: end for
```

which indicates whether state $\mathbf{s}_t$ is safe ($g(\mathbf{s}_t; \chi) = 0$) or risky ($g(\mathbf{s}_t; \chi) = 1$) regarding collisions. During a robot's training phase (Figure 3.2-top), this decision function allows a robot to request human intervention (*i.e.,* expert mode) only in risky state $\mathbf{s}_t$ while deploying a learned policy (*i.e.,* auto mode) during the others.

3.4.4. DP-OnIL Overview

This section describes DP-OnIL's algorithmic flow. As shown in Figure 3.2, the robot's policy is learned by iterating two phases: (i) generating training datasets through on-policy data collection with collision risk intervention criteria (Figure 3.2, top), and (ii) learning the robot's policy and the precision estimator using the accumulated training datasets (Figure 3.2, bottom).

Specifically, an initial dataset, $\mathcal{D} = \{\mathbf{a}^*_t, \mathbf{s}_t, v^*_t\}^T_{t=1}$, is only collected by an expert's policy π_{θ^*}. The initial parameters of policy θ^L_0 and precision estimator λ_0 are obtained by optimizing (2.3) and (3.3) on $\mathcal{D}$. Under this initialization, meta-policy $\pi_{\theta^M_k}$ collects a training dataset for K iterations, as described in Chapter 3.4.3. At each k th iteration, θ^L_{k-1} is used for the robot policy in meta-policy $\pi_{\theta^M_k}$. The states that are performed in the expert mode and the expert's actions

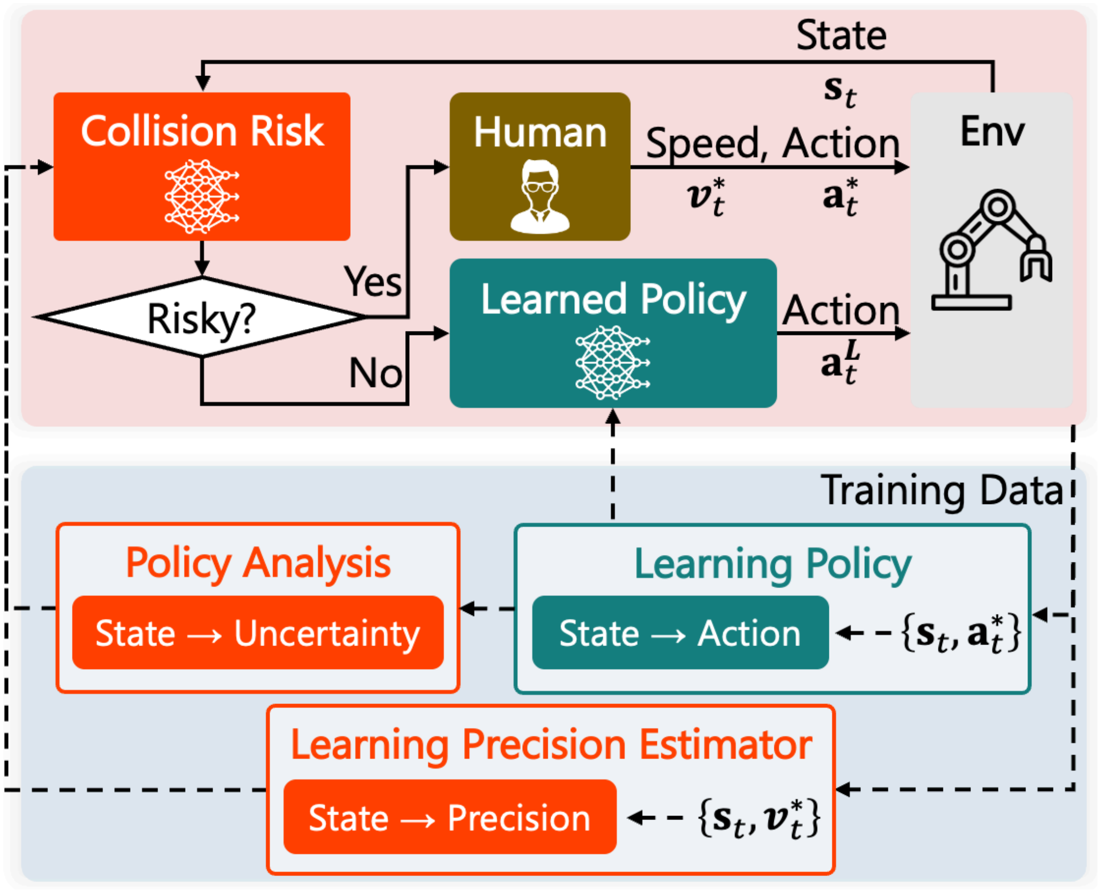

Figure 3.2. Overview of on-policy learning with DP-OnIL: (top): While a robot is executing a task with its policy, if $\mathbf{s}_t$ is too risky, a human controls it until the risk is sufficiently lowered. (bottom): Policy and precision estimator are iteratively learned from training data accumulated through on-policy data collections. Collision risk is computed with analyzed uncertainty of learned policy and estimated precision.

and speeds are collected during each k th iteration:

$$\mathcal{D}_k = \{\mathbf{a}_t^*, \mathbf{s}_t, v_t^* \mid g(\mathbf{s}_t; \chi) = 1\}_{t=1}^{T} \sim p(\boldsymbol{\tau} \mid \pi_{\theta_k^M}). \tag{3.6}$$

These collected data are added to dataset $\mathcal{D}$: $\mathcal{D} = \mathcal{D} \cup \mathcal{D}_k$. After each iteration, the parameters of learned policy θ_k^L and precision estimator λ_k in the k th iteration are optimized using equations (2.3) and (3.3)on accumulated dataset $\mathcal{D}$. A summary of DP-OnIL is shown in Algorithm 2.

3.5. Simulation

In this section, we validated whether our proposed method can effectively achieve an automation performance of a robot more safely than the prior algorithms in the following two simulation domains: (i) an aperture-passing task (Figure 3.3) and (ii) a ring-threading task with a 6-DOF UR5e robot (Figure 3.6).

Evaluation Metrics: The DP-OnIL performance is considered during the training and deployment test phases. For the former, *the on-policy demonstration performance* was evaluated as the probability of the task's success over all the training episodes of the on-policy IL approaches. For the latter, *the robot-autonomous performance* was evaluated as the probability of the task's successful deployment of the learned policy after training without expert assistance. Both metrics were assessed in both simulations (Chapter 3.5.1,Chapter 3.5.2) and real-robot experiments (Chapter 3.6).

Comparison Methods: Our methods (DP-OnIL$_{\mathrm{UCB}}$ and DP-OnIL$_{\mu}$) are compared as a baseline to the following other imitation learning methods:

- Behavior Cloning (BC) [17]: A conventional imitation learning that learns a policy without any interactions;
- Dataset Aggregation (DAgger) [18]: a conventional IIL that randomly requests human intervention;
- EnsembleDAgger [27]: A state-of-the-art IIL that only uses policy decision uncertainty $\sigma^2_{\theta^L}$ as Risk($\mathbf{s}_t$);
- ThriftyDAgger [29]: A state-of-the-art IIL where a precision estimator is learned through collision experiences;

- HG-DAgger [23]: A state-of-the-art IIL where an algorithmic expert decides when to intervene or not.

Our evaluation assumes an example problem where the ratio of states assigned as risky is sufficient and fair across risk-aware approaches (EnsembleDAgger, ThriftyDAgger, and DP-OnIL). To achieve this, we set the threshold χ for each method at the value of approximately the top 20% of the estimated risk in the training dataset, similar to previous works [28–30]. Its sensitivity is analyzed in Chapter 3.5.1. See Chapter A for how the hyperparameters of each method are set.

Demonstration Setting: Initially, speed transformation function f_v is defined by the Euclidean norm of the difference in position-related states $\mathbf{s}_t^{pos} \in \mathbf{s}_t$, which are generally included as the state space of robotic tasks (*e.g.,* positions of agent center or end effector): $f_v(\mathbf{s}_t, \mathbf{s}_{t+1}) = \|\mathbf{s}_{t+1}^{pos} - \mathbf{s}_t^{pos}\|_2^2$. Under this initial setting, demonstrations are provided by an algorithmic expert, especially where a human-like risk-sensitive movement [15] is implemented as shown in Figure 3.3 and Figure 3.6. Such movement is simulated by specifying agent's action for each state: fast in open areas (*e.g.,* far from walls), and slow in small clearance areas (*e.g.,* aperture traversal), while injecting state-dependent Gaussian noise [33] as described in Chapter 3.4.1. For an algorithmic expert in HG-DAgger, the timing of the intervention is also specified to prevent failure during interactions in Chapter 3.5.1.

3.5.1. Aperture-passing Simulation

An aperture-passing task involving multiple narrow apertures was initially performed in the OpenAI gym [35] environment (Figure 3.3. In this experiment, on-policy demonstration and robot-autonomous performances are evaluated in a challenging environment that includes states where such physical contacts are likely to occur as passing through narrow apertures, although no contacts are allowed for task success.

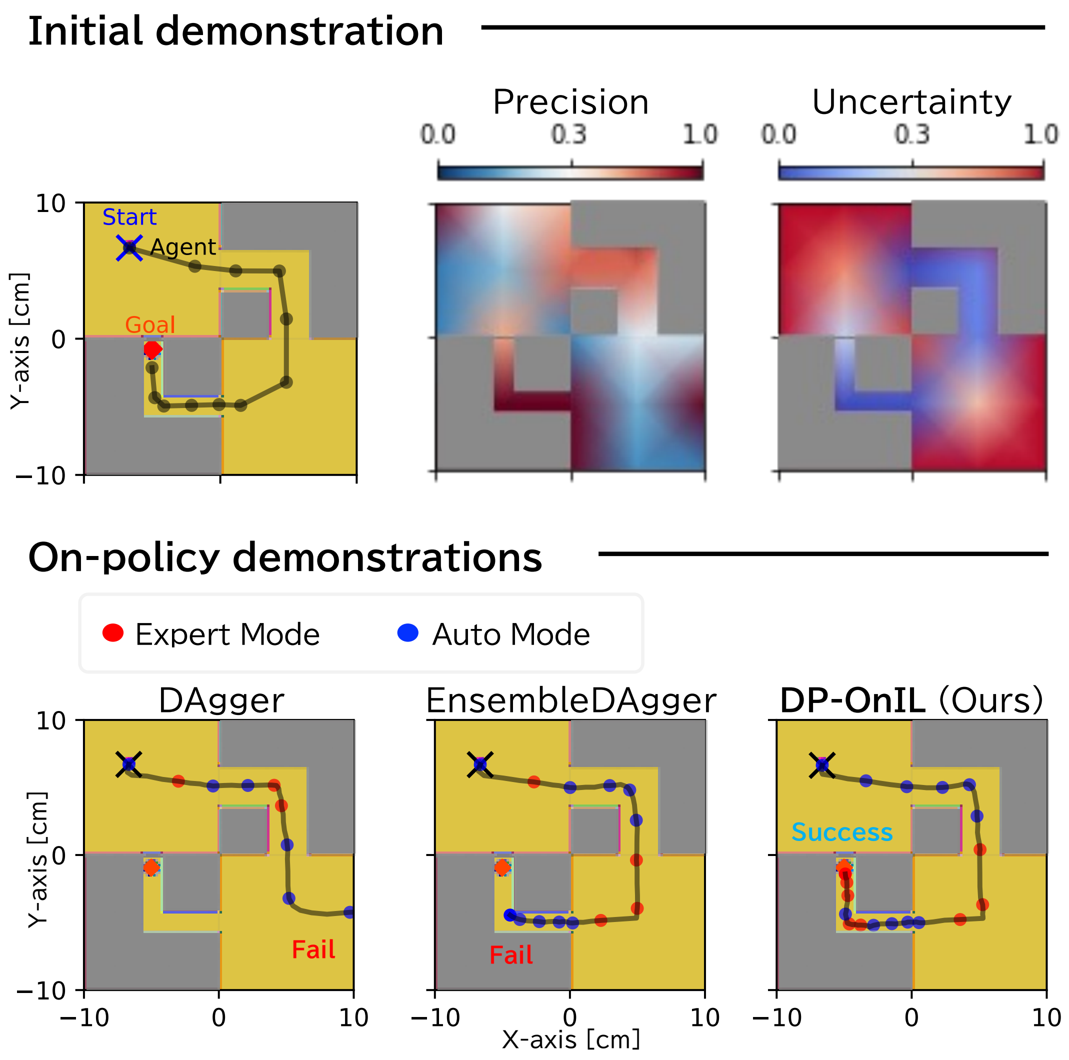

Figure 3.3. Qualitative analysis of aperture-passing simulation (On-policy Demonstration Phase): Uncertainty and precision results across state space are obtained using a policy and a precision estimator learned from initial demonstration dataset. Both measurements are normalized to clarify variations across states. Based on these indicators, on-policy demonstration trajectories of on-policy IL algorithms (DAgger, EnsembleDAgger, DP-OnIL (Ours)) are compared, where red and blue circles represent states with expert and auto modes.

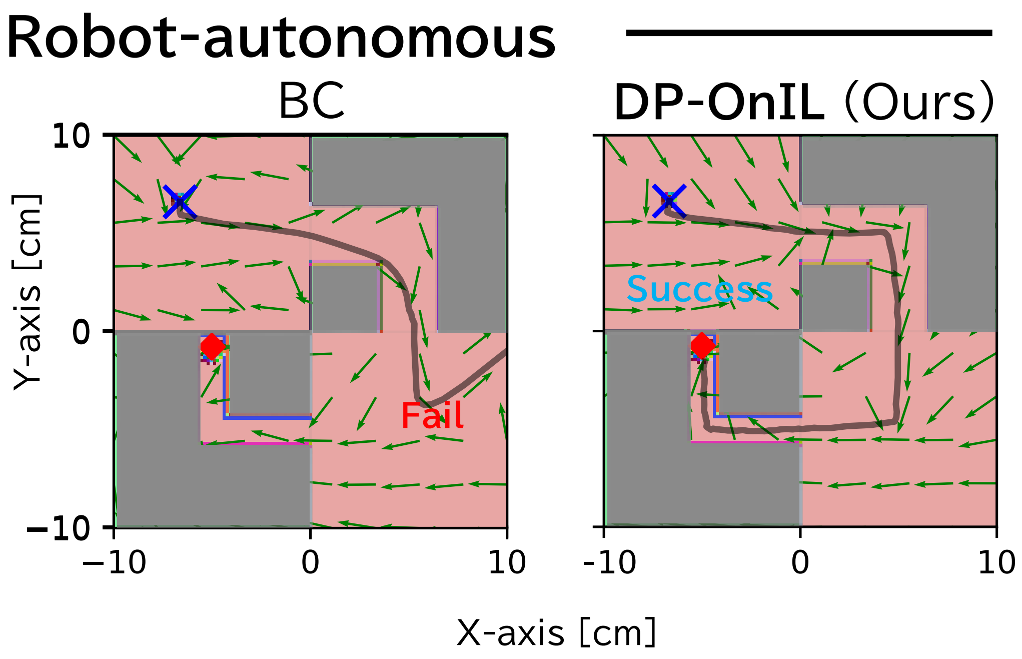

Figure 3.4. Qualitative analysis of aperture-passing simulation (Robot-autonomous Phase): Comparison of the 2D vector fields of the policies learned by BC and DP-OnIL (ours) and their execution trajectories.

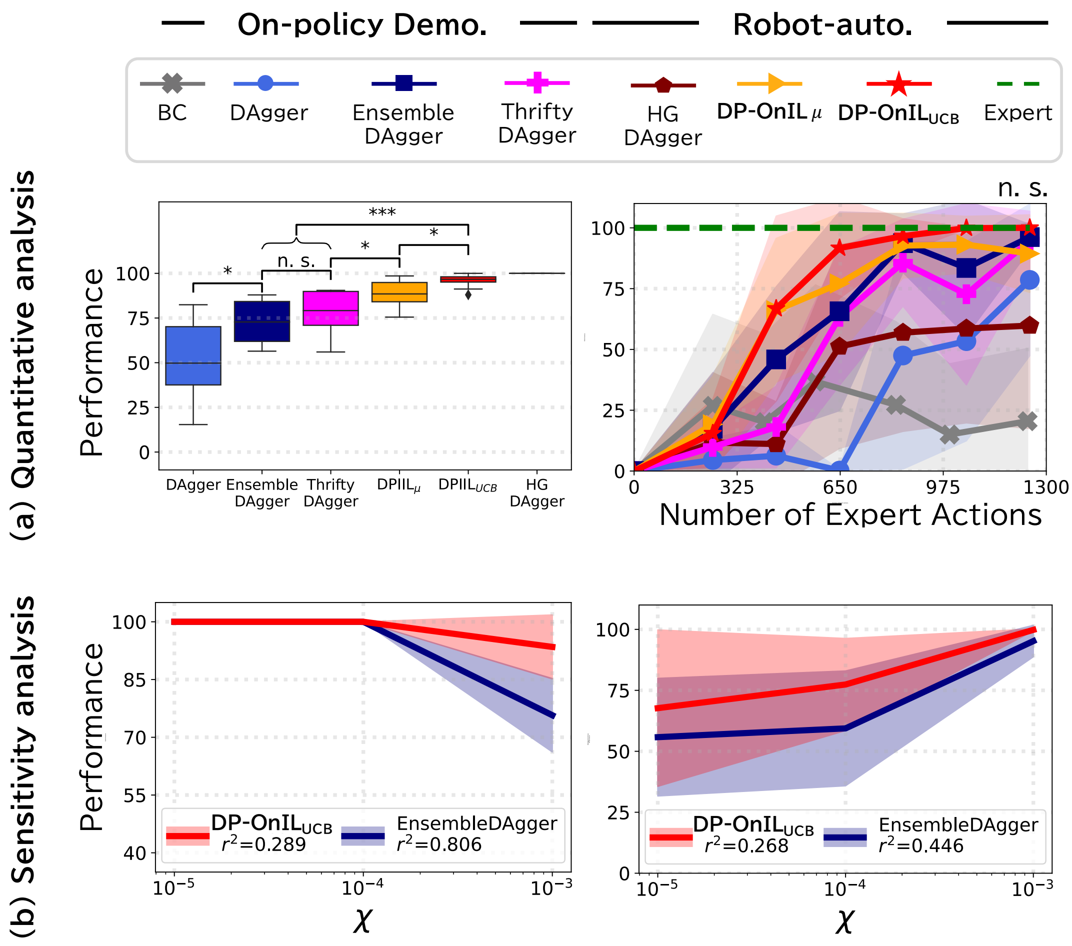

Figure 3.5. Quantitative analysis of aperture-passing simulation: **(a) Quantitative analysis**: Averaged performance of on-policy demonstration (left) and robot-autonomous (right) are evaluated by repeating each experiment 10 times with random seeds. (left): On-policy demonstration performance is measured as a box plot of average success probability during training phases across entire trials of each on-policy IL approach. Significant differences by t-test are observed between proposed method and a baseline ($* : p < 5e{-2}, *** : p < 5e{-4}$). (right): Comparing robot-autonomous performance for number of expert actions used to train by conducting 100 test episodes of each learned policy. No significant differences by t-test are observed between our methods and a state-of-the-art on-policy IL (EnsembleDAgger). **(b) Sensitivity analysis**: On-policy demonstration and robot-autonomous performances are measured as χ values fixed at $\chi \in [10^{-5}, 10^{-3}]$ for each experiment; square of correlation coefficient r^2 [36] between hyperparameter χ and each performance is measured as sensitivity indicator.

Task Setting

The task goal is to move the agent (black circles with a 0.25 cm radius) clock-wise from the starting position through the apertures (each of which has a width of 3.0 cm and 1.5 cm sequentially) to the goal without colliding with the walls (gray). The system state and action are the agent's position (*e.g.,* x, y-axis coordinates) and velocity (*e.g.,* x, y-axis). The initial state is deviated by additive uniform noise $\epsilon_{\mathbf{s}_0} \sim U(-2\text{ cm}, 2\text{ cm})$.

Learning Setting

Under these experimental parameters, we collected three initial demonstration trajectories (248 state-action pairs) by the expert policy for all the comparisons. This dataset is used to optimize initial learned policy $\pi_{\theta^L_0}$ and precision estimator Pre_{λ_0} until (2.3) and (3.3) converge. For DP-OnIL and each on-policy IL comparison method, an on-policy demonstration is performed with a meta-policy that switches the control between the expert and the learned policy, only collecting state-action pairs in which the expert controlled (*i.e.,* expert mode). After collecting 100 state-action pairs, the policy and precision estimator were updated by optimizing equations (2.3) and (3.3) on the accumulated dataset. If the agent collides with a wall, fails to reach the goal position within the time limit (200 steps), or moves beyond the task space, it is considered a failure. This process is denoted as one k iteration in Algorithm 2 and is repeated $K = 5$ times in this experiment. For BC, demonstration datasets are additionally provided by expert policy only until the number of expert actions is roughly equivalent to the other on-policy IL algorithms.

Result

The results are shown in Figure 3.3, Figure 3.4, and Figure 3.5.

Qualitative Analysis: The on-policy demonstration trajectories of the on-policy IL methods are compared in (Figure 3.3). In DAgger, the timing of an expert's intervention is randomly decided during on-policy demonstrations. Even if the agent has drifted away from the demo trajectories, expert intervention may not be requested timely, leading to failures (*e.g.,* leaving the task space). EnsembleDAgger requests expert intervention when the uncertainty of the policy decision is high due to a lack of demo data. Although this intervention design allows the robot to avoid drastic deviations from the demo trajectories, it cannot detect a collision risk in narrow states where slight deviations are unacceptable; expert intervention is not requested, resulting in failure (*e.g.,* collisions). In contrast, our method (DP-OnIL) implicitly estimates the precision of the environment by observing the expert's demonstrations. When the estimated precision is applied to detect the collision risk, expert interventions are encouraged in narrow states, resulting in successful interventions that avoid collisions.

In terms of robot-autonomous performance, learned policies of BC and DP-OnIL are compared in (Figure 3.4). In BC, the policy learned only near the initial trajectories, accumulating errors and failing execution. In contrast, DP-OnIL can train the policy that recovers to the initial trajectory through interaction, resulting in successful execution.

Quantitative Analysis: Our methods are compared with other baseline schemes in terms of the interactive and robot-autonomous performances (Figure 3.5(a)). In terms of interaction performance, DAgger had poor performance (52%) since its robot cannot be aware of any risks during the learned-policy execution. Although EnsembleDAgger has better performance (73%) by considering the uncertainty of policy decisions and promoting expert intervention in highly uncertain states, it has next poor performance since it does not ask experts to intervene in states where collisions may occur, as predicted by our qualitative analysis. Despite utilizing precision estimation, ThriftyDAgger performs (78%) similarly to EnsembleDAgger since it requires sufficient collision experience to estimate precision properly. In comparison, both our methods (DP-OnIL$_\mu$ and DP-OnIL$_{\text{UCB}}$) had significantly better performances (89% and 96%) than the others by using precision estimation without the collision experience, nearing the performance of an oracle (HG-DAgger) where an algorithmic expert decides when to intervene optimally.

In terms of robot-autonomous performance, BC performed poorly (21%) as predicted by our qualitative analysis. HG-DAgger monotonically increases the performance of the learned policy, but its performance is the worst (60%) among the On-policy IL methods. This is because the conservative expert repeatedly intervenes in a certain area and cannot generalize to a wider range of states. The next worst On-policy IL method is DAgger (79%), since if the robot fails the task during the interactive demonstrations, it won't be able to continue training on the rest of the task progress, reducing the efficacy of interactive learning. In contrast, risk-aware approaches can significantly improve performance (EnsembleDAgger: 96%, ThriftyDAgger: 95%, DP-OnIL$_\mu$: 89%). One of our methods (DP-OnIL$_{\mathrm{UCB}}$) had the best performance (100%) across all the iterations, suggesting that DP-OnIL increases the interaction safety and ensures efficiency.

Sensitivity Analysis of χ: We analyzed and compared hyperparameter χ's sensitivity from the prior risk-aware on-policy approach (EnsembleDAgger) and our best method (DP-OnIL$_{\mathrm{UCB}}$) (Figure 3.5(b)). EnsembleDAgger, which uses the uncertainty of the policy decision as risk, is sensitive to χ, and the on-policy demonstration and robot-autonomous performances are mutual trade-offs in a range of $\chi \in [10^{-4}, 10^{-3}]$. In contrast, our method (DP-OnIL$_{\mathrm{UCB}}$), which uses precision that is combined with uncertainty as a collision risk, is more robust to a wider range of χ in the on-policy demonstration performance and has sufficient robot-autonomous performance at $\chi = 10^{-3}$.

3.5.2. Ring-threading Simulation

To evaluate DP-OnIL's scalability, a second experiment was conducted for learning a ring-threading task with a 6-DOF UR5e robot in a Robosuite [37] environment (Figure 3.6). This task has two challenges that surpass an aperture-passing task: (i) various physical contact scenarios (*e.g.,* robot vs. object, object vs. object) can occur dynamically on high dimensional state-action space, and (ii) the ring and robot positions are randomly initialized.

Task Setting

The goal is to grasp a ring with the random initial positions and insert it through a peg with a fixed position, regardless of the physical contact. The dimension of the state is 51D, consisting of the robot's joint angles and the ring's position, and the action is 6D, specifying the end-effector's translation (*e.g.,* x, y, z-axes), rotation (*e.g.,* y, z-axes), and gripper manipulation (*e.g.,* open or closed). See a

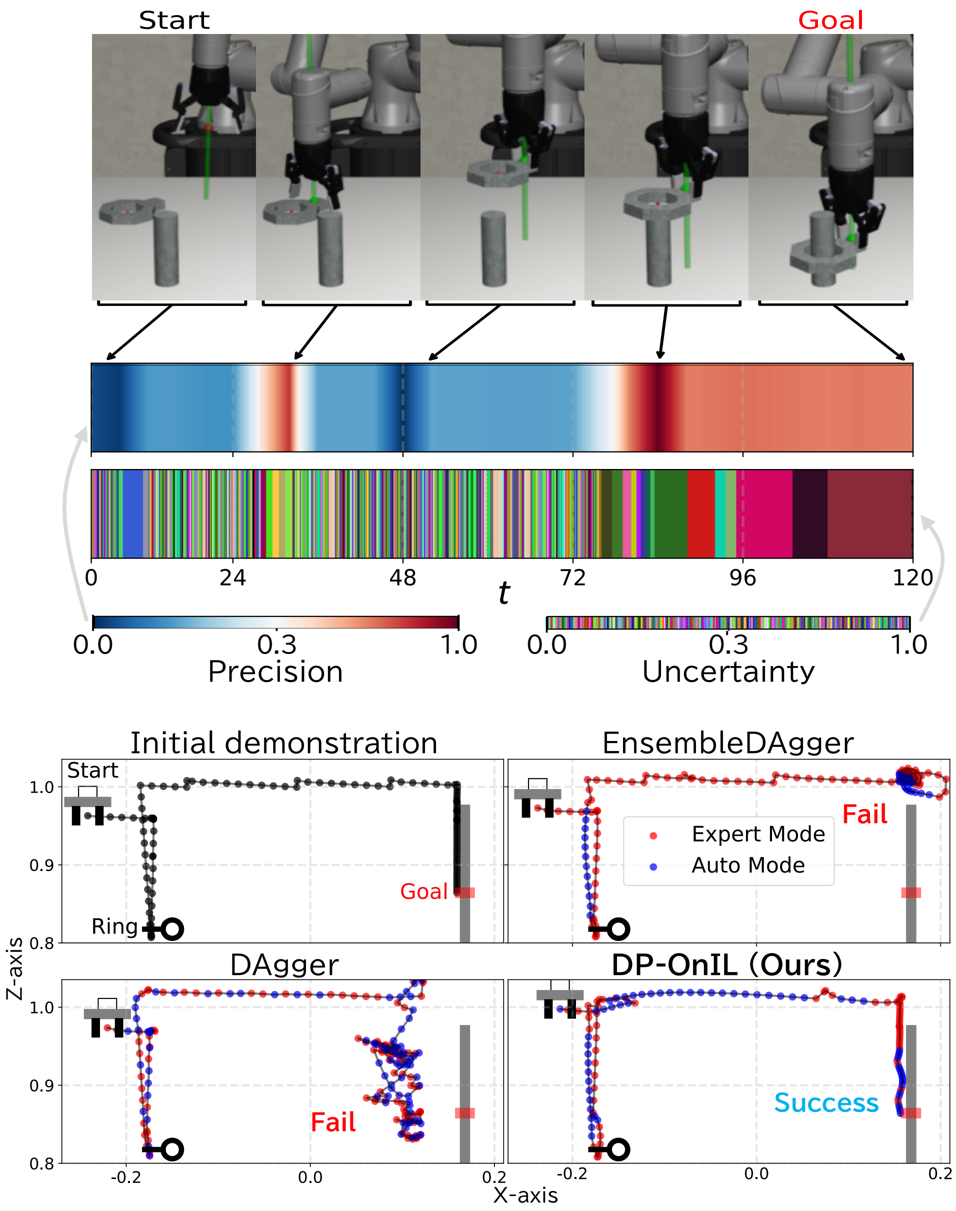

Figure 3.6. Qualitative analysis of ring-threading simulation: Algorithmic expert's demonstration includes two high-precision phases as a robot reaches to grasp a ring and inserts it into a peg. Precision and uncertainty results were obtained by analyzing a demo trajectory's sequence using a precision estimator and a policy learned on the initial demo dataset. Based on this expert, on-policy demonstration trajectories of on-policy IL algorithms (DAgger, EnsembleDAgger, DP-OnIL (Ours)) were compared.

previous work [37] for details about the state composition and randomizing the initial position of the robot and ring.

Learning Setting

The procedure here is similar to Chapter 3.5.1, but due to the task's complexity, the amount of training data is increased by a factor of 10. The number of initial demonstration trajectories collected by the expert policy is 30 (4,414 state-action pairs), and the number of state-action pairs collected by the expert mode in each iteration is 1,000. Accordingly, the amount of training data for BC also increased. In addition, the time limit (200 steps) is this task's only failure condition for evaluating the performances under various physical contacts.

Result

The results can be seen in Figure 3.6 and Figure 3.7.

Qualitative Analysis: We compared the on-policy demonstration trajectories of the on-policy IL methods (Figure 3.6). As described in the previous qualitative analysis (Chapter 3.5.1), the randomized intervention timing of DAgger may induce a robot to fall into a state where the task is infeasible even in the expert mode (*e.g.,* robotic arms getting tangled up), resulting in failure. Although the uncertainty-based intervention of EnsembleDAgger prevents vast deviations from the demo trajectory, it cannot detect precision to request an expert's intervention in high-precision areas, resulting in repeatedly failing to thread a ring due to slight deviations. Contrarily, the DP-OnIL implicitly detects environmental precision from expert demonstrations to promote interventions in high-precision areas (*e.g.,* near a peg), resulting in successful on-policy demonstrations.

Quantitative Analysis: The overall results (Figure 3.7) show a similar trend to the previous task (Chapter 3.5.1), although due to an increase in the task complexity, even DP-OnIL$_{\text{UCB}}$, which had the best performance in the previous results (Figure 3.5(a)), required more than 10 times the amount of training data to exceed the 90% robot-autonomous performance (93.3%). The other methods fail to even surpass 90% despite the extra data. As the amount of training data increased, the overall number of interactions also increased. Our methods

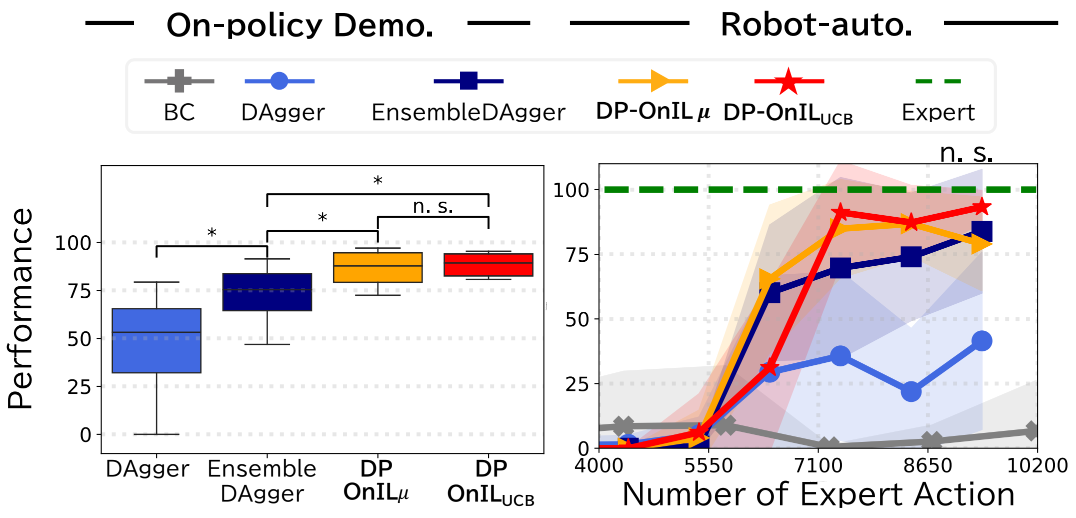

Figure 3.7. Quantitative analysis of ring-threading simulation: Averaged performances of on-policy demonstration (left) and robot-autonomous (right) were evaluated by repeating each experiment 10 times with random seeds. (left): On-policy demonstration performance is measured as a box plot of average success probability during training phases across entire trials of each on-policy IL approach. Significant differences by t-test are observed between proposed method and a baseline ($* : p < 5e{-}2$). (right): Comparing robot-autonomous performance for number of expert actions used to train by conducting 100 test episodes of each learned policy. No significant differences by t-test are observed between our methods and a state-of-the-art on-policy IL (EnsembleDAgger).

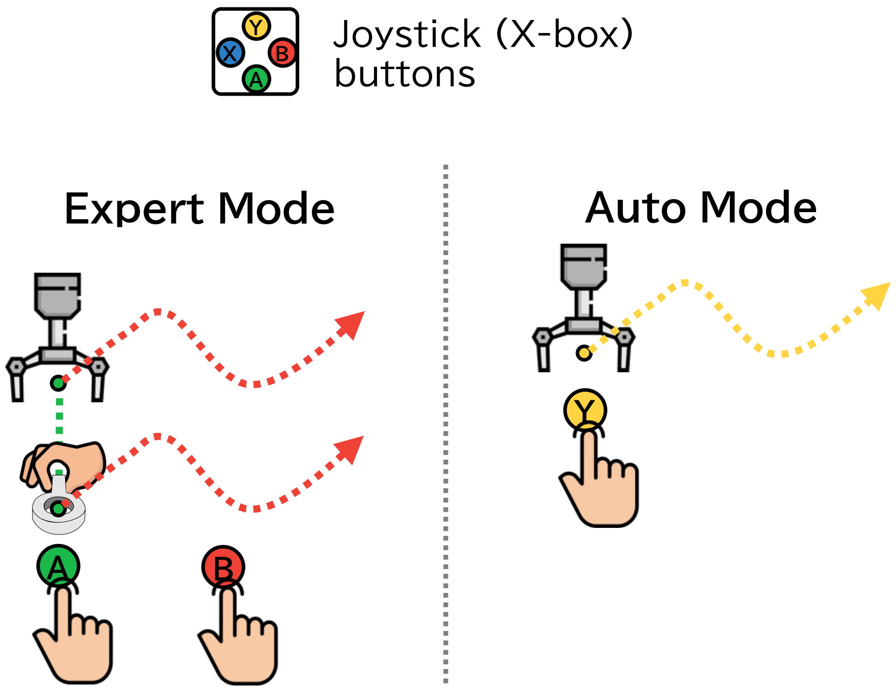

Figure 3.8. Illustration of user interface using buttons on a joystick (X-box). In expert mode, pressing the "A" button synchronizes the position of the robot's end effector with that of the human-held ring. While the "B" button is pressed, the robot follows the movement of the ring. If the "B" button is released, the robot stops moving, and synchronization must be redone by pressing the "A" button again. In auto mode, while the "Y" button is pressed, the robot is moved by learned policy. Note, "Y" button is only set to ensure safety in verification evaluations, not as the requirement of our method (DP-OnIL).

(DP-OnIL$_{\text{UCB}}$ and DP-OnIL$_{\mu}$) still have significantly higher interactive performance, and the other methods have larger variance than the previous results (Figure 3.5(a)) due to increased interactions. These findings suggest that DP-OnIL can effectively address safety concerns in the interactive policy learning of clearance-limited tasks while ensuring efficiency.

3.6. Real-Robot Experiments with Human Experts

In this section, we verified the applicability of our method in various real-world scenarios (Figure 3.9 and Figure 3.10) by conducting an experiment that trains the 6-DOF UR5e robot by human demonstrations of the following two assembly tasks:

(i) a shaft-reaching task: We assessed the robot's skill to reach and grasp a shaft while avoiding fixed obstacles (Figure 3.9). Successfully performing this task within the time limit (150 steps) is challenging since the environment is prone to physical contact (*e.g.,* robot vs. obstacles);

(ii) ring-threading task: We assessed the robot's skill of inserting a ring into a peg without bumping into another peg for the assembly (Figure 3.10). This scenario is more complicated than the shaft-reach task since the clearance for inserting the ring is smaller (only 2 mm), requiring more precise control and a larger time limit (200 steps).

Task Setting

The system state dimension is 12D, which consists of the robot's joint angles and the 3D coordinates of its arm and each task's target assembly part (*e.g.,* a shaft, a peg). Markers are attached to each object (*e.g.,* a shaft, a peg, obstacles) and its coordinates are captured by a motion capture system (OptiTrack Flex13). An action is defined as the velocity of the robot arm in the x, y, and z-axes. The initial robot end-effector position is deviated with additive uniform noise: (i) the shaft-reaching task: $\epsilon_{\mathbf{s}_0} \sim U(-0.05 \text{ m}, 0.05 \text{ m})$, and (ii) the ring-threading task: $\epsilon_{\mathbf{s}_0} \sim U(-0.02 \text{ m}, 0.02 \text{ m})$.

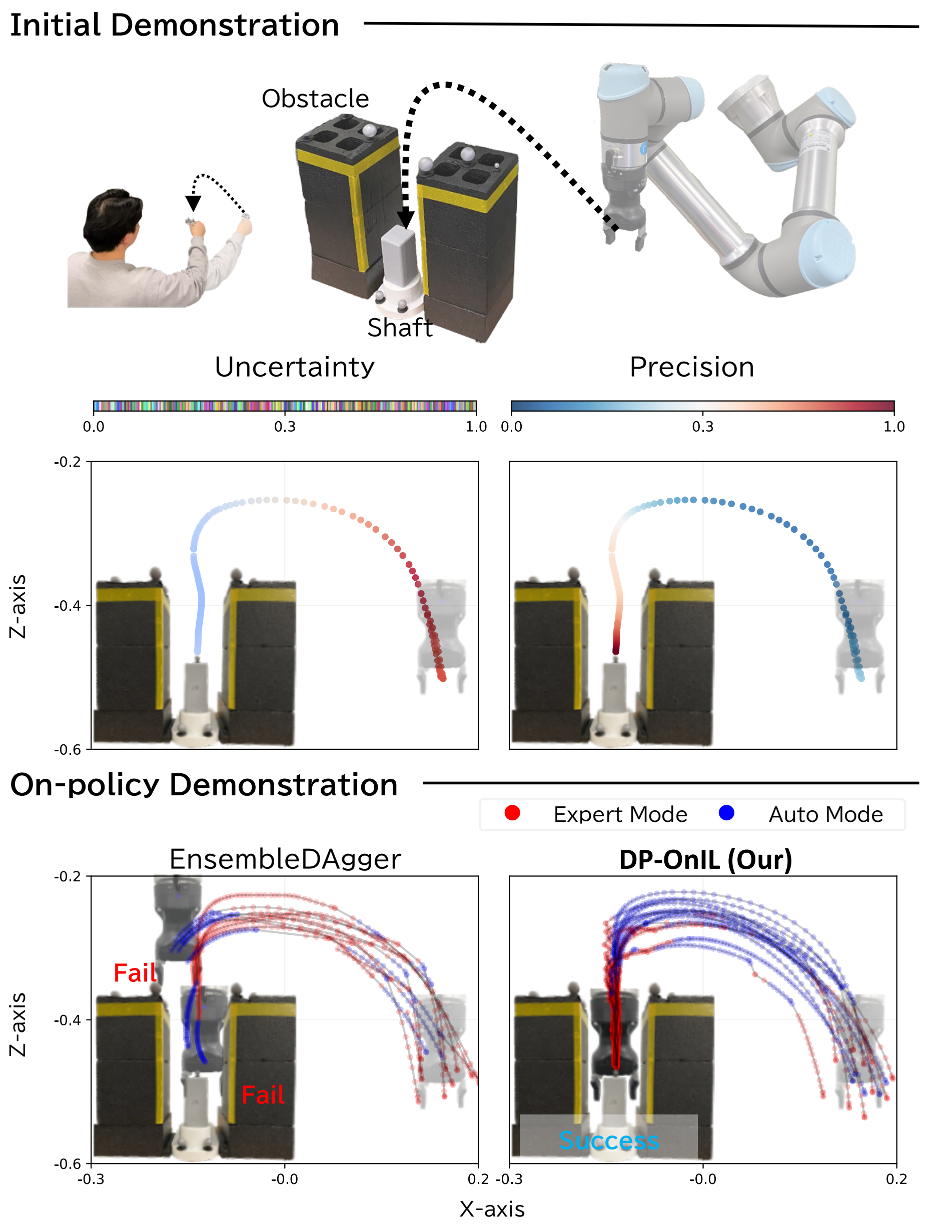

Figure 3.9. Real-robot experiments of the shaft-reaching task: Experimental environments were conducted for a 6-DOF robotic arm (UR5e) reaching a shaft by avoiding obstacles. Precision and uncertainty results were obtained by analyzing initial demonstration trajectories and normalized to visualize variations across states. On-policy demonstrations of EnsembleDAgger and DP-OnIL (Ours) show trajectories at the on-policy learning phase.

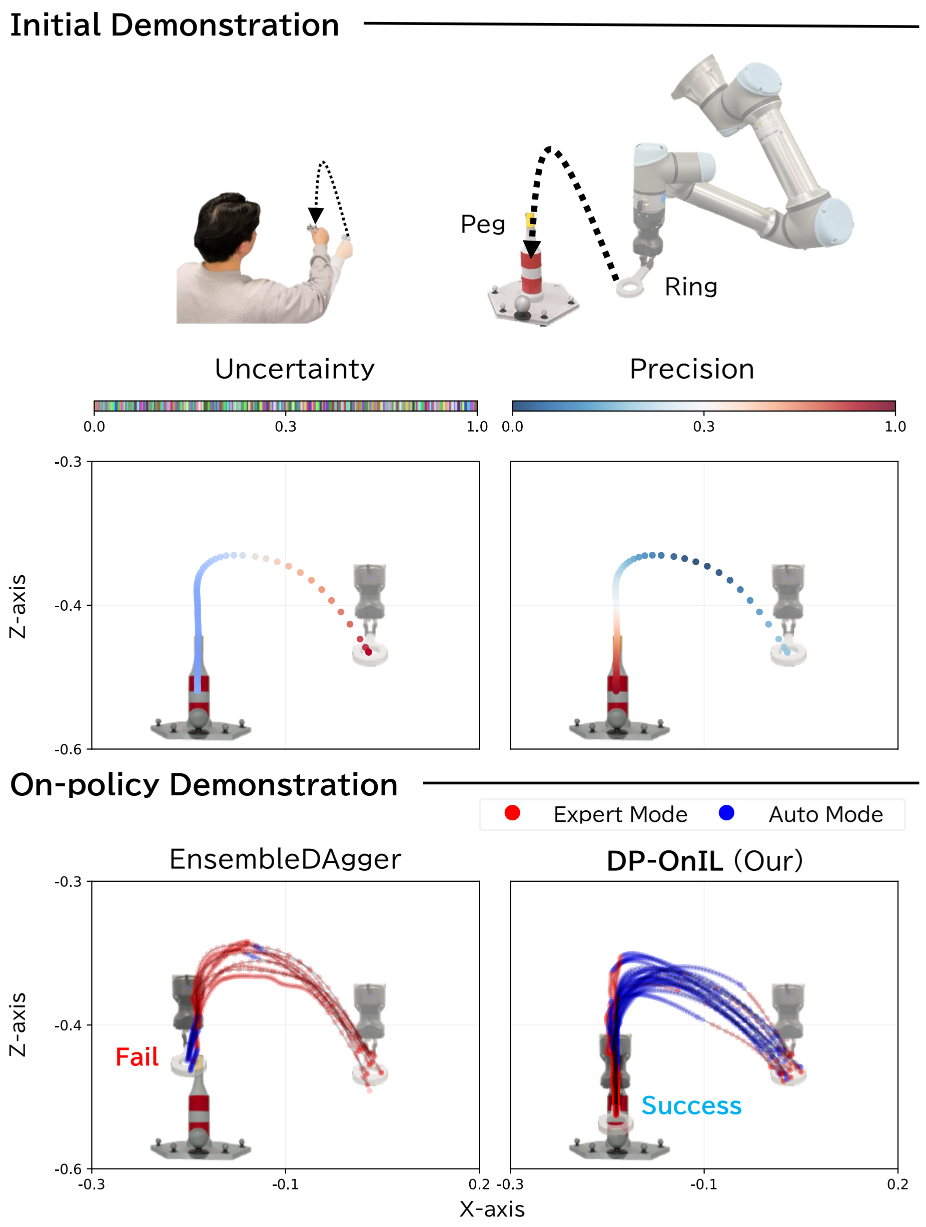

Figure 3.10. Real-robot experiments of the ring-threading task: Experimental environments were conducted for a 6-DOF robotic arm (UR5e) threading a ring into a peg. Precision and uncertainty results were obtained by analyzing initial demonstration trajectories and normalized to visualize variations across states. On-policy demonstrations of EnsembleDAgger and DP-OnIL (Ours) show trajectories at the on-policy learning phase.

Learning Setting

In a similar procedure to Chapter 3.5.1, the human initially collects 5 demonstration trajectories. The number of state-action pairs collected by the human expert in each iteration equals the time limit of each task and the number of iteration is 2 ($K = 2$). Accordingly, the amount of BC training data is roughly similar to the other on-policy IL comparisons.

Comparison Methods: In real-world evaluations, DAgger was excluded from the comparison since its random intervention criteria is too risky (Chapter 3.5.2). Therefore, two approaches were compared with out methods (DP-OnIL$_{\mathrm{UCB}}$ and DP-OnIL$_{\mu}$):

- BC [17]: a conventional imitation learning;
- EnsembleDAgger [27]: a state-of-the-art risk-aware on-policy IL.

Moreover, to ensure sufficient human analysis, more human actions are encouraged by setting threshold χ as 50% of the overall training states that are classified as expert modes.

Demonstration Setting: Demonstrations of each task were performed using a teleoperation system (Figure 3.8) that synchronizes the robot's end effector with the position of a ring grasped by a human demonstrator. Thus, a robot follows a human hand's movements in a real-time manner. We used four human subjects with robotics experience. To obtain sufficient expert performance from them, the following curriculum was applied. Before starting each experiment, all subjects practiced teleoperating the robot by performing several task scenarios, ranging from wide clearance (*e.g.,* obstacle-free) to narrow clearance (*e.g.,* obstacle-present), until they became achieving success in each scenario consecutively. These interactions increased their understanding of environmental precision. In addition, during task demonstrations, the subjects were informed of their remaining time by bells at every 1/3 interval of the time limit.

Results

The results are seen in Figure 3.9, Figure 3.10, and Table 3.1.

Table 3.1. Real-robot experiments results of performance comparison: Performance of each learning model is mean and standard deviation of results of four subjects. Robot-autonomous performance of policies learned by each learning model was measured over ten test executions. Since BC does not employ a meta-policy, we annotated it as N/A in on-policy demonstration performances. Our methods are significantly better than task results marked * (t-test, $p < 5e{-}2$).

Learning Models	On-policy Demo. Perf. [%]		Robot-auto. Perf. [%]	
	Shaft-reach.	Ring-thread.	Shaft-reach.	Ring-thread.
BC	N/A	N/A	$0.0^{*} \pm 0.0$	$0.0^{*} \pm 0.0$
Ensemble DAgger	$41.1^{*} \pm 19.4$	$39.8^{*} \pm 19.1$	$42.5^{*} \pm 30.3$	$55.0^{*} \pm 26.9$
DP-OnIL$_{\mu}$ (ours)	100.0 ± 0.0	100.0 ± 0.0	82.5 ± 13.0	85.0 ± 11.18
DP-OnIL$_{\text{UCB}}$ (ours)	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0

Qualitative Analysis: The on-policy demonstration trajectories of the on-policy IL methods are compared in Figure 3.9 and Figure 3.10. EnsembleDAgger uses the uncertainty of the policy decisions as intervention criterion and requests human intervention in highly uncertain areas (*e.g.,* near the starting position). However, the uncertainty of the policy decision alone does not recognize the latent collision risks in the limited clearance areas due to obstacles. Therefore, the robot is operated in the auto mode in narrow areas, and no human intervention is requested even when a collision is imminent, resulting in task failure. In contrast, the proposed method (DP-OnIL) uses human demonstrations to capture environmental precision and incorporates it into an intervention criterion to recognize collision risks during the on-policy data collection. Accordingly, the robot is operated in the expert mode during times of high collision risks (*e.g.,* near obstacles), thereby reducing their risk.

Quantitative Analysis: The results (Table 3.1) show that BC has zero robot-autonomous performance in both the clearance-limited tasks. This is because, as noted in a previous work [18], policies learned by BC easily lead a robot to deviate from human-demonstrated trajectories, and such deviations are not

Table 3.2. Real-robot experiments results of human stress comparisons: The total number of interventions (mode switching from auto to expert) is measured as the factor of human stress.

Learning Models	Total Number of Interventions	
	Shaft-reach.	Ring-thread.
BC	N/A	N/A
EnsembleDAgger	16.5 ± 8.5	20.3 ± 4.6
DP-OnIL$_{\mu}$ (ours)	12.25 ± 4.76	18.2 ± 2.6
DP-OnIL$_{\text{UCB}}$ (ours)	10.5 ± 2.7	16.0 ± 2.1

allowed in either task. EnsembleDAgger outperformed BC, although it did not exceed 55% in either one since frequent failures during interaction (less than 50% of the interactive performance) make training on the task's later part insufficient. Notably, our method (DPIIL) significantly improves both the interactive and robot-autonomous performances by at least 30% compared to EnsembleDAgger in both tasks, without increasing the total number of interventions (*i.e.,* human stress Table 3.2).

3.7. Summary of Chapter 3

In this chapter, we presented DP-OnIL, a safe on-policy IL algorithm that leverages human risk sensitivity to mitigate the risk of collisions during on-policy demonstration. Human risk sensitivity is exhibited through the speed-accuracy trade-off in demonstrations, and we built on this psychological finding to introduce a model that captures and estimates the demonstrator-perceived precision as a risk in an on-policy demonstration. This approach enables the robot to automatically request human intervention where there is a high risk of collision during an on-policy demonstration, thus ensuring safety while maintaining the effectiveness of the on-policy IL. The efficacy of our method (DP-OnIL) was evaluated in learning on various assembly tasks with limited clearances on simulation and real environment. The results suggest that through the utilization of risk sensitivity, our algorithm can effectively learn limited clearance tasks with substantial improvements in the safety of on-policy IL.

4. Leveraging Human Risk-sensitivity to Ensure Demonstration Feasibility of Disturbance-injected Imitation Learning

4.1. Robustness and Demonstration Feasibility

As described in Chapter 2, a major issue limiting application of learned policies is the problem of *covariate shift* [38]. Specifically, environment variations (*e.g.,* manipulator starting position) induces differences between the policy distribution as learned by the manipulator and the actual task distribution during application.

An intuitive approach to robustifying learned policies against sources of error, without needing to a priori specify task-relevant learning parameters, is to exploit phenomenon similar to persistence excitation [39]. In this, disturbances are injected into the expert's demonstrated actions, and the recovery behavior of the expert is learned given this perturbation. In an imitation learning context, DART [20] exploits this phenomenon for learning a deterministic policy model with a single optimal action (Chapter 2.3). Additionally, DART is well suited to creating a richer dataset, by concurrently determining the optimal disturbance level to be injected into the demonstrated actions during policy learning.

However, the applicability of algorithms proposed to implement DART [20] is limited, since DART employs a naïve disturbance model which cannot regulate the level of disturbance regarding given states (*i.e.,* state-independent distur-

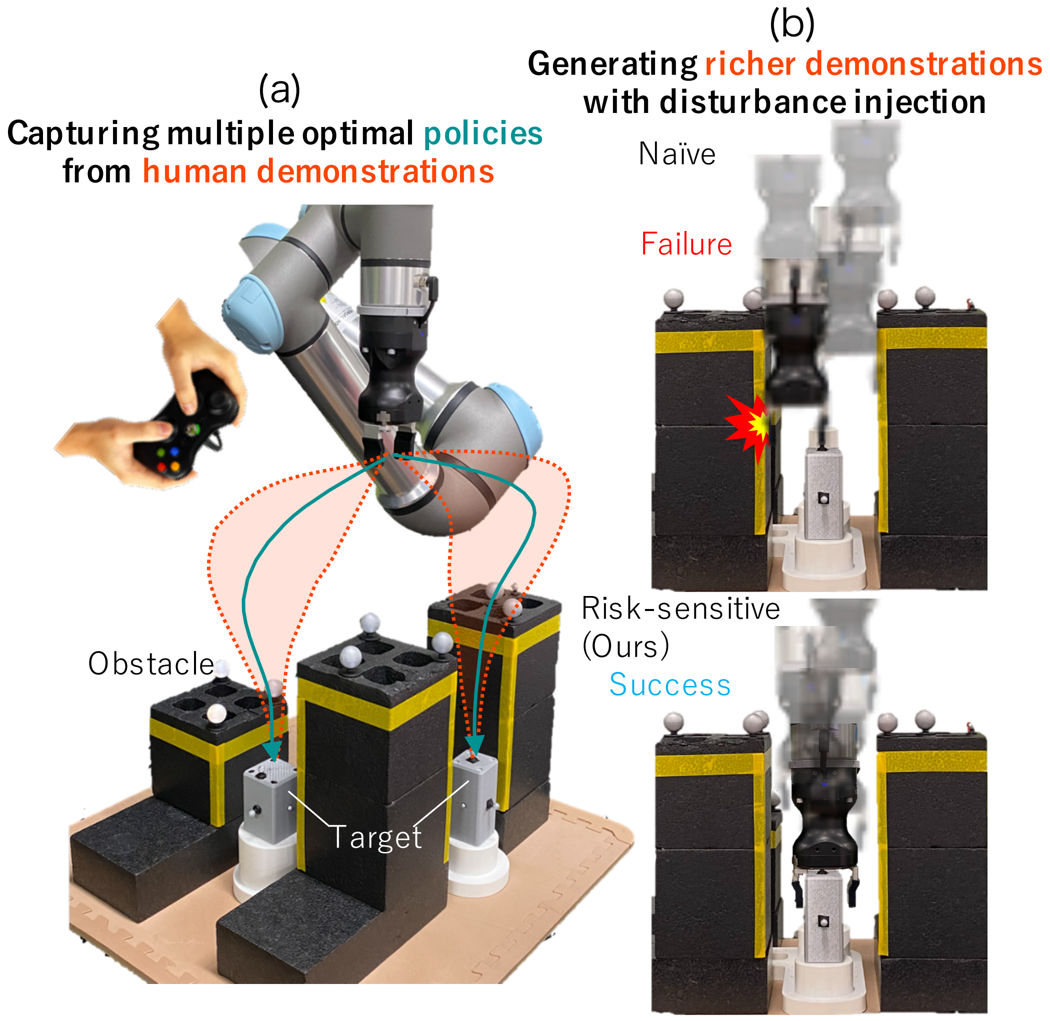
(a)
Capturing multiple optimal policies from human demonstrations
Obstacle
Target
(b)
Generating richer demonstrations with disturbance injection
Naïve
Failure
Risk-sensitive (Ours)
Success

Figure 4.1. Illustration of the critical functions of the proposed method. (a): Multiple optimal policies are captured from complex human demonstrations, which may involve multiple optimal actions. (b): Generating richer (more exploratory) demonstrations by injecting disturbances into expert's actions. Risk-sensitive disturbance models, which regulates its level response to risks of states, is employed to ensure demonstration feasibility.

bance). For robotics tasks with local precision involving small clearance, such as a Figure 4.1, applying a uniform level of disturbance may hinder collecting human demonstration datasets. For example, using a large level of disturbance regardless of state can effectively reduce the covariate shifts, but it may lead to unintended collisions, making human demonstration unfeasible. On the other hand, to be safe, simply limiting a level of disturbance to a small level does not sufficiently reduce the covariate shift.

This study aims to develop a disturbance injection approach for robustifying policies while maintaining demonstration feasibility. A natural approach to addressing this in an imitation learning context, is to explore how human demonstrators approach this problem. Demonstrators, when aware of environmental risks, decrease movement velocity to increase action accuracy [15], based on a *speed-accuracy trade-off* [16]. Inspired by such risk-sensitive behavior, this paper proposes a state-dependent disturbance model, which regulates the disturbance level to be small at risky states (*e.g.,* close to obstacles). As such, our disturbance injection robustifies policies, while maintaining demonstration feasibility. Specifically, a Heteroscedastic Gaussian Process (HGP) [40], which can accurately infer probabilistic regression models with input-dependent variance, and is employed as a state-dependent disturbance model in this chapter.

4.2. Flexibility

Learning generalized optimal action policies from human demonstrations, which often contain complex behaviors (*e.g.,* multiple optimal actions for a task), requires elaborate policy models with non-linearity and stochasticity. However, classical imitation learning approaches, including DART, commonly assume a deterministic policy model, which outputs only a single action from observations, due to its simplicity of application. This oversight leads to undesirable policy learning in real-world scenarios where humans stochastically choose from multiple optimal actions as shown in Figure 4.2-(a). For DART (Chapter 2.3), in particular, approximation errors of learned policy can lead to a cascade of effects that induce learning larger disturbance levels (2.8), limiting demonstration feasibility. To this end, in this section, existing methods to address this lack of

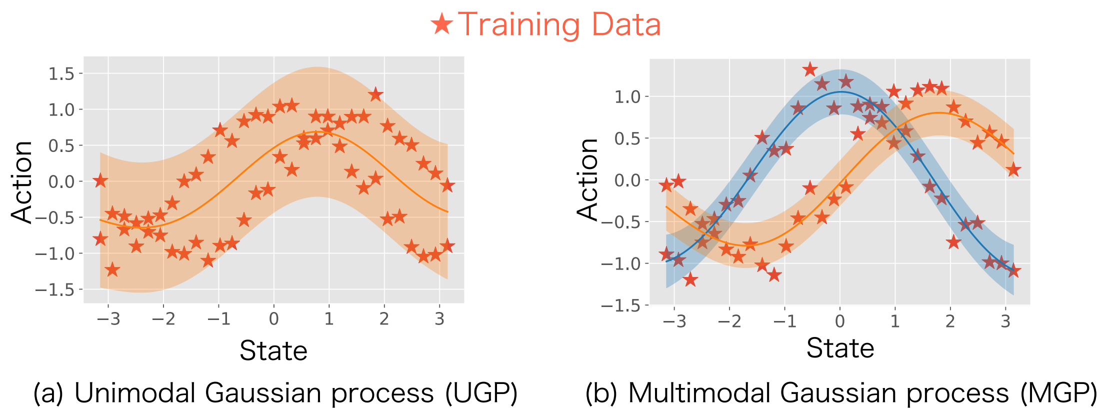

Figure 4.2. Comparing flexibility of policy model on a problem which have multiple optimal actions over one state. (a): Standard unimodal GPs may learn undesirable actions since it can not capture multiple actions. However, (b): Multimodal GPs successfully capture expert' demonstrated actions which have multiple optimal actions.

flexibility are explored.

Dynamic Movement Primitives (DMPs)

Classical approaches to modelling uses dynamical frameworks for learning trajectories from demonstrations, *e.g.,* Dynamic Movement Primitives (DMPs). DMPs represent demonstrated movement with combination of a point attractor term and a nonlinear forcing term $\mathcal{F}$ as follows:

$$\ddot{\mathbf{s}} = \Lambda_{\mathbf{s}}^{a}(\Lambda_{\mathbf{s}}^{b}(g - \mathbf{s}) - \dot{\mathbf{s}}) + \mathcal{F}. \tag{4.1}$$

where, g is the goal state and $\Lambda_{\mathbf{s}}^{a}, \Lambda_{\mathbf{s}}^{b}$ are gain parameters that determined the damping and spring behavior, respectively. In this, the forcing term $\mathcal{F}$ over time is defined by a nonlinear system called a canonical system [41] with radial basis functions.

As such, DMPs can generalize the learned trajectories to new situations (*i.e.,* goal location or speed) while retaining control stability, which is achieved by a point attractor term performing a spring-damper system stabilizing actions [41–43]. However, this generalization depends on heuristics (*e.g.,* the appropriate number of basis functions regarding the complexity of trajectories), and is thus unsuitable for learning state-dependent feedback policies.

Gaussian Mixture Regression (GMR)

A common method for modeling these imitation learning policies, which overcomes the issues of DMP by using the Gaussian Mixture Model (GMM) [44]. The GMM is a probabilistic model that assumes that N data points $\mathbf{X} = \{\mathbf{s}_i, a_i\}_{i=0}^{N}$ are generated from a mixture of a finite number of a Gaussian distributions with unknown parameters (*e.g.,* means $\{\boldsymbol{\mu}_m\}_{m=1}^{M}$, covariances $\{\boldsymbol{\Sigma}_m\}_{m=1}^{M}$, mixing coefficients $\{\boldsymbol{z}_m\}_{m=1}^{M}$) as:

$$p(\mathbf{X}) = \sum_{m=1}^{M} \boldsymbol{z}_m \mathcal{N}(\mathbf{X} \mid \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m) \tag{4.2}$$

In addition, parameters of a joint distribution $p(\mathbf{X})$ is optimized by the *Expectation-Maximization* (EM) algorithm [45]. In this, the GMM is used as a basis function to capture non-linearities during learning and has been utilized in imitation learning that deals with human demonstrations [46].

In the Gaussian Mixture Regression (GMR) model, the conditional probability distribution $p(\mathbf{a} \mid \mathbf{S})$ is derived in terms of the Bayesian theorem and regression functions from each model. As such, the GMR is an intuitive means to learn trajectories or policies from demonstrators in the state-action-space, non-parametrically, without imposing a priori structure. However, a major constraint limiting the applicability of GMR is that hyperparameter tuning (including selecting the optimal number of Gaussians initialization conditions) does not scale well to high-dimensional systems, becoming computationally intensive for robotic manipulation [47].

Variational Auto-Encoders (VAE)

In data-driven manner, nonlinearities can also be captured flexibly using Variational Auto-Encoders (VAE), which is a generative model that can embed high-dimensional features in latent variables [48]. Furthermore, Conditional VAE (CVAE) can learn multi-modality by conditioning latent variables on a decoder [49], and has been applied to capture multiple optimal actions from human demonstrations [50, 51]. However, such CVAE-based methods typically require large amounts of data to capture multi-modality, and even though such multi-modality is obtained using latent variables, learned policies may be sub-optimal for the high precision task; since latent variables are randomly sampled from a standard Gaussian prior distribution [52].

Gaussian Processes (GPs)

As an alternative method for modeling these imitation learning policies, is to use Gaussian processes. For a set of N input state $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_\mathrm{N}\}$, when the joint probability distribution $p(\mathbf{a})$ of the corresponding action $\mathbf{a} = \{a_1, a_2, \cdots, a_\mathrm{N}\}$ follows a multivariate Gaussian distribution, it is called the Gaussian processes (GPs) [53]. It has a meaning as a probability model to infer the latent function f between $\mathbf{s}$ and a.

In the Gaussian process regression (GPR) model, using the Gaussian noise $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ and function f sampled from the GPs, assume as follows:

$$a_n = f(\mathbf{s}_n) + \varepsilon_n \tag{4.3}$$

$$f \sim \mathcal{GP}(\mathbf{0}, \mathrm{k}(\mathbf{s}, \mathbf{s}')). \tag{4.4}$$

where $\mathrm{k}(\cdot, \cdot)$ is a kernel function. Let $f_n = f(\mathbf{s}_n)$ be the latent variable indicating the output of the function for the input data x_n, and let $\mathbf{f} = \{f_1, \cdots, f_\mathrm{N}\}$ represent the latent variable for all the input data. A multivariate Gaussian distribution is assumed for the relationship between the data output $\mathbf{a}$ and the function output $\mathbf{f}$ as:

$$\begin{aligned} p(\mathbf{a} \mid \mathbf{f}) &= \mathcal{N}(\mathbf{a} \mid \mathbf{f}, \sigma^2 \mathbf{I}_\mathrm{N}) \\ p(\mathbf{f} \mid \mathbf{S}; \omega) &= \mathcal{N}(\mathbf{f} \mid \mathbf{0}, \mathbf{K}; \omega). \end{aligned} \tag{4.5}$$

where $\mathbf{I}_\mathrm{N}$ is the identity matrix of size N and $\mathbf{K} = \mathrm{k}(\mathbf{S}, \mathbf{S})$ is the kernel gram matrix with a kernel hyperparameter ω. In addition, to obtain the joint distribution of $\mathbf{a}$, the latent function $\mathbf{f}$ is marginalized as follows:

$$\begin{aligned} p(\mathbf{a} \mid \mathbf{S}; \omega) &= \int p(\mathbf{a} \mid \mathbf{f}) p(\mathbf{f} \mid \mathbf{S}; \omega) d\mathbf{f} \\ &= \mathcal{N}(0, \mathbf{K} + \sigma^2 \mathbf{I}_\mathrm{N}; \omega). \end{aligned} \tag{4.6}$$

Note, the kernel hyperparamter ω is used to optimize the maximised log marginal likelihood as:

$$\hat{\omega} = \arg\max_{\omega} \log p(\mathbf{a} \mid \mathbf{S}; \omega) \tag{4.7}$$

According to the definition of GP, the relationship between the joint distribution of $\mathbf{a}$ and the unknown output a_* is also a multivariate Gaussian distribution as:

$$\begin{bmatrix} \mathbf{a} \\ a_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma^2\mathbf{I}_{\mathrm{N}} & \mathbf{k}_* \\ \mathbf{k}_*^\top & k_{**} + \sigma^2 \end{bmatrix}\right) \tag{4.8}$$

where $[\mathbf{k}_*]_n = \mathrm{k}(\mathbf{s}_n, \mathbf{s}_*)$ and $k_{**} = \mathrm{k}(\mathbf{s}_*, \mathbf{s}_*)$ Then, the predictive distribution of the a_* conditioning on observed data is derived as:

$$p(a_* \mid \mathbf{s}_*, \mathbf{S}, \mathbf{a}) = \mathcal{N}(a_* \mid \mu_*, \sigma_*^2) \tag{4.9}$$

$$\mu_* = \mathbf{k}_*^\top(\mathbf{K} + \sigma^2\mathbf{I}_N)^{-1}\mathbf{a} \tag{4.10}$$

$$\sigma_*^2 = \sigma^2 + k_{**} - \mathbf{k}_*^\top(\mathbf{K} + \sigma^2\mathbf{I}_N)^{-1}\mathbf{k}_* \tag{4.11}$$

In this, Gaussian process regression (GPR) deals with implicit (high-dimensional) feature spaces with kernel functions. It thus can directly deal with high-dimensional observations without explicitly learning in this high-dimensional space [53–56].

In particular, Overlapping Mixtures of Gaussian Processes (OMGP) [57] learns a multi-modal distribution by overlapping multiple GPs as shown in Figure 4.2-(b), and has been employed as a policy model with multiple optimal actions on flexible task learning of robotic policies [21]. To further reduce a priori tuning, Infinite Overlapping Mixtures of Gaussian Processes (IOMGP) [58] requires only an upper bound of the number of GPs to be estimated. As such, IOMGP is an intuitive means of learning flexible multi-modal policies from unlabeled human demonstration data and is employed in this paper.

To the authors' knowledge, there is no unified framework for imitation learning that can simultaneously consider robustness, demonstration feasibility, and flexibility. Our insight into this stems from the difficulty of formulating all three elements as a single framework. For example, in flexible policy learning, using non-parametric probabilistic policy models (*e.g.,* [21]) effectively captures multiple optimal actions from real human demonstrations where the number of optimal actions in each state cannot be specified a priori. However, the previously proposed disturbance injection method [20] optimizes the disturbance level by minimizing the covariate shift, which corresponds to the maximum likelihood estimation based on the assumption of a deterministic policy model and a fixed disturbance level parameter. Thus, such flexible policy learning cannot be directly integrated into the previous framework.

To address this difficulty, we propose to reformulate it as a non-parametric Bayesian inference problem, which employs the objective function of robustification as the likelihood and other non-parametric flexible policy and risk-sensitive disturbance models as the prior distribution. As such, we presents a novel Bayesian imitation learning framework that learns a probabilistic policy model capable of being both flexible to variations in demonstrations and robust to sources of error in policy application by injecting risk-sensitive disturbances in next section.

4.3. Proposed Method

In this section, a novel Bayesian imitation learning framework is proposed (Figure 4.3) to learn a probabilistic policy via expert demonstrations with disturbance injection. Specifically, flexibility, robustness, and risk-sensitivity are incorporated as a single formulation in a Bayesian manner; thus, it is referred to as Bayesian Disturbance Injection (BDI). The general form of BDI is derived in Chapter 4.3.1. As an overview, a non-parametric mixture model is utilized as a policy prior for capturing multiple optimal actions from human demonstration. A heteroscedastic model is employed as a disturbance prior for regulating disturbance level regarding states as shown in Figure 2.2-(b). The disturbance optimization term (2.8) is employed as a likelihood for minimising the covariate shift. This combination derives an imitation learning method, which learns a multi-modal policy and an injection disturbance distribution by Bayesian inference. Given this model, the predictive distribution is induced in a Bayesian form. A specific implementation of BDI, which employ IOMGP [58] as a policy prior and HGP [40] as a disturbance prior, is derived from Chapter 4.3.2.

4.3.1. Bayesian Disturbance Injection (BDI)

Bayesian treatment is employed to learn probabilistic policies and disturbances in a single incorporated framework. As such, each goal function of the learning a policy (2.9) and disturbances (2.8) are formulated as a single likelihood. In

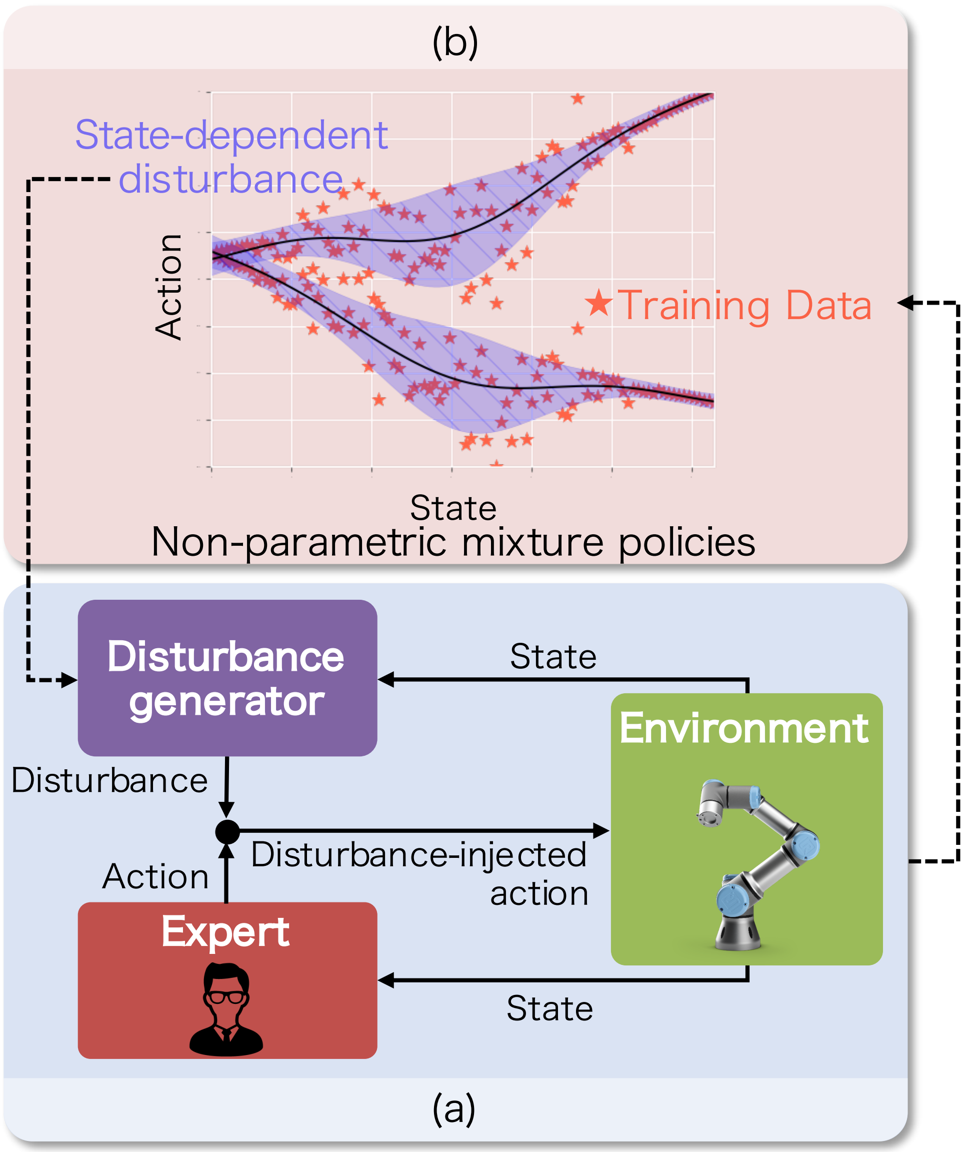

Figure 4.3. Overview of MHGP-BDI, learning robust multi-modal policy with state-dependent disturbance injection. (a): Collect and accumulate training datasets by injecting disturbance into the expert's demonstration actions. (b): Optimize the disturbance at which level can be regulated in a state-dependent manner. These processes (a) and (b) are repeated to obtain a robust multi-modal policy finally.

addition, prior distributions of policy and disturbances are defined, and their respective posterior distributions are obtained via Bayesian inference.

To capture complex human behaviors as involving uncertainties, the probabilistic policy model which output action a_t from the state $\mathbf{s}_t$ with Gaussian disturbance $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ is defined as: $a_t = f(\mathbf{s}_t) + \epsilon_t$, where $f(\cdot)$ is an output of a latent non-linear function. By applying this policy model to the objective function of policy learning (2.9), a log-likelihood function that integrates policy and disturbances is derived as follows:

$$J(\pi^*, \mathbf{f}, \sigma^2 \mid \boldsymbol{\tau}) = \sum_{t=0}^{T-1} \log p(a_t^* \mid f(\mathbf{s}_t), \sigma^2), \tag{4.12}$$

where, $\mathbf{f} = [f(\mathbf{s}_t)]_{t=0}^{T-1}$ is a set of a latent function outputs. Note that this log likelihood function (4.12) is equal to the objective function of disturbance learning (2.8) if the mean and variables are swapped in a Gaussian distribution (the value of the distribution remains the same).

In addition, to infer a policy and disturbances in a non-parametric way from iteratively accumulated state-action pairs ($\{\mathbf{a}^*, \mathbf{S}\} = \{a_n^*, \mathbf{s}_n\}_{n=1}^N$, where $N = \sum_{j=1}^k N_j$, N_j is a size of the dataset that collected at j-th iteration), the prior distribution of a policy and disturbances are defined as $p(\mathbf{f} \mid \mathbf{S})$ and $p(\sigma^2)$, respectively. Accordingly, posterior distributions of a policy and disturbances are simultaneously inferred by Bayesian inference as:

$$p(\mathbf{f}, \sigma^2 \mid \mathbf{a}^*, \mathbf{S}) = \frac{p(\mathbf{a}^* \mid \mathbf{f}, \sigma^2) p(\mathbf{f} \mid \mathbf{S}) p(\sigma^2)}{\pi^*(\mathbf{a}^* \mid \mathbf{S})}. \tag{4.13}$$

A summary of the BDI is shown in Algorithm 3.

4.3.2. Multi-modal Heteroscedastic Gaussian Process BDI (MHGP-BDI)

Formulation

To learn a multi-modal policy, the policy prior is considered as the product of infinite GPs, inspired by IOMGP. In addition, to learn state-dependent disturbances that can regulate its level respond to states, the prior of disturbances is considered as a state-dependent variance GP prior, inspired by HGP. Intuitively,

Algorithm 3 Bayesian Disturbance Injection (BDI)

```
Input: $\sigma_1^2$
Output: $p(\mathbf{f}, \sigma^2 \mid \mathbf{a}^*, \mathbf{S})$
1: for $k = 1$ to $K$ do
2:     Get dataset through the disturbance injected expert:
       $\{a_t^*, \mathbf{s}_t\}_{t=1}^{N_k} \sim p(\boldsymbol{\tau} \mid \pi^*, \sigma_k^2)$
3:     Aggregate datasets :
       $\mathbf{a}^* \leftarrow \mathbf{a}^* \cup \{a_t^*\}_{t=1}^{N_k}$ , $\mathbf{S} \leftarrow \mathbf{S} \cup \{\mathbf{s}_t\}_{t=1}^{N_k}$
4:     Update $p(\mathbf{f}, \sigma^2 \mid \mathbf{a}^*, \mathbf{S})$
5: end for
```

Figure 4.4 shows a probabilistic policy model in which expert's actions $\mathbf{a}^*$ are estimated by $\mathbf{f}^{(m)}, \mathbf{Z}, \mathbf{g}$. The latent function $\mathbf{f}^{(m)}$ is the output of m-th GP given state $\mathbf{S}$. To allocate the expert's n-th action a_n^* to the m-th latent function $\mathbf{f}^{(m)}$, the indicator matrix $\mathbf{Z} \in \mathbb{R}^{N \times \infty}$ is defined. To estimate the optimal number of GPs, a random variable v_m quantifies the uncertainty assigned to $\mathbf{f}^{(m)}$. In addition, to learn an injection disturbance which can regulate its level in a state-dependent way, a state-dependent disturbance level $\sigma^2(\mathbf{s}_n) = e^{g(\mathbf{s}_n)}$ is introduced, where $g(\cdot)$ is an output of GP given a state $\mathbf{s}_n$.

Policy prior: the set of latent functions is denoted as $\{\mathbf{f}^{(m)}\} = \{\mathbf{f}^{(m)}\}_{m=1}^{\infty}$ and a GP prior is given by :

$$p(\{\mathbf{f}^{(m)}\} \mid \mathbf{S}, \{\boldsymbol{\omega}^{(m)}\}) = \prod_{m=1}^{\infty} \mathcal{N}(\mathbf{f}^{(m)} \mid \mathbf{0}, \mathbf{K}_{\mathbf{f}}^{(m)}; \omega^{(m)}), \tag{4.14}$$

where $\mathbf{K}_{\mathbf{f}}^{(m)} = \mathrm{k}_{\mathbf{f}}^{(m)}(\mathbf{S}, \mathbf{S})$ is the m-th kernel Gram matrix with the kernel function $\mathrm{k}_{\mathbf{f}}^{(m)}(\cdot, \cdot)$ and a kernel hyperparameter $\omega_{\mathbf{f}}^{(m)}$. Let $\{\boldsymbol{\omega}_{\mathbf{f}}^{(m)}\} = \{\omega_{\mathbf{f}}^{(m)}\}_{m=1}^{\infty}$ be the set of hyperparameters of infinite number of kernel functions.

To infer the optimal number of GPs from the above GP mixtures (4.14), the Stick Breaking Process (SBP) [59] is used as a prior of $\mathbf{Z}$, which can be interpreted as an infinite mixture model as follows:

$$p(\mathbf{Z} \mid \mathbf{v}) = \prod_{n=1}^{N} \prod_{m=1}^{\infty} \left(v_m \prod_{j=1}^{m-1} (1 - v_j) \right)^{\mathbf{Z}_{nm}}, \tag{4.15}$$

$$p(\mathbf{v} \mid \beta) = \prod_{m=1}^{\infty} \mathrm{Beta}\left(v_m \mid 1, \beta\right). \tag{4.16}$$

Note that the implementation of variational Bayesian learning approximates infinite-dimensional inference with a predefined upper bound of M. In this process, v_m is a random variable indicating the probability that the data corresponds to the m-th GP. Thus, it is possible to estimate the optimal number of GPs with a high probability of allocation starting from an infinite number of GPs. β is a hyperparameter of SBP denoting the level of concentration of the data in the cluster.

Disturbance prior: the above policy model differs from the IOMGP model for regression [58]; our model employs a state-dependent disturbance level $e^{g(\mathbf{s}_n)}$ where the values are determined in response to the state. To learn a state-dependent disturbance, the disturbance prior is considered as a heteroscedastic Gaussian disturbance, inspired by HGP [40]. Accordingly, a GP prior is placed on a latent function $\mathbf{g} = \{g(\mathbf{s}_n)\}_{n=1}^{N}$, which represent a level of disturbance as:

$$p(\mathbf{g} \mid \mathbf{S}; \omega_{\mathbf{g}}) = \mathcal{N}(\mathbf{g} \mid \mu_0 \mathbf{1}_N, \mathbf{K}_{\mathbf{g}}; \omega_{\mathbf{g}}), \tag{4.17}$$

where, μ_0 is mean of disturbance distribution, $\mathbf{1}_{N_i}$ is a vector whose size is N_i and all components are one, and $\mathbf{K}_{\mathbf{g}}$ is kernel Gram matrix with a kernel hyperparameter $\omega_{\mathbf{g}}$.

Likelihood: the likelihood function, as in (4.12), for the variables $(\{\mathbf{f}^{(m)}\}, \mathbf{g}, \mathbf{Z})$ in the policy and disturbance models, is derived as follows:

$$\begin{aligned} &p(\mathbf{a}^* \mid \mathbf{g}, \{\mathbf{f}^{(m)}\}, \mathbf{Z}) \\ &= \prod_{n=1}^{N} \prod_{m=1}^{\infty} \mathcal{N}(a_n^* \mid \mathbf{f}_n^{(m)}, e^{\mathbf{g}_n})^{\mathbf{Z}_{nm}}. \end{aligned} \tag{4.18}$$

This formulation is described in a graphical model that defines the relationship between the variables as shown in Figure 4.4, and the joint distribution of the model as :

$$\begin{aligned} &p(\mathbf{a}^*, \mathbf{g}, \{\mathbf{f}^{(m)}\}, \mathbf{Z}, \mathbf{v} \mid \mathbf{S}; \Omega) \\ &= p(\mathbf{a}^* \mid \mathbf{g}, \{\mathbf{f}^{(m)}\}, \mathbf{Z}) p(\mathbf{g} \mid \mathbf{S}; \omega_{\mathbf{g}}) \cdot \\ &\quad p(\{\mathbf{f}^{(m)}\} \mid \mathbf{S}; \{\boldsymbol{\omega}_{\mathbf{f}}^{(m)}\}) p(\mathbf{Z} \mid \mathbf{v}) p(\mathbf{v} \mid \beta), \end{aligned} \tag{4.19}$$

where $\Omega = (\{\boldsymbol{\omega}_{\mathbf{f}}^{(m)}\}, \omega_{\mathbf{g}}, \mu_0, \beta)$ represents a set of hyperparameters.

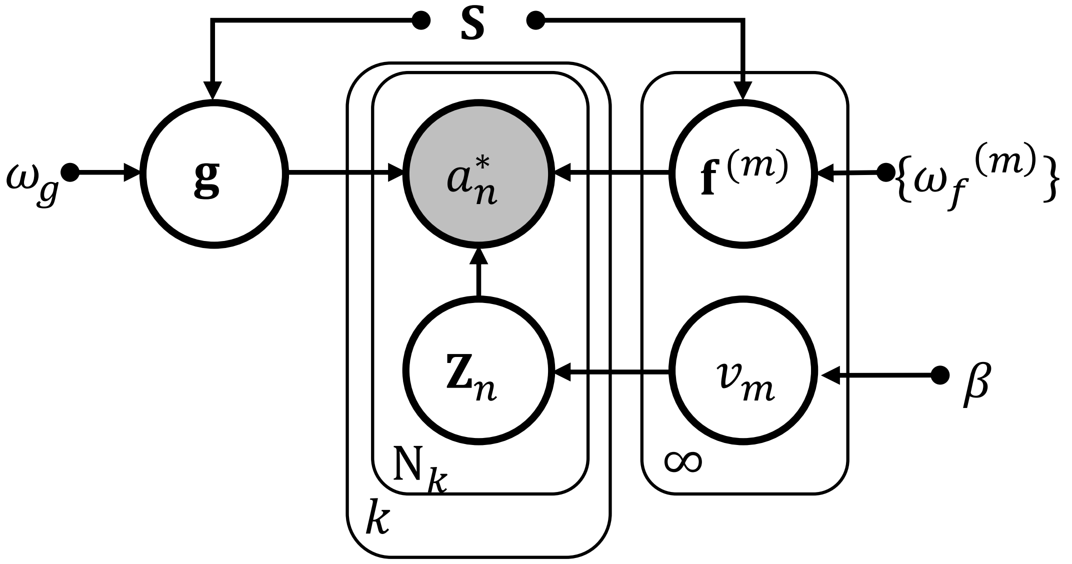

Figure 4.4. Graphical model of policy with state-dependent injection disturbance.

Optimization of Policies and Injection Disturbance via Variational Bayesian Inference

Bayesian inference is a framework that estimates the posterior distributions of the policies and their predictive distributions for new input data rather than point estimates of the policy parameters. To obtain the posterior and the predictive distributions, the marginal likelihood is calculated as :

$$\begin{aligned} &p(\mathbf{a}^* \mid \mathbf{S}; \Omega) \\ &= \int p(\mathbf{a}^*, \mathbf{g}, \{\mathbf{f}^{(m)}\}, \mathbf{Z}, \mathbf{v} \mid \mathbf{S}; \Omega) \mathrm{d}\mathbf{g} \mathrm{d}\{\mathbf{f}^{(m)}\} \mathrm{d}\mathbf{Z} \mathrm{d}\mathbf{v}. \end{aligned} \tag{4.20}$$

However, it is intractable to calculate the log marginal likelihood of (4.20) analytically. Therefore, the variational lower bound is derived as the objective function of variational learning. The true posterior distribution is approximated by the variational posterior distribution, which maximizes the variational lower bound. Such the variational lower bound $\mathcal{L}(q, \Omega)$ is derived by applying the

Jensen inequality to the log marginal likelihood, as:

$$\begin{aligned}
&\log p(\mathbf{a}^* \mid \mathbf{S}; \Omega) \\
&\geq \int q \log \frac{p(\mathbf{a}^*, \mathbf{g}, \{\mathbf{f}^{(m)}\}, \mathbf{Z}, \mathbf{v} \mid \mathbf{S}; \Omega)}{q} \mathrm{d}\mathbf{g}\mathrm{d}\{\mathbf{f}^{(m)}\}\mathrm{d}\mathbf{Z}\mathrm{d}\mathbf{v} \\
&= \mathcal{L}(q, \Omega),
\end{aligned} \tag{4.21}$$

where, $q = q(\mathbf{g}, \{\mathbf{f^{(m)}}\}, \mathbf{Z}, \mathbf{v})$ represents a set of variational posteriors.

As a common fashion of variational inference, the variational posterior distribution is assumed to be factorized among all latent variables (known as the *mean-field approximation* [60]) as follows:

$$q(\mathbf{g}, \{\mathbf{f^{(m)}}\}, \mathbf{Z}, \mathbf{v}) = q(\mathbf{g})q(\mathbf{f}^{(m)})q(\mathbf{Z}) \prod_{m=1}^{\infty} q(v_m). \tag{4.22}$$

In addition, to compute the variational lower bound in closed form, the posterior of $\mathbf{g}$ is restricted to a multivariate Gaussian distribution. Furthermore, to reduce the computational complexity and facilitate the optimization problem, similar to Gaussian approximation [61], a positive variational parameter $\mathbf{\Lambda} = \mathrm{diag}\{\lambda_n\}_{n=1}^{N}$ is employed as :

$$q(\mathbf{g}) = \mathcal{N}(\mathbf{g} \mid \boldsymbol{\mu}_{\mathbf{g}}, \mathbf{\Sigma}_{\mathbf{g}}), \tag{4.23}$$

$$\boldsymbol{\mu}_{\mathbf{g}} = \mathbf{K}_{\mathbf{g}} \left(\mathbf{\Lambda} - \frac{1}{2}\mathbf{I} \right) \mathbf{1}_N + \mu_0 \mathbf{1}_N, \tag{4.24}$$

$$\mathbf{\Sigma}_{\mathbf{g}}^{-1} = \mathbf{K}_{\mathbf{g}}^{-1} + \mathbf{\Lambda}, \tag{4.25}$$

where, $\mathbf{I}$ is an identity matrix.

Therefore, the optimization formulation is derived by utilizing the *Expectation-Maximization* (EM)-like algorithm. The variational posterior distributions q are optimized with fixed hyperparameters Ω' in E-step, and the hyperparameters Ω' are optimized with fixed variational posterior distributions q in M-step with:

$$\hat{q}, \hat{\Omega}' = \arg\max_{q, \Omega'} \mathcal{L}(q, \Omega'), \tag{4.26}$$

where, $\Omega' = (\Omega, \mathbf{\Lambda})$ represents a set of variational hyperparameters. See Chapter B.1 for details of q update laws and Chapter B.2 for details of lower bound of marginal likelihood. In addition, a summary of the proposed method is shown in Algorithm 4; and Table 4.1 shows the computational complexity of each optimization.

Algorithm 4 Multi-modal Heteroscedastic Gaussian Process BDI (MHGP-BDI)

```
Input: $M, \sigma_1^2$
Output: $\hat{q}, \hat{\Omega}'$
 1: for $k = 1$ to $K$ do
 2:     Get dataset through the disturbance injected expert:
        $\{a_t^*, \mathbf{s}_t\}_{t=1}^{N_k} \sim p(\boldsymbol{\tau} \mid \pi^*, \sigma_k^2)$
 3:     Aggregate datasets :$\mathcal{D} \leftarrow \mathcal{D} \cup \{a_t^*, \mathbf{s}_t\}_{t=1}^{N_k}$
 4:     while $\mathcal{L}(q, \Omega')$ is not converged do
 5:         while $\mathcal{L}(q, \Omega')$ is not converged do
 6:             Update $q(\mathbf{f}^{(m)})$, $q(\mathbf{Z})$,and $q(v_m)$ alternately
 7:         end while
 8:         Optimize $\Omega'$ with fixed $q$:
            $\hat{\Omega}' \leftarrow \arg\max_{\Omega'} \mathcal{L}(q, \Omega')$
 9:     end while
10: end for
```

Table 4.1. Computational complexity of each optimization in MHGP-BDI: N and M are number of training data sets and upper bound of mixtures, respectively.

	$q(\mathbf{g})$	$q(\mathbf{v})$	$q(\{\mathbf{f}^{(m)}\})$, $q(\mathbf{Z})$, $\mathcal{L}$
MHGP-BDI	$\mathcal{O}(N^3)$	$\mathcal{O}(M^2N)$	$\mathcal{O}(MN^3)$

Predictive Distribution

Using variational parameter $\boldsymbol{\Lambda}$ optimized by maximizing (4.26), the predictive disturbance $q(g_*)$ on a new state $\mathbf{s}_*$ can be obtained as:

$$\begin{aligned} q(g_*) &= \int p(g_* \mid \mathbf{s}_*, \mathbf{S}, \mathbf{g}) q(\mathbf{g}) \mathrm{d}\mathbf{g} \\ &= \mathcal{N}(g_* \mid \mu_{g*}, \sigma_{g*}^2), \qquad (4.27) \\ \mu_{g_*} &= \mathbf{k}_{g*}^\top (\boldsymbol{\Lambda} - \mathbf{I}/2) \mathbf{1}_N + \mu_0, \qquad (4.28) \\ \sigma_{g_*}^2 &= k_{g**} - \mathbf{k}_{g*}^\top (\mathbf{K_g} + \boldsymbol{\Lambda}^{-1})^{-1} \mathbf{k}_{g*}, \qquad (4.29) \end{aligned}$$

where $\mathbf{k}_{g*} = \mathrm{k}_g(\mathbf{s}^*, \mathbf{S})$, and $\mathrm{k}_{g**} = \mathrm{k}_g(\mathbf{s}^*, \mathbf{s}^*)$. As such, a level of disturbance injected at the next iteration $k+1$ is calculated as: $\sigma_{k+1}^2(\mathbf{s}_*) = e^{\mu_{g*}}$.

In addition, using the hyperparameters Ω' and the variational posterior distributions q optimized by variational Bayesian learning, the predictive distribution

Table 4.2. Computational complexity of each prediction in MHGP-BDI: N is number of training data sets.

	$q(g_*)$	$p(a_*^{(m)} \mid \mathbf{s}_*, \mathbf{S}, \mathbf{a}^*)$
MHGP-BDI	$\mathcal{O}(N^3)$	$\mathcal{O}(N^3)$

of the m-th action $a_*^{(m)}$ on a current state $\mathbf{s}_*$ is derived as:

$$\begin{aligned} & p(a_*^{(m)} \mid \mathbf{s}_*, \mathbf{S}, \mathbf{a}^*) \\ & \approx \int p(a_*^* \mid \mathbf{f}^{(m)}, g_*, \mathbf{s}_*) q(\mathbf{f}^{(m)}) q(g_*) \mathrm{d}\mathbf{f}^{(m)} \mathrm{d}g_* \\ & = \int \mathcal{N}(a_*^* \mid \mu_*^{(m)}, c_*^{2(m)} + \exp(g_*)) \mathcal{N}(g_* \mid \mu_{g*}, \sigma_{g_*}^2) \mathrm{d}g_*; \end{aligned} \tag{4.30}$$

however, it is analytically intractable to compute. Alternatively, using a Gauss-Hermite quadrature rule [62], mean $\mu_*^{(m)}$ and variance $\sigma_*^{2(m)}$ of the predictive distribution (4.30) can be approximated as:

$$\mu_*^{(m)} = \mathbf{k}_{f_*}^{(m)\top} (\mathbf{K}_\mathbf{f}^{(m)} + \mathbf{R}^{-1})^{-1} \mathbf{a}^*, \tag{4.31}$$

$$\sigma_*^{2(m)} = c_*^{2(m)} + \exp(\mu_{g_*} + \sigma_{g_*}^2/2), \tag{4.32}$$

$$c_*^{2(m)} = k_{f**}^{(m)} - \mathbf{k}_{f*}^{(m)\top} (\mathbf{K}_\mathbf{f}^{(m)} + \mathbf{R}^{-1})^{-1} \mathbf{k}_{f*}^{(m)}, \tag{4.33}$$

where $\mathbf{k}_{f*}^{(m)} = \mathrm{k}_f^{(m)}(\mathbf{s}^*, \mathbf{S})$, and $\mathrm{k}_{f**}^{(m)} = \mathrm{k}_f^{(m)}(\mathbf{s}^*, \mathbf{s}^*)$; and Table 4.2 shows the computational complexity of each prediction. Additionally, m is chosen as the value that maximizes the inverse of the predicted variance $\sigma_*^{2(m)}$ as:

$$\hat{m} = \arg\max_m \frac{1}{\sigma_*^{2(m)}}, \tag{4.34}$$

as such, meaning the $\hat{m}$-th GP is selected, due to its minimal uncertainty.

4.4. Simulation

In this section, the proposed methodology (MHGP-BDI) is evaluated in regards to the following questions, to examine key objectives of capturing human behavior characteristics in a simulated precision wall-avoidance task: (i) flexibility: how does capturing multiple optimal human actions affect imitation learning of robotic tasks?, (ii) robustness: how does injecting disturbances into human demonstrations affect the applicability of learned policies?, and (iii) risk-sensitivity: how does injecting disturbance into human action command affect human demonstrations' feasibility?

Evaluation Metrics: Performance of MHGP-BDI is considered during the training phase and execution phase. For the former, *demonstration feasibility*, or the success rate of collecting training data with a human expert in the loop, is evaluated. On the latter, *execution performance*, or the success rate of deploying the learned policy after training, is evaluated. These metrics are reported in the wall-avoidance simulation study (Chapter 4.4.1) and the real robot assembly study (Chapter 4.5). By comparing both performances across different algorithms, each algorithm is evaluated for how effectively it obtains policy performance while ensuring the demonstration feasibility.

Table 4.3. Comparison models in terms of flexibility, robustness, and demonstration feasibility.

Learning Models	Flexibility	Robustness	Demonstration Feasibility
BC [17]	✗	✗	✓
DART [20]	✗	✓	✗
CVAE-BC [51]	✓	✗	✓
UGP-BC	✗	✗	✓
UGP-BDI	✗	✓	✗
UHGP-BDI	✗	✓	✓
MGP-BC	✓	✗	✓
MGP-BDI [63]	✓	✓	✗
MHGP-BDI (Proposed)	✓	✓	✓

Comparison Methods: To evaluate the proposed method (MHGP-BDI), comparisons are made between 8 baselines. Each baseline's features (flexibility, robustness, and demonstration feasibility) are represented in Table 4.3. Specifically, these algorithms are implemented as:

- **Behavior Cloning (BC)** [17]: Conventional supervised imitation learning as described in Chapter 2.1 using a neural network policy model,

- **Disturbances for Augmenting Robot Trajectories (DART)** [20]: Robust imitation learning by injecting disturbance into expert as described in Chapter 2.3 using a neural network policy model,
- **Conditional Variational AutoEncoders BC (CVAE-BC)** [51]: Multi-modal imitation learning based on BC algorithm using a CVAE policy model,
- **Uni-modal GP Behavior Cloning (UGP-BC)**: BC using standard uni-modal GPs [53],
- **Multi-modal GP BC (MGP-BC)**: BC using infinite overlapping mixtures of GPs (IOMGP),
- **UGP-BDI**: BDI using standard uni-modal GPs and state-independent disturbance model with a constant disturbance level of σ^2,
- **Uni-modal Heteroscedastic GP BDI (UHGP-BDI)**: BDI using standard uni-modal GPs and Heteroscedastic Gaussian Processes (HGP) as state-dependent disturbance model $\sigma^2(\mathbf{s}_t)$,
- **MGP-BDI** [63]: BDI using IOMGP policy model and state-independent disturbance model which level parameter as σ^2.

See Chapter C.2 for how the hyperparameters of each method are set. Note, in all experiments, demonstrations are performed without injecting disturbances in the first iteration (*i.e.,* $\sigma_1^2 = 0$); since initially, there is no available evidence of which level of disturbance is suitable.

4.4.1. Wall-avoidance Task

Initially, a wall-avoidance task involving multiple apertures is presented (Figure 4.5-(a)). In this experiment, demonstrations are conducted in an environment involving states in which physical contact (*e.g.,* collisions of an agent and walls) is likely to occur, and the demonstration feasibility (*e.g.,* avoiding collision) will be evaluated. The learned policy is evaluated through test execution episodes to evaluate its flexibility capturing multiple optimal actions from demonstrations

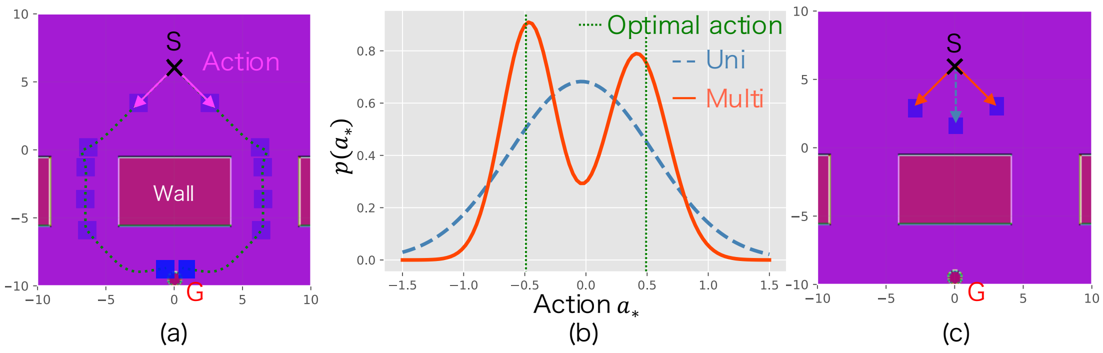

Figure 4.5. Wall-avoidance Task (Wide). (a): Environment of passing through a multiple aperture. S and G represent a starting and a goal position, respectively. An algorithmic supervisor's demonstrated movement, which includes the cautious phase (*e.g.,* move slow when a robot is close to an aperture), is captured as multiple frames with a 0.025 frame rate. (b), (c): Comparing flexibility between multi-modal approaches and uni-modal approaches. (b) The predictive distribution of x-axis action a_* in a given starting position state. (c) Movements of multi-modal approaches and uni-modal approaches at policy application phase.

(*e.g.,* multiple paths through an aperture to reach the goal), and robustness against environmental variations (*e.g.,* starting positions of the agent or inertial of the agent) that may induces the covariate shift.

Setup

In the wall-avoidance task environment (Figure 4.5-(a)), the aim is for the agent (blue square with width and height 1.4 cm and 1.5 cm respectively) to move from the starting position (black cross) through one of the two apertures to the goal position (red circle) without colliding with the wall (grey square). The system state is the agent's position (*e.g.,* x, y-axis coordinates), and the action is the agent's velocity (*e.g.,* x, y-axis).

Expert demonstrations are provided by an algorithmic supervisor, specifically human-like cautious behavior [15] is generated by a classical PID controller. This behavior is simulated by adjusting the agent's velocity during task execution: high velocity (high p-gain) in open regions far from apertures, and low velocity during aperture traversal, as shown in Figure 4.5-(a).

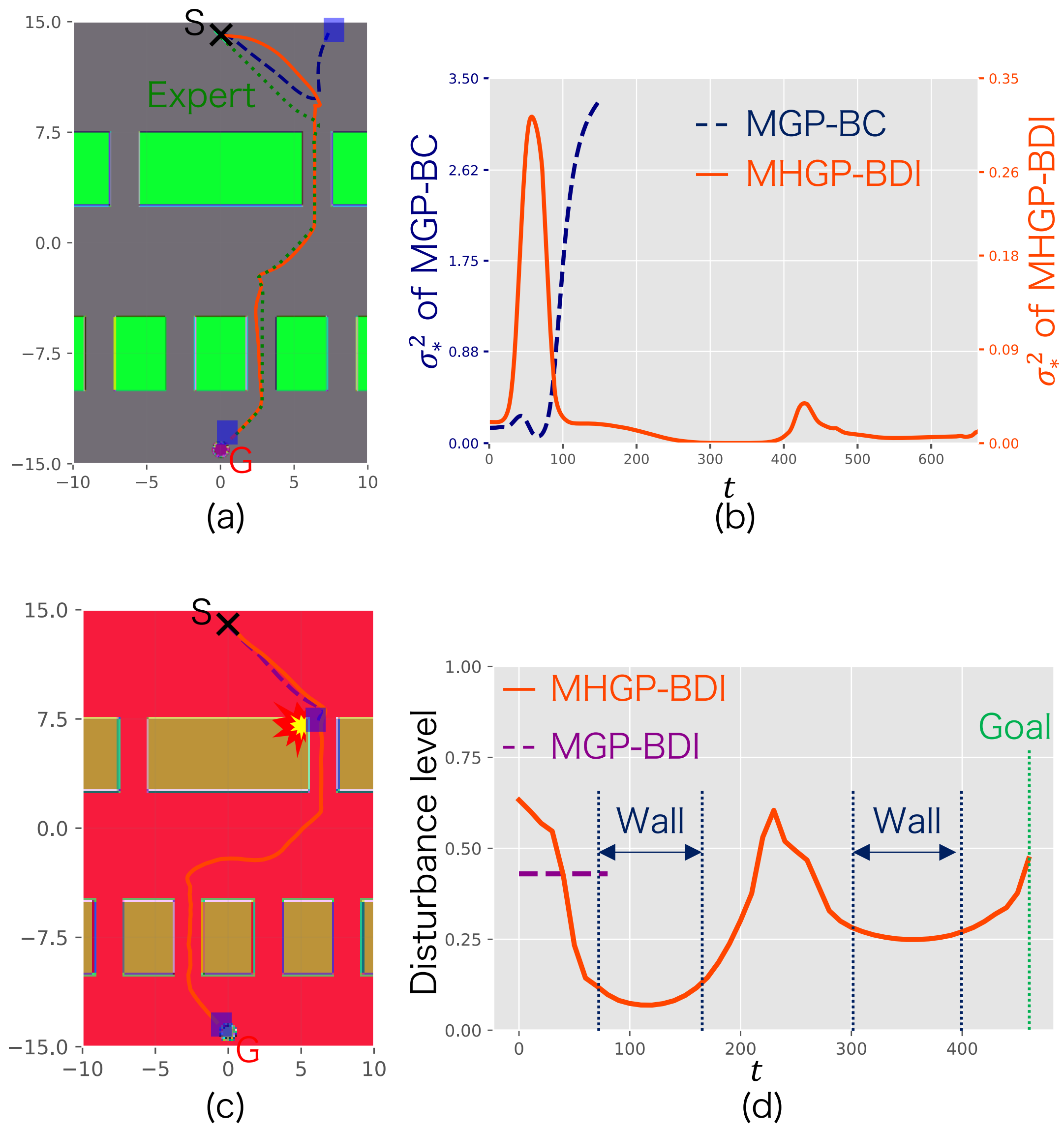

Figure 4.6. Wall-avoidance Task (Complex). (a), (b): Comparing robustness between MHGP-BDI and MGP-BC. (a) Generated trajectory from policy learned by MHGP-BDI and MGP-BC, and (b) sequentially depicts the predictive action variance σ_*^2 (*i.e.*, norm of the XY-axis σ_*^2) of both policy at each step. This result shows that as the agent deviates from the expert's trajectory towards the perpendicular distance, the confidence decreases as the data becomes more sparse. (c), (d) : Demonstration feasibility comparison of MHGP-BDI and MGP-BDI. (c) Demonstration trajectory with injecting a state-dependent disturbance (MHGP-BDI) and a state-independent disturbance (MGP-BDI), and (d) sequentially depicts the level of disturbance injected at each step.

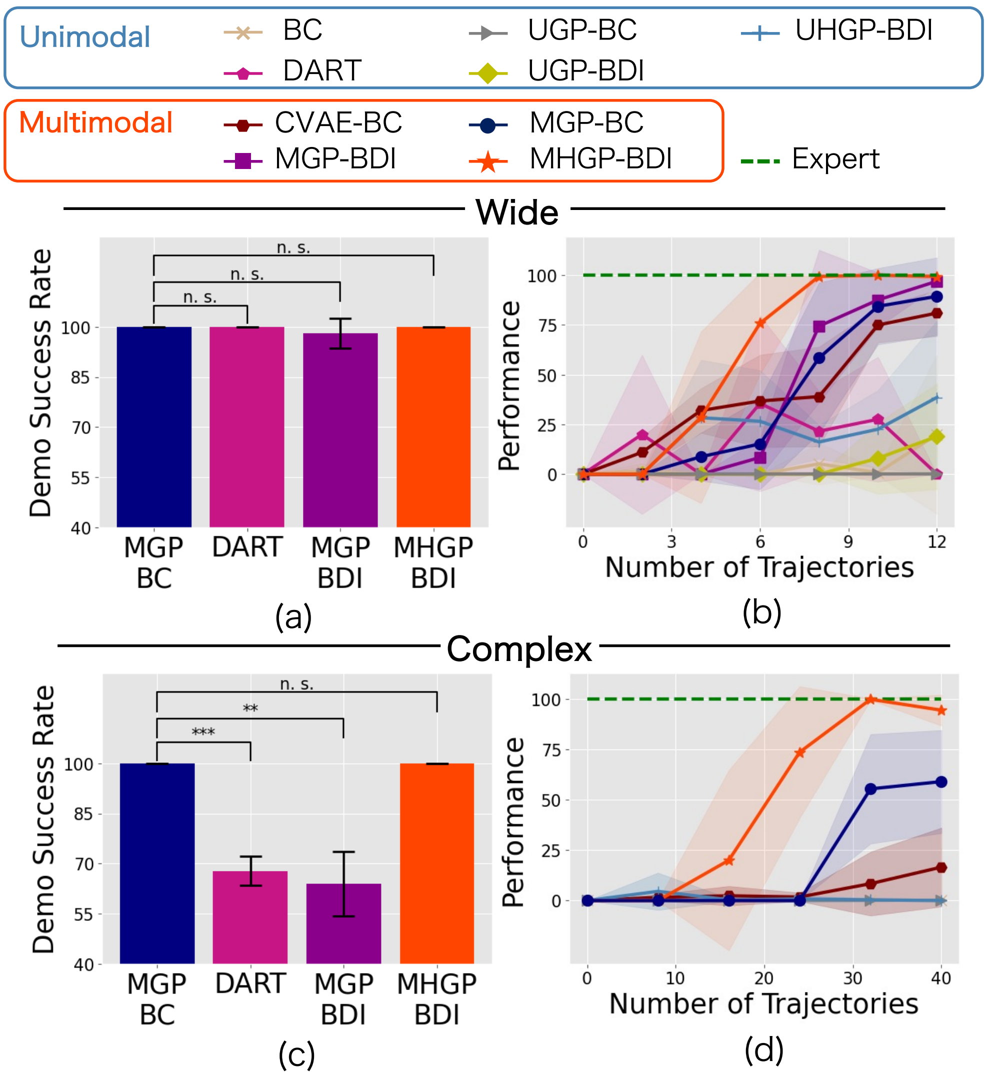

Figure 4.7. Wall-avoidance Task Results (Wide & Complex). (a), (c): Comparing the demonstration success rate for representative learning methods (MGP-BC, DART, MGP-BDI, and MHGP-BDI) of each robustification method. The demonstration success rate of each comparison method is measured as the mean and standard deviation of the demonstration success probability for the entire trials of the final learning iteration. Significant differences by t-test were observed between the proposed method and baselines ($** : p < 0.005, *** : p < 0.0005$). Note, uni-modal methods exhibit similar results in these experiments. It is seen that demo success rate is more related to robustifying than flexibility; thus, these results focus on comparing between robustifying approaches. (b), (d): Comparing task performance with the number of trajectories. The task performance of policy application is measured as the mean and standard deviation of the task success probability by conducting five learning trials and testing each learned policy 100 times.

Wide: Under these experimental parameters, expert demonstrations of passing through each aperture (aperture width is 5.0 cm) is provided in sequence. If the agent collides with a wall or fails to reach the goal position within the time limit (400 steps), it is considered as a failure, and data is discarded, and the demonstration is restarted. After collecting 2 demonstration trajectories, the data are used to optimize the policy and the disturbance until the optimization equation (4.26) converges. In contrast, learning methods that fail to collect demonstrations more than 5 times are considered a learning failure and are not included in the task performance comparison. This process is defined as one iteration of k in Algorithm 4, and is repeated K times, adding the successful demonstrations to the training dataset and continuously updating the policy and disturbance until the fixed number of iterations is reached. In this experiment, K is empirically chosen to stop learning when the average of injected disturbance level is sufficiently small (*i.e.,* learned policy from each comparison is achieved at $K = 6$). During the test execution stage, each element of the initial state is deviated by an additive uniform noise $\epsilon_{\mathbf{s}_0} \sim \mathcal{U}(-0.05 \text{ cm}, 0.05 \text{ cm})$, and positions of the walls and goal remain constant.

Complex: To evaluate the proposed method's scalability, a second experiment is also presented for a more complex task, as shown in Figure 4.6-(a),(c). In this, apertures with a smaller width (2.0 cm) is placed in the environment, and a secondary wall with four apertures is additionally placed below the first wall of the previous experiment. The clearance for moving the agent is smaller in the both layer apertures (0.5 cm), requiring more precise control to avoid collision. Additionally, this secondary layer creates new traversal branches, inducing additional multiple optimal actions and requiring longer steps to accomplish the task. Due to the increased task complexity, the time limitation is increased to 1500 step and the maximum number of demonstration trajectories for updating the policy and disturbance estimates is increased to 8 and the maximum number of iterations is $K = 5$ (total 40 trajectories). Additionally, during the test execution stage, each element of the initial state is deviated by the wider additive uniform noise $\epsilon_{\mathbf{s}_0} \sim \mathcal{U}(-0.1 \text{ cm}, 0.1 \text{ cm})$.

Results

This section presents the qualitative and quantitative analysis of this simulation. The qualitative analysis is presented in terms of (i) flexibility, (ii) robustness, (iii) demonstration feasibility. In addition, the quantitative analysis is presented with previously described evaluation metrics. The results of this simulation are shown in Figure 4.5, Chapter 4.6, Chapter 4.7.

(i) Flexibility: Initially, to evaluate the ability of the agent to flexibly learn in scenarios with multiple-optimal actions (Figure 4.5-(a)), policies are learned for each of the comparison methods, and generated action distributions are shown in Figure 4.5-(b). In this, it is seen that the uni-modal policy learned by UHGP-BDI fails to capture multiple optimal actions at the starting position (S) of the task. Note, all other uni-modal GP-based methods (UGP-BC, UGP-BDI) exhibit very similar Gaussian distributions. Specifically, as seen in Figure 4.5-(c), uni-modal approaches learn a mean-centered policy from the demonstrations, resulting in an incorrect average direction and inability to reach any aperture. However, policies learned by MHGP-BDI can correctly capture the multi-modal distribution (Figure 4.5-(b)) and learn the two optimal actions (Figure 4.5-(c)). Note, all other multi-modal GP-based methods (MGP-BC, MGP-BDI) exhibit very similar Gaussian mixture distributions.

(ii) Robustness: To evaluate the effect of demonstrations on policy learning and application (*i.e.,* the test execution phase), initially the successful demonstrations from the MGP-BC method are used for policy learning. The results for applying policy learning is seen in (Figure 4.6-(a)), where immediately the agent poorly performs the task by veering away from trained trajectory, and does not recover back to the optimal trajectory. This demonstrates the error compounding problem, whereby the lack of robustness in the learned model causes the agent to visit unexplored and unrecoverable states. This effect can be seen in (Figure 4.6-(b)), whereby the action variance of MGP-BC is dramatically increased during policy learning, in a failed attempt to mitigate the problem. As such, the confidence of the policy learned by MGP-BC decreases monotonically after the 60 th time step and fails the task (time-limitation). Note that the other multi-modal neural network-based approach (CVAE-BC) exhibits a similar phenomenon.

In contrast, in the MHGP-BDI method, error compounding is minimised by injecting disturbances into demonstrations, thereby collecting recovery actions under conditions that drift from an optimal trajectory. Accordingly, when applying the policies learned in the MHGP-BDI method, even though the agent similarly immediately drifts, it can recover to an optimal trajectory and complete the task (Figure 4.6-(a)). Even if there is a momentary decrease in confidence due to environmental variations, the policy exhibits a high confidence (Figure 4.6-(b)). Note that confidence is relatively lower when passing through the first aperture (60 th time step) than the second aperture (440 time step), since the perpendicular distance from the expert's trajectory to the agent is larger, induced by the environmental variations (*e.g.,* random starting position and inertial effects).

(iii) Demonstration Feasibility: Given this demonstration of flexibility and robustification, the disturbance injection approaches are then evaluated in terms of their ability to limit collisions. Specifically, the ability of methods which utilizes either a state-independent (MGP-BDI) or a state-dependent (MHGP-BDI) disturbance, is evaluated in aperture traversal. In Figure 4.6-(c), it is seen that state-independent methods, which do not regulate disturbance, collide with the walls, due to its constant level of disturbance (as seen in Figure 4.6-(d)). As such, state-independent robustification (MGP-BDI) injects disturbances that are unsafe, and render this method unable to collect supervisor demonstrations, and the learning process cannot proceed any further. Note that the other state-independent approach (DART) exhibit similar phenomenon. In contrast, MHGP-BDI equipped with a state-dependent disturbances, successfully navigates the tasks-space, by reducing the level of disturbance to about 23% of that of the MGP-BDI when it comes close to aperture ($t = 86$) (Figure 4.6-(d)). This cautious-like behavior enables the agent to pass through the aperture safely, and complete the demonstrations.

Quantitative Evaluation: To evaluate the stability of these approaches, these experiments were repeated five times. The averaged demo success probabilities for representative learning methods (MGP-BC, DART, MGP-BDI, and MHGP-BDI) of each robustification method are shown in Figure 4.7-(a), (c). In addition, the averaged task execution performance of each learned policy is shown in Figure 4.7-(b), (d).

In the wide aperture experiments, (Figure 4.7-(a)), the demonstration feasibility is not significantly different from MGP-BC even with the disturbance injection learning approaches (DART, MGP-BDI and MHGP-BDI), since the aperture size is sufficiently large. However, (Figure 4.7-(b)), the uni-modal policy schemes (BC, DART, UGP-BC, UGP-BDI and UHGP-BDI) all fail to learn the multi-modal task and as expected produce low performance (under 50%) results, due to lack of flexibility (as discussed in Figure 4.5-(b), (c)). Note, DART, UGP-BDI, and UHGP-BDI gain additional robustness over the standard uni-modal approaches; since some deviated states, induced by control errors due to failure to capture multiple optimal actions, may be covered by disturbance injection. Thus its performance increases monotonically in the early stages; however it eventually cannot exceed 50% due to the limitation of learning flexibility. In comparison, the multi-modal policy schemes (CVAE-BC, MGP-BC, MGP-BDI and MHGP-BDI) improve the learning performance by nearly 100% with increasing number of trained trajectories.

In the complex aperture experiments, (Figure 4.7-(d)), even multi-modal BC approaches (CVAE-BC, MGP-BC) using a flexible multi-modal policy, learned policies' task performance cannot exceed 60%, due to the lack of robustness (as discussed in Figure 4.6-(a), (b)). However, if disturbances are injected into demonstrations in a state-independent manner (DART, MGP-BDI), this perturbation may cause physical contact at narrow apertures, and lead to demonstration failure (as discussed in Figure 4.6-(c), (d)). This failure is seen in DART and MGP-BDI; both have a low demonstration success rate in the complex simulation (Figure 4.7-(c)), with demonstration success decreased by 32% compared to the wide-version. Accordingly, DART and MGP-BDI are removed from the comparison of learning performance in the complex aperture experiments (Figure 4.7-(d)), since they failed 5 times demonstrations during learning iteration. In contrast, MHGP-BDI, which can learn a state-dependent disturbances, has a 100% demonstration success rate for both simulations, and consistently shows superior learning efficiency and obtain policies with high task performance (nearly 100%, only very small failures due to some specific starting position or given environmental noise).

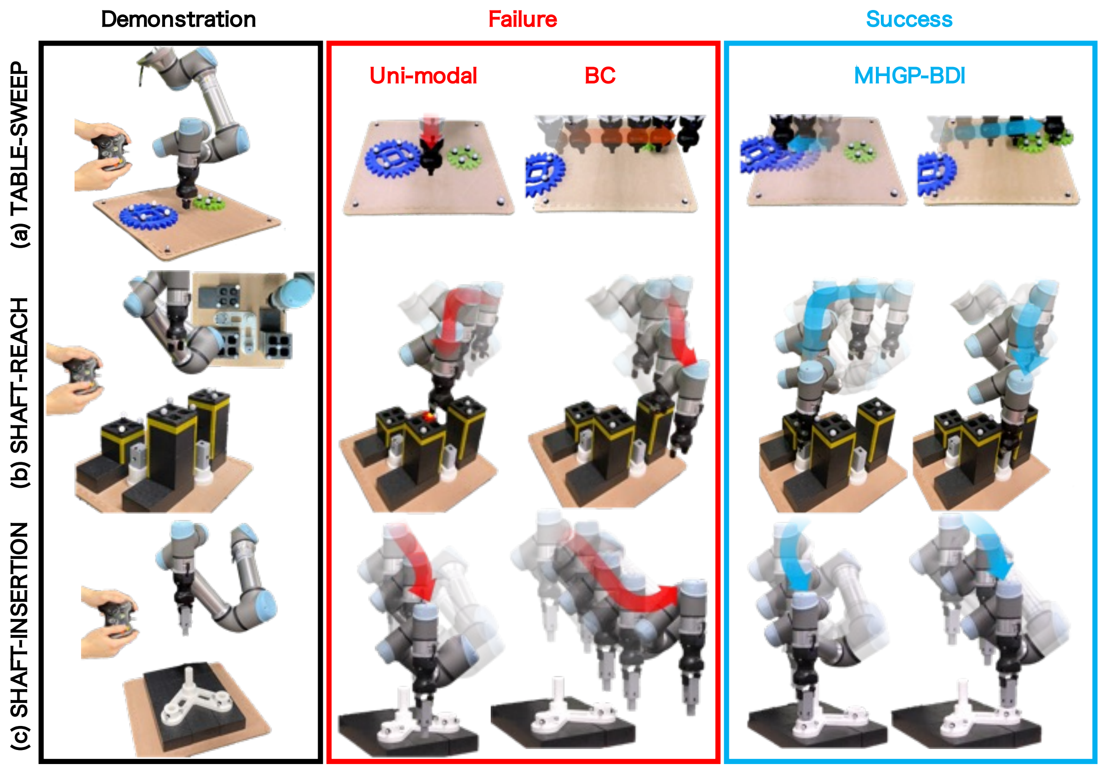

Figure 4.8. Real Robot Experiments Setup. Experimental environments for 6-DOF robotic arm (UR5e) assembly tasks with human expert are conducted as: (a) sweeping gears on the table, (b) reaching to a shaft with avoiding obstacles, (c) inserting a shaft into a hole. Test execution scenes of learned policies: **Failure**: (Uni-modal) Due to the inability to capture the multiple optimal actions, these approaches learn mean-centred policy, resulting in (a) sweeping a centre of the gears, (b) colliding to an obstacle between shafts, (c) putting a shaft onto the centre of the holes. (BC) Even though the approach can capture multiple optimal actions, without disturbance injection in demonstrations, policies are vulnerable to environmental variations, resulting in a robot departure from the demonstrated states; thus, the robot (a) cannot sweep gears completely or ((b), (c)) go out of the task space. **Success**: Our proposed method (MHGP-BDI) provides policies that are learned by capturing optimal actions or initiating recovery actions by injecting optimized disturbances, which allow the robot to successfully (a) sweep the whole gears, (b) reach to both shafts and (c) insert a shaft into the holes, in a given any starting position. Our supplementary video can be seen at: `https://youtu.be/NeJy8pfkrC4`.

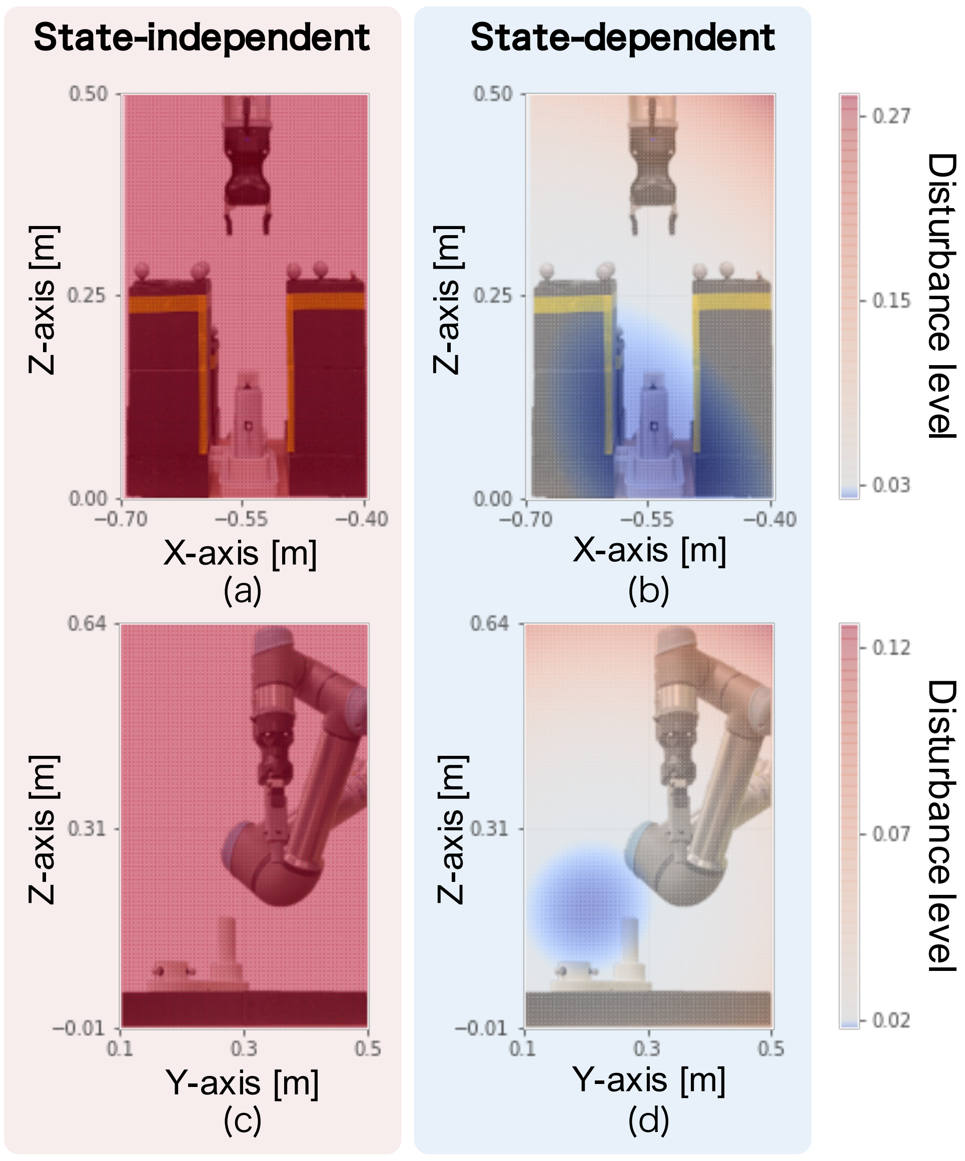

Figure 4.9. Comparison of Two Types of Disturbances. Both disturbances (state-independent and state-dependent) are injected into a human demonstration during a shaft-reach task ((a), (b)) and a shaft-insertion task ((c), (d)). Graphs showing the disturbance level with regards to the end-effector position with fixed y-coordinate ((a), (b) fixed Y-axis = 0.23 m) and the grasped shaft position with fixed X-coordinate ((c), (d) fixed X-axis = −0.7 m) : state-independent disturbances have a uniform level in any state (left), and state-dependent disturbances, obtained by MHGP-BDI, have a spectrum of level depending on the state (right). Colors of disturbance level are normalized by the amount of clearances for each task.

Table 4.4. Real Robot Experiments Results. Each learning model's demonstration success is measured at the last iteration of learning each task (10 demonstration attempts). If a human fails demonstration (*e.g.,* robot crashes with obstacles or fails to complete the task within the time limit) over 5 times at a single iteration then finish the learning process, failing to obtain a policy. Such learning models are not able to measure the task execution performance of learned policy; thus it is annotated as N/A. The test execution performance of policies learned by each learning model has been measured over 10 test executions.

Learning Models	Demonstration Success			Test Execution Performance		
	Table-sweep	Shaft-reach	Shaft-insertion	Table-sweep	Shaft-reach	Shaft-insertion
BC	10/10	10/10	10/10	0/20	0/10	0/10
DART	10/10	4/10	4/10	0/20	N/A	N/A
CVAE-BC	10/10	10/10	10/10	16/20	4/10	5/10
UGP-BC	10/10	10/10	10/10	0/20	0/10	0/10
UGP-BDI	10/10	0/10	0/10	0/20	N/A	N/A
UHGP-BDI	10/10	10/10	10/10	0/20	0/10	0/10
MGP-BC	10/10	10/10	10/10	10/20	7/10	5/10
MGP-BDI	10/10	2/10	1/10	20/20	N/A	N/A
MHGP-BDI (Proposed)	10/10	10/10	10/10	20/20	10/10	10/10

4.5. Real Robot Experiments

In this section, three experiments are conducted to demonstrate the proposed method's applicability on various scenarios as shown in Figure 4.8. MHGP-BDI is applied to a 6-DOF UR5e (Universal Robotics) robot to learn three assembly tasks:

- **Table-sweep task**: the robot's ability to reach multiple objects and sweep them out of the table, is evaluated. Demonstrations of sweeping two gears on the table are provided by a human, as shown in Figure 4.8-(a). The state of the system is defined as the relative 2D coordinate from the robotic arm to two gears ($Q = 4$); an action is defined as the velocity of the robotic arm in the x and y axis.

- **Shaft-reach task**: the robot's ability to avoid fixed obstacles and reach a shaft to grasp it, is evaluated. Demonstrations of reaching one of the assembly supplies (*e.g.,* shaft) without colliding with fixed obstacles are provided by a human, as shown in Figure 4.8-(b). The state of the system is defined as the relative 3D coordinate between the robot arm and two shafts ($Q = 6$), an action is defined as the velocity of the robot arm in the x, y and z axis. This is a more difficult task than the table-sweep task, as: (1) the state-action space is larger to deal with a more general setting, (2) the environment is prone to physical contact (*e.g.,* collision with obstacles).

- **Shaft-insertion task**: the robot's ability for inserting a shaft into a hole for assembly, is evaluated. Demonstrations of inserting the assembly supplies (*e.g.,* shaft) into one of the holes (on both side of white "L" shaped base) are provided by a human, as shown in Figure 4.8-(c). The state of the system is defined as the relative 3D coordinate between the robot arm and two holes ($Q = 6$), an action is defined as the velocity of the robot arm in the x, y and z axis. This scenario is more complicated than the shaft-reach task as: (1) Physical contacts are involved, requiring more sensitive behavior, (2) the clearance for inserting shaft is smaller (only 1 mm), requiring more precise control.

In the following experiments, learned policies are evaluated in terms of ability to flexibly learn tasks with multiple optimal actions (*e.g.,* the order in which to interact with the objects), and well as robustness to environmental covariance shift inducing disturbances (*e.g.,* friction between the objects and environment, inducing variations in movement). Here, the test execution performance of the learned policies is measured by 10 deployment tests of the final learned policy for each learning method. In addition, human demonstrations are evaluated in terms of feasibility for completing a task. For example, if the robot collides with obstacles or fails to complete a task during the demonstration stage within the time limit (400 steps), it is considered a failure, and the demonstration is instead repeated. Suppose a human fails demonstration over 5 times at a single learning iteration. In that case, it is considered learning failure (terminate the learning process), and such learning methods are removed from the task performance comparison. Here, the demonstration success is measured by conducting 10 demonstration attempts with same conditions (*e.g.,* disturbance model) used at the last learning iteration.

To measure the state of the system, markers are attached to each object (gear, shaft, hole) and tracked through a motion capture system (OptiTrack Flex13). In addition, to validate the robustness of the policy to deviations from optimal trajectories, each element of the initial state is deviated with additive uniform noise: (1) table-sweep task: $\epsilon_{\mathbf{s}_0} \sim \mathcal{U}(-0.05 \text{ m}, 0.05 \text{ m})$ (2) shaft-reach/insertion task: $\epsilon_{\mathbf{s}_0} \sim \mathcal{U}(-0.005 \text{ m}, 0.005 \text{ m})$ The assembly model used [64] (*e.g.,* gears, shafts, base) is a standardized benchmark task for robotic assembly.

4.5.1. Table-sweep Task

Setup

Initially, two gears and the robot arm are placed at fixed coordinates on a table. The human expert performs demonstrations in which the objects are swept off the table. Two demonstrations from these initial conditions are then performed, capturing both variations in the order of which the objects are swept from the table. The method optimizes a policy and disturbances until (4.26) is converged. This process is repeated $K = 4$ times (8 trajectories).

The learned policies' performance is evaluated according to the number of gears swept out of the table at the end of the test execution episode.

Result

The results of this experiment are seen in Table 4.4. In the table-sweep task, the expert can successfully perform demonstrations using any of the proposed methods, even when disturbances (*i.e.,* state-dependent or state-independent) are injected; since the environment does not involve any obstacles in which disturbances may induce risks (*e.g.,* collisions or confusion in decision making).

Given these successful demonstrations, task performance is then evaluated in Table 4.4. In this, it is seen that the uni-modal policy methods (BC, DART, UGP-BC, UGP-BDI, UHGP-BDI) all fail. Specifically, in terms of flexibility, it is seen that instead of capturing multiple optimal actions at the start of sweeping, instead a mean-centered policy is learned that fails to reach either gears (Figure 4.8-(a)

Uni-modal failure). As such, they have a zero task execution performance, and demonstrate a lack of flexibility. In comparison, the multi-modal policy methods (CVAE-BC, MGP-BC, MGP-BDI and MHGP-BDI) correctly learn that there are multiple optimal actions (*e.g.,* move to blue or green gear), and outputs actions to sweep the two gears accordingly (Figure 4.8-(a)Success). However, while CVAE-BC and MGP-BC incorporate flexibility, it has a low task performance (80% and 50%, respectively). This is due the dynamic behavior of gears varying between the test execution and training due to environmental variations (*e.g.,* friction between gears and the table), thereby introducing error compounding and resulting in the robot being unable to sweep the remaining gear after the first sweep (Figure 4.8-(a) BC failure). In contrast, while the proposed disturbance-injected methods also experiences some uncertainty, it recovers and successfully sweep gears (Figure 4.8-(a) Success); thus MGP-BDI and MHGP-BDI show greatly improved performance (both are 100%).

4.5.2. Shaft-reach Task

Setup

Prior to the start of a demonstration, two shafts and robot arm are placed at fixed positions between the obstacles (black blocks) on the table. Following the same procedure as outlined in Chapter 4.4.1, the human expert performs demonstrations in which the robot arm reach to each shaft alternatively. When the robot arm collides with an obstacle, it is considered a failure. After collecting two demonstrations, a policy and disturbances are optimized until (4.26) is converged. This process is repeated $K = 4$ times (8 trajectories).

The learned policies' performance is evaluated according to the success of the test execution episode, determined by whether the robot arm grasped the shaft at the end of the episode.

Result

The results of this experiment are seen in Table 4.4. In this, it is seen that the state-independent disturbance injection methods (DART, UGP-BDI and MGP-BDI) have a poor demonstration success rate, 40%, 0% and 20% respectively.

Specifically, to examine this result, the learned disturbance is visualised in the state-space (Figure 4.9-(a), (b)). In this, it is seen that state-independent methods generate disturbances with a uniform level, and as such inducing physical contacts (*e.g.,* collide with obstacle) at the demonstration. In contrast, a state-dependent disturbance injection methods (UHGP-BDI and MHGP-BDI) can regulate disturbance level small when robot arm close to obstacles (Figure 4.9-(b)), both have a 100% demonstrations success rate.

At the policy execution phase, it is seen that the uni-modal policy methods (BC, UGP-BC, UHGP-BDI) both fail to correctly learn policies to account for multiple optimal actions in the environment; thus robot collide with obstacle between the two shafts as shown in Figure 4.8(b)-Uni-modal failure. As such, they have a 0% success rate, and demonstrate a lack of flexibility. In contrast, the multi-modal policy methods (CVAE-BC, MGP-BC and MHGP-BDI) show improved performance (40%, 70% and 100%, respectively). However, it is clear that even when incorporating flexibility, the success rate for BC is poor; since environmental variation (*e.g.,* starting position), the robot may deviate from trained states (Figure 4.8(b)-BC failure), demonstrating a lack of robustness.

4.5.3. Shaft-insertion Task

Setup

Before the start of a demonstration, the “L" shaped base and shaft grasped robot arm are placed at fixed starting positions in the environment. This task involves physical contact (*e.g.,* between shaft and base) and requires a scheme to protect the experimental environment, including a robot and objects. As such, an impedance control [65] is implemented, that cancels the force by adding reverse direction velocity when the shaft collides with the base.

Following the same procedure as outlined in Chapter 4.4.1, the human expert performs demonstrations in which the robot arm inserts the shaft into each hole alternatively. After collecting four demonstration, a policy and disturbances are optimized until (4.26) is converged. This process is repeated $K = 3$ times (12 trajectories).

The learned policies' performance is evaluated according to the success of the

Table 4.5. Shaft-insertion Task Results (Multiple Subjects). Experimental results of robotic shaft insertion varied by four expert subjects. To obtain sufficient expert demonstrations, test subjects are practiced velocity control and make a smooth trajectory with simple instructions (*e.g.,* move the shaft from the starting point to the hole while sequentially decelerating the robotic arm), before performing the demonstrations. Each multi-modal approach (MGP-BC, MGP-BDI, and MHGP-BDI) has been validated during the demonstration and test execution phases. The success rate of each learning model is the mean and standard deviation of the results from four subjects.

Subjects	Demonstration Success			Test Execution Performance		
	MGP BC	MGP BDI	MHGP BDI	MGP BC	MGP BDI	MHGP BDI
#1	10/10	1/10	10/10	5/10	N/A	10/10
#2	10/10	0/10	10/10	3/10	N/A	9/10
#3	10/10	1/10	10/10	6/10	N/A	10/10
#4	10/10	0/10	10/10	2/10	N/A	9/10
Success Rate (%)	100 ± 0	5.0 ± 5.0	100 ± 0	40.0±15.8	N/A	95.0±5.0

test execution episode, determined by whether the shaft is in the hole at the end of the episode.

Result

The results of this experiment are seen in Table 4.4. In the demonstration phase, methods that employ state-independent disturbance injections (DART, UGP-BDI and MGP-BDI) have a uniform strong level of disturbance in any state (seen Figure 4.9-(c)). This disturbances make it challenging to insert the shaft; thus leading to a poor demonstration success rate (40% , 0% and 10%, respectively). In contrast, a state-dependent disturbance injection methods (UHGP-BDI and MHGP-BDI) regulates the disturbance level when the shaft is close to the hole (Figure 4.9-(d)), and as such has a superior demonstrations success rate (both are 100%). This allows for both enriching the demonstrations in clear open spaces, and allowing for precision manipulation in tasks that require fine control, such as physical contact.

At the test execution of learned policies, it is seen that, as expected, the uni-modal policy methods (BC, UGP-BC, UHGP-BDI) learned mean-centered policies that generate movements between the two holes and fail the task (Figure 4.8(c)-Uni-modal failure); they have a 0% success rate, demonstrating a lack of flexibility. Furthermore, similar to Chapter 4.5.2, incorporating flexibility without robustification (CVAE-BC and MGP-BC), causes the robot to deviate from trained states (Figure 4.8(c)-BC failure), giving a poor success rate (50% and 50%, respectively). In contrast, policies learned by MHGP-BDI can output multiple optimal actions while robust to sources of error as shown in Figure 4.8. In particular, despite the small clearance in the hole's vicinity, it is seen that the robot can overcome with precise control, resulting in improved performance (100%).

In addition, to evaluate intersubject robustness of the methods, four human experts with experience in robotics are used to compare multi-modal approaches (MGP-BC, MGP-BDI, MHGP-BDI); with results shown in Table 4.5. Note, for the sake of simplicity and fairness of analysis, this experiment is conducted between GP-based multi-modal imitation learning approaches. In this, injecting state-independent disturbances into demonstrations results in demonstration-infesibility for all subjects; thus, MGP-BDI has a poor demonstration success rate ($5.0 \pm 5.0\%$). Test execution performance of MGP-BC similarly demonstrates poor average success rate ($40 \pm 15.8\%$), due to error compounding similar to previous experiments. However, these results show a higher intrasubject variance, due to the inherent differences between human-specific strategies. In contrast, MHGP-BDI consistently obtains a superior success rate on both demonstrations and test executions ($100 \pm 0\%$ and $95 \pm 5.0\%$, respectively) with multiple subjects.

4.6. Summary of Chapter 4

This algorithm presents a novel paradigm on imitation learning, by focusing on learning human risk sensitivity, and demonstrating its importance and usage. Our proposal Bayesian imitation learning framework injects risk-sensitive disturbances into an expert's demonstration to learn robust multi-action policies. This framework captures intrinsic human risk sensitivity and allows for learning reduced covariate shift policies by collecting training data on an optimal set of states without losing demonstration feasibility. The effectiveness of the proposed method is verified on several simulations and real robotic tasks with human demonstrations.

5. Discussions

This dissertation presented a novel mechanism for enhancing imitation learning that captures and exploits human risk sensitivity. Our idea is implemented as two distinct imitation learning algorithms (DP-OnIL and BDI). As demonstrated in the experimental results, our proposed methods of combining human risk-sensitivity via speed-accuracy trade-off are effective for imitation learning and improve the robot's generalizability and applicability. This chapter discusses several open issues and possible future works of our proposed mechanism, and guidance on implementing risk estimation derived from human risk sensitivity as below.

5.1. Open Issues

In this section, we discuss some open issues and limitations of the presented methods that require further study and investigation.

5.1.1. Diverse Risk Sensitivities of Human Demonstrators

Although this thesis assumes a human demonstrator that has high sensitivity to risks, in practice, this situation may vary across individuals. For example, a demonstrator who emphasizes swiftly performing tasks at the expense of safety may operate the robot at high speeds even when high precision is required. Such human sensitivities can be captured as latent variables [66], and our future work will explore how this changes performances of our methods (DP-OnIL and BDI).

5.1.2. Unable to Guarantee Collision Avoidance

By inferring environmental information only from human data, this thesis avoided the expensive costs (*e.g.,* sensors, measuring system design) typically required for direct sensing of the environment. However, in practice, it is impossible to guarantee collision avoidance without directly sensing the environment. Therefore, in the future, our method can be extended to integrate technologies that map the environment around the robot using distance sensors [67, 68] to ensure collision avoidance at a minimal cost.

5.1.3. Diverse User Interface

This thesis uses two types of user interfaces: In Chapter 3, leader-following teleoperation, which can directly reflect human movement speed; in Chapter 4, joystick teleoperation, which indirectly reflects human decision-making speed. In both examples, the validity of the proposed methods is verified using the respective user interface. However, there are numerous user interfaces for controlling robots, and in imitation learning, the choice of user interface significantly impacts the demonstrator's task performance (*e.g.,* task completion time [69]). Therefore, in the future, we will investigate how variations in user interface can affect our proposed risk estimation.

5.1.4. Diverse Quality of Demonstrations

While the standard imitation learning assumes that the demonstrator is capable of outputting the optimal behavior in any given state, in robotic tasks that require high precision, such as a needle threading task [70], even a human demonstrator rarely succeeds at one time without any mistakes. Applying imitation learning in such tasks requires a lot of time and cost for collecting demonstration data. To alleviate this contradiction, human demonstrations can be parameterized with weighted values of task achievement and enabling to learn from demonstration data that contains mistakes [71–73].

5.1.5. Tasks Involving Long-term Process

While standard imitation learning only deals with low-level control abilities, such as determining velocity from a given robot's position, many tasks in daily human life have long-term processes and are divided into symbolic sub-tasks, which require high-level planning ability to determine the sequence of sub-tasks [74]. In applying conventional imitation learning to such tasks, even changing one sub-task requires re-learning the entire task from the beginning, which is inefficient and imposes a heavy burden on a human demonstrator. To address this contradiction, the hierarchical policy model [75, 76], which can simultaneously execute high-level planning in symbolic task space and low-level control in geometric task space, can be employed to learn a human high-level planning ability.

5.1.6. Environmental Uncertainty

While environmental uncertainty is not directly addressed in this paper, uncertainty and fuzziness of environment are another important challenges in applications to real systems. In the field of adaptive control, to cope with various uncertainties on environment, attempts to achieve more accurate control [77, 78] or safe-conscious engineering [79] commonly propose a probabilistic model to capture complex system dynamics' uncertainty or adapt a dynamics model to an unknown environment through iterative online learning. In light of this, BDI can be extended for employing the Model Predictive Control (MPC)-type policy model [80], in which a sequence of states and actions are estimated with a stochastic dynamics model at each time step; thereby, enabling BDI to account for environmental uncertainty.

5.1.7. Computational Complexity of GP (Chapter 4)

Since the main purpose of our experiment is to investigate the effect of our method on several tasks with underlying contradictions in the demonstration data, only the Gaussian Processes (GPs) are employed as an inference tool for generating policies or disturbance of BDI. While GP allows several advantages to our method, it still suffers from computational complexity, which significantly increases with the number of data points N as Table 4.1. As such, BDI as applied to long-term tasks with real-time control systems is limited by this underlying policy generation method. To address this, GPs' kernels can be approximated with randomized Fourier features from the fastfood algorithm [81], which lowers the computational time of computing the inverse kernel matrix from $O(N^3)$ to $O(NW^3)$, where W is a dimension of feature space.

5.2. Other Information Inherent in Human Demonstration

One of the key conceptual ideas of this thesis is to identify auxiliary information from human data within the fundamental paradigm of robotic imitation learning. To this end, our applications have shown their effectiveness by inferring environmental risk information from human data and leveraging it appropriately for imitation learning. In the future, our ideas can be extended to incorporate approaches of extracting other information (*e.g.,* stress [82]) from human data and leveraging it for imitation learning.

5.3. Guidance of Risk Estimation Derived from Human Risk Sensitivity

This section offers a step-by-step guide on how to leverage human risk sensitivity in risk estimation for generalizing to diverse robotic tasks. The main idea of our risk estimation is that the human demonstrations reflects environmental risk information based on their understanding of the environment. Therefore, it is crucial to design a training curriculum that allows humans to deepen their understanding of the environment and to develop models that decode environmental information from human data.

We concluded four steps for implementing risk estimation derived from human risk sensitivity onto new robotic systems and tasks:

1. **Construct a robotic teleoperation system:** To collect a dataset of human demonstrations, a robot should allow humans to control it to perform desired tasks. Therefore, constructing a robot teleoperation system for humans to complete the task demonstration is essential.

2. **Define risk factors involved in tasks:** Based on the given task, define factors that pose a risk (*e.g.,* collision) and make it difficult to perform the task.

3. **Design a demonstrator training curriculum:** Design practice tasks from low-risk (*e.g.,* no obstacles) to high-risk (*e.g.,* with obstacles) based on defined risk factors. Plan a curriculum for practicing tasks from low risk to high risk until sufficient success is achieved (*e.g.,* 5 successes in a row). This allows to ensure that human demonstrators understand the environment and robotic tasks.

4. **Design a risk estimation model:** Investigate human data related to defined risk factors based on human psychological findings, and design a risk estimation model to predict risk factors from collected human demonstration dataset.

6. Conclusion

Advancements in robot technology have boosted manufacturing industries; however, generating robotic manipulation is still challenging due to the necessity of explicit programming or skillful human operation, both of which are high-cost. In this situation, Imitation Learning (IL) is an attractive scheme for learning robot manipulators via observations of expert demonstrations, where a human performs a desired behavior by operating a robot.

The standard form of IL is training an agent to learn a policy that outputs expert actions from given states of the environment by relying solely on state-action pairs from human demonstrations. However, actual human demonstrations have intrinsic behavioral characteristics that enhance task performance, and these cannot be captured solely depending on simplified behavioral representations of state-action policy.

This thesis presents a novel mechanism, "Leveraging Human Behavioral Characteristics for Enriching Imitation Learning," that designs imitative learning algorithms that utilize human behavioral characteristics, embodying principles for capturing and exploiting actual demonstrator behavioral characteristics. Specifically, we develop methods to capture behavioral characteristics inherent in human demonstrations based on behavioral psychology. By typifying human behavioral characteristics, we establish an enriched imitation learning framework that captures and leverages human behavioral characteristics within the imitation learning paradigm. Through this human behavioral characteristic integration of imitation learning, robots can deepen their understanding of human behavior, improving robot generalizability and applicability.

The effectiveness of our methods is verified through two distinct imitation learning algorithms. The success of our algorithms on risk-sensitive simulations and real-robot experiments for various assembly tasks proves that, through improved risk-sensitivity, the task execution performance of policy as well as demonstration feasibility are significantly better than comparison methods.

A. Appendix: Hyperparameter of Comparisons in Chapter 3

The hyperparameter of comparisons are described in Table A.1, Table A.2, Table A.3, Table A.4, Table A.5, and Table A.6.

Table A.1. Hyperparameters of DP-OnIL.

Hyperparameters	Simulations		Real Environments	
	Aperture-pass.	Ring-thread.	Shaft-reach.	Ring-thread.
optimizer	Adam	Adam	Adam	Adam
activation function	Hardswish	Hardswish	Hardswish	Hardswish
# of hidden layers	2	2	2	2
# of hidden units per layer	64	64	64	64
minibatch size	100	100	100	100
learning rate of π_{θ^L}	1×10^{-3}	1×10^{-2}	1×10^{-3}	1×10^{-3}
weight decay of π_{θ^L}	1×10^{-5}	1×10^{-5}	0.0025	0.0025
# of network of π_{θ^L}	5	5	5	5
output acti. func. of π_{θ^L}	Identity	Identity	Identity	Identity
learning rate of μ_λ and σ^2_λ	1×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-3}
weight decay of μ_λ and σ^2_λ	0.0025	0.0025	0.0025	0.0025
output acti. func. of μ_λ	Sigmoid	Sigmoid	Sigmoid	Sigmoid
output acti. func. of σ^2_λ	Identity	Identity	Identity	Identity

Table A.2. Hyperparameters of EnsembleDAgger.

Hyperparameters	Simulations		Real Environments	
	Aperture-pass.	Ring-thread.	Shaft-reach.	Ring-thread.
optimizer	Adam	Adam	Adam	Adam
activation function	Hardswish	Hardswish	Hardswish	Hardswish
# of hidden layers	2	2	2	2
# of hidden units per layer	64	64	64	64
minibatch size	100	100	100	100
learning rate of π_{θ^L}	1×10^{-3}	1×10^{-2}	1×10^{-3}	1×10^{-3}
weight decay of π_{θ^L}	1×10^{-5}	1×10^{-5}	0.0025	0.0025
# of network of π_{θ^L}	5	5	5	5
output acti. func. of π_{θ^L}	Identity	Identity	Identity	Identity

Table A.3. Hyperparameters of BC.

Hyperparameters	Simulations		Real Environments	
	Aperture-pass.	Ring-thread.	Shaft-reach.	Ring-thread.
optimizer	Adam	Adam	Adam	Adam
activation function	Hardswish	Hardswish	Hardswish	Hardswish
# of hidden layers	2	2	2	2
# of hidden units per layer	64	64	64	64
minibatch size	100	100	100	100
learning rate of π_{θ^L}	1×10^{-3}	1×10^{-2}	1×10^{-3}	1×10^{-3}
weight decay of π_{θ^L}	1×10^{-5}	1×10^{-5}	0.0025	0.0025
# of network of π_{θ^L}	1	1	1	1
output acti. func. of π_{θ^L}	Identity	Identity	Identity	Identity

Table A.4. Hyperparameters of DAgger.

Hyperparameters	Simulations	
	Aperture-pass.	Ring-thread.
optimizer	Adam	Adam
activation function	Hardswish	Hardswish
# of hidden layers	2	2
# of hidden units per layer	64	64
minibatch size	100	100
learning rate of π_{θ^L}	1×10^{-3}	1×10^{-2}
weight decay of π_{θ^L}	1×10^{-5}	1×10^{-5}
# of network of π_{θ^L}	1	1
output acti. func. of π_{θ^L}	Identity	Identity

Table A.5. Hyperparameters of ThriftyDAgger.

Hyperparameters	Simulations
	Aperture-pass.
optimizer	Adam
activation function	Hardswish
# of hidden layers	2
# of hidden units per layer	64
minibatch size of π_{θ^L}	100
learning rate of π_{θ^L}	1×10^{-3}
weight decay of π_{θ^L}	1×10^{-5}
# of network of π_{θ^L}	5
output acti. func. of π_{θ^L}	Identity
minibatch size of Q-function	50
discount factor of Q-function	0.9999

Table A.6. Hyperparameters of HG-DAgger.

Hyperparameters	Simulations
	Aperture-pass.
optimizer	Adam
activation function	Hardswish
# of hidden layers	2
# of hidden units per layer	64
minibatch size	100
learning rate of π_{θ^L}	1×10^{-3}
weight decay of π_{θ^L}	1×10^{-5}
# of network of π_{θ^L}	1
output acti. func. of π_{θ^L}	Identity

B. Appendix: Mathematical Details of Chapter 4

B.1. Update laws of variational posteriors q

The analytical solution of variational posterior $q^*(\mathbf{f}^{(m)})$ is given by the following derivation:

$$\log q^*(\mathbf{f}) = \int \Big\{ \log p(\mathbf{a}^*, \mathbf{g}, \{\mathbf{f}^{(m)}\}, \mathbf{Z}, \mathbf{v} \mid \mathbf{S}; \boldsymbol{\theta}) \Big\} \cdot q(\mathbf{g})q(\mathbf{Z})q(\mathbf{v})\mathrm{d}\mathbf{g}\mathrm{d}\mathbf{Z}\mathrm{d}\mathbf{v} + \mathrm{Const}, \tag{B.1}$$

where, Const is a constanct term for normalizing distributions. Accordingly, $q^*(\mathbf{f}^{(m)})$ is obtained by solving (B.1) as:

$$q^*(\mathbf{f}^{(m)}) = \mathcal{N}(\mathbf{f}^{(m)} \mid \boldsymbol{\mu}_{\mathbf{f}}^{(m)}, \mathbf{C}^{(m)}), \tag{B.2}$$

$$\boldsymbol{\mu}_{\mathbf{f}}^{(m)} = \mathbf{C}^{(m)}\mathbf{B}^{(m)}\mathbf{a}^*, \tag{B.3}$$

$$\mathbf{C}^{(m)} = (\mathbf{K}_{\mathbf{f}}^{-1(m)} + \mathbf{B}^{(m)})^{-1}, \tag{B.4}$$

$$\mathbf{B}^{(m)} = \mathrm{diag}\{r_{nm}/\mathbf{H}_{nn}\}, \tag{B.5}$$

$$\mathbf{H} = \mathrm{diag}\{\exp([\boldsymbol{\mu}_{\mathbf{g}}]_n - [\boldsymbol{\Sigma}_{\mathbf{g}}]_{nn}/2)\}. \tag{B.6}$$

As similar to (B.1), the analytical solution of variational posterior $q^*(\mathbf{Z})$ is given by the following derivation:

$$\log q^*(\mathbf{Z}) = \int \Big\{ \log p(\mathbf{a}^*, \mathbf{g}, \{\mathbf{f}^{(m)}\}, \mathbf{Z}, \mathbf{v}) \Big\} \cdot q(\mathbf{f})q(\mathbf{g})q(\mathbf{v})\mathrm{d}\mathbf{g}\mathrm{d}\mathbf{f}\mathrm{d}\mathbf{v} + \mathrm{Const}. \tag{B.7}$$

Therefore, $q^*(\mathbf{Z})$ is obtained by solving (B.7) as:

$$q^*(\mathbf{Z}) = \prod_{n=1}^{N}\prod_{m=1}^{\infty} r_{nm}^{\mathbf{Z}_{nm}}, \tag{B.8}$$

$$r_{nm} = \frac{\rho_{nm}}{\sum_{m=1}^{\infty}\rho_{nm}}, \tag{B.9}$$

$$\begin{aligned}\log\rho_{nm} = &-\frac{1}{2\mathbf{H}_{nn}}\{(a_n^* - [\boldsymbol{\mu}_{\mathbf{f}}^{(m)}]_n)^2 + [\mathbf{C}^{(m)}]_{nn}\}\\ &-\frac{1}{2}\log(2\pi\mathbf{H}_{nn}) - \psi(\alpha_m+\beta_m)\\ &+\psi(\alpha_m) + \sum_{j=1}^{m-1}\{\psi(\beta_j) - \psi(\alpha_j+\beta_j)\},\end{aligned} \tag{B.10}$$

where, $\psi(\cdot)$ is the digamma function.

As well as, the analytical solution of variational posterior $q^*(\mathbf{v})$ is given by the following derivation:

$$\begin{aligned}\log q^*(\mathbf{v}) = \int\Big\{\log p(\mathbf{a}^*, \mathbf{g}, \{\mathbf{f}^{(m)}\}, \mathbf{Z}, \mathbf{v})\Big\}\cdot\\ q(\mathbf{f})q(\mathbf{g})q(\mathbf{Z})\mathrm{d}\mathbf{g}\mathrm{d}\mathbf{f}\mathrm{d}\mathbf{Z} + \text{Const.}\end{aligned} \tag{B.11}$$

As such, $q^*(v_m)$ is obtained by solving (B.11) as:

$$q^*(v_m) = \text{Beta}(v_m \mid \alpha_m, \beta_m), \tag{B.12}$$

$$\alpha_m = 1 + \sum_{n=1}^{N} r_{nm}, \tag{B.13}$$

$$\beta_m = \beta + \sum_{j=m+1}^{\infty}\sum_{n=1}^{N} r_{nj}, \tag{B.14}$$

where, Beta is the beta function.

B.2. Lower bound of marginal likelihood $\mathcal{L}(q, \Omega')$

The lower bound of the marginal likelihood $\mathcal{L}(q, \Omega')$ is analytically obtained by the following derivation:

$$
\begin{aligned}
&\mathcal{L}(q, \Omega') \\
&= \sum_{m=1}^{\infty} \log \mathcal{N}(\mathbf{a}^* \mid \mathbf{0}, \mathbf{K}_{\mathbf{f}}^{(m)} + \mathbf{B}^{-1(m)}) \\
&\quad + \sum_{n=1}^{N} \sum_{m=1}^{\infty} \Big[r_{nm} \Big\{ \psi(\alpha_m) - \psi(\alpha_m + \beta_m) - \frac{1}{2} [\boldsymbol{\mu}_{\mathbf{g}}]_n \\
&\quad + \sum_{j=1}^{m-1} \{\psi(\beta_j) - \psi(\alpha_j + \beta_j)\} - \frac{1}{2} \log 2\pi - \log r_{nm} \Big\} \\
&\quad - \frac{1}{2} \log\{[\mathbf{B}^{(m)}]_{nn} / 2\pi\} \Big] - \sum_{m=1}^{\infty} \mathrm{KL}(q(v_m) \mid\mid p(v_m)) \\
&\quad - \mathrm{KL}(\mathcal{N}(\mathbf{g} \mid \boldsymbol{\mu}_{\mathbf{g}}, \boldsymbol{\Sigma}_{\mathbf{g}}) \mid\mid \mathcal{N}(\mathbf{g} \mid \mu_0 \mathbf{1}, \mathbf{K}_{\mathbf{g}})),
\end{aligned} \tag{B.15}
$$

where,

$$
\begin{aligned}
&\mathrm{KL}(q(v_m) \mid\mid p(v_m)) \\
&= \log\{\mathrm{Beta}(v_m | 1, \beta) / \mathrm{Beta}(v_m | \alpha_m, \beta_m)\} \\
&+ (\alpha_m - 1)\psi(\alpha_m) + (\beta_m - \alpha)\psi(\beta_m) \\
&+ (1 - \alpha_m + \alpha - \beta_m)\psi(\alpha_m + \beta_m),
\end{aligned} \tag{B.16}
$$

and

$$
\begin{aligned}
&\mathrm{KL}(\mathcal{N}(\mathbf{g} \mid \boldsymbol{\mu}_{\mathbf{g}}, \boldsymbol{\Sigma}_{\mathbf{g}}) \mid\mid \mathcal{N}(\mathbf{g} \mid \mu_0 \mathbf{1}, \mathbf{K}_{\mathbf{g}})) \\
&= \frac{1}{2} \Big[\log\{|\mathbf{I} + \mathbf{K}_{\mathbf{g}} \boldsymbol{\Lambda}|\} - 1 + \mathrm{tr}\{(\mathbf{I} + \boldsymbol{\Lambda} \mathbf{K}_{\mathbf{g}})^{-1}\} \\
&\qquad + \mathbf{1}^\top \left(\boldsymbol{\Lambda} - \frac{1}{2} \mathbf{I} \right)^\top \mathbf{K}_{\mathbf{g}} \left(\boldsymbol{\Lambda} - \frac{1}{2} \mathbf{I} \right) \mathbf{1} \Big].
\end{aligned} \tag{B.17}
$$

C. Appendix: Additional Analysis of Chapter 4

C.1. Computational complexity analysis of MHGP-BDI

As described in Table 4.1 and Table 4.2, the computational complexity of MHGP-BDI is mainly related to the number of training data sets (N) and the upper bound of mixtures (M). To analyze the impact of N and M on computational complexity of MHGP-BDI in practice, *optimization time*, duration for one optimization loop, and *prediction time*, average time for 10-step prediction, were measured in a wide version of wall avoidance simulation task; the results show that the computational complexity of MHGP-BDI is increased as N and M increase, as described in Table C.1. All experiments were ran on an Intel CPU Core i9-9900 K.

C.2. Hyperparameters

Details of hyperparameters of MHGP-BDI and all comparison models are provided as below.

C.2.1. MHGP-BDI

The hyperparameters of MHGP-BDI are as follow: $M, \omega_{\mathbf{f}}, \omega_{\mathbf{g}}, \mu_0, \mathbf{\Lambda}, \beta$. These hyperparameters are empirically chosen with certain heuristic methodologies in this paper. Such hyperparameters' selection and sensitivity analysis are described as below.

Table C.1. Computational complexity analysis results: optimization time and prediction time in MHGP-BDI

N	M	Optimization time [sec]	Prediction time [sec]
713	2	2.3	0.0055
711	5	5.2	0.013
719	10	9.9	0.022
1409	2	12.3	0.022
1407	5	27.0	0.053
1415	10	49.2	0.10

The maximum number of mixtures GPs (M) and the concentration level parameter of SBP (β) are both related to the flexibility of the policy model. To spread out data to multiple GPs, β is initialized as $\beta = 100$ in all experiments. In addition, M is initialized based on the computational complexity of MHGP-BDI. Such as, a larger M makes it better for capturing multiple optimal behaviors from human demonstrations; however, the computational complexity of the algorithm increases as M grows, as shown in Table C.1. Therefore, to ensure convergence within a reasonable time frame period, $M = 5$ in all experiments.

$\omega_{\mathbf{f}}$ and $\omega_{\mathbf{g}}$ are parameters of kernel function to regress policy's and disturbance's latent function ($\mathbf{f}$ and $\mathbf{g}$, respectively). In all experiments, Radial Basis Function (RBF) kernel ($k(x, y) = \exp\{-(\|x-y\|^2)/(2\omega^2)\}$), which is most commonly used kernel in GP regression [53], is employed for all GPs. Each parameter is initialized using the maximum and the minimum of state as: $|\max(\mathbf{S}) - \min(\mathbf{S})|$.

The initial mean of disturbance level (μ_0) and the positive variational parameter ($\mathbf{\Lambda}$) are related to action variation of human demonstrations. In all experiments, $\mathbf{\Lambda} = \text{diag}\{\lambda_n\}_{n=1}^N$ is initialized as $\lambda_n = 1/2$. In addition, μ_0 is initialized with variance of actions as: $\text{var}(\mathbf{a}^*) \times 0.01$.

To analyze sensitivity to hyperparameters, one-at-a-time parameter sensitivity [36] is employed, in which the demonstration success rate and the test execution performance are measured in the wide version of wall-avoidance simulation task with varying one parameter at a time while holding the others fixed. Note, to simplify analysis, β and $\mathbf{\Lambda}$ are initialized as $\beta = 100$ and $\lambda_n = 1/2$ in all

Table C.2. MHGP-BDI hyperparameters sensitivity analysis results. The demonstration success probability of each learning model is measured for entire learning iteration with one learning trial. The test execution performance of policy application is measured as the task success probability by conducting one learning trial and testing each final learned policy 100 times. Success rate for all demonstrations are 100 %.

Parameters	Initial Value	Test Execution Performance
M	2	99%
	5	100%
	10	100%
$\omega_{\mathbf{f}}$ $\omega_{\mathbf{g}}$	$\lvert \max(\mathbf{S}) - \min(\mathbf{S})\rvert \times 0.1$	100%
	$\lvert \max(\mathbf{S}) - \min(\mathbf{S})\rvert$	100%
	$\lvert \max(\mathbf{S}) - \min(\mathbf{S})\rvert \times 10$	70%
μ_0	$\mathrm{var}(\mathbf{a}^*) \times 0.001$	100%
	$\mathrm{var}(\mathbf{a}^*) \times 0.01$	100%
	$\mathrm{var}(\mathbf{a}^*) \times 0.1$	100%

experiments. As described in Table C.2, MHGP-BDI is robust to a wide range of hyperparameters.

C.2.2. Other GP-based comparisons

The hyperparameter of other GP-based comparisons are described in Table C.3.

C.2.3. Neural networks-based comparisons

The hyperparameter of neural networks-based comparisons are described in Table C.4 and Table C.5. These hyperparameters are set based on original papers [20, 51], but some parameters are tuned to improve performance in our domain.

Table C.3. Hyperparameters of other GP-based comparisons. Since these methods employ the GP model, parameters are selected the same as in MHGP-BDI, but if the parameters are not available in the implementation, it is annotated as N/A.

Learning Models	Hyperparameters					
	M	β	$\omega_{\mathbf{f}}$	$\omega_{\mathbf{g}}$	μ_0	λ_n
UGP-BC	1	N/A	$\lvert\max(\mathbf{S})-\min(\mathbf{S})\rvert$	N/A	N/A	N/A
UGP-BDI	1	N/A	$\lvert\max(\mathbf{S})-\min(\mathbf{S})\rvert$	N/A	N/A	N/A
UHGP-BDI	1	N/A	$\lvert\max(\mathbf{S})-\min(\mathbf{S})\rvert$		$\mathrm{var}(\mathbf{a}^*)\times 0.01$	1/2
MGP-BC	5	100	$\lvert\max(\mathbf{S})-\min(\mathbf{S})\rvert$	N/A	N/A	N/A
MGP-BDI	5	100	$\lvert\max(\mathbf{S})-\min(\mathbf{S})\rvert$	N/A	N/A	N/A

Table C.4. Hyperparameters of BC and DART.

Hyperparameter	Value
optimizer	Adam
learning rate	1×10^{-2}
weight decay	1×10^{-5}
number of hidden layers	2
number of hidden units per layer	64
number of sample per minibatch	128
activation function	Tanh

Table C.5. Hyperparameters of CVAE-BC.

Hyperparameter	Value
optimizer	Adam
learning rate	1×10^{-3}
weight decay	1×10^{-5}
number of hidden layers	2
number of hidden units per layer	64
number of sample per minibatch	64
activation function	ReLU
number of latent dimension	5

Acknowledgements

I am immensely grateful to everyone who supported me throughout my PhD journey. Without their encouragement, guidance, and unwavering support, I would not have been able to complete this journey successfully.

During my Ph.D. journey, I was fortunate enough to have many great mentors, and I would like to thank Brendan Michael, Hikaru Sasaki, and Yoshihisa Tsurumine for their invaluable guidance. Most of all, I would like to express my sincere gratitude to my supervisor, Dr. Takamitsu Matsubara, whose expertise, patience, and constant feedback were instrumental in my research. His mentorship taught me many valuable life lessons and helped me achieve this work.

I would also like to acknowledge my colleagues Naoto Komeno, Tomoya Yamanokuchi, Yuki Kadokawa, Cheng-Yu Kuo, Tahara Hirotaka, Wataru Hatanaka, Wendyam Eric Lionel Ilboudo and Yuhwan Kwon for their support and motivation throughout the journey. My five years at NAIST were filled with endless discussions and creative energy, and it has been an honor to work with such talented and dedicated people. Their valuable insights and suggestions have shaped my research and contributed significantly to the success of my PhD program. I will always cherish the memories of the awesome moments during a trip to Jeonju.

Additionally, I want to express my heartfelt thanks to the secretary, Hiromi Nishimura. Her consistent support and assistance were invaluable to me in carrying out my research work. Thanks to her efficient and scrupulous management.

Finally, I would like to express my heartfelt gratitude to my parents: Inho Oh and Ilsoon Kang, and my brother Haechang Oh, for their unwavering support and encouragement. Their love, faith in me, and unwavering support during the most challenging time of my Ph.D. journey were a source of strength and motivation.

Sincerely,
Hanbit Oh

Bibliography

[1] Z. Pan, J. Polden, N. Larkin, S. Van Duin, and J. Norrish, "Recent progress on programming methods for industrial robots," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 87–94, 2012.

[2] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous robots*, vol. 40, no. 3, pp. 429–455, 2016.

[3] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International journal of robotics research*, vol. 37, no. 4-5, pp. 421–436, 2018.

[4] D. A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems*, vol. 1, p. 305–313, 1989.

[5] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics.," in *Advances in Neural Information Processing Systems*, vol. 27, pp. 1071–1079, 2014.

[6] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1–8, 2018.

[7] Y. Wang, C. C. Beltran-Hernandez, W. Wan, and K. Harada, "Hybrid trajectory and force learning of complex assembly tasks: A combined learning framework," *IEEE Access*, vol. 9, pp. 60175–60186, 2021.

[8] S. Dadhich, U. Bodin, and U. Andersson, "Key challenges in automation of earth-moving machines," *Automation in Construction*, vol. 68, pp. 212–222, 2016.

[9] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," in *Neural Information Processing Systems. Deep Learning Symposium*, 2016.

[10] A. Giusti, J. Guzzi, D. C. Cireşan, F.-L. He, J. P. Rodríguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. D. Caro, D. Scaramuzza, and L. M. Gambardella, "A machine learning approach to visual perception of forest trails for mobile robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 661–667, 2016.

[11] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer Handbook of Robotics*, pp. 1371–1394, Springer, 2008.

[12] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[13] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[14] J. S. Dyrstad, E. Ruud Øye, A. Stahl, and J. Reidar Mathiassen, "Teaching a robot to grasp real fish by imitation learning from a human supervisor in virtual reality," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 7185–7192, 2018.

[15] A. J. Nagengast, D. A. Braun, and D. M. Wolpert, "Risk sensitivity in a motor task with speed-accuracy trade-off," *Journal of neurophysiology*, vol. 105, no. 6, pp. 2668–2674, 2011.

[16] W. A. Wickelgren, "Speed-accuracy tradeoff and information processing dynamics," *Acta psychologica*, vol. 41, no. 1, pp. 67–85, 1977.

[17] M. Bain and C. Sammut, “A framework for behavioural cloning.,” in *Machine Intelligence*, pp. 103–129, 1995.

[18] S. Ross, G. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 627–635, 2011.

[19] A. Venkatraman, M. Hebert, and J. A. Bagnell, “Improving multi-step prediction of learned time series models,” in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, p. 3024–3030, 2015.

[20] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg, “DART: Noise injection for robust imitation learning,” in *Proceedings of the Conference on Robot Learning*, pp. 143–156, 2017.

[21] H. Sasaki and T. Matsubara, “Multimodal policy search using overlapping mixtures of sparse Gaussian process prior,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2433–2439, 2019.

[22] C. Celemin, R. Pérez-Dattari, E. Chisari, G. Franzese, L. de Souza Rosa, R. Prakash, Z. Ajanović, M. Ferraz, A. Valada, J. Kober, *et al.*, “Interactive imitation learning in robotics: A survey,” *Foundation and Trends® in Robotics*, vol. 10, no. 1-2, pp. 1–197, 2022.

[23] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, “HG-DAgger: Interactive imitation learning with human experts,” in *IEEE International Conference Robotics Automation*, pp. 8077–8083, 2019.

[24] E. Chisari, T. Welschehold, J. Boedecker, W. Burgard, and A. Valada, “Correct me if i am wrong: Interactive learning for robotic manipulation,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3695–3702, 2022.

[25] J. Spencer, S. Choudhury, M. Barnes, M. Schmittle, M. Chiang, P. Ramadge, and S. Srinivasa, “Learning from interventions: Human-robot interaction as both explicit and implicit feedback,” in *Proceedings of Robotics: Science and Systems*, 2020.

[26] R. Hoque, L. Y. Chen, S. Sharma, K. Dharmarajan, B. Thananjeyan, P. Abbeel, and K. Goldberg, "Fleet-DAgger: Interactive robot fleet learning with scalable human supervision," in *Conference on Robot Learning*, pp. 368–380, 2023.

[27] K. Menda, K. Driggs-Campbell, and M. J. Kochenderfer, "Ensembledagger: A bayesian approach to safe imitation learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5041–5048, 2019.

[28] J. Zhang and K. Cho, "Query-efficient imitation learning for end-to-end simulated driving," in *Proceedings of the AAAI Conference on Artificial Intelli.*, p. 2891–2897, 2017.

[29] R. Hoque, A. Balakrishna, E. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg, "ThriftyDAgger: Budget-aware novelty and risk gating for interactive imitation learning," in *Conference on Robot Learning*, pp. 598–608, 2021.

[30] R. Hoque, A. Balakrishna, C. Putterman, M. Luo, D. S. Brown, D. Seita, B. Thananjeyan, E. Novoseller, and K. Goldberg, "LazyDAgger: Reducing Context Switching in Interactive Imitation Learning," in *IEEE International Conference on Automation Science and Engineering*, pp. 502–509, 2021.

[31] H.-I. Lin and C. G. Lee, "Speed-accuracy optimization for skill learning," in *IEEE International Conference Robotics Automation*, pp. 2506–2511, 2009.

[32] L. Murphy and P. Newman, "Risky planning: Path planning over costmaps with a probabilistically bounded speed-accuracy tradeoff," in *IEEE International Conference Robotics Automation*, pp. 3727–3732, 2011.

[33] C. M. Harris and D. M. Wolpert, "Signal-dependent noise determines motor planning," *Nature*, vol. 394, no. 6695, pp. 780–784, 1998.

[34] D. Nix and A. Weigend, "Estimating the mean and variance of the target probability distribution," in *Proceedings of IEEE Int. Conf. on Neural Networks*, vol. 1, pp. 55–60, 1994.

[35] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," in *arXiv:1606.01540*, 2016.

[36] D. M. Hamby, "A review of techniques for parameter sensitivity analysis of environmental models," *Environmental monitoring and assessment*, vol. 32, no. 2, pp. 135–154, 1994.

[37] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín, "robosuite: A modular simulation framework and benchmark for robot learning," in *arXiv:2009.12293*, 2020.

[38] S. Ross and D. Bagnell, "Efficient reductions for imitation learning," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 661–668, 2010.

[39] S. Sastry and M. Bodson, *Adaptive control: stability, convergence and robustness.* Courier Corporation, 2011.

[40] M. Lazaro-Gredilla and M. Titsias, "Variational heteroscedastic Gaussian process regression," in *Proceedings of the International Conference on Machine Learning*, pp. 841–848, 2011.

[41] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural computation*, vol. 25, no. 2, pp. 328–373, 2013.

[42] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, "Learning movement primitives," in *Robotics research. The International Symposium*, pp. 561–572, Springer, 2005.

[43] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with Gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[44] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intelligent service robotics*, vol. 9, no. 1, pp. 1–29, 2016.

[45] C. M. Bishop, *Pattern recognition and machine learning.* springer, 2006.

[46] M. Kyrarini, M. A. Haseeb, D. Ristić-Durrant, and A. Gräser, “Robot learning of industrial assembly task via human demonstrations,” *Autonomous Robots*, vol. 43, no. 1, pp. 239–257, 2019.

[47] Y. Huang, L. Rozo, J. Silvério, and D. G. Caldwell, “Kernelized movement primitives,” *The International Journal of Robotics Research*, vol. 38, no. 7, pp. 833–852, 2019.

[48] D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” *International Conference on Learning Representations*, 2013.

[49] K. Sohn, H. Lee, and X. Yan, “Learning structured output representation using deep conditional generative models,” *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[50] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine, “Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3758–3765, IEEE, 2018.

[51] A. Z. Ren, S. Veer, and A. Majumdar, “Generalization guarantees for imitation learning,” in *Proceedings of the Conference on Robot Learning*, pp. 1426–1442, 2020.

[52] F.-I. Hsiao, J.-H. Kuo, and M. Sun, “Learning a multi-modal policy via imitating demonstrations with mixed behaviors,” *arXiv preprint arXiv:1903.10304*, 2019.

[53] C. E. Rasmussen, “Gaussian processes in machine learning,” in *Summer School on Machine Learning*, pp. 63–71, Springer, 2003.

[54] M. Arduengo, A. Colomé, J. Lobo-Prat, L. Sentis, and C. Torras, “Gaussian-process-based robot learning from demonstration,” *arXiv preprint arXiv:2002.09979*, 2020.

[55] A. P. Shon, K. Grochow, and R. P. Rao, “Robotic imitation from human motion capture using Gaussian processes,” in *5th IEEE-RAS International Conference on Humanoid Robots, 2005.*, pp. 129–134, IEEE, 2005.

[56] D. Nguyen-Tuong, M. Seeger, and J. Peters, “Model learning with local gaussian process regression,” *Advanced Robotics*, vol. 23, no. 15, pp. 2015–2034, 2009.

[57] M. Lázaro-Gredilla, S. Van Vaerenbergh, and N. D. Lawrence, “Overlapping mixtures of Gaussian processes for the data association problem,” *Pattern recognition*, vol. 45, no. 4, pp. 1386–1395, 2012.

[58] J. C. Ross and J. G. Dy, “Nonparametric mixture of Gaussian processes with constraints,” in *Proceedings of the International Conference on Machine Learning*, pp. 1346–1354, 2013.

[59] J. Sethuraman, “A constructive definition of Dirichlet priors,” *Statistica sinica*, pp. 639–650, 1994.

[60] G. Parisi, *Statistical field theory.* Addison-Wesley, 1988.

[61] M. Opper and C. Archambeau, “The variational Gaussian approximation revisited,” *Neural computation*, vol. 21, no. 3, pp. 786–792, 2009.

[62] Q. Liu and D. A. Pierce, “A note on Gauss — Hermite quadrature,” *Biometrika*, vol. 81, no. 3, pp. 624–629, 1994.

[63] H. Oh, H. Sasaki, B. Michael, and T. Matsubara, “Bayesian Disturbance Injection: Robust imitation learning of flexible policies,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 8629–8635, 2021.

[64] Siemens, “Robot learning challenge,” 2017. Last accessed 27 February 2021.

[65] V. Duchaine and C. M. Gosselin, “General model of human-robot cooperation using a novel velocity based variable impedance control,” in *Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC’07)*, pp. 446–451, 2007.

[66] A. Majumdar, S. Singh, A. Mandlekar, and M. Pavone, “Risk-sensitive inverse reinforcement learning via coherent risk models.,” in *Proceedings of Robotics: Science and Systems*, 2017.

[67] Y. Liu, A. Xu, and Z. Chen, "Map-based deep imitation learning for obstacle avoidance," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8644–8649, IEEE, 2018.

[68] L. Sharma, M. Everett, D. Lee, X. Cai, P. Osteen, and J. P. How, "Ramp: A risk-aware mapping and planning pipeline for fast off-road ground robot navigation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5730–5736, IEEE, 2023.

[69] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, *et al.*, "Roboturk: A crowdsourcing platform for robotic skill learning through imitation," in *Conference on Robot Learning*, pp. 879–893, PMLR, 2018.

[70] I. Jourdan, E. Dutson, A. Garcia, T. Vleugels, J. Leroy, D. Mutter, and J. Marescaux, "Stereoscopic vision provides a significant advantage for precision robotic laparoscopy," *Journal of British Surgery*, vol. 91, no. 7, pp. 879–885, 2004.

[71] D. S. Brown, W. Goo, and S. Niekum, "Better-than-demonstrator imitation learning via automatically-ranked demonstrations," in *Proceedings of the Conference on Robot Learning*, pp. 330–359, 2019.

[72] L. Chen, R. Paleja, and M. Gombolay, "Learning from suboptimal demonstration via self-supervised reward regression," in *Proceedings of the Conference on Robot Learning*, pp. 1262–1277, 2020.

[73] H. Tahara, H. Sasaki, H. Oh, B. Michael, and T. Matsubara, "Disturbance–injected Robust Imitation Learning with Task Achievement," in *Intl. Conference on Robotics and Automation*, pp. 2466–2472, 2022.

[74] R. E. Fikes and N. J. Nilsson, "STRIPS: A new approach to the application of theorem proving to problem solving," *Artificial Intelli.*, vol. 2, no. 3-4, pp. 189–208, 1971.

[75] R. Fox, R. Berenstein, I. Stoica, and K. Goldberg, "Multi-task hierarchical imitation learning for home automation," in *IEEE International Conference on Automation Science and Engineering*, pp. 1–8, 2019.

[76] D. Xu, R. Martín-Martín, D.-A. Huang, Y. Zhu, S. Savarese, and L. F. Fei-Fei, "Regression planning networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[77] X. Xin, Y. Tu, V. Stojanovic, H. Wang, K. Shi, S. He, and T. Pan, "Online reinforcement learning multiplayer non-zero sum games of continuous-time Markov jump linear systems," *Applied Mathematics and Computation*, vol. 412, p. 126537, 2022.

[78] Z. Zhuang, H. Tao, Y. Chen, V. Stojanovic, and W. Paszke, "Iterative learning control for repetitive tasks with randomly varying trial lengths using successive projection," *International Journal of Adaptive Control and Signal Processing*, vol. 36, no. 5, pp. 1196–1215, 2022.

[79] P. Cheng, H. Wang, V. Stojanovic, S. He, K. Shi, X. Luan, F. Liu, and C. Sun, "Asynchronous fault detection observer for 2-d Markov jump systems," *IEEE Transactions on Cybernetics*, pp. 1–12, 2021.

[80] M. Pereira, D. D. Fan, G. N. An, and E. Theodorou, "MPC-inspired neural network policies for sequential decision making," *arXiv preprint arXiv:1802.05803*, 2018.

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

[82] B.-G. Lee and W.-Y. Chung, "Wearable glove-type driver stress detection using a motion sensor," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1835–1844, 2016.

Publication List

Journal

(i) Hanbit Oh, Hikaru Sasaki, Brendan Michael and Takamitsu Matsubara, "Bayesian Disturbance Injection: Robust imitation learning of flexible policies for robot manipulation," *Neural Networks*, vol.158, no.4 pp.42-58, January 2023

[Corresponding to Chapter 4]

(ii) Hanbit Oh, Takamitsu Matsubara, "Leveraging Demonstrator-perceived Precision for Safe Interactive Imitation Learning of Clearance-limited Tasks," *IEEE Robotics and Automation Letters*, vol. 9, no. 4, pp.3387-3394, April 2024

[Corresponding to Chapter 3]

International Conference (Reviewed)

(i) Hanbit Oh, Hikaru Sasaki, Brendan Michael and Takamitsu Matsubara, "Bayesian Disturbance Injection: Robust Imitation Learning of Flexible Policies," *IEEE International Conference on Robotics and Automation*, pp.8629-8635, May 2021

[Corresponding to Chapter 4]

Domestic Conference

(i) Hanbit Oh，佐々木光，松原崇充，無限重複混合ガウス過程に基づく頑健・柔軟な模倣学習，第38回日本ロボット学会学術講演会，1C2-02，October 2020.

(ii) Hanbit Oh, Hikaru Sasaki, Brendan Michael, and Takamitsu Matsubara, "Bayesian Disturbance Injections: Safely learning robust and flexible policies，" 第39回日本ロボット学会学術講演会，2A3-04，September 2021.

Others

Journal

[1] Hirotaka Tahara, Hikaru Sasaki, Hanbit Oh, Edgar Anarossi, and Takamitsu Matsubara, “Disturbance Injection under Partial Automation: Robust Imitation Learning for Long-horizon Tasks,” *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp.2724-2731, May 2023

International Conference (Reviewed)

[1] Hirotaka Tahara, Hikaru Sasaki, Hanbit Oh, Edgar Anarossi, and Takamitsu Matsubara, “Disturbance Injected Robust Imitation Learning with Task Achievement,” *IEEE International Conference on Robotics and Automation*, pp.2466-1115, May 2022.

[2] Hirotaka Tahara, Hikaru Sasaki, Hanbit Oh, Edgar Anarossi, and Takamitsu Matsubara, “Disturbance Injection under Partial Automation: Robust Imitation Learning for Long-horizon Tasks,” *IEEE/RSJ International Conference on Intelligent Robots and Systems* , October 2023.
(as presentation option of Robotics and Automation Letters)

Domestic Conference

[1] 田原熙昂，Oh Hanbit，佐々木光，松原崇充 (奈良先端大)，タスク達成度を考慮した教示者に摂動を加えるロバスト模倣学習，第21回計測自動制御学会システムインテグレーション部門講演会，3D3-13，December 2020.

[2] 田原熙昂，Oh Hanbit，佐々木光，松原崇充 (奈良先端大)，Coarse2Fine ロバスト模倣学習，ロボティクス・メカトロニクス講演会，2P1-B01，June 2022.