2016 IEEE-RAS 16th International Conference on
Humanoid Robots (Humanoids)
Cancun, Mexico, Nov 15-17, 2016

Co-active Learning to Adapt Humanoid Movement for Manipulation

Ren Mao[1], John S. Baras[1], Yezhou Yang[2], and Cornelia Fermüller[3]

***Abstract*— In this paper we address the problem of interactive robot movement adaptation under various environmental constraints. A common approach is to adopt motion primitives to generate target motions from demonstrations. However, their generalization capability is weak for novel environments. Additionally, traditional motion generation methods do not consider versatile constraints from different users, tasks, and environments. In this work, we propose a co-active learning framework for learning to adapt the movement of robot end-effectors for manipulation tasks. It is designed to adapt the original imitation trajectories, which are learned from demonstrations, to novel situations with different constraints. The framework also considers user feedback towards the adapted trajectories, and it learns to adapt movement through human-in-the-loop interactions. Experiments on a humanoid platform validate the effectiveness of our approach.**

I. Introduction

Trajectory learning from human demonstrations has been studied in the field of Robotics for decades because of its wide range of applications in both industrial and domestic environments. A popular approach uses so-called Motion Primitives (MPs) to parameterize the observed human motion and reproduce similar motions with different initial and target states. However, it is widely known that general MPs methods, such as Dynamic Movement Primitives (DMPs) [1], exhibit limited capability for generalizing to new environments involving other constraints. Moreover, the learning used in standard MPs does not allow incorporating user preferences, such as preferred movements under geometric constraints. However, humanoid applications in real world environments would greatly benefit from a practical robot movement learning framework that take user preferences and environmental constraints into consideration.

Let's start with a common example. A human user teaches a humanoid how to transfer a bottle in different situations. Using an off-the-shelf approach, the robot can learn the motion by acquiring MPs from demonstrations and applying them to generate new trajectories. However, solely following the generated trajectories may fail in a slightly altered environment, such as when a bowl is blocking the path as illustrated in Fig. 2(a). Here we assume that these constraints are presented to the robot only during the testing phase, and not during the training phase. In this work, we propose an optimization based framework for adapting trained movements to novel environments. The first goal of our system is

[1]R. Mao and J. Baras are with the Department of Electrical and Computer Engineering and the ISR, [3]C. Fermüller is with the Department of Computer Science and the UMIACS, University of Maryland, College Park, Maryland, USA. [2]Y. Yang is with School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, USA. {neroam, baras} at umd.edu, yz.yang at asu.edu and fer at umiacs.umd.edu.

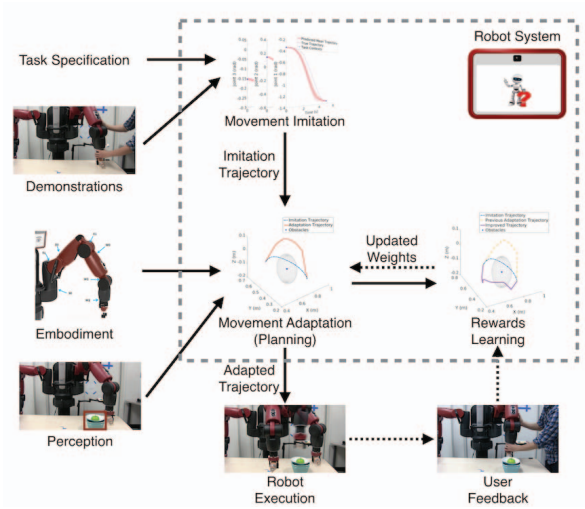


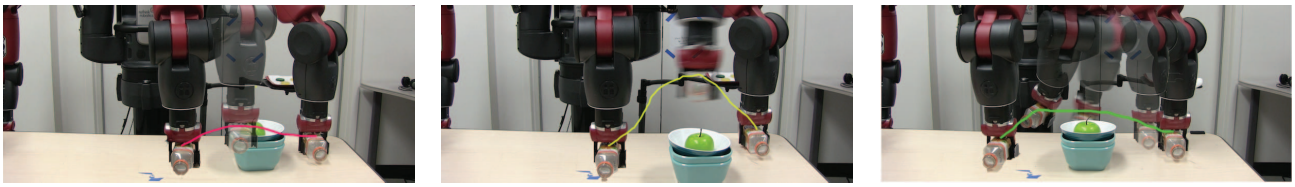
Fig. 1. System for learning movement adaptation for manipulation tasks. Dashed lines indicate feedback.

to generate adapted trajectories, as shown in Fig. 2(b), that can: 1) follow demonstrated trajectories for the purpose of preserving movement patterns, and 2) fulfill novel constraints perceived from the environment during the testing phase.

Moreover, new environmental constraints perceived during the testing phase could be more complex than simply encountering an obstacle. Building on the last example, this time, let's consider the situation where the target bottle is leaking. Ideally an intelligent robot that understands the situation should avoid moving the bottle over the bowl, but follow the movement path *around* it. We could simply adjust the objective function in movement adaptation. But what if in another scenario the robot is asked to transfer a knife while avoiding obstacles *above* them to prevent potential scratches? Constraints of this nature are not only associated with the context of the task, i.e, leaking bottle or knife as the manipulated object, but also with the user's preference, i.e, avoiding the bowl in a certain manner. To account for these preferences, a human-in-the-loop on-line adaptation system is necessary. In the optimization framework for generating trajectories presented in this paper, we firstly treat the reward weights as adjustable parameters that adapt the quality of the trajectory. Then based on user feedback, the framework learns the preferred behavior, that fulfills constraints, by updating the reward weights. Therefore, the learned behavior can be generalized to different situations with similar constraints. As illustrated in Fig. 2(c), after a few iterations of on-line learning, the robot is able to generate a trajectory adapted in accordance with the learned preferences.

This paper proposes an approach for interactive learning

978-1-5090-4718-5/16/$31.00 ©2016 IEEE



(a) (b) (c)

Fig. 2. Baxter Transferring Leaking Bottle: (a) Movement imitation, failed to avoid the bowl; (b) Movement adaptation with initial weights, successfully avoided the bowl with a path above it but spilled water into the bowl; (c) Movement adaptation with weights learned for user preferences, successfully avoided the bowl with a path around it and avoided spilling water in the bowl.

of movement adaptation for manipulation tasks. Fig. 1 illustrates the proposed system. The main contributions of this work are: 1) A system to generalize robots' movements learned from demonstrations to fulfill constraints perceived in a new environment. It is able to adapt trajectories according to user preferences; 2) An approach for robot learning to adapt trajectories by updating reward weights based on users' feedback. The user thus can co-actively train the robot in-the-loop by demonstrating desired trajectories; 3) An implementation of the optimization schema for the skill of "transferring objects," considering obstacles and different geometric user preferences for the movements. We validate the implementation on a humanoid platform (Baxter), and the experimental results support our claims.

II. Related Work

Various approaches have been proposed in robotics for learning manipulation movements. A well known approach is imitation learning [2], which focuses on mimicking human demonstrations, and this approach works well when learning from demonstration (LfD) techniques [3], [4] are applicable. However, it only allows to reproduce learned movements in similar environments.

Approaches [5] for encoding the trajectory as motion primitives have been proposed for various forms of generalization and modulation, such as Gaussian mixture regression and Gaussian mixture models [6], [7]. In [8], a mixture model was used to estimate the entire movement skill from several sample trajectories. Another class of approaches employs Hidden Markov models [9]. One popular representation to encode motion from demonstrated trajectories, originally introduced in [1], is Dynamic Movement Primitives (DMPs). It consists of differential equations with a non-linear learnable component that allows modeling of almost arbitrarily complex motion. Recently, Probabilistic Movement Primitives (ProMPs) [10] was proposed as an alternative representation. It learns a trajectory distribution from multiple demonstrations and modulates the movement by conditioning on desired target states. Incorporating the variance of demonstrations, the ProMPs approach handles noise from different demonstrations and provides increased flexibility for reproducing movement. However, all these approaches hardly deal with novel environments such as involving different obstacles. In our work, we first train the robot using ProMPs, and then generalize these trained motion primitives to newly introduced environmental constraints.

In order to enable MPs to adapt to novel environments with obstacles [11], [12], Kober et al. [4] proposed an augmented version of DMPs which incorporates perceptual coupling to an external variable. Ghalamzan et al. [13] proposed a three-tiered approach that can generalize noisy task demonstrations to new situations with obstacles. They generated the nominal path with DMPs and then adapted the trajectory to avoid obstacles by formulating an optimal control problem regarding the reward function learned from demonstrations via inverse optimal control. This approach allows users to teach a robot the desired response to different objects but requires offline training in the environment containing the obstacles for the reward function. However, in practice, the human users often have different preferences for various environments and tasks, while it is extremely challenging to provide the optimal training trajectories in every situation. To account for this, in our approach, the human users can interactively provide sub-optimal suggestions on how to improve the trajectory and the robot learns the preference for different constraints, and incorporates it to generate more applicable trajectories.

User preferences for a robot's trajectories have been studied in the field of human robot interaction (HRI). Sisbot et al. [14] proposed to model user specified preferences as constraints such as the distance of the robot from the user. Then a path planner fulfilling those user preferences was provided. Ashesh Jain et al. [15] proposed a co-active learning method to learn user preferences over generated trajectories for manipulation tasks by iteratively taking user sub-optimal feedback, and the optimal trajectory was selected based on the learned reward function. In our work, we adopt the co-active learning paradigm and further propose a reward formulation to model user preferences over constraints for movement generation. Then we integrate it with movement adaptation through optimization based planning.

III. Co-active Learning for Movement Generalization

For the problem of robot learning from demonstrations [3], a common practice is to offline learn the skills by encoding the trajectories with movement patterns such as DMPs [16]. During the testing phase, they can then be used to generalize the movement to novel situations with slight alterations. However, this generalization capability does not apply to novel environments with different obstacles or to a new task contexts with a variety of manipulated objects. In this paper, we propose a complementary framework for generalizing off-line learned movement skills to novel situations, and in

addition we incorporate on-line learning preferences of how to generalize from human's feedback co-actively.

While facing a novel situation, the robot is given a manipulation task context $\boldsymbol{x}_c$ that describes the initial and target states of the robot, and the locations of relevant objects in the environment. It could compute an imitation movement trajectory $\boldsymbol{y}_D$ by generalizing offline learned skills and execute if the new environment does not have obstacles and there are no other constraints inherited from the task.

To further generalize learned movement skills to more challenging situations, the robot has to generate an adapted trajectory $\boldsymbol{y}$ based on the task contexts $\boldsymbol{x}_c$ and the computed imitation trajectory $\boldsymbol{y}_D$. Here we use a reward function $f^*(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D)$ to reflect how much reward the adapted trajectory $\boldsymbol{y}$ can achieve for different contexts. This way, we can adapt the movement by solving an optimal control problem by maximizing the reward function f^*. The reward function consists of a Imitation Reward f_D describing the tendency to follow the imitation trajectory $\boldsymbol{y}_D$, a Control Reward f_C describing the smoothness of executing the adapted trajectory $\boldsymbol{y}$ and a Response Reward f_E describing the expected response given the environment. To learn the reward function which controls how the robot adapts trajectories under new contexts, we apply a co-active learning technique [15] in which the user only corrects the robot by providing an improved trajectory $\bar{\boldsymbol{y}}$ and then the robot updates the parameter $\boldsymbol{w}$ of $f(\cdot; \boldsymbol{w})$ based on the user's feedback. It is worth noting that this feedback only indicates $f^*(\bar{\boldsymbol{y}}, \boldsymbol{x}_c, \boldsymbol{y}_D) > f^*(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D)$, and $\bar{\boldsymbol{y}}$ may be non-optimal trajectories. With iterations of improvement, the robot could learn a function that approximates the oracle $f^*(\cdot)$ tightly.

IV. Our System

Overall, after the robot has offline learned the movement skill from demonstrations, when facing a different task context $\boldsymbol{x}_c$ in a novel environment, the testing phase includes three stages: 1) Movement Imitation, which computes an imitation trajectory $\boldsymbol{y}_D$ by generalizing demonstrated movement to new initial and target states; 2) Movement Adaptation, which generates an adapted trajectory $\boldsymbol{y}$ under new task and environment contexts by maximizing the given reward function; 3) Rewards Learning, which updates the parameters of estimated reward function according to the user's feedback through co-active learning. We formulate each stage in our framework presented in Fig. 1 as following.

A. Movement Imitation

At the beginning, our system offline learns movement skills in an environment without obstacle or other constraints. In this work, we adopt the Probabilistic Movement Primitives (ProMPs) [10] for imitation learning. It obtains a distribution over trajectories from multiple demonstrations, which captures the variations, and can be easily generalized to new initial and target states while imitating the movement.

To be specific, we consider that a robot's end-effector has d degrees of freedom (DOF) along with its arm, with its state denoted as $\boldsymbol{y}(t) = [y_1(t), \ldots, y_d(t)]^T$. The trajectory of the robot's end effector is represented as a sequence $\boldsymbol{\mathcal{T}} = \{\boldsymbol{y}(t)\}_{t=0,\ldots,T}$. We model each dimension i of $\boldsymbol{y}(t)$ using linear regression with n Gaussian time-dependent basis functions ψ and a n-dimensional weight vectors w_i as $y_i(t) = \psi(t)^T w_i + \epsilon_y$ where $\epsilon_y \sim \mathcal{N}(0, \sigma_y^2)$ denotes zero-mean i.i.d. Gaussian noise. With the underlying weight vectors $\boldsymbol{w} = [w_1^T, \ldots, w_d^T]^T$, the probability of observing a trajectory $\boldsymbol{\mathcal{T}}$ can be given by

$$p(\boldsymbol{\mathcal{T}}|\boldsymbol{w}) = \prod_t p(\boldsymbol{y}(t)|\boldsymbol{w}) = \prod_t \mathcal{N}(\boldsymbol{y}(t)|\boldsymbol{\Psi}(t)^T \boldsymbol{w}, \boldsymbol{\Sigma_y}), \quad (1)$$

where $\boldsymbol{\Psi}(t) = \text{diag}(\overbrace{\psi(t), \ldots, \psi(t)}^{d})$ and $\boldsymbol{\Sigma_y} = \sigma_y^2 \boldsymbol{I}_{d\times d}$.

1) Learning from Demonstrations: For each demonstration, the trajectory can be easily represented by a weight vector $\boldsymbol{w}$. To capture trajectory variations from multiple demonstrations, a Gaussian distribution $p(\boldsymbol{w}; \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu_w}, \boldsymbol{\Sigma_w})$ over the weights w is estimated. Therefore, the distribution of the trajectory $p(\boldsymbol{\mathcal{T}}|\boldsymbol{w})$ can be represented as

$$p(\boldsymbol{\mathcal{T}}; \boldsymbol{\theta}) = \int p(\boldsymbol{\mathcal{T}}|\boldsymbol{w}) p(\boldsymbol{w}; \boldsymbol{\theta}) d\boldsymbol{w} \quad (2)$$

$$= \prod_t \mathcal{N}(\boldsymbol{y}(t)|\boldsymbol{\Psi}(t)^T \boldsymbol{\mu_w}, \boldsymbol{\Psi}(t)^T \boldsymbol{\Sigma_w} \boldsymbol{\Psi}(t)^T + \boldsymbol{\Sigma_y}) \quad (3)$$

We can then estimate the parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu_w}, \boldsymbol{\Sigma_w}\}$ by using maximum likelihood estimation as suggested in [10].

2) Trajectory Generation: In novel situations, the trajectory could be modulated by conditioning with different observed states. By adding an observation vector of $\boldsymbol{Y^*} = [\boldsymbol{y}_0^{*T}, \boldsymbol{y}_T^{*T}]^T$ indicating the desired initial state $\boldsymbol{y}_0^*$ and target state $\boldsymbol{y}_T^*$ with accuracy $\boldsymbol{\Sigma}_{\boldsymbol{y}}^*$, we apply Bayes theorem and represent conditional distribution for $\boldsymbol{w}$ as

$$\begin{aligned} p(\boldsymbol{w}|\boldsymbol{Y^*}) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}'_{\boldsymbol{w}}, \boldsymbol{\Sigma}'_{\boldsymbol{w}}) \propto \mathcal{N}\left(\boldsymbol{Y}^*|\boldsymbol{\Psi}^{*T}\boldsymbol{w}, \boldsymbol{\Sigma}_{\boldsymbol{Y}}^*\right) p(\boldsymbol{w}) \\ \boldsymbol{\mu}'_{\boldsymbol{w}} &= \boldsymbol{\mu_w} + \boldsymbol{\Sigma_w}\boldsymbol{\Psi}^* \left(\boldsymbol{\Sigma}_{\boldsymbol{Y}}^* + \boldsymbol{\Psi}^{*T}\boldsymbol{\Sigma_w}\boldsymbol{\Psi}^*\right)^{-1} \left(\boldsymbol{Y}^* - \boldsymbol{\Psi}^{*T}\boldsymbol{\mu_w}\right) \\ \boldsymbol{\Sigma}'_{\boldsymbol{w}} &= \boldsymbol{\Sigma_w} - \boldsymbol{\Sigma_w}\boldsymbol{\Psi}^* \left(\boldsymbol{\Sigma}_{\boldsymbol{Y}}^* + \boldsymbol{\Psi}^{*T}\boldsymbol{\Sigma_w}\boldsymbol{\Psi}^*\right)^{-1} \boldsymbol{\Psi}^{*T}\boldsymbol{\Sigma_w} \end{aligned} \quad (4)$$

where $\boldsymbol{\Psi}^* = [\boldsymbol{\Psi}(0), \boldsymbol{\Psi}(T)]$ and $\boldsymbol{\Sigma}_{\boldsymbol{Y}}^* = \text{diag}(\boldsymbol{\Sigma}_{\boldsymbol{y}}^*, \boldsymbol{\Sigma}_{\boldsymbol{y}}^*)$ are augmented for observation vector $\boldsymbol{Y}^*$. With a conditional distribution of $\boldsymbol{w}$, we can generate conditional trajectory distribution and easily evaluate the mean $\boldsymbol{y}_D$ and the variance $\boldsymbol{\Sigma}_D$ of the trajectory $\boldsymbol{\mathcal{T}}$ for any time point t according to Eq.(1) and Eq.(2). Therefore, the mean $\boldsymbol{y}_D(t)$ can be used as the imitation trajectory in movement adaptation and the variance $\boldsymbol{\Sigma}_D(t)$ can be used to indicate which parts of the trajectory are more flexible to adapt. A larger variance reflects higher variations in demonstrations. It means more flexibility for modifying the corresponding part of the trajectory.

It is worth mentioning that, although we adopt ProMPs for movement imitation in this work, the proposed Movement Adaptation framework can be integrated into any other movement imitation learning technique.

B. Movement Adaptation

As mentioned before, if the environment of a new situation is exactly the same as the one during demonstration when ProMPs are learned, e.g, no obstacle, safety constraints or

other new considerations, the robot can perform the movement optimally by directly following the imitation trajectory $\boldsymbol{y}_D \in \mathbb{R}^d$ generated by learned ProMPs in discrete time.

In this work, we want to have a system that can adapt to an environment with novel constraints. Thus, we model the movement adaptation as an optimal control problem with fixed time horizon T in discrete time. The output of the adaptation system is a new trajectory $\boldsymbol{y} \in \mathbb{R}^d$ in discrete time. The input consists of the task context $\boldsymbol{x}_c$ which is obtained from the perception module, the imitation trajectory $\boldsymbol{y}_D$ which is generated from learned ProMPs, and the reward function $f(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D)$ which represents the reward of the adapted trajectory $\boldsymbol{y}$ corresponding to the new situation.

1) Optimization with Constraints: Let's consider that the perception module detects N_{obj} objects in the environment, which may be obstacles. Each object is abstracted as a sphere in the space represented by its center location and semi-diameter $\{\boldsymbol{O}_k, d_k\}, k = 1, \ldots, N_{obj}$. We assume that the reward function can be modeled as accumulated sum of rewards from each state $\boldsymbol{y}(t)$ at time step t:

$$f(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D) = \sum_{t=0}^{T} f_t(\boldsymbol{y}(t), \boldsymbol{x}_c, \boldsymbol{y}_D). \tag{5}$$

Because we are only modulating the trajectory, the adaptation system can be modeled as linear dynamics with the control signal $\boldsymbol{a} \in \mathbb{R}^m$, as it does not involve real physical dynamics. Considering the embodiment of robotic end-effectors, we can compute the end-effector's position in spatial space $\boldsymbol{E}(\boldsymbol{y})$ following the kinematics modeling [17]. Then, considering obstacles avoidance, the target optimal policy $\pi^* = \{\boldsymbol{a}(t)^*\}_{t=0,\ldots,T-1}$ can be defined from Eq. (6) with constraints.

$$\pi^* = \arg\max_{\pi} \textstyle\sum_{t=0}^{T} f_t(\boldsymbol{y}(t), \boldsymbol{x}_c, \boldsymbol{y}_D) \tag{6}$$

$$\begin{aligned}
\text{subj. to} \quad & \forall t = 0, \cdots, T-1 && (7)\\
& \boldsymbol{z}(t+1) = \boldsymbol{A}\boldsymbol{z}(t) + \boldsymbol{B}\boldsymbol{a}(t) && (8)\\
& \boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{z}(t) && (9)\\
& \boldsymbol{U} \geq \boldsymbol{y}(t) \geq \boldsymbol{L} && (10)\\
& \|\boldsymbol{E}(\boldsymbol{y}(t)) - \boldsymbol{O}_k\|^2 \geq d_k^2, \quad \forall k = 1, \cdots, N_{obj} && (11)\\
& \boldsymbol{y}(T) = \boldsymbol{y}_D(T), && (12)
\end{aligned}$$

where $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ are system matrices, Eq.(12) constrains the final position of the adapted trajectory, Eq.(10) constrains it within feasible limits, and Eq.(11) ensures it can avoid obstacles safely by keeping a minimum distance d_k between the robot's end-effector and any object.

2) Model Predictive Control: In order to find an optimal solution of such a system with continuous state and action spaces, we adopt Model Predictive Control which computes the optimal actions in a finite prediction horizon. Therefore, by considering a prediction time horizon T_p, the optimal action $\boldsymbol{a}(i)^*$, at time step $i = 0, \ldots, T-1$, can be solved by (13). At each step i, the optimal actions $\{\boldsymbol{a}(i)^*, \cdots, \boldsymbol{a}(i+T_p-1)^*\}$ for T_p decision steps in the future are computed but only the action for the current step $\boldsymbol{a}(i)^*$ is performed. Therefore, it can deal with changing environments as these changes could be considered in the next decision steps.

$$\begin{aligned}
& \max_{(\boldsymbol{a}(i), \cdots, \boldsymbol{a}(i+T_p-1))} \textstyle\sum_{t=i+1}^{i+T_p} f_t(\boldsymbol{y}(t), \boldsymbol{x}_c, \boldsymbol{y}_D)\\
\text{subj. to} \quad & \forall t = i, \cdots, i+T_p-1\\
& \boldsymbol{z}(t+1) = \boldsymbol{A}\boldsymbol{z}(t) + \boldsymbol{B}\boldsymbol{a}(t)\\
& \boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{z}(t)\\
& \boldsymbol{U} \geq \boldsymbol{y}(t) \geq \boldsymbol{L}\\
& \|\boldsymbol{E}(\boldsymbol{y}(t)) - \boldsymbol{O}_k\|^2 \geq d_k^2, \quad \forall k = 1, \cdots, N_{obj}\\
& \boldsymbol{y}(T) = \boldsymbol{y}_D(T).
\end{aligned} \tag{13}$$

3) Reward Function: In order to adapt robot movements to perform well in novel situations, considering only hard constraints such as obstacle avoidance, Eq.(11), does not suffice. Thus, we further model a reward function $f(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D)$ that reflects the amount of rewards that an adapted trajectory $\boldsymbol{y}$ can gain within the context $\boldsymbol{x}_c$ and $\boldsymbol{y}_D$. As the reward function $f(\boldsymbol{y})$ is assumed temporally discrete in Eq.(5), we model the reward function $f_t(\boldsymbol{y}(t))$ at t by three parts:

$$f_t(\boldsymbol{y}(t); \boldsymbol{w}) = f_{D,t}(\boldsymbol{y}(t); \boldsymbol{w}_D) + f_{C,t}(\boldsymbol{y}(t); \boldsymbol{w}_C) + f_{E,t}(\boldsymbol{y}(t); \boldsymbol{w}_E), \tag{14}$$

where the Imitation Reward f_D models the tendency to follow the imitation trajectory $\boldsymbol{y}_D$, the Control Reward f_C models the smoothness of executing the adapted trajectory $\boldsymbol{y}$ and the Response Reward f_E characterizes the expected response to the environment. Meanwhile, $\boldsymbol{w} = [\boldsymbol{w}_D^T, \boldsymbol{w}_C^T, \boldsymbol{w}_E^T]^T$ are parameters that affect the behavior of the movement adaptation. Next we describe each reward function in detail.

a) Imitation Reward: The Imitation Reward characterizes how well the adapted trajectory can imitate the demonstrations by the distance between points on $\boldsymbol{y}$ and $\boldsymbol{y}_D$. Recall that we have the variance $\boldsymbol{\Sigma}_D(t)$ of the imitation trajectory $\boldsymbol{y}_D$ in IV-A.2, which indicates how flexible we can adapt the trajectory. Considering $\boldsymbol{\Sigma}_D(t) = \text{diag}(\sigma_1^2(t), \ldots, \sigma_d^2(t))$ to be diagonal for the sake of simplicity, we model the Imitation Reward by the weighted distance:

$$f_{D,t}(\boldsymbol{y}(t); \boldsymbol{w}_D) = -(\boldsymbol{y}(t) - \boldsymbol{y}_D(t))^T \boldsymbol{V}(t)(\boldsymbol{y}(t) - \boldsymbol{y}_D(t)) \tag{15}$$

$$\boldsymbol{V}(t) = \text{diag}(\boldsymbol{w}_D)\text{diag}(e^{-\sigma_1^2(t)}, \ldots, e^{-\sigma_d^2(t)}), \tag{16}$$

where $\boldsymbol{V}(t)$ is a weight matrix consisting of parameters $\boldsymbol{w}_D$ and $\{e^{-\sigma_i^2(t)}\}$ in which the variances learned from demonstrations $\boldsymbol{\Sigma}_D(t)$ are modeled to affect adaptation rewards.

b) Control Reward: The Control Reward f_C characterizes the smoothness of executing the adapted trajectory $\boldsymbol{y}$ using the following formulation:

$$f_{C,t}(\boldsymbol{y}(t); \boldsymbol{w}_C) = -\boldsymbol{w}_C \|(\boldsymbol{y}(t) - \boldsymbol{y}(t-1))\|^2, \tag{17}$$

where $\boldsymbol{w}_C$ is the parameter to weigh this reward.

c) Response Reward: The Response reward f_E describes the expected response to the environment, such as safety considerations for obstacles and objects under manipulation. Here we give intuitive examples. Although we can ensure minimum distance to avoid obstacles using Eq.(13), as human users we still expect the robot to transfer a cup full of water *around* a laptop instead of *above* it, to avoid potential spills. Another example is that the user would prefer

that the robot when manipulating sharp objects, such as knives, keeps a relatively larger distance from the human for safe. All the above preferences are specific to objects under manipulation and the exact environment. Thus, we set the Response Reward such that the better the adapted trajectory fulfills the preferences, the higher the reward is.

To formally represent the Response Reward, let us consider a scenario with N_{obj} obstacles on the table. The leftmost and rightmost locations of the table are $\boldsymbol{B}_1, \boldsymbol{B}_2$ and the table surface is $\boldsymbol{S}$, we then can formulate it as follows:

$$f_{E,t}(\boldsymbol{y}(t); \boldsymbol{w}_E) = -\left(\sum_{k=1}^{N_{obj}} \boldsymbol{w}_{O,k}^T \boldsymbol{\phi}_{O,k} + w_B \phi_B + w_S \phi_S\right) \tag{18}$$

$$\begin{aligned} \boldsymbol{\phi}_{O,k}^T &= \left[-\|\boldsymbol{E}(\boldsymbol{y}(t)) - \boldsymbol{O}_k\|, (\boldsymbol{E}(\boldsymbol{y}_D(t)) - \boldsymbol{E}(\boldsymbol{y}(t)))^T\right] \\ &\cdot \quad \exp\left(-\frac{\|\boldsymbol{E}(\boldsymbol{y}(t)) - \boldsymbol{O}_k\|^2}{d_k}\right) \end{aligned} \tag{19}$$

$$\phi_B = \sum_{i=1}^{2} \exp\left(-\frac{\|\boldsymbol{E}(\boldsymbol{y}(t)) - \boldsymbol{B}_i\|^2}{d_{min}}\right) \tag{20}$$

$$\phi_S = \|\boldsymbol{E}(\boldsymbol{y}(t)) - S\|^2, \tag{21}$$

where $\boldsymbol{\phi}_{O,k}$ represents the feature vector for preferences in avoiding obstacle $\boldsymbol{O}_k$, of which the first element denotes avoiding distance and the second element denotes the deviation direction. The preferred deviation direction is given as reward weights and the inner product indicates the rewards of deviation considering the given preference. The exponential decay function is applied so that the features are only effective when the robot's end-effector is close to the obstacles. ϕ_B and ϕ_S are features related to safety by considering boarders and surface of the table. $\boldsymbol{w}_E = [\boldsymbol{w}_{O,1}^T, \ldots, \boldsymbol{w}_{O,N_{obj}}^T, w_B, w_S]^T$ are weights corresponding to the features respectively.

Given a set of parameters $\boldsymbol{w} = [\boldsymbol{w}_D^T, \boldsymbol{w}_C^T, \boldsymbol{w}_E^T]^T$, the MPC module generates an adapted trajectory by maximizing $f(\cdot; \boldsymbol{w})$. The robot could follow the trajectory and execute the task facing the novel situation. However, the generated trajectory may not be sufficiently satisfying from a user's perspective, since the given or initialized parameters may not be accurate for modeling the rewards. To accommodate this issue, after the movement execution, our system allows the user to provide a better trajectory as feedback to update the parameters during the following Rewards Learning section.

C. Rewards Learning

In this section, we describe how our system learns the reward function. Let us assume there is an oracle reward function $f^*(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D)$ that reflects exactly how much reward the adapted trajectory $\boldsymbol{y}$ can gain for each context. The goal of this module is to estimate such a reward function $f(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D; \boldsymbol{w})$, where $\boldsymbol{w}$ are the parameters to be learned, that approximate the oracle reward $f^*(\cdot)$ tightly.

By rewriting Eq.(5) and Eq.(14) for the entire trajectory, we can have the reward function in a linear form represented by features and weights:

$$f(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D; \boldsymbol{w}) = \boldsymbol{w}_D^T \boldsymbol{\phi}_D + \boldsymbol{w}_C^T \boldsymbol{\phi}_C + \boldsymbol{w}_E^T \boldsymbol{\phi}_E \tag{22}$$

$$\boldsymbol{\phi}_D = [\phi_{D,1}, \ldots, \phi_{D,d}]^T, \phi_{D,i} = -\sum_{t=0}^{T} (y_i(t) - y_{D,i}(t))^2 \, e^{-\sigma_i^2(t)} \tag{23}$$

$$\boldsymbol{\phi}_C = -\sum_{t=1}^{T} \|(\boldsymbol{y}(t) - \boldsymbol{y}(t-1))\|^2 \tag{24}$$

$$\boldsymbol{\phi}_E = -\sum_{t=0}^{T} \left[\boldsymbol{\phi}_{O,1}^T(\boldsymbol{y}(t)), \ldots, \boldsymbol{\phi}_{O,N_{obj}}^T(\boldsymbol{y}(t)), \phi_B(\boldsymbol{y}(t)), \phi_S(\boldsymbol{y}(t))\right]^T \tag{25}$$

where $\boldsymbol{\phi}_D, \boldsymbol{\phi}_C, \boldsymbol{\phi}_E$ represent features of the entire trajectory corresponding to Imitation, Control and Response Rewards.

Since the user only provides a feedback trajectory $\bar{\boldsymbol{y}}$ and the system can not directly observe the reward function, we apply the co-active learning technique [15] in which the robot iteratively updates the parameter $\boldsymbol{w}$ of $f(\cdot; \boldsymbol{w})$ based on user's feedback. Note that this feedback only needs to indicate $f^*(\bar{\boldsymbol{y}}, \boldsymbol{x}_c, \boldsymbol{y}_D) > f^*(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D)$ and $\bar{\boldsymbol{y}}$ could be non-optimal trajectories. Algorithm 1 gives our learning algorithm.

Algorithm 1 Rewards Learning for Movement Adaptation

Initialize $\boldsymbol{w}^{(0)} = [\boldsymbol{w}_D^{(0)T}, \boldsymbol{w}_C^{(0)T}, \boldsymbol{w}_E^{(0)T}]^T$
for Iteration $i = 0$ to T_l **do**
  Task Context and Environment Perception: $\boldsymbol{x}_c^{(i)}$
  Movement Imitation:
    $\boldsymbol{y}_D^{(i)}, \boldsymbol{\Sigma}_D^{(i)} \leftarrow p(\boldsymbol{\mathcal{T}} | \boldsymbol{x}_c^{(i)})$
  Movement Adaptation:
    $\boldsymbol{\pi}^{*(i)} = \arg\max_{\boldsymbol{\pi}} f(\boldsymbol{y}, \boldsymbol{x}_c^{(i)}, \boldsymbol{y}_D^{(i)}; \boldsymbol{w}^{(i)})$
    $\boldsymbol{y}^{(i)} \leftarrow \boldsymbol{\pi}^{*(i)}$
  Movement Execution: $\boldsymbol{y}^{(i)}$
  if User Provides Feedback: $\bar{\boldsymbol{y}}^{(i)}$ **then**
    $\alpha^{(i)} = 1/\sqrt{i}$
    $\boldsymbol{w}_D^{(i+1)} = \boldsymbol{w}_D^{(i)} + \alpha^{(i)}(\boldsymbol{\phi}_D(\bar{\boldsymbol{y}}^{(i)}, \boldsymbol{y}_D^{(i)}) - \boldsymbol{\phi}_D(\boldsymbol{y}^{(i)}, \boldsymbol{y}_D^{(i)}))$
    $\boldsymbol{w}_C^{(i+1)} = \boldsymbol{w}_C^{(i)} + \alpha^{(i)}(\boldsymbol{\phi}_C(\bar{\boldsymbol{y}}^{(i)}) - \boldsymbol{\phi}_C(\boldsymbol{y}^{(i)}))$
    $\boldsymbol{w}_E^{(i+1)} = \boldsymbol{w}_E^{(i)} + \alpha^{(i)}(\boldsymbol{\phi}_E(\bar{\boldsymbol{y}}^{(i)}, \boldsymbol{x}_c^{(i)}) - \boldsymbol{\phi}_E(\boldsymbol{y}^{(i)}, \boldsymbol{x}_c^{(i)}))$
    Weights Projection:
      $\bar{\boldsymbol{w}}^{(i+1)} = [\boldsymbol{w}_D^{(i+1)T}, \boldsymbol{w}_C^{(i+1)T}, \boldsymbol{w}_E^{(i+1)T}]^T$
      $\boldsymbol{w}^{(i+1)} = \arg\min_{\boldsymbol{w} \in \boldsymbol{C}} \|\boldsymbol{w} - \bar{\boldsymbol{w}}^{(i+1)}\|^2$
  else $\boldsymbol{w}^{(i+1)} = \boldsymbol{w}^{(i)}$
  end if
end for

Note that α is a learning rate, which decays along iterations, and $\boldsymbol{C}$ in the weights projection part is a bounded set to ensure that the updated parameters $\boldsymbol{w}$ are in a feasible space. After iterations of improvements, the robot can learn an estimated reward function $f(\cdot; \boldsymbol{w}^*)$ that approximates the oracle reward $f^*(\cdot)$ as proven in [18]. By maximizing the estimated reward function $f(\boldsymbol{y}, \boldsymbol{x}_c, \boldsymbol{y}_D; \boldsymbol{w}^*)$, the robot can generate an adapted trajectory $\boldsymbol{y}$ that maximizes the rewards facing situation $\boldsymbol{x}_c$ based on imitation trajectory $\boldsymbol{y}_D$.

V. Experiments

To validate the system described above, we design and conduct the following experiments on a Baxter humanoid platform. The Baxter robot is asked to do manipulation tasks such as cleaning on a table top. The workspace is defined with the surface as $\boldsymbol{S} = (0, 0, -0.1)$, the leftmost location as $\boldsymbol{B}_1 = (0, 0.8, 0)$ and the rightmost location as $\boldsymbol{B}_2 = (0, -0.8, 0)$ described in meters in robot spatial space, where the coordination system is shown in Fig. 4(a). It needs to learn transferring the manipulated object between different locations while avoiding obstacles in desired manners.

A. Movement Imitation

In the first stage of the experiments, we have our robot learn off-line the movement skill from kinethestic demonstrations with no obstacles on the table. All trajectories are sampled discretely and normalized to $T = 200$ steps for transferring movement in joint space, and the left arm of the Baxter has $d = 7$ degrees of freedom for joints from shoulder to wrist denoted as joint 1 to 7. The training trajectories are encoded by ProMPs with $n = 10$ Gaussian basis functions so that it can be generalized to different initial and target states.

Fig. 3(a) shows an example of our generated imitation trajectory in spatial space for new task contexts using ProMPs. Fig. 3(b) shows the corresponding imitation trajectory of joint 1 (first shoulder joint) in joint space. The blue crosses here are desired new initial and target states, and the shaded area is the estimated variance for the imitation trajectory, which reflects the variations of demonstrations. True trajectory here means a trajectory recorded from user demonstration in the testing scenario for comparison. It is not hard to see that the predicted mean of the imitation trajectory well generalizes to new initial and target states and follows the same movement pattern as the prior mean trajectory learned from demonstrations. Therefore, the robot can perform the task well by following this imitation trajectory if there are no obstacles or other safety constraints.

B. Learning Adaptation

We consider the situation, where the robot, while facing the task of transferring a leaking bottle, finds a bowl filled with food as obstacle on the table. We assume that the bowl's center location $\boldsymbol{O}_1$ and minimum safety distance d_1 are obtained through perception. For movement adaptation, we set the prediction horizon $T_p = 11$ in the model predictive control and select system matrices $\boldsymbol{A} = 0.9 \cdot \boldsymbol{I}, \boldsymbol{B} = \boldsymbol{C} = \boldsymbol{I}$ to make the system stable in the prediction window as suggested in [13]. The limits of joints could be found from the Baxter hardware specification. The minimum safety distance to the table boarder is set as $d_{min} = 0.1$. And the weights for reward function are initialized to $\boldsymbol{w}_D = 30 \cdot \boldsymbol{1}, \boldsymbol{w}_C = 10, \boldsymbol{w}_E = \boldsymbol{0}$. We apply the native Matlab Gradient-based optimization method *fmincon* to solve the optimization at each time step.

Fig. 3(c) shows the movement imitation for transferring the leaking bottle, which failed to avoid the obstacle even though the trajectory generalizes to a novel initial and target states. Fig. 3(d) shows the movement adaptation with initial weights. There is no preference specified in the reward function about how to avoid obstacles. Therefore, even though the adapted trajectory could avoid the obstacle successfully, it may be not an ideal trajectory.

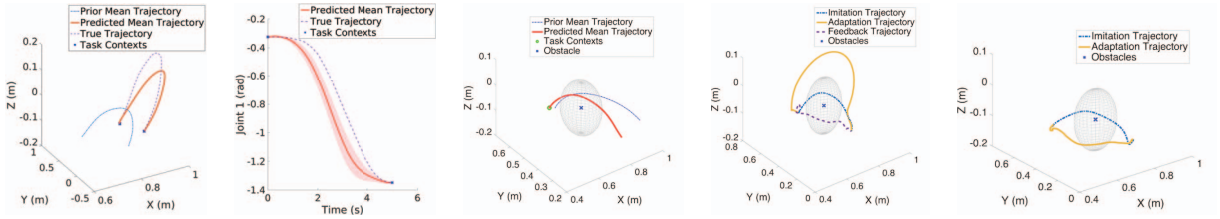
To learn the user preference, we then provide feedback via kinethestic demonstration illustrated in Fig. 4(a) and the feedback trajectory is shown in Fig. 3(d) as dashed line to indicate user preferences. Following Algo. 1, the robot iteratively updates the rewards weights based on the user feedback. Weights are limited via projection in the feasible set $\boldsymbol{C}$ where $\boldsymbol{w}_D \in [1, 100]^7, \boldsymbol{w}_C \in [1, 100], \boldsymbol{w}_E \in [0, 100]$ except that the last two parameters in $\boldsymbol{w}_{O,k}$ indicating preferred deviation direction could be $[-100, 100]$. To quantitatively validate the performance of our method in movement adaptation, we consider the metric of cumulative error between the adapted trajectory and the feedback trajectory $e(i) = \frac{1}{T}\sum_{t=0}^{T}\left(\bar{\boldsymbol{y}}^{(i)}(t) - \boldsymbol{y}^{(i)}(t)\right)^2$ as the learning error at iteration i. Since the metric is affected by different situations such as obstacles' locations, we consider the feedback trajectory as fixed and let the robot iteratively learn several times to see how it performs, and we record the "learning curve" under the same feedback. From Fig. 4(b), we can see that the error decreases and converges after several iterations, and it only requires a few iterations to achieve an adapted trajectory as desired preference according to the feedback. After learning, the robot uses the updated weights for movement adaptation in a different situation with novel initial/target states and obstacles' locations. Fig. 3(e) shows the adapted trajectory based on the updated weights after one iteration, where it successfully avoids the obstacle via the desired direction.

In a second scenario where a robot is transferring a knife around some fragile obstacle, the user may prefer the robot to avoid the obstacle above it instead of around it. With the same methods here, we could also generate adapted trajectories as shown in Fig. 5(a) and Fig. 5(c) for initial weights. With the user provided feedback, the robot successfully learns the specified preferences for movement adaptation and generates the improved adapted trajectories for different situations as shown in Fig. 5(b) and Fig. 5(d).

VI. Conclusion and Future Work

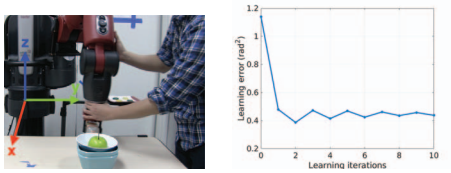
We presented a framework for learning to adapt robot end effector movement for manipulation tasks. The proposed method generalizes offline learned movement skills to novel situations considering obstacle avoidance and other task-dependent constraints. It adapts the imitation trajectory generated from demonstrations, while maintaining the learned movement pattern and considering variations in the geometry. Here we considered as variations, avoiding obstacles with movements in desired directions, and keeping certain distances for a safety margin within a workspace. The methods also provides a way to learn how to adapt the movement in on-line interactions with user's feedback.

Another interesting way to incorporate environmental constraints would be to consider visual information of objects and the environment as an indication of the preferences for movement adaptation. For instance, the deviation direction



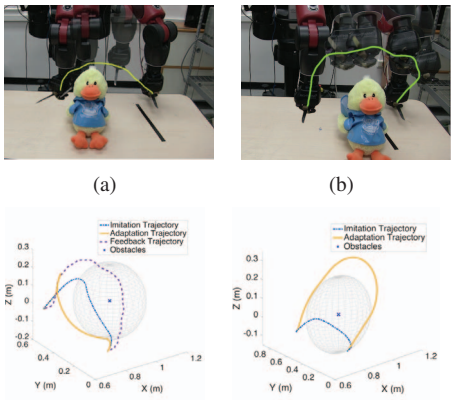
(a) (b) (c) (d) (e)

Fig. 3. Learning to Transfer a Leaking Bottle: (a) Imitation trajectory in spatial space; (b) Imitation trajectory for joint 1 in joint space, shaded area indicating the predicted variance.(a) Movement Imitation failed to avoid the obstacle; (b) Movement adaptation with initial weights successfully avoided the obstacle by a path above it but has a potential danger of spilling water, feedback trajectory is provided afterwards; (c) Movement adaptation for a different situation with updated weights after learning from feedback trajectory, successfully avoids the obstacle through a path around. Corresponding execution on the Baxter platform is given by Fig. 2.



(a) (b)

Fig. 4. Rewards Learning from User Feedback for Transferring Leaking Bottle: (a) User feedback via kinethestic demonstration; (b) Learning curve for adaptation under the same feedback.



(a) (b) (c) (d)

Fig. 5. Baxter Learning to Adapt Movement for Transferring Knife: (a) (c) Movement adaptation with initial weights using a path around the duck doll successfully avoided it but risked scratches; afterwards feedback trajectory is provided for preferences; (b) (d) Movement adaptation for different situations, with updated weights after learning from feedback trajectory, successfully avoided the duck doll using a path above it as desired.

for avoiding a knife could be directly inferred from the location and orientation of its blade from visual input. In ongoing work we are further investigating the possibility of directly learning the preferences to adapt movement from visual perception for the task context.

ACKNOWLEDGMENT

This work was supported by DARPA (through ARO) grant W911NF1410384 and by NSF through grants CNS-1544787 and SMA-1540917.

REFERENCES

[1] 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.

[2] T. Asfour, P. Azad, F. Gyarfas, and R. Dillmann, "Imitation learning of dual-arm manipulation tasks in humanoid robots," *International Journal of Humanoid Robotics*, vol. 5, no. 02, pp. 183–202, 2008.

[3] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 763–768.

[4] J. Kober, B. Mohler, and J. Peters, "Learning perceptual coupling for motor primitives," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, 2008, pp. 834–839.

[5] A. Gams, A. J. Ijspeert, S. Schaal, and J. Lenarčič, "On-line learning and modulation of periodic movements with nonlinear dynamical systems," *Autonomous robots*, vol. 27, no. 1, pp. 3–23, 2009.

[6] F. Guenter, M. Hersch, S. Calinon, and A. Billard, "Reinforcement learning for imitating constrained reaching movements," *Advanced Robotics*, vol. 21, no. 13, pp. 1521–1544, 2007.

[7] S. Calinon, F. D'halluin, E. L. Sauser, D. G. Caldwell, and A. G. Billard, "Learning and reproduction of gestures by imitation," *Robotics & Automation Magazine, IEEE*, vol. 17, no. 2, pp. 44–54, 2010.

[8] S. M. Khansari-Zadeh and A. Billard, "Imitation learning of globally stable non-linear point-to-point robot motions using nonlinear programming," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 2676–2683.

[9] T. Inamura, I. Toshima, H. Tanie, and Y. Nakamura, "Embodied symbol emergence based on mimesis theory," *The International Journal of Robotics Research*, vol. 23, no. 4-5, pp. 363–377, 2004.

[10] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, "Probabilistic movement primitives," in *Advances in neural information processing systems*, 2013, pp. 2616–2624.

[11] D.-H. Park, P. Pastor, S. Schaal *et al.*, "Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*. IEEE, 2008, pp. 91–98.

[12] H. Hoffmann, P. Pastor, D.-H. Park, and S. Schaal, "Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 2587–2592.

[13] A. M. Ghalamzan E., C. Paxton, G. D. Hager, and L. Bascetta, "An incremental approach to learning generalizable robot tasks from human demonstration," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5616–5621.

[14] E. A. Sisbot, L. F. Marin, and R. Alami, "Spatial reasoning for human robot interaction," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. IEEE, 2007, pp. 2281–2287.

[15] A. Jain, S. Sharma, T. Joachims, and A. Saxena, "Learning preferences for manipulation tasks from online coactive feedback," *The International Journal of Robotics Research*, p. 0278364915581193, 2015.

[16] R. Mao, Y. Yang, C. Fermuller, Y. Aloimonos, and J. S. Baras, "Learning hand movements from markerless demonstrations for humanoid tasks," in *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*. IEEE, 2014, pp. 938–943.

[17] Z. Ju, C. Yang, and H. Ma, "Kinematics modeling and experimental verification of baxter robot," in *Control Conference (CCC), 2014 33rd Chinese*. IEEE, 2014, pp. 8518–8523.

[18] G. Ciná and U. Endriss, "Proving classical theorems of social choice theory in modal logic," *Autonomous Agents and Multi-Agent Systems*, pp. 1–27, 2016.