Ergodic Imitation with Corrections: Learning from Implicit Information in Human Feedback

Junru Pang, Quentin Anderson-Watson, Kathleen Fitzsimons

***Abstract*—As the prevalence of collaborative robots increases, physical interactions between humans and robots are inevitable—presenting an opportunity for robots to not only maintain safe working parameters with humans but also learn from these interactions. To develop adaptive robots, we first aim to analyze human responses to different errors through a study in which users are asked to correct any errors that the robot makes in various tasks. With this characterization of corrections, we can treat physical human-robot interactions (pHRI) as informative instead of ignoring physical interactions or leaving robots to return to the originally planned behaviors when interactions end. We incorporate physical corrections into existing Learning from Demonstration (LfD) frameworks, which allow robots to learn new skills by observing human demonstrations. We demonstrate that learning from physical interactions can improve task-specific performance metrics. The results reveal that including information about the behavior being corrected in the update improves task performance significantly compared to adding corrected trajectories alone. In a user study with an optimal control-based LfD framework, we also find that users are able to provide less feedback to the robot after each interaction update to the robot's behavior. Utilizing corrections could enable advanced LfD techniques to be integrated into commercial applications for collaborative robots by enabling end-users to customize the robot's behavior through intuitive interactions rather than by modifying the behavior in software.**

***Index Terms*—Robotics, Human-Robot-Interaction, Adaptive System**

I. Introduction

Perhaps the most common strategy to handle physical human-robot interaction (pHRI) is through designing controllers that avoid contact with humans during collaboration [1] or implementing some form of impedance control—treating human-robot interaction as a disturbance. When that interaction is intentional, robots could utilize those 'disturbances' to adapt their behavior in dynamic and unpredictable environments. Human guidance allows robots to improve decision-making processes, correct mistakes, and align actions with user-desired behavior. Utilizing the knowledge from intuitive feedback through physical corrections can reduce barriers to translating existing research on Learning from Demonstration (LfD) into commercial robot applications. In such frameworks, a human partner can provide information on task goals or preferences that are difficult to capture in a small number of demonstrations. Human feedback could enable the autonomy to continually develop a more comprehensive definition of the desired behavior as long as that feedback is interpreted correctly by the autonomy. This may reduce robot mistakes and improve human-robot trust [2], [3].

Human feedback can be explicit expressions of human preferences as in binary queries [4], [5], performance ranking [6], and corrections [7]. These active learning approaches treat users as an oracle that provides a reward signal indicating successful task completion. In performance ranking, a user scores trajectories that the IRL framework uses to maximize the expected discounted sum of rewards [8]. Binary queries can train a classifier that determines task success such that the classifier can be used as a reward function for sparse reinforcement learning [4]. Physical interactions and ranking information have also been combined to infer the human's preferred trajectory features through a learned model of the human's scoring function [7].

Information can also be inferred online from user corrections captured through a teleoperation interface [9], [10]. Physical corrections to an impedance-controlled robot arm [7], [11], [12] can be combined with IOC frameworks, such that the corrections are extrapolated to apply a deformation to the rest of the trajectory. Alternatively, long-term performance in the task can be improved by modifying the control policy [13] or the parameterized task definition [14], [15].

Though some recent work has aimed to use interventions indirectly to train a residual policy [16] or to generate synthetic interventions [17], most methods for learning from physical corrections treat the interaction as a source of explicit trajectory adjustment as in [18]. In the present work, we aim to understand what implicit information may be contained in the physical corrections and assess how this can be leveraged to improve robot learning from interactions. First, we collect data on human corrections to a set of pre-defined errors with and without online imitation learning in a user study. We analyze the interaction time, interaction force, and the delay between the start of the error and the user intervention. We observed a significant delay between the onset of the error and the user's response. In contrast to [11]–[15], these results suggest that we could leverage the implicit and explicit information encoded in the pHRI to update a control objective. Therefore, we propose a framework in which human corrections not only cause the deformation on the rest of the trajectory but also indicate that the previously planned robot trajectory is a demonstration of "what not to do". We adopted the imitation learning framework developed in [19] to change the robot's ongoing behavior and future iterations. The timing of human corrections and relative interaction forces observed in the study of pre-defined errors was used to select parameters for the interaction update. The experimental performance of the interaction-based learning framework is presented on a 7-DOF

Junru Pang, Quentin Anderson-Watson, and Katie Fitzsimons are with Department of Mechanical Engineering, Pennsylvania State University, University Park, PA 16802, USA [jvp6149 | qxa5031 | k-fitzsimons]@psu.edu

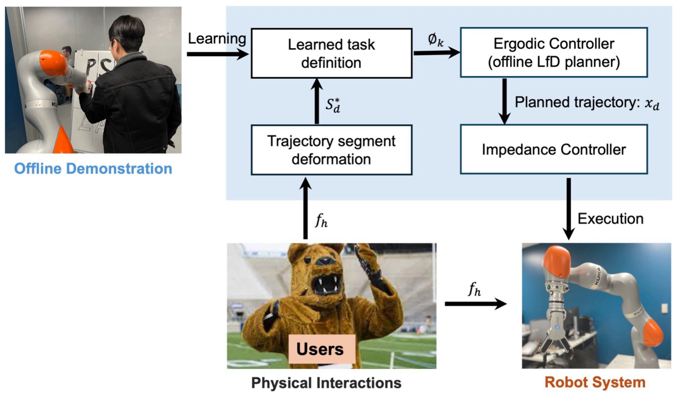

Fig. 1: Learning from Interaction Framework: While the robot is executing the learned task, its sensors monitor external forces for user interaction. When the external force is detected, a deformed trajectory is calculated to update the current task execution and update the learned task definition.

robot arm. The result demonstrates that online updates to the task definition reduced the amount of time and interaction force that humans spend correcting the robot's performance. Finally, we explored multiple ways to incorporate implicit information. The algorithm is tested under different assumptions about the implicit information. The results show that including implicit information from the originally planned robot trajectory could improve robot performance more than explicit corrections alone. In addition, utilizing correction requires fewer demonstrations than offline LfD for similar performance outcomes. The contributions of this work are 1) a user study characterizing human corrections to robot errors, 2) a framework for incorporating implicit information from the corrections into an LfD method such that updates are made online, and 3) an evaluation of the LfD framework under various interpretations of the implicit information contained in the corrections.

II. Related Work

Compared to manually programming a time series of desired joint angles, LfD has been used to reduce the need for end-users to be able to write complex and tedious programs since the 1980s [20]. Demonstration data can be collected via observations [21], kinesthetic teaching [22], or teleoperation [23], [24]. This data is used to learn a reference task definition for optimal control methods [25], [26], or a control policy can be learned directly through reinforcement learning (RL) methods [27]–[29]. For both Inverse Optimal Control (IOC) and RL-based methods, the demonstrations provided often must be expert-provided or represent a near-optimal execution of the task [30]–[32].

Probabilistic methods can account for small errors in demonstrations with small-scale noise in the probabilistic representation [33]–[37]. To further relax these constraints on demonstration data, several recent LfD approaches enable robots to learn from sub-optimal demonstrations [38], [39]. Failed demonstrations [40]–[43] and demonstrations of what not to do [19] can also yield control policies that successfully replicate the intended task. In the present study, we aim to understand how humans correct errors such that information from these corrections can update the control policies *online*.

A. Ergodic Imitation

We use Ergodic Imitation [19] to control the robot so that its end-effector trajectories reflect the demonstrated task. By integrating suboptimal and failed demonstrations, ergodic imitation allows robots to learn robust behaviors, even in scenarios where demonstrations contain errors or imperfections. This flexibility reduces the need for expert-level demonstrations, enabling the system to handle potentially noisy human inputs.

Ergodic imitation is an IOC method in which tasks are defined through spatial distributions in a state-based feature space, and ergodic control [44], [45] is used to minimize the difference between the temporal statistics of the robot trajectory and the spatial distribution defining the task. This difference, ergodicity, can be measured using the spectral approach [46], which characterizes ergodicity by comparing spatial Fourier coefficients of the trajectory $x(t)$ to coefficients of a reference distribution $\phi(x)$—giving us the distance from ergodicity.

The ergodicity of the robot trajectory is quantified by the sum ε of the weighted square distance between the Fourier coefficients of the distribution ϕ_k and the coefficients of the spatial Fourier transform of the trajectory c_k:

$$c_k = \frac{1}{T}\int_0^T F_k(x(t))dt, \tag{1}$$

where $F_k(x)$ is the Fourier basis function. The spatial statistics of each demonstration trajectory d are represented by the coefficients computed in Eq. 1, and the learned distribution is defined by coefficients ϕ_k, which are computed by a weighted average:

$$\phi_k = \sum_{j=1}^{m} w_j c_{k,j}, \tag{2}$$

where the weighting factor w_j normalizes each demonstration based on either the length of the trajectory or the relative quality of the demonstration. Therefore, demonstrations of what not to do (negative demonstrations) are combined with a negative weight—$w_j < 0$ in Eq. 2.

B. Trajectory Deformation

Trajectory deformation can be used to modify a planned robot trajectory beyond the deviation resulting from an instantaneous physical interaction between the human and an impedance-controlled robot. The deformation of the trajectory is computed independently along each axis [11]. The original trajectory segment, S_d, is represented as a vector, with entries describing the position of the system at regular time intervals between the contact time, t_c, and the end of the prediction window, t_p. The deformed trajectory, S_d^*, is computed by

$$S_d^* = S_d + \mu D f_h(t_c), \tag{3}$$

where f_h is the force applied by the human and $\mu \in [0, 1]$ is a parameter that changes the sensitivity of the trajectory deformation to the force magnitude. When μ is relatively small, the user needs to input a larger force to achieve smaller deformations. For the larger value of μ, a smaller force will cause larger trajectory deformations. The unitless term, D, can be used to tune the deformed trajectory. For instance, when D is computed as:

$$D = (A^T A)^{-1}, \tag{4}$$

where A is a finite differencing matrix, it ensures that the resulting trajectory is minimum jerk [47].

In the present work, S_d is the segment of the originally planned trajectory, x_{ec}, that is recomputed when a human-applied force f_h exceeds a measurement threshold. We selected a moderate value, $\mu = 0.5$, for the sensitivity of the deformation calculation to the applied force and use the finite difference matrix A from [47] that results in deformed trajectories with the minimum jerk.

III. Methods

In Experiment 1, we investigate how humans choose to correct robot errors through physical interactions by analyzing the timing and forces applied during corrections to pre-defined trajectory errors in 3 tasks performed by a 7-DOF robot arm. We hypothesized that collision or safety errors would lead participants to interact with the robot earlier in the deviation from the desired trajectory, with high forces and longer interaction times compared to errors in which the robot deviated slightly from the nominal trajectory. Using the results of Experiment 1, we evaluate different ways to incorporate the implicit and explicit information in these corrections into an online learning from interaction algorithm in Experiment 2. Below, we first describe the online learning algorithm in section III-A, followed by the experimental design and data analysis of the user study (Experiment 1). This is followed by an explanation of the alternative update methods developed as a result of Experiment 1 that are used in Experiment 2, which evaluates the impact of positive demonstrations—deformed trajectories—and demonstrations of what not to do—the current robot trajectory and the trajectory that had been planned prior to the correction.

A. Incorporating Corrections into Ergodic Imitation

Ergodic imitation learning is implemented as in [19], where an initial definition of the task is learned from a set of demonstrations D with positive/negative labels from an offline training phase. For the MPC method used to generate the end-effector trajectory, we define the dynamic model of the form:

$$\dot{x} = f(x, u) = g(x) + h(x)u \tag{5}$$

where $x \in \mathbb{R}^n$ is the state of the agent and $u \in \mathbb{R}^m$ is the control input. The system must be control-affine to use the non-linear MPC method in [48]. We define the task objective:

$$J = q\varepsilon + \int_0^T \frac{1}{2} u(t) R u(t) dt, \tag{6}$$

where q is a weight on the ergodic metric, and R is a weight on the control effort. The ergodic metric ε is a measure of the trajectory's distance from ergodicity:

$$\varepsilon = \sum_{k_1=0}^{K} \cdots \sum_{k_n=0}^{K} \Lambda_k |c_k - \phi_k|^2, \tag{7}$$

where the coefficient $\Lambda_k = 1/((1 + ||k||^2)^s)$, $s = (n+1)/2$. This quantifies how closely the robot trajectory $x(t)$ matches the spatial statistics of the target distribution $\phi(x)$ by computing a weighted sum of the squared difference between the spatial Fourier coefficients of the target distribution, ϕ_k (Eq. 2), and the coefficients of the time-averaged trajectory, c_k, computed using Eq. 1.

During trajectory execution, user inputs above a threshold force were used to compute a trajectory segment using Eq. 3. This trajectory segment and portions of the originally planned trajectory are used to recalculate the Fourier coefficients of the target distribution, ϕ_k. The spatial Fourier coefficients of the deformed trajectory are computed using Eq. 1. The reference distribution, ϕ_k is recomputed as the weighted average of the previous demonstrations and the additional demonstration data with a positive weight assigned to the deformed trajectory and negative weight assigned to the originally planned trajectory according to Eq. 2. This incremental update ensures that the modified distribution reflects the applied corrections. The MPC then re-optimizes the controls $u(t)$ to track a new trajectory that minimizes the objective in Eq. 6. The full algorithm is outlined in Algorithm 1, where t_c denotes the start time of each interaction, t_p is a fixed offset after t_c, and $f_{threshold}$ is predefined.

B. Experiment 1: User Study on Impact of Error Types

To evaluate human corrective actions, we used three tasks: a peg-in-hole task, pouring a cup with obstacle avoidance, and drawing & erasing on a whiteboard as shown in Fig. 2.

Algorithm 1 Ergodic Imitation with Online Corrections

```
Input: initial time $t_0$, initial state $x_0$, set of demonstrations $D = \{d_1, ..., d_m\}$ with positive/negative labels $\{e_1, ..., e_m\}$, final time $t_f$
    Define: impedance parameters, ergodic cost weight $q$, highest order of coefficients $K$, control weight $R$, search domain bounds $\{L_1, ...L_n\}$, sampling time $t_s$, time horizon $T$, Interaction force threshold $f_{threshold}$
Output: ergodic trajectory $x_{ec} \rightarrow X$
2:  Initialize: nominal control $u_{nom}$, $i = 0$
    Generate distribution $D(s)$ from demonstration set $D$.
4:  Calculate $\phi_k$ from distribution $D(s)$
    while $t_i < t_f$ do
6:      Compute $u_i^*$ using MPC
        Apply $u_i^*$ for $t \in [t_i, t_i + t_s]$ to get $x \forall t \in [t_i, t_i + t_s]$.
8:      if $f_h > f_{threshold}$ then
            Calculate $S_d^*$ using Equation 3
10:         $x_{ec} = \begin{cases} x_{ec} & \text{if } t < t_c \text{ or } t > t_p, \\ S_d^* & \text{if } t_c \leq t \leq t_p \end{cases}$
            Assigning weights $w_j$ based on methods described in section III-C
12:         Update $\phi_k$ using equation 2: $\phi_k = \sum_{j=1}^{m} w_j c_{k,j}$
            Define $t_{i+1} = t_i + t_s, x_{i+1} = x(t_{i+1})$
14:     end if
        $i \leftarrow i + 1$
16: end while
```

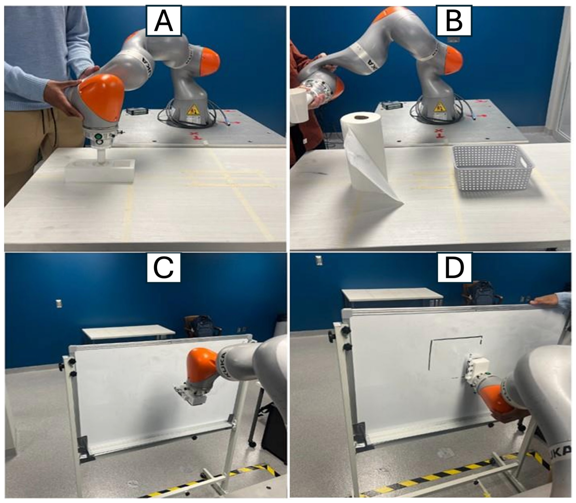

Fig. 2: **Pre-defined Error Tasks**. (A)Peg-in-hole task (B) Pour task: Kuka delivers the cup and pours the foam balls into the assigned location without collision ; (C) Draw & Erase task: Kuka draws a square on the whiteboard; (D) Draw & Erase task: Kuka erases the square drawn before.

Within the 3 tasks, we defined four types of errors: no error, substandard, task failure, and safety. In no-error trajectories, the robot successfully completed the task as quickly as possible. Substandard errors occur when the robot completes the task, but with a less direct and often longer trajectory than the no-error example. During task failure errors, the robot does not complete the task at all. Safety errors included trajectories where the robot collides with an object in the workspace or enters the space near the participant. We collect data on the magnitude of forces used to correct erroneous behaviors and the amount of time users spend providing corrections. We hypothesized that the type of error would be a significant factor in the forces applied by the participant.

1) Tasks Descriptions: In the **peg-in-hole task**, the robot inserts a round peg into the round hole on a test fixture. The test fixture has three holes of different shapes and sizes. This task had two task-failure and two safety errors. In the no-error trajectory, the robot inserted the peg directly into the circular hole in the peg box. In contrast, for the two task failure errors, the robot inserted the peg into the wrong hole, or the robot slightly missed the hole. For the two safety errors, the peg hit the side of the box, or the peg moved out towards the participant before inserting itself into the box.

In the **pouring task**, the robot arm delivers a cup and pours foam balls into a basket without colliding with an obstacle. The task had four predefined mistakes: one substandard error, one task failure error, and two safety errors. The substandard trajectory had the robot moving over an obstacle before pouring the contents of the cup into a bowl with an exaggerated motion to avoid the obstacle. In the task failure trajectory, the robot moved over the obstacle but poured the cup early. The two safety errors involve the robot colliding with the obstacle and the basket itself.

In the **draw & erase task**, the robot arm draws a square and then erases it on the same whiteboard. The end effector was designed to allow the robot to easily switch between drawing and erasing by simply rotating the end effector. This task had three task failure trajectories. For the no-error trajectory, the robot draws a square and then erases the square immediately. For the three task failure trajectories, the robot draws a diamond instead of a square before erasing, draws the correct square, but does not erase the square completely. Finally, the robot attempts to draw and erase the square; however, the end-effector is not close enough to make contact with the board.

2) User Study Protocol & Analysis: 8 right-handed participants (3 male, 5 female) ranging in age from 18-35 were recruited to participate in this study. Before interacting with the robot, the researchers explained the types of tasks and errors that the participant could expect to see. Participants were given the opportunity to interact with the robot during practice tasks before data collection began. The participants familiarized themselves with how the robot moved and how much force could be applied to the robot to change its motion. We employ two pHRI strategies in this user study: a baseline impedance controller and ergodic imitation with interaction-based updates. For ergodic imitation with interaction-based updates, the robot would react to human corrections by re-planning a new trajectory during execution.

We employed the impedance control mode of `iiwa_stack` ROS package [49] in both strategies. The stiffness values (x,y,z) were set as [300, 300, 300] N/m, and the angular stiffness values are set as [500, 500, 500] Nm/rad. The damping ratio for the Cartesian impedance control for all degrees of freedom is set as 0.7. The impedance control parameters remain constant during task execution.

During data collection, the participant would be introduced to a task and observe and correct each error instance 8 times for a total of 40 trials for the peg-in-hole and the pouring task, and a total of 32 trials for the draw & erase task. Prior to starting the trials for each task, the correct trajectory was shown to the participants one time. To prevent participants from anticipating the error, the order in which the error types were presented was randomized. The participant would complete all trials for one task before moving on to the next task. The order of tasks was counterbalanced across all participants. Participants were given the option to either physically correct the robot's movement or hit the emergency stop button.

Experimental data collected during the trials consisted of the Cartesian position of the robot's end-effector and its joint torques as a function of time. The total interaction time for each trial was calculated by summing the time in the trial during which the robot's trajectory deviated from the programmed trajectory. The average interaction force was calculated by measuring the mean joint torque applied to the robot during the deviations from the programmed trajectory. A one-factor repeated measures ANOVA was used to analyze the effect of error type on the interaction time and the interaction force for each task.

C. Incorporating User Study Results into Algorithm Design

The results of Experiment 1 showed a significant delay between the start of a deviation from the normative trajectory and the user intervention (see Sec. IV-A)—inspiring the design of Experiment 2 to understand the impact of including the original robot behavior as a negative demonstration.

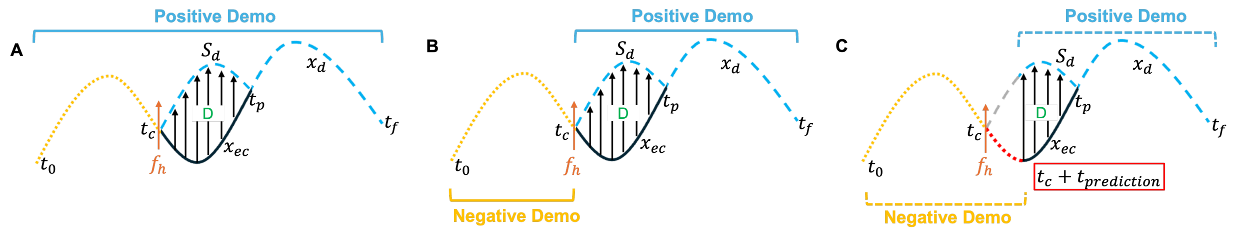

Fig. 3: (A) PositiveDeformedBehvior: The original LfD trajectory x_{ec} (black) is altered by human input force f_h according to Eq. 3 such that the robot follows the deformed trajectory x_d (blue dash) which is used as the positive demonstration; (B) PreviousNegBehavior: x_d is divided at $t = t_c$ into the negative demonstration trajectory before the correction(red dash) and positive demonstration occurring after the correction(blue dash). (C) PredictedNegBehavior: The negative demonstration is taken as the trajectory before the correction and the planned trajectory x_{ec} from t_c to $t_c + t_{prediction}$

The method used to incorporate corrected trajectories as demonstrations depends on how one interprets the implicit information in the interaction. While the deformed trajectory is an explicitly positive demonstration (Fig. 3), the implicit information could be contained in the robot's executed performance (Fig. 3B) or the robot's original planned behavior(Fig. 3C).

We explore three methods for incorporating corrective feedback into the learned task definition. When detecting user input force, f_h, at t_c, the deformed trajectory x_d is calculated based on Eq. 3, as shown in Fig. 3. The deformed trajectory is always included as a positive demonstration in the task definition as described in section III-C (Fig. 3A). In addition, we can incorporate the robot behavior preceding the correction as a negative demonstration, as shown in Fig. 3B. If one assumes the user makes some prediction about how the robot's behavior would have evolved without their intervention, the trajectory that was planned for immediately after the correction is also a negative demonstration (Fig. 3C).

Deformed trajectory used as positive demonstration: The deformed trajectory is computed with Eq. 3 and is used to update the desired trajectory x_d of the impedance controller. The deformed trajectory is also added to the task distribution learned during the offline training phase ϕ_k by assigning it the positive weight w_j in the summation given in Eq. 2. This method is referred to as ***PositiveDeformedBehavior*** in the following sections.

Deformed trajectory used as a positive demonstration with preceding behavior as a negative demonstration: In Fig. 3B, the robot trajectory is split at t_c. As with the method of incorporating the deformed trajectory as a positive demonstration, we assume that the deformed trajectory S_d^* is positive. Because the correction implies that some aspect of the robot's task execution must be fixed, the robot behavior prior to t_c is incorporated into the learned task distribution as a negative demonstration $w_j < 0$. The underlying assumption is that the user's correction is reactive to observed negative behavior. This method is referred to as ***PreviousNegBehavior*** in the following sections.

Deformed trajectory used as a *positive* demonstration and predicted negative demonstrations: User corrections may be a reaction to observed negative behavior, but they may also be a result of a user's anticipation of negative behavior. Therefore, we also incorporated the presumed positive trajectory deformation with the previous behavior and a portion of the originally planned trajectory for some time $t_{prediction}$ into the future as a negative demonstration. This method is referred to as ***PredictedNegBehavior*** in the following sections.

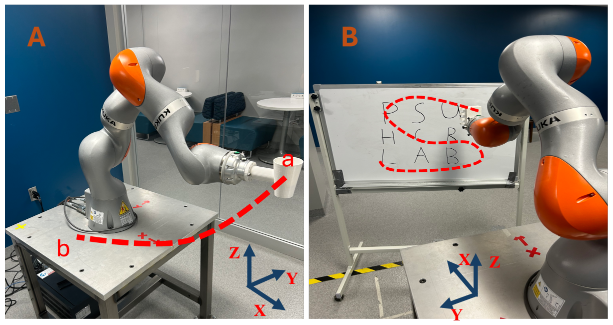

Fig. 4: **Experimental Tasks**. (A) Delivery task: Kuka is responsible for delivering the cup from point A to point B; (B) Cleaning task: Kuka is responsible for cleaning the writing on the whiteboard.

D. Experiment 2: Study on Performance of Explicit and Implicit Update Methods

We implement ergodic imitation with online corrections on a Kuka LBR iiwa operating in impedance control mode as shown in Fig. 1. In the impedance control mode, stiffness values for (x, y, z) are set as [1500,100,100] N/m. The angular stiffness values are set as [300, 300, 300] Nm/rad. The damping ratio for the Cartesian impedance control for all degrees of freedom is set as 0.7. The parameters of the impedance control mode remain constant during the training and execution phases. To communicate with Kuka and analyze the motion data, including Cartesian position and wrench, we used the `iiwa_stack` ROS package [49]. In the experiments, there are two phases: training and learning. After an offline training period in which up to 5 kinesthetic demonstrations were provided by the researchers, the robot attempts to execute the tasks based on the provided demonstrations. During the online learning period, physical corrections were applied to the 6th joint of the Kuka up to 4 times, depending on the ratio being tested. For the delivery task, these corrections altered the angle of the cup, while the corrections in the erasing task increased the coverage of the whiteboard. The experiments used the following hyperparameters: $R = 0.01 \cdot I_{3\times3}, q = 500, K = 10, T = 1.0, t_s = 1/60, f_{threshold} = 4.5N$, and $t_{prediction} = 1.5$, which were chosen empirically.

We evaluate the performance of the three methods for incorporating corrective action in two tasks—transporting a filled cup (Fig. 4A) and cleaning a whiteboard (Fig. 4B). We show how changing the ratio of offline demonstrations and corrective actions affects task performance by analyzing task-specific metrics that capture task success. We tested the following offline:collection ratios: 5:0, 1:4, 3:2, and 4:1. The average performance is computed from 20 task executions occurring after all demonstration data had been incorporated into the learned task definition.

1) Task Descriptions: We simplified the pouring task and the draw and erase task from experiment 1 to a cup delivery task and an erase-only task, so that we could assess the performance of one primary metric in each task. In the **delivery task**, the robot arm learns to deliver a cup to users as shown in Fig. 4A, which takes 8 seconds. In addition to reaching the target location, the robot must keep the cup upright during the task execution. Corrections are provided when the cup spills the material contained in it. Therefore, the main performance metric that we evaluate is the angle of the cup, where 0 radians is the upright position. The angle of the robot end effector (e.g., the angle of the cup) is used in the ergodic metric.

In the **cleaning task**, the robot arm is trained to clean the whiteboard shown in Fig. 4B, which takes 19 seconds. During executions, users provide corrections when areas of the board have not received sufficient coverage. We evaluate the performance of the learned trajectories by comparing the pixel values of an image of the whiteboard before and after cleaning. The percentage of the original image that retains a high pixel value is reported as the uncleaned area. The Cartesian coordinates of the end-effector, (x,y,z), are used in the ergodic metric.

IV. Results

A. Statistical Results of Error Types

User interaction time across the three tasks—Peg Task, Pour Task, and Draw Task is shown in Fig. 5. Three one-factor, repeated measures ANOVA were performed using R [50], to assess the impact of error types in each of the tasks. The main effect of error types was statistically significant in the peg task ($p = 0.0202, F = 2.9636$), but the error type was not a significant factor on the interaction time in the pour ($p = 0.6650, F = 0.5970$) and draw tasks ($p = 0.0649, F = 2.4452$). To further evaluate the effect of error type, a post-hoc pair-wise t-test was performed for the peg task. In the peg task, there was a statistically significant difference between the safety error 1 and the task failure error 2, and between the safety error 1 and the safety error 2. Overall, these results suggest that the task type and error type may influence differences in interaction time. Interestingly, there were no significant differences between the 'no error' error type and the other error types. In other words, participants spend a similar amount of time correcting trajectories with error as with trajectories without error.

Three one-factor, repeated measures ANOVA were also performed on the interaction forces used to correct the predefined errors in each of the three tasks. The main effect of error types was statistically significant in each task — Peg task: $p = 1.86 \times 10^{-106}$, $F = 347.8440$; Pour task: $p = 1.73 \times 10^{-80}$, $F = 185.0599$; and Draw task: $p = 3.82 \times 10^{-25}$, $F = 51.3442$.). To further evaluate the effect of error type, a pair-wise t-test was performed for each task. In the peg task, there was a significant difference in all pairwise comparisons of error types. In the pour task, there were significant differences between most error types except for the comparison between No Error and Task Failure Error. In the draw task, there were significant differences between each pair of error types except for the comparison between Task Failure Error 1 and Task Failure Error 3. Overall, we observe that the interaction force varies significantly across different error types, indicating that users exert different levels of corrective force depending on the nature of the error. It

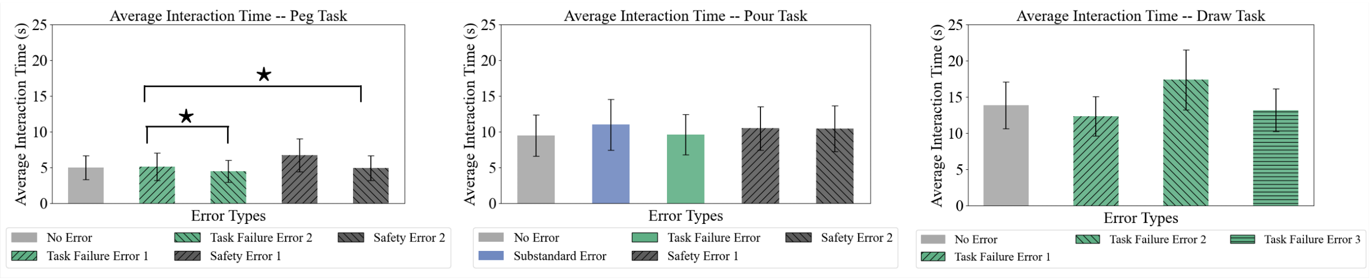

Fig. 5: **User Interaction Time on predefined errors:** Peg Task has the lowest average interaction time across all conditions. The safety-related errors in the peg and pour tasks exhibit higher interaction time, which suggests that safety errors require more prolonged engagement. The Draw Task demonstrates the longest interaction time among the three tasks. (*:$p < 0.05$;**:$p < 0.005$;***:$p < 0.0001$)

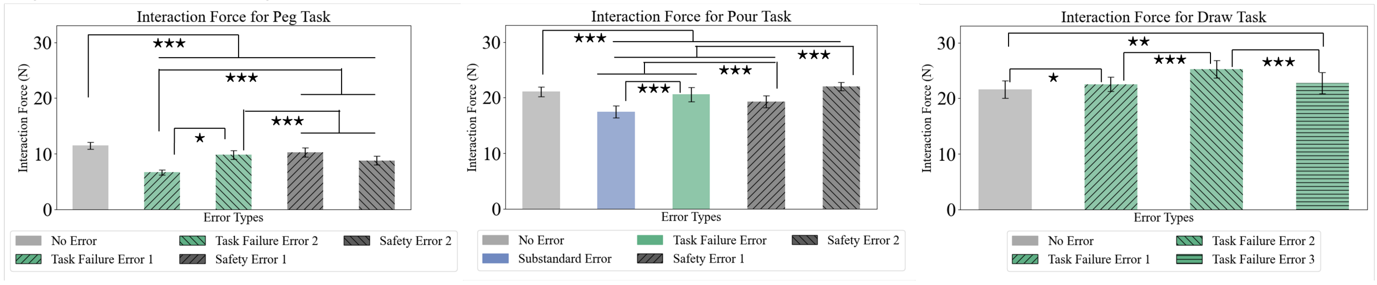

Fig. 6: **User Interaction Force on predefined errors:** The elevated force levels of the Pour Task (Safety Error 1) indicate that users prefer spending more effort to correct safety-related errors in this task. In contrast, the Peg Task has the lowest interaction forces since the task is relatively easy to correct and finish. (*:$p < 0.05$;**:$p < 0.005$;***:$p < 0.0001$)

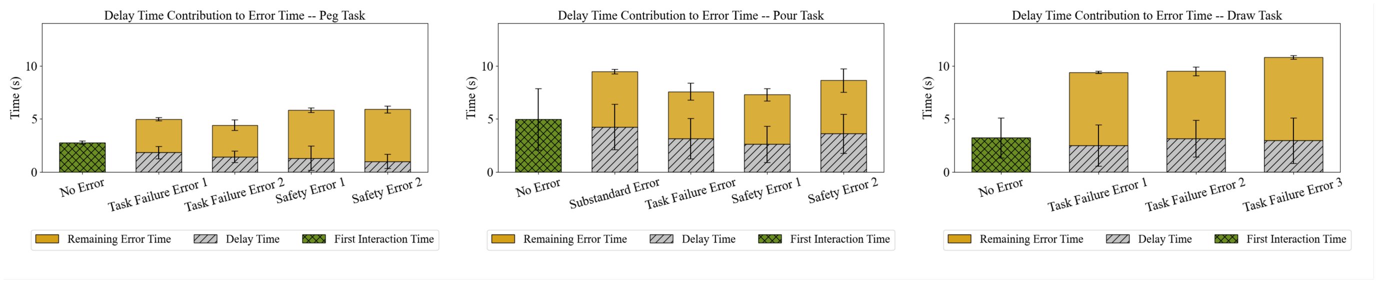

Fig. 7: **Combination of Delay and Total Error Time Comparison for Peg Task, Pour Task, and Draw Task**: We analyzed the temporal delay between the start of errors and the initiation of user interventions. Results revealed that safety-related errors tend to have a shorter delay before user intervention compared to task failures.

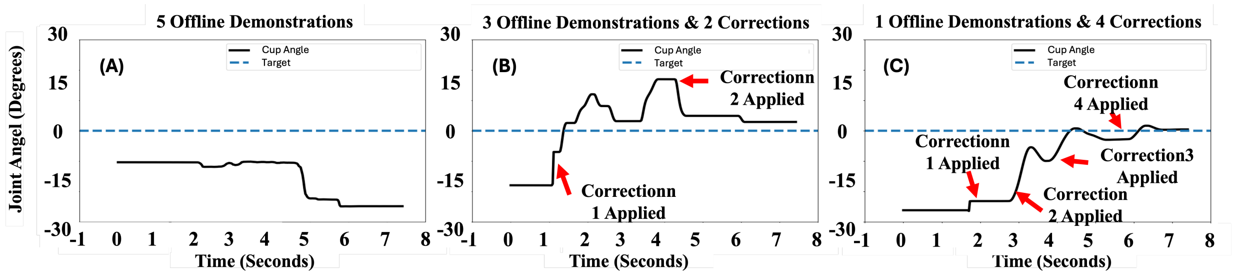

Fig. 8: In the case of transporting the cup, each correction improves the task definition such that the robot's task execution more accurately reflects the goal of keeping the cup upright.

reveals that user interaction force is an informative signal reflecting the relative need for trajectory modification. This magnitude is leveraged in the trajectory deformation step of the online update. However, as seen in Fig. 6, users may still provide strong corrections to 'no error' trajectories, sometimes even more than errors that would result in task failure.

We computed the delay between the onset of the trajectory error and the user interaction. Fig. 7 shows the results of the delay within the total error time for different error types of each task. We also present the time from the start of the 'no error' trajectory to the first interaction, which was about three seconds for each task. Typically, users responded within 2-4 seconds of error onset, indicating that a negative behavior occurs prior to user intervention. This motivated the use of the previous negative behavior as demonstrations of what not to do in Experiment 2.

B. Sample Response with Online Corrections

Fig. 8 shows the sample response of the cup angle (the seventh robot joint angle) during the delivery task. The blue dashed line represents the target angle, which is 0 degrees. The seventh joint angle of the robot is expected to remain close to this target angle to prevent spilling. When trained with five offline demonstrations, the angle stayed around 10 degrees and increased to over 20 degrees when the task was completed. With three offline demonstrations, the robot started at approximately 20 degrees. After two corrections, the joint angle gradually approached the target angle but remained above 10 degrees for a period. When trained with one offline demonstration, the robot began at an even larger angle. Following three corrections, the joint angle gradually approached the target angle and remained stable around it for some time.

Fig. 9 illustrates the sample response of the trajectories and task definitions during the cleaning task. Regions of the task distribution where more time is spent have a higher color intensity than regions where less time is spent. The robot was expected to clean as many areas as possible. When trained with five offline demonstrations, the task definition covered fewer areas. However, as more user corrections were incorporated, the task distribution became more diffuse, resulting in an increase in the cleaned area. This improvement is also evident from the robot's trajectories.

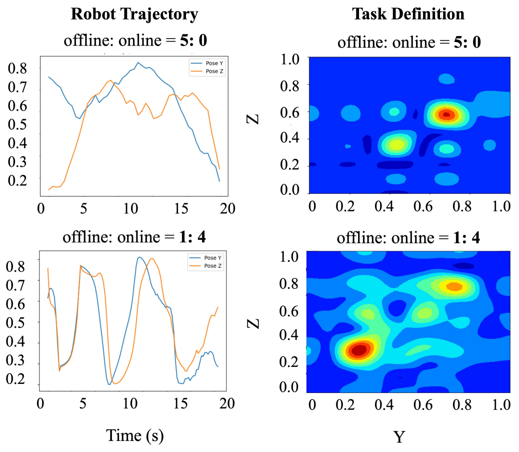

Fig. 9: This example illustrates how the learned task distribution evolves with different correction ratios. As corrections are applied, the task distribution becomes more diffuse, resulting in increased area coverage with online corrections versus offline demonstrations. The end effector trajectories in the y–z plane are shown on the left, and the corresponding learned task distributions are shown on the right.

C. Human Interaction Time is Decreased with Online Updates

Fig. 10 shows the results of the interaction time of each trial. In general, participants interacted less with the robot in the first 2 trials, resulting in lower interaction times, perhaps because they were still gaining familiarity with the robot by hands-off observation. In subsequent trials, interaction time increased somewhat. Overall, enabling online updates via corrections

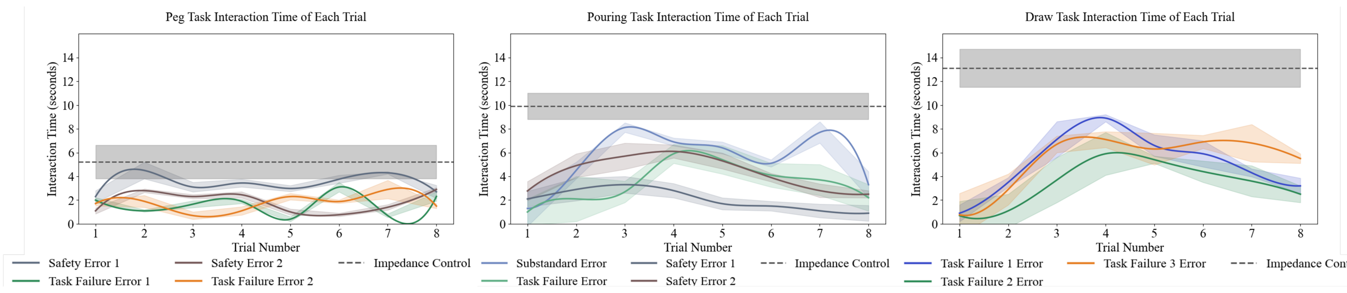

Fig. 10: **Interaction Time Comparison**: We explored the meaning of robots learning from physical human-robot interactions (pHRI) and compared it to learning from impedance. Results revealed that learning from pHRI significantly reduced participant interaction time across all experimental tasks. The user interaction time also decreased gradually with the increased number of trials.

reduced the interaction time from trial 3 to trial 8 in most tasks and error types, except the peg-in-hole tasks, where interaction times were relatively flat. All trials with online updates were lower than the average interaction time observed in the study of predefined errors without online updates.

D. Online Learning Frameworks Improve Performance of Cup Delivery

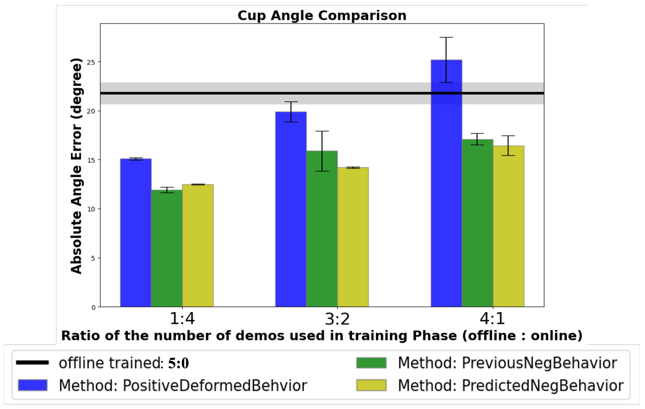

Fig. 11: Comparison of cup angle from 2 control strategies: offline LfD (black line) and continual learning (bars) with different ratios of trained demonstrations. The average performance is computed from 20 task executions completed after all offline and online demonstrations were incorporated into the learned task definition.

The angle error for ergodic imitation using only offline demonstrations was over 20 degrees after inputting all demonstrations. This is plotted as the horizontal black line in Fig. 11, where the grey shading represents the standard error for this baseline learning method. When the task was learned from a mix of offline demonstrations and online corrections instead of offline demonstrations alone, the angle error decreased significantly. The addition of the deformed trajectory as a positive demonstration (PositiveDeformedBehavior) with 4 corrections and one offline demonstration reduced the angle error by more than 30% compared to offline demonstrations alone. The performance of PositiveDeformedBehavior improves with an increased number of corrections. Specifically, the cup angle trained with four corrections and one offline demonstration decreased by approximately 40% compared to the angle trained with four offline demonstrations and only one correction. The number of corrections does not significantly impact PreviousNegBehavior and PredictedNegBehavior. Overall, methods that utilize information about the negative aspects of the original trajectory perform better than the method using only the deformed trajectory as a positive demonstration. By combining positive (explicit information) with negative (implicit information) aspects of behavior, performance improves by nearly 50% compared to the offline approach and 20% compared to using the deformed trajectory as a positive demonstration only, when four corrections are provided.

E. Cleaned area is increased with Online Learning Frameworks

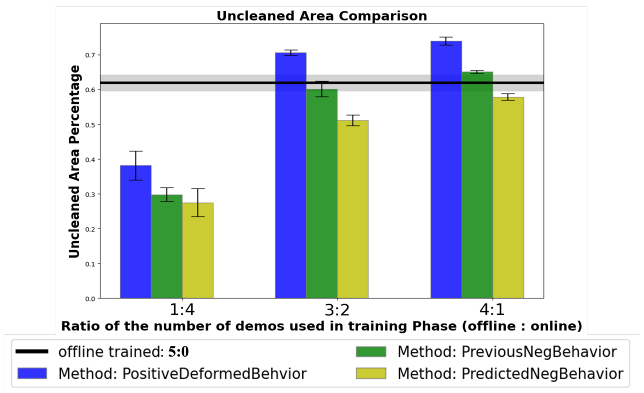

Fig. 12: Uncleaned area percentage is calculated from the original writing area divided by the uncleaned area. The highest percentage scale is 1. The average performance is computed from 20 task executions completed after all offline and online demonstrations were incorporated into the learned task definition.

In the cleaning task, a smaller percentage of uncleaned areas indicates better robot performance. Shown in Fig. 12, the performance of the offline method trained with five demonstrations is used as the baseline, with the grey shading representing the standard error for this baseline. Generally, the robot arm with all online update control strategies shows improved performance as more online corrections are provided during the learning phase. Methods incorporating negative aspects perform slightly better than those using only the deformed trajectory as a positive demonstration. Among the online updates via corrections methods, PredictedNegBehavior outperforms the others. Specifically, the uncleaned area percentage for PredictedNegBehavior, trained with four online corrections, decreased by approximately 50% compared to the offline baseline.

V. Discussion and Conclusions

The aim of the present study was to characterize human corrections by analyzing responses to a set of pre-defined errors with varying levels of impact on task performance and safety. Our primary metrics were the total interaction time and interaction force. Differences in error types were not good predictors of the interaction time, but there were clear differences in the average interaction force depending on the error type. The differences in interaction force could be at least partially explained by the fact that more severe errors would require stronger forces to overcome the impedance controller that was tracking the erroneous trajectory. Yet, if this were the only explanation, we would expect the no error condition to have the lowest forces and lowest interaction times. Surprisingly, this was not the case. Users spent similar amounts of time providing corrections to ‘no error’ trajectories as they did to trajectories that posed a risk of collision or trajectories that failed at the task. The order in which these trajectories were presented was randomized, so it is possible that participants simply began anticipating errors even if there were none. In other words, the users did not trust the robot to make no errors, so they intervened more often. This highlights the need for responsive and adaptive controls on the robot, not only to improve performance but also to reduce the burden on users to monitor and correct robot partners.

In addition to the differences in the total time and effort used to correct the robot, we evaluated the timing of these corrections relative to the initial deviation of the erroneous trajectories from the nominal trajectory. We observed a consistent delay between the onset of the error and the user correction. While the correction provides the explicit demonstration of the user’s preferred behavior, this delay suggests that there is implicit information about what not to do in the trajectory that was executed immediately before the correction. In the second experiment presented, multiple ways of interpreting human feedback are explored to incrementally update the task definition of an imitation learning framework. The deformed trajectories reflect the user intention and are incorporated as demonstrations to refine learned task distributions. User intention is interpreted into two components: “Negative”, representing undesired behavior, which corresponds to the robot’s planned motion before correction, and “Positive”, representing the intended behavior, which corresponds to the robot’s planned motion after correction with the deformed region. This method of incorporating corrections is implemented on a 7-DOF robot arm in two tasks—one requiring high accuracy and one requiring diffuse coverage. We demonstrate that incorporating these corrections can enhance robot learning by efficiently updating the task definitions.

Notably, the information contained in negative demonstrations helps robots achieve better performance with fewer user corrections. In the experiment comparing continual learning methods with offline LfD, the results show that continual learning frameworks can help robots update the original task definitions based on the information from pHRI, where PreviousNegBehavior and PredictedNegBehavior are more effective than PositiveDeformedBehavior alone. When performing a complicated task, the continual learning frameworks’ performance shows a greater dependency on the information contained in the original task definition and corrected task definition compared to the simple task. The deformed trajectory contains explicit information in the form of a positive demonstration. Implicit information is contained in the mistakes of the originally planned behavior and can be considered a negative demonstration or an example of what not to do. The impact of these corrections is limited in simple tasks like peg-in-hole, but provides larger performance gains in more complex tasks. This may be because it is difficult to capture all the relevant features in a small number of demonstrations of more complex tasks. In the user study evaluating three tasks with twelve errors in total, enabling online updates via corrections improved the robot’s performance to match human preferences with fewer interventions. Additionally, it decreased interaction time as the number of trials increased. The results from the interaction time experiments demonstrate that the robot’s ability to perform online updates using information encoded in pHRI can gradually reduce the user’s burden of correcting ongoing mistakes.

In the current work, we empirically demonstrate the benefit of incorporating implicit information from human corrections. However, future work could apply an information-theoretic approach to quantify the implicit information extracted from these interactions—potentially reducing the number of hyper-parameters that need to be selected by the programmer. One might expect that, with a less uniform participant pool than in the present study, the force threshold, deformation sensitivity, and response time might vary, especially in disparate settings like home or manufacturing. Another limitation of the present work is that while the robot can generalize over a variety of conditions of the same task, it cannot generalize across tasks. One could apply this method to other probabilistic task definitions, such as ProMPs, where one would have to use some heuristic to determine how to apply corrective feedback to an individual motion primitive in a library of learned behaviors. This would enable the proposed methods to be used in multi-step manipulation tasks or implemented into more complex robotics systems, e.g., humanoid or quadruped robots.

In this paper, we presented a user study of responses to robot errors and proposed a method for learning from pHRI to enhance the robot’s in-task online update ability, which has the potential to enable real-world robot designs to be more reliable and practical. We use both implicit and explicit information from user physical interactions. We interpreted the explicit information as the robot’s deformed trajectory and implicit information as the robot’s previous behavior—learning from “what not to do”. In the user study, user interaction time is reduced after interaction-based updates. Compared with the offline LfD approach, the performance is gradually improved as more corrections are implemented, especially when utilizing the information about the negative aspects of the original behavior.

References

[1] J. Mainprice and D. Berenson, “Human-robot collaborative manipulation planning using early prediction of human motion,” in *Int. Conf. Intelligent Robots and Systems*. IEEE, 2013, pp. 299–306.

[2] S. Ososky, D. Schuster, E. Phillips, and F. Jentsch, “Building appropriate trust in human-robot teams,” *AAAI Spring Symposium - Technical Report*, pp. 60–65, 01 2013.

[3] D. J. Brooks, M. Begum, and H. A. Yanco, “Analysis of reactions towards failures and recovery strategies for autonomous robots,” in *Int. Symp. Robot & Human Interactive Communication*. IEEE, 2016, pp. 487–492.

[4] A. Singh, L. Yang, K. Hartikainen, C. Finn, and S. Levine, “End-to-end robotic reinforcement learning without reward engineering,” in *Robotics: Science and Systems*, 2019.

[5] X. Wang, K. Lee, K. Hakhamaneshi, P. Abbeel, and M. Laskin, “Skill preferences: Learning to extract and execute robotic skills from human feedback,” in *Conf. Robot Learning*, 2022, pp. 1259–1268.

[6] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei, “Reward learning from human preferences and demonstrations in atari,” *Adv. Neural Information Processing Systems*, vol. 31, 2018.

[7] A. Jain, B. Wojcik, T. Joachims, and A. Saxena, “Learning trajectory preferences for manipulators via iterative improvement,” *Adv. Neural Information Processing Systems*, vol. 26, 2013.

[8] B. Burchfiel, C. Tomasi, and R. Parr, “Distance minimization for reward learning from scored trajectories,” in *AAAI Conf. Artificial Intelligence*, vol. 30, no. 1, 2016.

[9] N. Ye, A. Somani, D. Hsu, and W. S. Lee, “Despot: Online pomdp planning with regularization,” *J. Artificial Intelligence Research*, vol. 58, pp. 231–266, 2017.

[10] 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 *Robotics: Science and Systems*, 2020.

[11] D. P. Losey and M. K. O’Malley, “Trajectory deformations from physical human–robot interaction,” *Trans. Robotics*, vol. 34, no. 1, pp. 126–138, 2017.

[12] J. Y. Zhang and A. D. Dragan, “Learning from extrapolated corrections,” in *Int. Conf. Robotics and Automation*. IEEE, 2019, pp. 7034–7040.

[13] M. Li, A. Canberk, D. P. Losey, and D. Sadigh, “Learning human objectives from sequences of physical corrections,” in *Int. Conf. Robotics and Automation*. IEEE, 2021, pp. 2877–2883.

[14] M. Karlsson, A. Robertsson, and R. Johansson, “Autonomous interpretation of demonstrations for modification of dynamical movement primitives,” in *Int. Conf. Robotics and Automation*. IEEE, 2017, pp. 316–321.

[15] M. Khoramshahi, A. Laurens, T. Triquet, and A. Billard, “From human physical interaction to online motion adaptation using parameterized dynamical systems,” in *Int. Conf. Intelligent Robots and Systems*. IEEE, 2018, pp. 1361–1366.

[16] Y. Jiang, C. Wang, R. Zhang, J. Wu, and L. Fei-Fei, “TRANSIC: Sim-to-real policy transfer by learning from online correction,” in *Conf. Robot Learning*, 2024.

[17] R. Hoque, A. Mandlekar, C. Garrett, K. Goldberg, and D. Fox, “Intervengen: Interventional data generation for robust and data-efficient robot imitation learning,” in *Int. Conf. Intelligent Robots and Systems*. IEEE, 2024, pp. 2840–2846.

[18] S. Gomez-Gonzalez, G. Neumann, B. Schölkopf, and J. Peters, “Adaptation and robust learning of probabilistic movement primitives,” *Trans. Robotics*, vol. 36, no. 2, pp. 366–379, 2020.

[19] A. Kalinowska, A. Prabhakar, K. Fitzsimons, and T. Murphey, “Ergodic imitation: Learning from what to do and what not to do,” in *Int. Conf. Robotics and Automation*. IEEE, 2021, pp. 3648–3654.

[20] G. Hayes and Y. Demiris, “A robot controller using learning by imitation,” *Citeseer*, vol. 676, 06 1995.

[21] D. C. Bentivegna and C. G. Atkeson, “Learning from observation using primitives,” in *Int. Conf. Robotics and Automation*, vol. 2. IEEE, 2001, pp. 1988–1993.

[22] P. Kormushev, S. Calinon, and D. G. Caldwell, “Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input,” *Advanced Robotics*, vol. 25, no. 5, pp. 581–603, 2011.

[23] J. DelPreto, J. I. Lipton, L. Sanneman, A. J. Fay, C. Fourie, C. Choi, and D. Rus, “Helping robots learn: a human-robot master-apprentice model using demonstrations via virtual reality teleoperation,” in *Int. Conf. Robotics and Automation*. IEEE, 2020, pp. 10 226–10 233.

[24] K. Liang, Y. Wang, L. Pan, Y. Tang, J. Li, Y. Lin, and M. Pan, “A robot learning from demonstration method based on neural network and teleoperation,” *Arabian J. Science and Engineering*, vol. 49, no. 2, pp. 1659–1672, 2024.

[25] T. Park and S. Levine, “Inverse optimal control for humanoid locomotion,” in *Robotics: Science and Systems*, 2013, pp. 4887–4892.

[26] A. Doerr, N. D. Ratliff, J. Bohg, M. Toussaint, and S. Schaal, “Direct loss minimization inverse optimal control.” in *Robotics: Science and Systems*, 2015.

[27] A. Boularias, J. Kober, and J. Peters, “Relative entropy inverse reinforcement learning,” in *Int. Conf. Artificial Intelligence & Statistics*, 2011, pp. 182–189.

[28] T. Brys, A. Harutyunyan, H. B. Suay, S. Chernova, M. E. Taylor, and A. Nowé, “Reinforcement learning from demonstration through shaping,” in *IJCAI Conf. Artificial Intelligence*, 2015.

[29] U. Trivedi, R. Alqasemi, and R. Dubey, “Robot learning from human demonstration of activities of daily living (adl) tasks,” in *Int. Mechanical Engineering Congress and Exposition*, vol. 6. ASME, 2021, p. V006T06A019.

[30] P. Abbeel and A. Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in *Int. Conf. Machine learning*, 2004, p. 1.

[31] S. Levine, Z. Popovic, and V. Koltun, “Nonlinear inverse reinforcement learning with gaussian processes,” *Adv. Neural Information Processing Systems*, vol. 24, 2011.

[32] J. Ho and S. Ermon, “Generative adversarial imitation learning,” in *Adv. Neural Information Processing Systems*, vol. 29, 2016.

[33] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, “Probabilistic movement primitives,” *Adv. Neural Information Processing Systems*, vol. 26, 2013.

[34] T. Kulak, J. Silvério, and S. Calinon, “Fourier movement primitives: an approach for learning rhythmic robot skills from demonstrations.” in *Robotics: Science and Systems*, 2020.

[35] S. Calinon, “A tutorial on task-parameterized movement learning and retrieval,” *Intelligent Service Robotics*, vol. 9, pp. 1–29, 2016.

[36] M. Schneider and W. Ertel, “Robot learning by demonstration with local gaussian process regression,” in *Int. Conf. Intelligent Robots and Systems*. IEEE, 2010, pp. 255–260.

[37] N. Jaquier, D. Ginsbourger, and S. Calinon, “Learning from demonstration with model-based gaussian process,” in *Conf. Robot Learning*, 2020, pp. 247–257.

[38] X. Wang and D. Klabjan, “Competitive multi-agent inverse reinforcement learning with sub-optimal demonstrations,” in *Int. Conf. Machine Learning*, 2018, pp. 5143–5151.

[39] A. Coates, P. Abbeel, and A. Y. Ng, “Learning for control from multiple demonstrations,” in *Int. Conf. Machine learning*, 2008, pp. 144–151.

[40] D. H. Grollman and A. Billard, “Donut as i do: Learning from failed demonstrations,” in *Int. Conf. Robotics and Automation*. IEEE, 2011, pp. 3804–3809.

[41] A. Colomé and C. Torras, “Dual reps: A generalization of relative entropy policy search exploiting bad experiences,” *Trans. Robotics*, vol. 33, no. 4, pp. 978–985, 2017.

[42] K. Shiarlis, J. Messias, and S. Whiteson, “Inverse reinforcement learning from failure,” in *Int. Conf. Autonomous Agents & Multiagent Systems*, 2016, p. 1060–1068.

[43] M. Zhao and M. Shimosaka, “Inverse reinforcement learning with failed demonstrations towards stable driving behavior modeling,” in *Intelligent Vehicles Symposium*. IEEE, 2024, pp. 2537–2544.

[44] I. Abraham, A. Prabhakar, and T. D. Murphey, “An ergodic measure for active learning from equilibrium,” *Trans. Automation Science and Engineering*, vol. 18, no. 3, pp. 917–931, 2021.

[45] L. M. Miller, Y. Silverman, M. A. MacIver, and T. D. Murphey, “Ergodic exploration of distributed information,” *Trans. Robotics*, vol. 32, no. 1, pp. 36–52, 2015.

[46] G. Mathew and I. Mezić, “Metrics for ergodicity and design of ergodic dynamics for multi-agent systems,” *Physica D: Nonlinear Phenomena*, vol. 240, no. 4-5, pp. 432–442, 2011.

[47] A. D. Dragan, K. Muelling, J. A. Bagnell, and S. S. Srinivasa, “Movement primitives via optimization,” in *Int. Conf. Robotics and Automation*. IEEE, 2015, pp. 2339–2346.

[48] A. Mavrommati, E. Tzorakoleftherakis, I. Abraham, and T. D. Murphey, “Real-time area coverage and target localization using receding-horizon ergodic exploration,” *Trans. Robotics*, vol. 34, no. 1, pp. 62–80, 2018.

[49] C. Hennersperger, B. Fuerst, S. Virga, O. Zettinig, B. Frisch, T. Neff, and N. Navab, “Towards mri-based autonomous robotic us acquisitions: a first feasibility study,” *Trans. Medical Imaging*, vol. 36, no. 2, pp. 538–548, 2016.

[50] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2023. [Online]. Available: https://www.R-project.org/