Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Learning Preferences for Intention-Aware Physical Human-Robot Cooperation

Linda Spaa, van der, Jens Kober, Michael Gienger

2024

Preprint:

This is a post-peer-review, pre-copyedit version of an article published in Autonomous Robots. The final authenticated version is available online at: https://doi.org/10.1007/s10514-024-10167-3

Simultaneously Learning Intentions and Preferences during Physical Human-Robot Cooperation

Linda van der Spaa[1,2*], Jens Kober[1] and Michael Gienger[2]

[1*]Dep. of Cognitive Robotics, Delft University of Technology, Mekelweg 2, 2628 CD, Delft, The Netherlands.

[2]Honda Research Institute Europe (Germany) DE, Carl-Legien-Straße 30, 63073, Offenbach (Main), Germany.

*Corresponding author(s). E-mail(s): L.F.vanderSpaa@tudelft.nl;
Contributing authors: J.Kober@tudelft.nl; michael.gienger@honda-ri.de;

Abstract

The advent of collaborative robots allows humans and robots to cooperate in a direct and physical way. While this leads to amazing new opportunities to create novel robotics applications, it is challenging to make the collaboration intuitive for the human. From a system's perspective, understanding the human intentions seems to be one promising way to get there. However, human behavior exhibits large variations between individuals, such as for instance preferences or physical abilities. This paper presents a novel concept for simultaneously learning a model of the human intentions and preferences incrementally during collaboration with a robot. Starting out with a nominal model, the system acquires collaborative skills step-by-step within only very few trials. The concept is based on a combination of model-based reinforcement learning and inverse reinforcement learning, adapted to fit collaborations in which human and robot think and act independently. We test the method and compare it to two baselines: one that imitates the human and one that uses plain maximum entropy inverse reinforcement learning, both in simulation and in a user study with a Franka Emika Panda robot arm.

Keywords: Inverse Reinforcement Learning, Physical Human-Robot Interaction, Human-Robot Collaboration, Human-Centered Planning

1 Introduction

Physical human-robot collaboration (pHRC) is becoming increasingly popular, as it has the potential to increase flexibility and efficiency in industrial automation (Hanna et al, 2022) as well as support people in home environments (Fitter et al, 2020). To realize a fluent and intuitive collaboration, such novel robot systems should ideally be capable of understanding the intentions of the human partner, and to adapt their behavior accordingly. From a system's perspective, autonomously learning to interpret human intentions will make it easier and more intuitive for humans to engage in joint tasks, an important step towards cooperative intelligence (Sendhoff and Wersing, 2020). This requires a learning algorithm that is fast, needs little data, and learns in a way that is safe for both the robot and its environment.

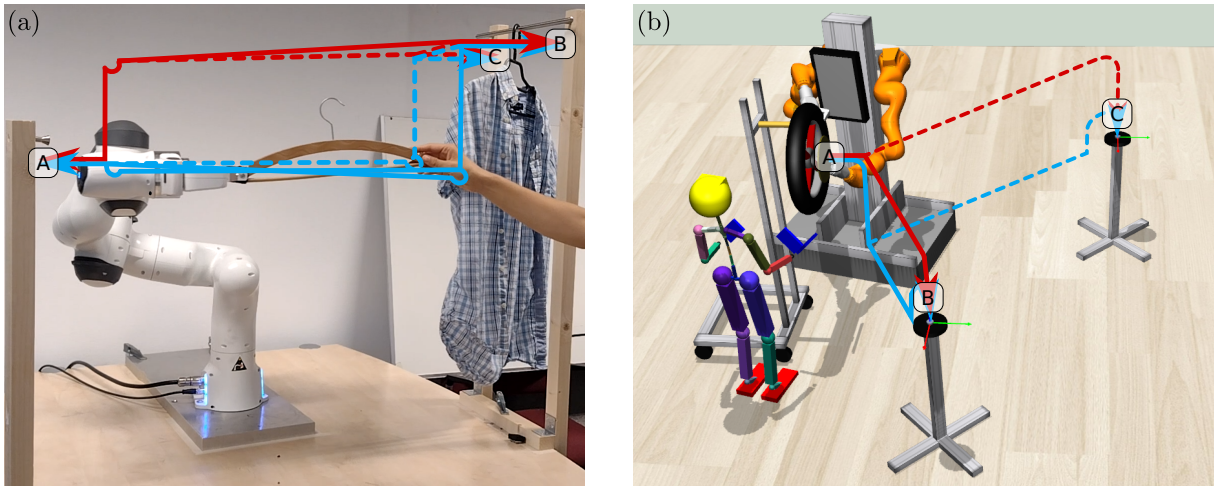


Fig. 1 Two cooperative scenarios: The robot needs to learn how best to assist the human to move the object between support points A, B, and C. Two colors of arrows indicate different paths along which the human may prefer to move the object (the clothes hanger on the left (a), the wheel on the right (b)). The dashed lines indicate how the preferences may generalize when the goal is different.

In cooperation, the success of a task depends on the combination of what all actors do. Moreover, *how* a task is best completed depends additionally on the individual actors' preferences and the interaction dynamics. A team learning to work together needs to learn how the 'system' (including their colleagues) is responding. They need to learn how to follow/express their preferences within the bounds imposed by both the task and their teammates' preferences and capabilities.

This paper addresses the challenge of enabling a robot to learn to cooperate with a human. In the setting we consider, the robot does not know the exact intention of the human and simultaneously attempts to act according to the human's preferences. We make an explicit distinction between *intentions*: *what* (sub)goal someone currently has, and *preferences*: *how* the person likes to approach the (sub)goal. Figure 1 shows two example scenarios: The robot needs to learn how to help the human move the object, a clothes hanger or a wheel, to the goal intended by the human.

We consider the problem on the abstract level where the two agents (human and robot) have prelearned/programmed skills, e.g., *grasp the object*, or, *pull towards* x, y, z *in space*. Compliant control lets the success of actions depend on the physical interaction. This allows us to focus on the learning problem in this paper, without simultaneously having to consider the specific mechanics of physical human-robot interaction.

This paper's first contribution is a novel method for learning a human preference model for intention-aware cooperation, from collaborative episodes. The method 1) learns a personalized model of a human partner from physically cooperating with this partner, from scratch or improving a nominal model; 2) models human preferences as an explicit function of intention, enforcing inherent intention awareness; 3) applies two-level Theory of Mind (ToM) reasoning to model the human's preferences separate from the robot's, resulting in explicit partner awareness. This allows the robot to optimize an objective different from the human for improved cooperative behavior. The process is iterative: after each collaborative episode, the robot updates its internal models based on the observed partner response and the intention observed in hindsight at the end of the episode. As its internal models improve, so does the robot's response. Since most optimization is done internally in the modeled environments, the robot requires very few experimental episodes for learning. We achieve this by combining existing Reinforcement Learning (RL) and Inverse Reinforcement Learning (IRL) methods in a novel way.

Secondly, we contribute by testing our method in a user study with a diverse group of mostly novice users. We compare our "Learner" to an "Imitator" baseline which lets the robot merely imitate its partner. Users were free to choose their

preferences (within the limits of the setup). We evaluate the user experience and the performance both quantitatively and qualitatively. In simulation, we additionally try our method in a scenario with increased complexity for further evaluations and insights for directions of future work.

Sec. 2 discusses the related literature. Then, the method is presented in Sec. 3. Implementation considerations are discussed in Sec. 4. We describe the scenarios shown in Fig. 1 in Sec. 5, on which we evaluate our method's performance in a user study in Sec. 6, and in additional simulations in Sec. 7, before we conclude in Sec. 8.

2 Related Work

Before we present our method to learn a behavioral model of a human partner for improved intention-aware planning, we will first discuss relevant literature in the three main directions related to our work: intention-aware planning, behavioral modeling, and model learning.

2.1 Intention-aware planning

Literature on intention estimation for human-robot cooperation (HRC) tends to fall into one of the following three categories: (sub)goal estimation – predicting which (sub)goal out of a set of possibilities the human is trying for (Karami et al, 2009; Malik et al, 2018); action prediction – predicting which (primitive) action the human will take next (Hawkins et al, 2014; Gienger et al, 2018; Belardinelli et al, 2022); motion extrapolation – predicting how fast the human will continue in which direction (Duchaine and Gosselin, 2007; Bai et al, 2015), or along which trajectory (Ranatunga et al, 2015; Park et al, 2019).

The last category is useful for collision avoidance (e.g., to independently navigate the same environment (Bai et al, 2015; Park et al, 2019)), and for motion following (e.g., steering a single tool (Duchaine and Gosselin, 2007; Ranatunga et al, 2015)). More abstract level planning needs higher-level action predictions. On the top level, an estimate of the goal the human wants to reach will allow a robot to plan further ahead. Somewhere in between are reaching and placement tasks, where the intention encodes both the motion and the goal (Koert et al, 2019), and, in the pHRI case, the interaction forces (Lai et al, 2022; Haninger et al, 2022).

Instead, we consider tasks consisting of a chain of actions, and different possible goals can each be reached in multiple ways. We seek to learn human preferences in (physical) cooperation while we have no direct access to the human partner's intention. Similar to Koppula et al (2016); Park et al (2019), we define the problem as a Markov Decision Process (MDP). Intentions can be incorporated as a 'hidden state', resulting in a Partially Observable MDP (POMDP) (Karami et al, 2009; Bai et al, 2015), or a Mixed Observability MDP (MOMDP) (Ong et al, 2009). This definition allows us to use standard techniques for learning a fitting robot policy, determining when it will take which action given the observations.

For robot-robot cooperation in the MDP domain, Multi-Agent Reinforcement Learning (MARL) techniques have been derived from single-agent techniques (Buşoniu et al, 2010). Some of those methods could be applied to human-robot cooperation problems, but have the disadvantage of requiring a large number of trials which is impracticable for learning in interaction.

2.2 Human behavior modeling

For directed cooperation, a robot needs a model of the agents with whom it should cooperate (Choudhury et al, 2019). Agents can be modeled by a black box model, such as a neural network (Schmerling et al, 2018; Zyner et al, 2019). Although such a model can give accurate predictions, collecting sufficient representative data in a pHRC scenario is expensive from a human perspective. More recently, Shih et al (2022) and Xie et al (2021); Wang et al (2022); Parekh et al (2022) solved this by learning a low-rank latent space in different ways from few demonstrations which allows for interpolation to predict previously unseen partner policies or strategies respectively. Shih et al (2022) and Parekh et al (2022) show the effectiveness of the approach with human subjects. Nevertheless, these models still require a considerable amount of sufficiently diverse data, which is challenging to obtain in pHRC. Furthermore, it is not straightforward to set up the representation learning such that the latent space covers all the preferences the human partner may have. As the methods do show great promise, it is

an interesting direction for future work to research how this approach could incorporate hidden but leading partner intentions, and how suitable training data could be generated for such methods to handle pHRC scenarios as addressed in this paper.

Alternatively, gray box models have a structure which offers insight into the prediction process and increases data efficiency, if a proper structure is provided. A simple single parameter can already improve a robot's cooperative skills (Nikolaidis et al, 2017a). More complex structures may be derived from dynamics (Stouraitis et al, 2020) or from Theory of Mind (Choudhury et al, 2019). ToM originates from the fields of psychology and philosophy (Baker and Tenenbaum, 2014) and reasons about the reasoning of others. For example, a robot may model a human as an agent with its own internal model of the task and the world. When such a model includes the human's reasoning about the robot's reasoning about them, etc. (infinite regress), it is no longer practical. Successful implementations limit the regress to one or two levels (Buehler and Weisswange, 2018; Sadigh et al, 2016; Malik et al, 2018). ToM can be considered as an IRL problem (Jara-Ettinger, 2019). We follow this example, using IRL to learn a mental model of the human partner, which we can then use to optimize our robot's collaborative actions.

2.3 Inverse Reinforcement Learning

Inverse Reinforcement Learning focuses on inferring the underlying reward function from demonstrated samples. However, the problem of reward reconstruction is ill-posed: more than one reward function could describe the same demonstrated policy. Maximum Entropy IRL (ME-IRL) offers a solution to this problem which is the least biased on the demonstrations (Ziebart et al, 2008; Zhifei and Joo, 2012). Derived methods have been applied successfully to learn from non-expert data (Boularias et al, 2011) or incrementally update the model as data comes in (Jin et al, 2011; Rhinehart and Kitani, 2018), or both and from physical interaction (Losey et al, 2022).

In Cooperative IRL (CIRL) as described in Hadfield-Menell et al (2016), the human and the robot optimize the same Q-function. This is also the case in Malik et al (2018), where the human and the robot are modeled as different actors. Instead, we explicitly treat our robot and human as independent agents by learning/keeping separate reward functions for each, without giving up on the overall cooperative objective.

Dynamic Game (DG) theory is used in autonomous driving to adequately respond to (independent) other road users, whose behavior can be estimated by a single strategically chosen parameter (Schwarting et al, 2019) or through Inverse DG (IDG) (Peters et al, 2023; Mehr et al, 2023). In pHRC, something in between was done by Nikolaidis et al (2017b), where the human objective function is estimated while the human is learning the "true global cooperative reward" initially known only by the robot. More recent work takes the DG theoretical approach in haptic shared control, either assuming a known human cost function (Musić and Hirche, 2020) or learning one through Inverse Optimal Control (Franceschi et al, 2023). On the same motor-control level, Hafs et al (2024) apply IDG to estimate the human cost function in collaboration with an exoskeleton. Development of IDG for pHRC has been very recent, and focused so far mainly on the motor-control level. To apply these methods to the more abstract task level considered in this paper, including a hidden intention state, is a different direction of research. It will be interesting to compare the methods in the future.

3 Method

In order to optimize the robot response in cooperation, we learn a model of the human partner's behavior, including their response to the robot. We break this loop into three interconnected learning processes, indicated by the ellipses in Fig. 2. We start off with a nominal (safe) robot policy π^{R} and an initial estimate of the human reward function R^{H}. After every episode we try the collaborative task, we update our human reward estimate on the observed human actions in ζ and intention ι. To the human reward estimate, we apply RL to compute the most likely human response $\hat{\pi}^{\mathrm{H}}$, which we then use to compute an improved robot response π^{R}. Thus, we iterate.

A key element of the presented method is the explicit modeling of the human's intention, a variable which is not directly observable but assumed to uniquely define a person's response. The intention ι is a discrete variable. We assume the set of possible intentions $\mathcal{I}$ is known. The

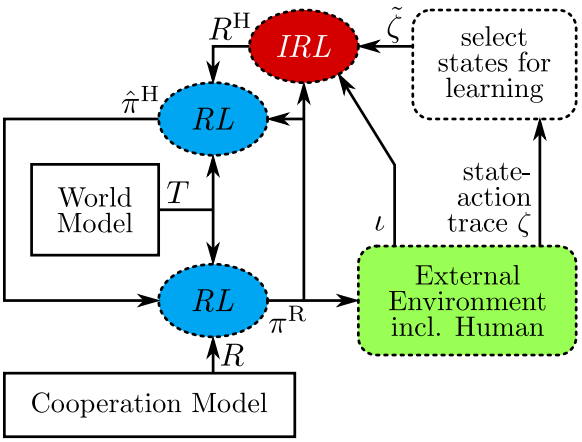


Fig. 2 Method overview, showing the learning processes in ellipses, other processes in dotted-lined rounded rectangles, models in solid-lined rectangles, and functions and data on the arrows. The human preference model R^{H} is updated by the IRL process based on observed state sequence ζ and intention ι. The two RL processes compute human policy estimate $\hat{\pi}^{\mathrm{H}}$ and robot policy π^{R}.

human preference model, captured in R^{H}, is a function of this intention, which allows it to be inferred by comparing the model to the observed actions. A real-valued parameter vector is updated in R^{H} after every episode to improve the feature match to the observed paths from the start to the intended goal.

To summarize, the *preferences* are captured by the human reward function R^{H} learned through IRL, while the *intentions* are captured by a variable ι that the robot cannot access directly but needs to infer from observed human actions.

We consider a discrete state-action space, where the state $\boldsymbol{s}$ is the combined state (for the robot, human, objects, environment, etc.) and we have separate actions for the robot a^{R} and the human a^{H}. The states and actions we employ in our experiments are detailed in Sections 4 and 5. As the model improves, so does the robot's response, decreasing cooperation effort.

First, Sec. 3.1 briefly recaps the necessary background on MDPs, Q-iteration, and soft-max policy optimization. Sec. 3.2 explains how these concepts have been modified to fit our collaborative case with hidden intention. Sec. 3.3 briefly discusses ME-IRL and its application to our multi-agent learner before Sec. 3.4 summarizes our algorithm.

3.1 MDPs and their model-based solution

An MDP is defined by the tuple $\{\mathcal{S}, \mathcal{A}, T, R, \gamma\}$, consisting of a state space $\mathcal{S}$ containing states s, action space $\mathcal{A}$ containing actions a, transition model $T(s' \mid s, a)$, reward function $R(s, a, s')$ and discount factor $\gamma \in [0, 1)$. In the model-based case, where the entire tuple is available, the value indicating the desirability of each state-action pair can be computed via Q-iteration:

$$Q(s,a) \leftarrow \sum_{s' \in S} T(s' \mid s,a)R(s,a,s') + \gamma V(s'), \quad (1)$$

with value function $V(s) = \max_{a \in \mathcal{A}} Q(s,a)$.

The Q-function is a sound basis for extracting a policy $\pi(a \mid s)$ an agent can use to decide which action to take in a state. We select our policies by taking the weighted soft-max as described by Tijsma et al (2016):

$$\pi(a_i \mid s) = \frac{e^{\tau Q(s,a_i)}}{\sum_a e^{\tau Q(s,a)}}. \quad (2)$$

The exponential relationship between an action's Q-value and its probability to be selected results in directed exploration around the optimal policy. Exploration can be decreased by weighting the Q-values by a temperature parameter $\tau \geq 1$. A small amount of directed exploration tends to speed up learning. It will mitigate modeling errors in cases multiple actions come up with similar values and which one shows up best depends heavily on an inaccurate model. This may very well happen in our case, since all internal MDPs depend on the human preference model, which is being learned.

For the robot, we do restrict exploration with a lower bound on the acceptable action Q-value: $\eta \ \max_{a^{\mathrm{R}}} Q^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}})$. This way, we prune potentially bad actions. Additionally, the bound can be set to only allow actions with a value at least as high as a baseline deterministic policy.

3.2 Multi-agent policy optimization with hidden intention

In our collaborative case, there are two necessary adaptations if we are to use the MDP principles of the previous subsection. First, we need to account for a collaborative partner whose actions

we assume we cannot control. Second, one state variable —the intention— is hidden for the robot. We assume the human knows their own intention, so we treat it as a regular state variable within the human model. However, the robot does not know this true intention, and we hence need to maintain the uncertainty over it in the robot model.

To solve the first problem, we extract the single-agent transition function for the agent we are interested in from the combined transition function $T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R}}, a^{\mathrm{H}})$, where a^{R} is the robot's action and a^{H} the human's. This is done by substituting the partner policy. For the robot transition function T^{R} we replace a^{H} by an estimate of the human policy $\hat{\pi}^{\mathrm{H}}$, resulting in $T^{\mathrm{R}}(\boldsymbol{s}' \mid \boldsymbol{s}, \iota, a^{\mathrm{R}}) = T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R}}, \hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota))$. For the human transition function T^{H} we replace a^{R} by the robot policy π^{R}, resulting in $T^{\mathrm{H}}(\boldsymbol{s}' \mid \boldsymbol{s}, \iota, a^{\mathrm{H}}) = T(\boldsymbol{s}' \mid \boldsymbol{s}, \pi^{\mathrm{R}}(\boldsymbol{s}, \iota), a^{\mathrm{H}})$. Note that the single-agent transition function is a function of intention ι because the partner policy depends on the intention.

In the human case, we find ourselves at the second level down the ToM. For predicting the human policy π^{H}, we need an estimate of how the human perceived the robot policy π^{R}. We cut the regression by using the most recent robot policy. It is an overestimate of what the human can know, but it is the best we have. If we would assume instead that the human models the robot as a random agent, the human would be modeled without any trust in the robot policy, which will make it much harder to learn policies that actually rely on the robot taking a certain action. Since our robot is learning, we cannot model human learning of the robot reward as in Nikolaidis et al (2017b), nor can we follow Tian et al (2023) and disregard the large effect of our robot's actions on the discrete state transitions within the human model. Modeling how the human partner would learn to trust the robot is out of scope of the current paper, although interesting to explore in future work.

The resulting human policy estimate $\hat{\pi}^{\mathrm{H}}(a^{\mathrm{H}} \mid \boldsymbol{s}, \iota)$ is used to obtain the robot transition function. The robot reward function $R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}, a^{\mathrm{H}}, \boldsymbol{s}')$, additionally depends on the human actions to explicitly encode cooperation objectives. Here, $\hat{\pi}^{\mathrm{H}}$ is substituted in the same way as in the robot transition function T^{R}, resulting in $R^{\mathrm{R}}(\boldsymbol{s}, \iota, a^{\mathrm{R}}, \boldsymbol{s}') = R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}, \hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota), \boldsymbol{s}')$.

The human policy estimate is a function of the intention, which the robot cannot observe directly. We assume that the human acts consistently under a given intention, which enables the robot to infer the intention from observations of the taken human actions. Since it is only one-dimensional and very small-sized, it is computationally feasible to resolve this second problem by computing the MDP and its solution for each possible intention $\iota \in \mathcal{I}$. For larger problems, we advise to adapt a MOMDP solver (Ong et al, 2009).

At runtime, a belief distribution is estimated over the possible intentions, using a Bayes filter:

$$b(\iota') = C\hat{\pi}^{\mathrm{H}}(a^{\mathrm{H}} \mid \boldsymbol{s}, \iota') \sum_{\iota \in \mathcal{I}} P(\iota' \mid \iota) b(\iota), \tag{3}$$

with normalizing constant C. The intention transition probability is the likelihood of the observed human action combined with the chance of keeping or changing the intention:

$$P(\iota' \mid \iota) = \begin{cases} \beta, & \iota' = \iota \\ \frac{1-\beta}{n-1}, & \iota' \neq \iota \end{cases} \tag{4}$$

with 'intention bias' $\beta \in [\frac{1}{n}, 1]$ and n possible intentions. The closer β is chosen to 1, the harder it is for the robot to understand, and thus adapt to, the situation when the estimated intention does not match the human's. This may happen because the human changed intention, or the estimate may have been wrong because of errors in the learned model. Smaller β results in faster robot adaptation (at runtime), but too small a β makes it impossible for the robot to effectively exploit its intention parameterized internal model.

Having the belief estimate, the final robot Q-function is obtained by superposition (Schweitzer and Seidmann, 1985):

$$Q^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}) = \sum_{\iota \in \mathcal{I}} b(\iota) Q^{\mathrm{R}}_{\iota}(\boldsymbol{s}, a^{\mathrm{R}}). \tag{5}$$

3.3 IRL human model updates

The IRL objective is to maximize the total reward of the optimal trajectory ζ^*_{ι}, which in our case depends on the intention ι. The reward

$$\sum_{\boldsymbol{s}_j \in \zeta^*_{\iota}} R(\boldsymbol{s}_j, \iota) = \boldsymbol{\theta}^T \boldsymbol{\phi}_{\zeta^*_{\iota}} = \boldsymbol{\theta}^T \sum_{\boldsymbol{s}_j \in \zeta^*_{\iota}} \boldsymbol{\phi}(\boldsymbol{s}_j, \iota) \tag{6}$$

is a linear combination of the features $\boldsymbol{\phi}$ observed in the trajectory given intention ι, weighed by

$\boldsymbol{\theta}$. Expert demonstrations $\tilde{\zeta}_{\iota,i}$ are assumed representative for the optimal trajectory. ME-IRL maximizes the log-likelihood of the observed trajectories (Ziebart et al, 2008):

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_i \log P(\tilde{\zeta}_i \mid \boldsymbol{\theta}, T), \tag{7}$$

This can be solved by gradient descent $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \lambda \nabla L$, where ∇L equals the difference in feature counts between the observed trajectories and the expected feature counts according to the model. The expected feature counts are computed by internal soft-max Q-iteration, using the human transition model T^{H}. Here, we only consider the intention observed during the episode. Like other incremental IRL methods (Jin et al, 2011; Rhinehart and Kitani, 2018), we perform a single gradient descent update after each episode.

In our interactive case, we must be somewhat selective in providing demonstration data to the learning algorithm. States are now visited because of both the human and the robot action. If the robot made a wrong choice and the human had to wait or correct, this should not be interpreted as optimal, just because it was observed. To resolve this, any loops between states that are visited multiple times are assumed to be caused undesirably by inexperience, and are therefore removed before updating the human model.

3.4 The combined algorithm

Algorithm 1 shows the full method. After initialization of the human model (L. 1), the lifelong learning loop starts. The robot models are extracted (L. 3–4) and the Q-functions are optimized per intention (L. 5). During a cooperative episode (initialized in L. 6), the robot Q-values are computed for the current state (L. 8). The policy in the current state is obtained (L. 9) using the bounded soft-max discussed in Sec. 3.2. When the robot and the human have performed their actions and the state is updated (L. 10), so are the robot belief (L. 11) and the state-action trace ζ (L. 12). The episode continues until a goal is reached (L. 13), then the state-action trace and the human intention (the reached goal) are returned to the model environment (L. 14). States are selected for learning (L. 17) and the human transition model is extracted (L. 18). The IRL step updates the feature weights of the human preference model (L. 19) using the state-action sequence observed during the latest episode and the human transition model given the observed intention. The human reward model (L. 20), Q-function (L. 21), and policy estimate (L. 22) are updated, and the cycle repeats.

Algorithm 1 Learning human-aware cooperation

Require: $T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R,H}})$, $R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R,H}}, \boldsymbol{s}')$, $\boldsymbol{\phi}(\boldsymbol{s}, \iota)$

1: $\boldsymbol{\theta} = \boldsymbol{\theta}^0$, $R^{\mathrm{H}}(\boldsymbol{s}, \iota) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{s}, \iota)$, $\hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota) \leftarrow \mathcal{U}(\boldsymbol{s})$
2: **while** True **do**
3: $T^{\mathrm{R}}(\boldsymbol{s}' \mid \boldsymbol{s}, \iota, a^{\mathrm{R}}) = T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R}}, \hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota))$
4: $R^{\mathrm{R}}(\boldsymbol{s}, \iota, a^{\mathrm{R}}, \boldsymbol{s}') = R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}, \hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota), \boldsymbol{s}')$
5: $Q^{\mathrm{R}}_\iota(\boldsymbol{s}, a^{\mathrm{R}}) \leftarrow \textsc{QIter}(T^{\mathrm{R}}, R^{\mathrm{R}})\ \forall \iota \in \mathcal{I}$
6: $\boldsymbol{s}_0$, $b_0(\iota) \leftarrow \mathcal{U}(\iota)$, $\zeta \leftarrow \emptyset$
7: **while** Collaborative Episode **do**
8: $Q^{\mathrm{R}}(\boldsymbol{s}_t, a^{\mathrm{R}}) = \sum_\iota b(\iota) Q^{\mathrm{R}}_\iota(\boldsymbol{s}_t, a^{\mathrm{R}})$
9: $\pi^{\mathrm{R}} \leftarrow \textsc{BoundedSoftMax}(Q^{\mathrm{R}})$
10: $\boldsymbol{s}_{t+1}, a^{\mathrm{H}}_t, a^{\mathrm{R}}_t \leftarrow \textsc{DoAction}(\pi^{\mathrm{R}})$
11: $b_{t+1} \leftarrow \textsc{UpdateBelief}(b_t, \boldsymbol{s}_t, a^{\mathrm{H}}_t)$
12: $\zeta \leftarrow \textsc{UpdateStateActionTrace}$
13: **if** $\textsc{isGoalState}(\boldsymbol{s})$ **then**
14: **return** $\zeta, \iota^{\mathrm{H}}$
15: **end if**
16: **end while**
17: $\tilde{\zeta} \leftarrow \textsc{SelectStatesForLearning}(\zeta)$
18: $T^{\mathrm{H}}(\boldsymbol{s}' \mid \boldsymbol{s}, \iota, a^{\mathrm{H}}) = T(\boldsymbol{s}' \mid \boldsymbol{s}, \pi^{\mathrm{R}}(\boldsymbol{s}, \iota), a^{\mathrm{H}})$
19: $\boldsymbol{\theta} \leftarrow \mathrm{IRL}(\tilde{\zeta}_\iota, T^{\mathrm{H}}(\iota^{\mathrm{H}}), \boldsymbol{\theta})$
20: $R^{\mathrm{H}}(\boldsymbol{s}, \iota) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{s}, \iota)$
21: $Q^{\mathrm{H}}(\boldsymbol{s}, \iota, a^{\mathrm{H}}) \leftarrow \textsc{QIter}(T^{\mathrm{H}}, R^{\mathrm{H}})$
22: $\hat{\pi}^{\mathrm{H}}(\boldsymbol{s}, \iota) \leftarrow \textsc{SoftMax}(Q^{\mathrm{H}})$
23: **end while**

4 Implementation

We test our method in two different scenarios in which a human and a robot cooperatively need to move an object from one support to another. The robot knows where the supports are, but not which one the human intends to move to. This section describes how we model such scenarios as an (MO)MDP for learning. In the final subsection (Sec. 4.6), we describe the baselines we compare our learning method to.

4.1 States

The physical states $\boldsymbol{s}$ are defined from the perspective of the manipulated object, defining its position $\boldsymbol{p}$, orientation $\boldsymbol{q}$, and affordance (Koppula et al, 2016) —in our case its manipulability μ: $\boldsymbol{s} = \begin{bmatrix} \boldsymbol{p}^T & \boldsymbol{q}^T & \mu \end{bmatrix}^T$. We consider positions in 3D (x, y, z). The orientation $\boldsymbol{q}$ can be a single angle or a quaternion. The manipulability defines how the object may be moved depending on by whom it is held (e.g., the object may only be moved if it is held by both human and robot). Concretely, μ is an integer encoding who is holding the object.

The object must be held by both human and robot anytime it is not resting on a support, which may be anything that will keep the object in a stable position without the help of an actor. Each support provides a possible start or goal state, with a specific object position and orientation. The human may intend to put the object on any of these supports.

In between these supports, a small number of strategically chosen waypoints define key locations in space. Examples are the position from which to mount the object onto a support —which is assumed to be the same as the position to which the object can be unmounted— or a position below such a "mounting point" at a height which is comfortable for the human to carry the object. The space in between waypoints is assumed to be free of obstacles.

Next to the physical state $\boldsymbol{s}$, there is the human intention ι encoding the desired goal to put down the object. This may be any of the available supports. The robot has no direct access to this variable. The initial intention estimate is set to zero at the initial support and distributed uniformly for the others.

4.2 Actions

The general set of high-level actions and their pre-conditions are listed in Table 1. It depends on the state which actions are allowed. In free space, where the object is only supported by the robot and the human, neither is allowed to let go. Rotation is allowed around a single axis at a time. Movement between waypoints is allowed either horizontally *or* vertically, along straight-line trajectories. We made this choice purely for demonstration purposes, to define easily distinguishable possible preferences while keeping the state space small. From and to a support, the motion is defined based on the geometry of the object and the support.

Figure 3 illustrates what the actions look like in the clothes hanger scenario. As long as the hanger is hanging at one of the supports (illustrated at B), the robot and the human can change their grasp on the object (a_1, a_2), each on their own side. If the hanger is held by both actors, it can be taken off the support (a_3). The hanger

Table 1 Actions and their necessary state conditions. The actions are illustrated in Fig. 3.

Action	State pre-conditions
a_0 wait/passive	
a_1 grasp	object resting, not held by actor
a_2 let go	object resting, held by actor
a_3 take off (support)	object resting, held by both actors
a_4 put on (support)	object at *mounting point*[1]
a_5 rotate[2]	object held in free space
a_6 move over[3]	object held in free space
a_7 move up/down[4]	object held in free space

[1] next to support, oriented correctly

[2] around a single axis

[3] to waypoint at same height

[4] to waypoint at same (x, y)-coordinate

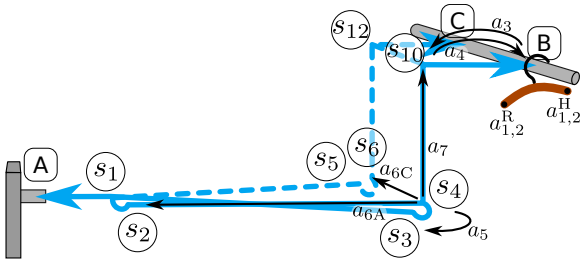


Fig. 3 Schematic overview of the different actions connecting a subset of states as defined on one preference in the clothes hanger scenario (Fig. 1a). States s_1 and s_2 are at the same position $\boldsymbol{p}$, but have a different orientation $\boldsymbol{q}$. The same holds for s_3 and s_4, and s_5 and s_6. The different states at each support differ only in μ. The actions are numbered as defined in Tab. 1. From s_4, a_6 can be taken in two different directions, towards A and C respectively (a_6 towards C is not part of the shown preference). Similarly, a_7 can be defined towards multiple different heights, as is the case in the wheel scenario. The clothes hanger scenario considers only two different heights.

can only be put on the support from the adjacent state if the orientation matches —the hanger can be mounted onto A from s_1, but not from s_2. In any state in free space $(s_1, s_2, \ldots, s_{12})$, actions a_5, a_6, a_7 are possible, as illustrated in s_4.

In simulation, we only consider the discrete states connected by the abstract actions. On the hardware, the way the robot grasps and lets go of the object is pre-programmed. The other actions are defined along straight-line trajectories, either in linear or in rotational space, and tracked applying disagreement-aware variable impedance (DAVI) control (Van der Spaa et al, 2022). In order to track the straight-line trajectories between our states in a robust way with our 7 DoF robot arm, we extend the DAVI controller with the following null-space component: We train a Gaussian Process (GP) (Williams and Rasmussen, 2006) on a small set of feasible arm configurations (one per gripper pose) to obtain an approximate inverse kinematics (IK) mapping. During actions, we control both the gripper position and orientation (6 DoF) and the joint configuration (7 DoF), one set of DoF with lower impedance than the other to resolve the redundant control. Close to state positions and orientations, the Cartesian impedance on the gripper dominates the joint impedance, to make sure the robot reaches the state. As the distance to the known states increases, so does the impedance on the joints, while the Cartesian impedance on the gripper is reduced. This way we smoothly bend our straight-line trajectories a little bit to avoid joint limits and allow the elbow of the robot arm to change side when necessary.

The robot always has the option not to act. In this "passive mode", the robot just compensates the gravity with zero stiffness and the human is free to drag the robot around with the object.

4.3 Transitions

The abstract physics of the problem, considered by the internal model, are simple: an object can be moved if both actors have a grasp on it. A robot action may either have the desired effect or no effect at all, if the human counteracts the action. The DAVI controller ensures smooth transitioning from active to passive in case of counteractive action, so the human is always in control of where the object is moved to.

If the robot is in passive mode, the human fully determines the transition. The human can also choose to passively follow the robot. Human partners are instructed only to do actions which end up in a valid next state.

4.4 Observations

In simulation, the abstract states and actions are observed directly. In the robot experiment, the closest abstract state is considered to be the state arrived at. This state is used to infer the action the human took to realize the transition, and to estimate the next abstract human action and choose a matching abstract robot action. The corresponding motion trajectory is always planned from the actual robot position and orientation, not from the abstract one the robot is expected to be at.

At the supports, there is the additional bound that the actual position should be close to the expected one. Furthermore, the velocity and interaction forces must be close to zero before letting go of the object is allowed, assuming the human will stop trying to move the object when it is stably supported, and using that as a sign that it is safe for the robot to let go.

In the experiment, the robot does not directly observe the human grasp on the object. Instead, we generally assume the human is holding the object, until the object has been non-moving for a number of seconds. We instruct our users to keep a hold on their end of the object when the robot is active on the task, and make sure not to activate the robot when they have not, so that the assumption holds.

4.5 Rewards

As described in Sec. 3, we have separate reward functions for the human and the robot. The human reward, $R^{\mathrm{H}}(\boldsymbol{s}, \iota)$, is defined by the learned preference model (Sec. 3.3), which is a function of features describing each state's relation to the start, the intended goal, and unintended alternative goal candidates. The features in the human preference model are the product of two Gaussian Radial Basis Functions (RBFs) and a binary component. The first set of RBFs are a measure of linear distance and are centered at points defined relative to the intended goal support, another support, or the world (e.g., comfortable carrying

height). These points cover the waypoints, but multiple waypoints relative to different supports map to the same feature point if the supports are not the intended goal. The second set of RBFs are a measure of angular distance and are centered at the allowed absolute orientations and at the final intended orientation. The standard deviations of the RBFs are chosen at 2 cm and 10° respectively. The binary component indicates the manipulability. With our choice of waypoints, this results in a total of 26 features. The feature vectors are normalized per state, $\boldsymbol{\theta}$ is scaled to $-1 \leq \boldsymbol{\theta}_i \leq 1$, and initialized at 0.1 at the intended support, -1 at the other supports, and 0 elsewhere.

The reward the robot receives for state transitions, $R^{\mathrm{R}}(\boldsymbol{s}, a^{\mathrm{R}}, a^{\mathrm{H}}, \boldsymbol{s}')$, punishes actions which are counteracted by the human, or do not change the state, by $r^- = -1$. Passive behavior, when a supportive alternative exists, is punished less severely, by $r^0 \in (-1, 0)$. The reward factor r^0 is deciding for the robot behavior. A smaller magnitude makes passive robot behavior more desirable as, relatively, the punishment for choosing a wrong action increases: the robot is "more afraid" of taking a wrong action. If $r^0 = r^-$, the robot does not care about taking wrong actions and the benefit of learning an internal model disappears. The effect of different values of r^0 is evaluated in Sec. 7.

4.6 Baseline Agents

Both in simulation and the real-world experiment, we compare our *Learner* agent to 1) an *Imitator*, 2) a *Passive* agent, and 3) a *plain ME-IRL* agent.

The Passive agent (Algorithm 2) is hard-coded to grasp at the start (L. 2–3) and let go when the object rests at a different support (L. 4–5). In between it just compensates the gravity, i.e., is in "passive mode" (L. 7). This passive policy is also the internal baseline the robot compares its actions to when computing its policy.

The Imitator agent (Algorithm 3) follows the passive policy in states where it has yet to observe a human action (L. 5b). Otherwise, it takes the action it has observed most recently when coming from the same start support (L. 5a), stored in the observed-state-action set after each episode Ξ (L. 14). This can capture most of the preference, but because it has no notion of intention, it will always have a chance of $\frac{n_\iota - 1}{n_\iota}$ —with n_ι the number of possible intentions— of choosing wrongly in the deciding state. If the human decides to return the object to the start support, the Imitator will not understand and keep trying to move elsewhere (if it observed an action in that state before, coming from the same start support). The way we defined the Imitator, allowing the start support to be also a goal support would mean that the robot will not hold on to the object to let the human move it away from the start, as it does not have an internal model to consider that option.

The plain ME-IRL agent (Algorithm 4) learns its policy applying plain ME-IRL (Ziebart et al, 2008) on the observed trajectories to learn its reward function, still incrementally updating only once after each episode. It has no explicit internal model of separate human rewards or a human policy. The robot does not try to optimize a cooperation reward as described in Sec. 4.5. Instead, it

Algorithm 2 Coded passive behavior

1: **procedure** BEPASSIVE($\boldsymbol{s}_0, \boldsymbol{s}$)
2: **if** $\boldsymbol{s} = \boldsymbol{s}_0$ **then**
3: **return** $a = \mathit{grasp}$
4: **else if** $\begin{bmatrix} \boldsymbol{p}^T & \boldsymbol{q}^T \end{bmatrix}^T \in \mathcal{S}_{\text{support}} \setminus \{\boldsymbol{s}_0\}$ **and** grasped by robot $\in \mu$ **then**
5: **return** $a = \mathit{let\ go}$
6: **else**
7: **return** $a = \mathit{wait}$
8: **end if**
9: **end procedure**

Algorithm 3 Baseline agent: Imitator

1: $\Xi \leftarrow \emptyset$
2: **while** True **do**
3: $\boldsymbol{s}_0, \zeta \leftarrow \emptyset$
4: **while** Collaborative Episode **do**
5: $a_t^{\mathrm{R}} = \begin{cases} \Xi(\boldsymbol{s}_0, \boldsymbol{s}_t), & (\boldsymbol{s}_0, \boldsymbol{s}_t) \in \Xi \\ \text{BEPASSIVE}(\boldsymbol{s}_0, \boldsymbol{s}_t), & (\boldsymbol{s}_0, \boldsymbol{s}_t) \notin \Xi \end{cases}$
6: $\boldsymbol{s}_{t+1}, a_t^{\mathrm{H}}, a_t^{\mathrm{R}} \leftarrow \text{DOACTION}(a_t^{\mathrm{R}})$
7: $\zeta \leftarrow \text{UPDATESTATEACTIONTRACE}$
8: **if** ISGOALSTATE($\boldsymbol{s}$) **then**
9: **return** ζ
10: **end if**
11: **end while**
12: $\tilde{\zeta} \leftarrow \text{SELECTSTATESFORLEARNING}(\zeta)$
13: **for** $(\boldsymbol{s}_t, a_t^{\mathrm{H}}) \in \tilde{\zeta}$ **do**
14: $\Xi \leftarrow \Xi \cup (\boldsymbol{s}_0, \boldsymbol{s}_t : a_t^{\mathrm{H}})$
15: **end for**
16: **end while**

Algorithm 4 Baseline agent: plain ME-IRL

Require: $T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R}}, a^{\mathrm{H}})$, $\boldsymbol{\phi}(\boldsymbol{s}, \iota)$
1: $T(\boldsymbol{s}' \mid \boldsymbol{s}, a) = T(\boldsymbol{s}' \mid \boldsymbol{s}, a, a)$
2: $\boldsymbol{\theta} = \boldsymbol{\theta}^0$, $R(\boldsymbol{s}, \iota) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{s}, \iota)$
3: $\Lambda(\iota | \boldsymbol{s}_0, \boldsymbol{s}) \leftarrow \mathcal{U}(\iota)\ \forall \boldsymbol{s}_0 \in \mathcal{S}_{\text{support}}, \forall \boldsymbol{s} \in \mathcal{S}$
4: **while** True **do**
5: $Q_\iota(\boldsymbol{s}, a) \leftarrow \textsc{QIter}(T, R)\ \forall \iota \in \mathcal{I}$
6: $\boldsymbol{s}_0$, $b_0(\iota) \leftarrow \mathcal{U}(\iota)$, $\zeta \leftarrow \emptyset$
7: **while** Collaborative Episode **do**
8: $Q(\boldsymbol{s}_t, a) = \sum_\iota b(\iota) Q_\iota(\boldsymbol{s}_t, a)$
9: $\pi^{\mathrm{R}} \leftarrow \textsc{BoundedSoftMax}(Q)$
10: $\boldsymbol{s}_{t+1}, a_t^{\mathrm{H}}, a_t^{\mathrm{R}} \leftarrow \textsc{DoAction}(\pi^{\mathrm{R}})$
11: $b_{t+1}(\iota) \leftarrow \Lambda(\boldsymbol{s}_0, \boldsymbol{s}_{t+1})$
12: $\zeta \leftarrow \textsc{UpdateStateActionTrace}$
13: **if** $\textsc{isGoalState}(\boldsymbol{s})$ **then**
14: **return** ζ, ι
15: **end if**
16: **end while**
17: $\tilde{\zeta} \leftarrow \textsc{SelectStatesForLearning}(\zeta)$
18: $\boldsymbol{\theta} \leftarrow \mathrm{IRL}(\tilde{\zeta}, T, \boldsymbol{\theta})$
19: $R(\boldsymbol{s}, \iota) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{s}, \iota)$
20: $\Lambda(\iota | \boldsymbol{s}_0, \boldsymbol{s}) \leftarrow \textsc{UpdateLikelihood}(\boldsymbol{s}_0, \tilde{\zeta}, \iota)$
21: **end while**

adopts what the Learner learns as R^{H} (Sec. 4.5) as its own and only $R(\boldsymbol{s}, \iota) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{s}, \iota)$ (L. 2, 18–19). This is the same basic principle as used in Losey et al (2022), which we consider to be the closest recent related work, as they also aim to learn a model of human objectives/preferences from pHRI. However, we do not adopt their adaptations for online learning during episodes. We follow classic ME-IRL in updating the model after observing a full episode, as we cannot observe the human intention (where the human wanted to go) before observing the final state of the episode. We initialize the feature weight $\boldsymbol{\theta}^0$ as for the Learner and use the same intention-parameterized features $\boldsymbol{\phi}(\boldsymbol{s}, \iota)$.

Since plain ME-IRL does not allow for a second independent actor, we let the agent assume maximum cooperation —i.e., both actors should take the same (or matching) actions— and use $T(\boldsymbol{s}' \mid \boldsymbol{s}, a) = T(\boldsymbol{s}' \mid \boldsymbol{s}, a^{\mathrm{R}} = a, a^{\mathrm{H}} = a)$, which no longer depends on the intention ι and therefore can be computed outside the learning loop (L. 1). Because both in reality and simulation the human actions override the robot actions, this should result in the desired states for each intention to be updated correctly in $R(\boldsymbol{s}, \iota)$ and the agent to plan matching cooperative actions to pass through those states. We use the same soft-max action selection as described for the robot in Sec. 3.1 (L. 5, 8–9).

By lack of a human policy model, the plain ME-IRL agent obtains an intention belief by taking the maximum likelihood estimate based on how often each intention occurred in the current state given the start state $\Lambda(\iota | \boldsymbol{s}_0, \boldsymbol{s})$ (L. 3, 11, 20). A more intelligent estimate would improve the agent's behavior. However, designing such an intention estimate is not the topic of this paper. It could be interesting for future work.

5 Scenarios

We test our learning algorithm in two different cooperative scenarios. The scenario of moving a clothes hanger (Fig. 1a) has a state space that allows human users a number of different preferences while moving a clothes hanger between three possible supports (intentions). We designed this scenario such that we could run it on a Franka Emika Panda robot arm, to test our algorithm in a user study.

The scenario of moving a wheel between stands (Fig. 1b) has a larger state and action space, that allows human preferences to include seemingly inefficient detours. In this scenario, we test our algorithm only in simulation. In simulation, we can also easily test the generalization to cases where the stands change position and height.

In both scenarios, we use the following learning parameters: For the iterative IRL, a learning rate $\lambda = 0.1$ is used. The robot action exploration is restricted by a soft-max temperature $\tau^{\mathrm{R}} = 5$, and, for the robot, with an additional bound $\eta = 0.9$. The human is assumed to explore even less, $\tau^{\mathrm{H}} = 25$. The intention bias is chosen at $\beta = 0.95$.

The plain ME-IRL baseline agent uses the same parameter values for action selection as the robot model in the Learner. The IRL learning rate is set at $\lambda = 0.03$ as a higher learning rate was found to destabilize the agent's learning.

5.1 Clothes Hanger Scenario

In the "Hanger Scenario", we use a quaternion to define the object(=hanger) orientation. We consider just a single rotation, around the vertical axis. There is no reason to not hold the hanger

with the hook on top, but the peg (A) (Fig. 1a) we can hang it on is oriented differently than the rail on which we have our support points B and C.

Supports B and C are at the same height, support A is considerably lower. To each of the supports, there is a mounting point (s_1, s_{10}, s_{12} in Fig. 3), a bit over a hanger 'radius' away in 'unhooking' direction, so that the hanger is sufficiently clear to be rotated. In between these mounting points, we define additional waypoints in space by recombining their (x, y) and z positions. With only the two distinct heights, there are 24 states and between 2 and 6 actions per state, including not acting.

We set $r^0 = -0.33$, which gives us balanced behavior: reasonably careful not to take wrong actions, yet not too afraid to act. For learning, we use discount factor $\gamma = 0.9$, for both Learner and plain ME-IRL agent.

5.2 Wheel Scenario

In the "Wheel Scenario", we describe the object(=wheel) orientation by a single angle, around the axis pointing from the robot to the human. The wheel can hang 'vertically' on the rack, or be placed 'horizontally' on one of two stands (Fig. 1b). The affordance μ, whether the robot and the human have a grasp on the wheel, is considered to be directly observable.

The rack and the two stands are all at different heights, respectively at 1.7, 1.0, and 1.2 m. Every episode, we initialize the positions of the stands at random, at a distance between 1.6 and 3.6 m from the rack and between 1.0 and 2.2 m from each other. Intermediate waypoints are defined as in the *hanger scenario*. Additionally, we define a "comfortable carrying height", at 0.95 m, below each mounting position. When working with real people, this height should be adjusted according to how tall the user is. If a point in space would collide with a stand, the point is projected in negative x-direction by a bit over a wheel radius distance.

Moving up or down to different heights, are all separate actions. At each height, there is the possibility to move over towards each of the other supports. All actions, of both robot and human, are assumed to be directly observable by the robot. In total, there are 36 states and up to 8 possible actions per state.

We test different r^0. To better allow our human model to capture detour preferences, and the robot model to support it, we lower our discount factor to $\gamma = 0.6$ (for both Learner and plain ME-IRL agent). As the human model is learned from demonstrations that reach a goal, and the robot receives punishment for not supporting the human, the learned policies still terminate releasing the wheel at a support, despite the low discount factor.

6 User Study: Clothes Hanger Scenario

6.1 Experiment

We did a user study with a Franka Emika Panda robot arm and 24 users (16 male, 8 female) of an age between 19 and 77 years old, with the median at 28 and the interquartile range between 25 and 35. Five of the participants had participated before in a user study involving a similar robot arm; one participant had multiple years experience with collaborative robot arms including the Franka Emika Panda, although not in a setting that involved physical interaction; one other participant had experience programming industrial robot arms; and there was one participant with experience with physical human-robot collaboration in terms of lane-keeping assistance.

The hanger scenario was explained to the participants, including that the robot would never be given the information of where the users were asked to hang the hanger next (i.e., the intention). The users were informed that the robot could perform only a few distinct actions between the supports and six distinct points in the intermediate space.

All participants went through the same familiarization phase in which they first moved the hanger around with the robot in passive/gravity compensation mode. Next, the robot would play a pre-programmed sequence of actions, letting the human follow and feel how it feels when the robot is maximally assistive. Then, the users were asked to follow the same sequence of motions they had observed the robot to lead previously, but this time with the robot trying to move elsewhere in each of the decision points in space. This way, the users would get comfortable disagreeing with the robot in case it would not follow their preference

Table 2 Experimental episodes

Nr.	from, to	remarks
1	B, A	initial behavior
2	A, B	
3	B, A	for the second time
4	A, C	starting from A with different intention
5	C, B	new region in the state space
6	B, C	starting from B with different intention
7	C, A	starting from C with different intention
8	A,(B)A	changing intention: turning back halfway
9	A, B	starting from A like in episode 2

or intention. The participants could try each of the 'modes' until they felt comfortable with whatever the robot would do during the actual experiment.

Now that the users felt somewhat familiar with the task and the robot, they were asked to specify their preference, segmenting the movement to the lower and rotated support in {moving over horizontally, moving down, rotating} in the order of their choice, as well as the way back. During the remainder of the experiment, they were instructed to stick as closely to this preference as they could manage, no matter what the robot would do.

The actual experiment then consisted of moving the hanger nine times to a next hanging point (as listed in Tab. 2), while the robot would update its internal model in between. The whole sequence took between 2.5 and 4 minutes. This was done once with the robot applying the proposed IRL method, and once running an imitator baseline. Half of the users experienced the Learner first, half of them the baseline, approximately alternating between participants, to average out the learning effect of the users. After each set of learning episodes, the users filled out a questionnaire on what they felt about the robot learning (on a 7-point Likert scale), and the NASA TLX questionnaire to assess their personal experience. A demonstration of the experiment can be found here: https://youtu.be/k-JYV4hyTs8.

The experiments were carried out in accordance with the guidelines and regulations of the lab and the equipment. All participants signed their written informed consent before participating. All collected data was anonymized before storage.

6.2 Hypotheses

The Learner tries right from the start to be of assistance. When in doubt of the user preference or intention, it does not act, as the punishment is less for letting the human lead than for choosing a wrong action, such that the waiting "action" has largest expected reward. However, once close to a support with little choice of actions left, it acts without needing to observe the human first. Once it has observed previous roll-outs of the task, the parameterized internal model tries to generalize the learned preferences across the possible intentions. So once a side of the rack appears to be chosen, the robot may provide assistance without before having observed the human move in that direction. However, the awareness of multiple possible intentions, with our choice of r^0, leads to there always being one state in which the robot leaves the initiative to the human, necessary to observe/predict the human's intention.

The Imitator does not do anything until the task starting from a support has been observed at least once. Then it copies what it observed the previous time. When entering a new region in the state space (coming from a specific support), it stays again passive. Without a notion of intentions, once starting the task from a support observed previously, the robot never hesitates to act. It provides maximum support if the human wants to go the same way. If not, the human has to 'fight' the robot, make it understand the desired action goes elsewhere. This will be the case in the state where the human chooses to take the turn to another support, and also in the case the human moves back to the start support. The Imitator is by design not capable of understanding the start support as goal support (Sec. 4.6).

Based on these differences, we expect the Learner to be overall more supportive in the sense of taking the right action at the right time and being less passive in tasks and states that were not observed before. We formulate the following hypotheses (w.r.t. the Imitator baseline):

H1. The Learner will be better able to support the human preferences and intention.

As a result, we expect:

H2. The Learner makes the task easier for the human, in terms of reducing both physical and perceived effort.

Furthermore, we test if:

H3. The user feels more comfortable when cooperating with Learner.

We test ***H1*** objectively by comparing the relative number of actions the robot initiated both correctly and wrongly. A large percentage of correct actions indicates a match of preference, while a mismatch of intention increases the number of wrong actions. Subjectively, we compare the questionnaire results on perceived understanding of preferences and intentions, learning speed, and trust.

To test ***H2*** objectively, we compare the force and torque exerted on the robot integrated over the duration of the task. For subjective evaluation, users graded how easy they felt the robot made the task, next to filling out the NASA TLX questionnaire. Additionally, the questionnaires allow us to evaluate ***H3***.

Next to these hypotheses, we will qualitatively check the convergence of the learned policy.

6.3 Results

Fig. 4 shows the percentage of 'correct', 'passive', and 'wrong' abstract actions taken by the robot, lightly colored for the Imitator and darker colored for our Learner. For the plain ME-IRL agent, we use the state sequences observed with the Learner, which are most clean of the influence of wrongly initiated robot actions, and compare the actions the plain ME-IRL agent would have taken. The results are shown in gray.

Actions are considered 'correct' if the next recognized proximal state corresponds to the state the robot started acting towards in the previous state. This means an action is registered as correct even when in between disagreement was detected and the robot aborted its action. Considerable 'false disagreements' were detected when users found the robot too slow or pulled the robot with some force to the same next state but not via the straight line the robot tried to track. On the other hand, users occasionally disagreed close enough to the state the robot was acting towards to have the action registered as 'correct' before moving on to where they wanted to go. In those cases, the wrong action taken is registered as a 'correct' plus a passive action. Because of this effect, we expect the number of wrong actions for the plain ME-IRL agent to register slightly lower if they were recorded with the actual users.

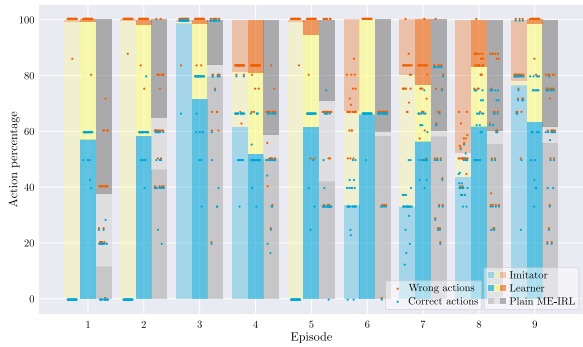


Fig. 4 Percentages of different actions taken by the Imitator, Learner, and plain ME-IRL agent, for the nine episodes of moving the hanger as tabulated in Tab. 2. The bars show the average percentage of correct actions taken (number of times when the next state recognized by the robot coincided with the state the robot decided to act towards in the previous state) in shades of blue and medium gray. The middle light color denotes the percentage of actions when the robot remained passive. The wrong actions (times when the next state did not match the initiated action) are shown in red/dark gray. To indicate the spread of the data, the dots represent the individual data points, where the wrong actions in red are counted from above.

Episode 3 is the episode in which pure imitation should give the optimal result (depending on the quality of the demonstration in Ep. 1). It is the only episode in which the intention matches the previous episode starting from the same start state. Indeed, we observe for this one (and only) episode that the Imitator outperforms the Learner. We see that the plain ME-IRL agent overfits considerably on the policy it thinks best. In many individual cases (dots in the figure), it does as well as the imitator in Ep. 3. In most episodes, it chooses more correct actions than the imitator, and in all except the first episode, its percentage of correct actions is closer to the Learner. However, as its learned model covers the full state space, already at initialization, it chooses more wrong actions than the Imitator in all episodes except Ep. 8, where the intention is changed. In Ep. 9, the Imitator has full state information, but no recent observation of the specific intention. In Ep. 4, many users moved quite close to support B before moving over to C. This resulted in the Imitator's action going to B being registered as correct, while the Learner waited to observe the intention, and then taking one wrong action believing the user might intend to go to B. In all the other episodes, we see the Learner take at

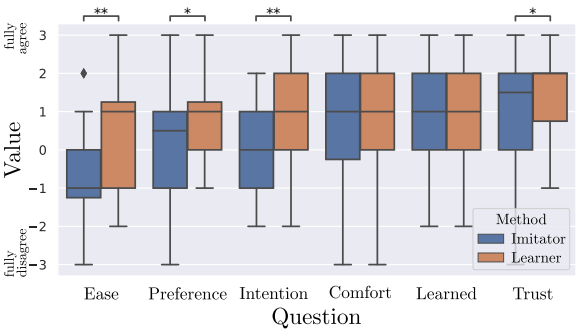


Fig. 5 Questionnaire results to statements from left to right: The robot made it *easier* for me to perform the task. The robot understood my *preference*s, how I wanted to do the task. The robot was supporting me to go where I wanted to go (*intention*). I was *comfort*able with what the robot was doing. The robot *learned* fast. I *trust*ed the robot. Significant differences between the methods are indicated by * ($p < 0.05$) and ** ($p < 0.01$).

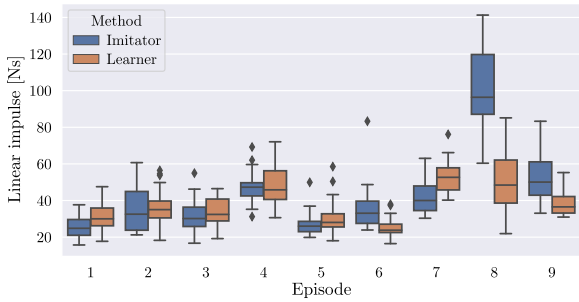


Fig. 6 Linear impulse (interaction forces integrated over time) with the Imitator and Learner for the nine episodes of moving the hanger as tabulated in Tab. 2.

Table 3 Results of the questionnaires for the Learner and Imitator compared with a one-tailed paired t-test. The statements are phrased for the Learner w.r.t. the Imitator, as perceived by the users. P-values < 0.05 are printed bold, indicating a significant result. Values between parentheses indicate the answers to a question were not normally distributed (with $p < 0.1$) for one or both of the methods.

Statement	p-Value
Robot made task *easier*	**0.006**
Robot understood *preferences*	**0.035**
Robot supported the user *intention*	**0.007**
User was *comfort*able with robot	(0.068)
User thought robot *learned* fast	0.156
User *trust*ed robot	**0.031**
Lower *mental demand*	0.760
Lower *physical demand*	0.143
Lower *temporal demand*	(0.966)
Higher *performance*	**0.036**
Lower *effort*	(0.086)
Lower *frustration*	(0.087)

least as many, and often considerably more, correct actions. The plain ME-IRL agent is seen to generalize its observations less well, as it chooses fewer correct actions in most episodes. Over all the episodes, the Learner takes as few or fewer wrong actions compared to both the imitator and the plain ME-IRL agent. These results support ***H1***.

We see ***H1*** further supported by how the users graded different aspects of the robot performance (Fig. 5). The results for the two methods are compared using a one-tailed paired t-test, testing if the Learner was perceived as a significant improvement over the Imitator. The p-values are tabulated in the top half of Tab. 3. Significant differences are indicated by * and ** in the figure.

The most significant results are: the users 1) found the task easier to perform with the Learner compared to the Imitator, and 2) felt their intentions were better understood by the Learner. Additionally, with $0.01 < p < 0.05$, the users also felt their preferences were better understood, and they trusted the Learner more than the Imitator.

As an objective measure of effort, we consider the forces and torques integrated over the duration of the tasks. The duration is measured from the moment the robot starts grasping until the robot has let go at the intended goal state. Since the Imitator never let go between Episodes 8 and 9, these episodes are separated manually. Time in which the robot lost grasp on the hanger and was not moving is subtracted. As the trend in the resulting linear and angular impulse look very similar, we show only the linear impulse in Fig. 6.

In general, comparing Figs. 6 and 4, we see that the registered impulse increased when the robot was more active, regardless of the quality of the actions taken, with the exception for Episode 3. This lack of support for ***H2*** may be largely due to the preference mismatch on the action level. People generally found the straight-line trajectories unnatural, and several users seemed to prefer the robot to go faster. The presented method focuses on preferences on the level of discrete states, extending it to additionally learn preferences on how to transition between those states (Avaei et al, 2023), will likely lead to improvement on this result.

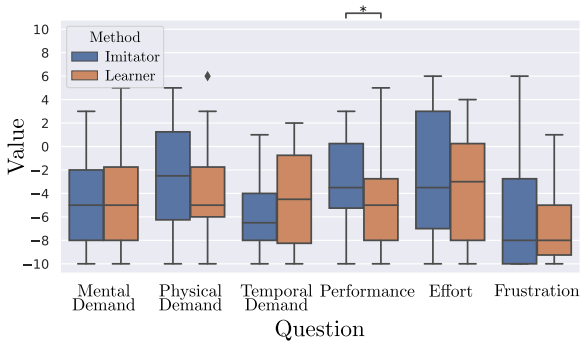


Fig. 7 Results of the NASA TLX questionnaire. A lower score is better. Significant differences between the methods are indicated by * ($p < 0.05$).

Subjectively, when explicitly questioned about the effort and demand of the task (Fig. 7, Tab. 3), the users did not grade the methods significantly different. However, they did feel they performed the task better with the Learner compared to the Imitator. Furthermore, the users very significantly found the task easier to perform with the Learner compared to the Imitator (Fig. 5, Tab. 3). This does provide some weak support to ***H2***.

People did not report a significant increase in comfort with the Learner, but they did trust the robot more and found it easier to cooperate with. We can interpret this as a weak support to ***H3***.

To check if this trust is well placed, we have a look at the policy the robot learned. We need to look at this per preference, as the policy the robot learned is preference specific. Since our users were free to choose their preferences, we have more data on some preferences than on others. Our users chose 9 different preferences in total. To get the best impression of the variance between the users and how well the robot was able to learn, we look at the most frequently chosen preference, which is marked in Fig. 1a by the blue lines and shown again in Fig. 3. Important states, in which different actions can be chosen, resulting in a different preference, are marked in Fig. 3 by s_{10} and s_4 for the intentions to go from support B to A, and by s_1 and s_3 from A to B.

Fig. 8 shows the learned action probabilities in those four critical states to those two intentions for the preference chosen by most users: From the rack to the peg (intention A): first move down (top left), then move over (top right), and finally rotate before hanging; and on the way back (intention B): first move over (bottom left), then rotate (bottom right), and finally move up before hanging the hanger on the rack. The actions shown in the figure are the actions defining the preference. The lines show the likelihood of the robot choosing the correct action compared to not taking an action (the dash-dotted line at 1.0). In blue, we show the expected relative action probability obtained from 100 simulations where the start and goal supports are chosen at random every episode, but the human follows the said preference perfectly.

From the simulation results, we see that our learner is able to capture some action preferences slower than others. This is due to the feature parameterization we chose. Nevertheless, in most of the critical states and with most of the users, the robot learns within a few episodes to recognize the correct action with a probability larger than the probability of staying passive.

We need to make a distinction here between the states on the left and on the right of Fig. 8. In states s_{10} and s_1, in Fig. 3, we see a dashed line, an alternative path, going to intention C. The action the human will take in these states depends very much on the intention, while the states visited up to these states gave no information of the intention. In states s_{10} and s_1, the Learner will not know where its partner wants to go. Not to accidentally choose a wrong action, we expect the Learner to learn to wait in these states. The small number of wrong actions shown in Fig. 4 indicates that this is indeed generally the case. It means that unless the Learner learns some really wrong behavior in these states, the final performance is not visibly affected by the preference model in these states. In these states we observe the largest effects of preference unlearning for certain intentions.

Specifically, we see the following four cases in Fig. 8 (from left to right, top to bottom): In s_{10} (or the equivalent state s_{12} when coming from C), the moving down action is only observed for intention A, in episodes 1, 3 and 7 (Tab. 2). In all other episodes, this preferred actions is slightly forgotten. This is also clearly illustrated by the wide blue band of the interquartile region of the simulated results. It suffices to move to A once in a while to keep the unlearning in check. Once moved down to the height of the goal, in s_4, moving over is quickly learned with little variance. This action is shared between all intentions as the next action

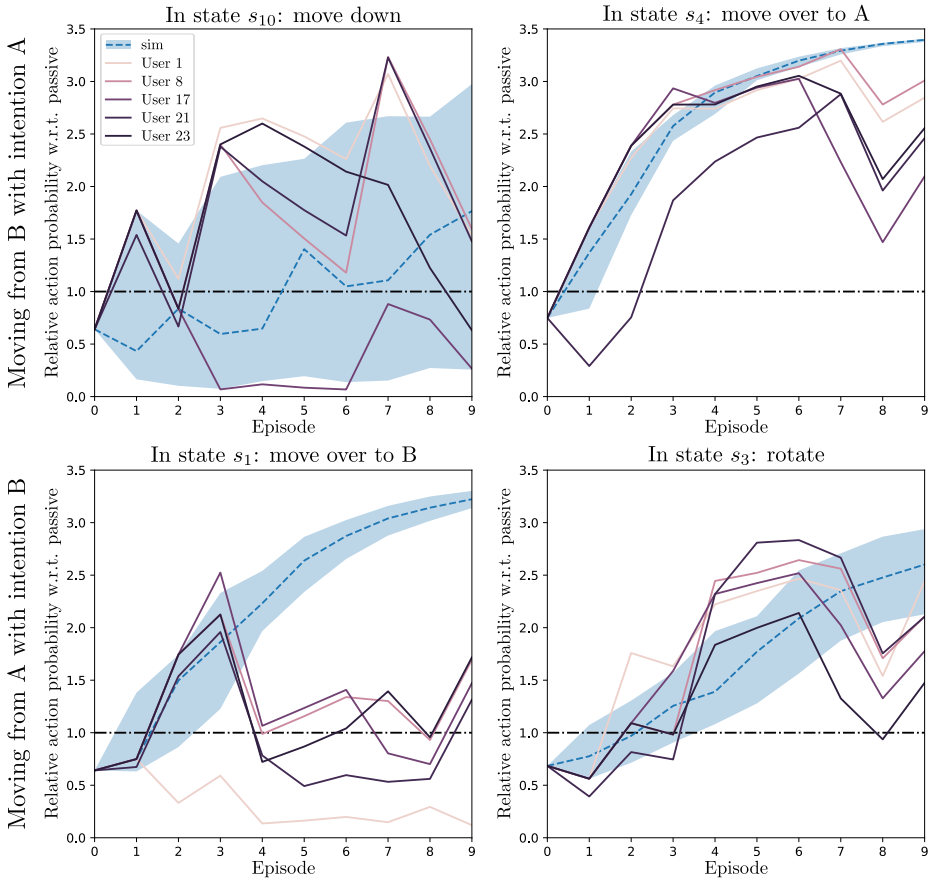


Fig. 8 The relative probability of the preferred action being chosen for a specific intention, in the four states defining the preference that was chosen by the largest number of participants, as it was learned over the episodes. The states s_{10}, s_4, s_1, s_3 are marked in Fig. 3. The solid lines show the data from the five individual users who had this preference. The blue area is the interquartile region of a 100 simulations that were run with random start and goal supports, the dashed line shows the median.

to take. In s_1 when going to C (Ep. 4), it turned out to be physically very hard to follow a more or less straight line to s_5. In all of the recorded cases, the users first moved to s_3 before continuing to s_5, confusing the robot, and unlearning that the human wants to move over directly in the direction of the intended goal. Our features could not capture the preference of choosing in state s_3 whether or not to continue on to C. In s_3, learning to rotate before moving up to the final intended height was somewhat harder to learn than the moving over to A in s_4. Mostly the episodes going to A were confusing here. Nevertheless, clear learning of this preference is observed.

Additionally, there is a large dip at Episode 8. There, the human changed intention halfway to go back. This option was not included in the simulations shown in blue. At runtime, we saw that the policy the robot learned is robust to such a change of intention. However, it did confuse the model in the learning update, as our model takes the final observed intention as the baseline intention for the entire episode.

7 Additional Simulation Study: Wheel Scenario

In this section, we demonstrate the effect of different choices of r^0, which we choose in Sec. 6 to maximize the learning effect, as well as the effect on the learning performance of people acting less as deterministic agents. Because we can test with a larger state and action space in simulation, we can now also investigate how well our Learner is able to capture "inefficient" preferences, visiting

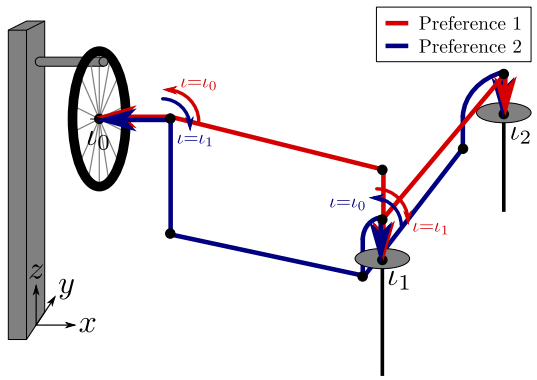


Fig. 9 Two preferences for moving the wheel between supports: 1. take the shortest path with the least changes of direction, rotating at the last moment when necessary; 2. rotate the wheel to horizontal at the first opportunity and move over at comfortable carrying height.

more intermediate states than strictly necessary. Also, we can easily move our supports around in the simulation to demonstrate that our state space parameterization lets our agent generalize between contexts, similar to Avaei et al (2023).

Both human and robot are simulated using the world model. The human can be controlled via a user interface, but for testing, we use pre-programmed human policies. Two human policies $\pi^{\mathrm{H}}(s, \iota)$ were provided to the simulator, characterizing the different preferences shown in Figure 9 (elaborating Fig. 1b). To these preferences, we can add a probability of the human being passive.

Of the different r^0 we tested, we present the results to the following values:

$r^0 = -0.25$ Low punishment – the robot will wait when unsure which action to take.

$r^0 = -0.50$ Medium punishment – the robot may try an action if it believes it could be better than waiting.

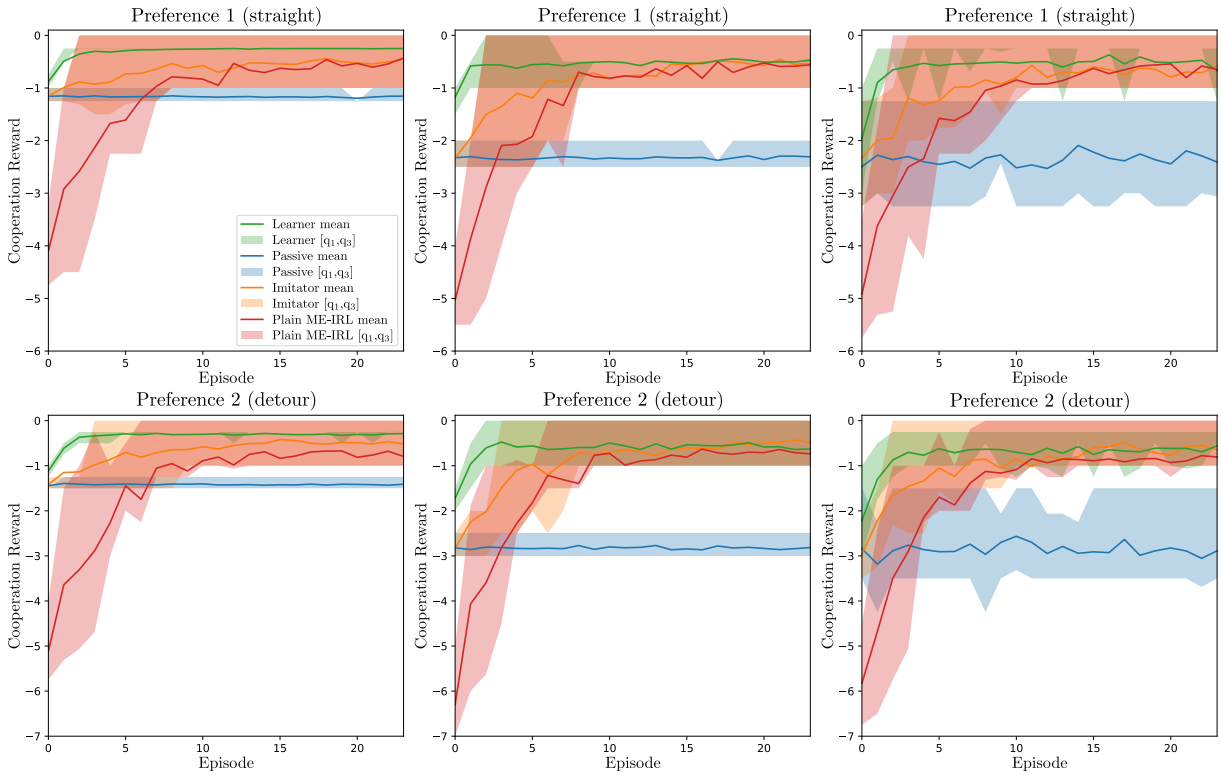
The start support and intention are chosen at random at the start of each episode. In each simulation, our Learner starts learning from an initial human model without initial preference (Sec. 4.5). The plain ME-IRL agent uses the same initial model, but as its own, as it does not have an explicit separate model for the human. Q-iteration on these models supplies the initial robot policy, which leads to non-passive initial robot behavior. The Imitator starts with an empty list of actions to imitate, making it start like the Passive agent.

Figures 10(a-b) show the mean and interquartile regions of the robot cooperation reward for a hundred simulations per preference, for $r^0 = -0.25$ and $r^0 = -0.5$ respectively. With a deterministic partner, we see our Learner converge within 4-8 episodes. The Imitator, after it has observed every combination of start and goal, settles down to take one wrong action with a 50% chance per episode: in the state where the human shows their intention. The plain ME-IRL agent takes somewhat longer to converge to the same result as the Imitator in the case Preference 1 and performs slightly worse for Preference 2, with the detour. Here, we do not consider the possibility of going back to the start support. Since moving between supports B and C requires one action less (the wheel does not need to be rotated), the passive policy shows an interquartile range corresponding to the one passive action difference.

For low waiting punishment, the learned robot policy converges to waiting only in the state where the human shows their choice of intended goal. For the longer route (Preference 2), this may take up to six episodes, for the shorter route, three episodes already suffice. This is really fast. For medium waiting punishment, the robot is less hesitant to take an action, even if it is not very certain it is correct, as long as it could provide a higher reward. For a higher waiting punishment, the Learner converges to a policy where, in the choice state, it randomly selects a goal, performing similarly to the Imitator. The optimal value for r^0 depends on the scenario, as well as on how careful or daring the human prefers their robot partner to behave.

For sufficiently low waiting punishment, the Learner outperforms the Imitator (and thus the plain ME-IRL agent). Depending on the objective (provide as much active support as possible or offer the least wrong support), the Imitator may approach the Learner's performance once it has seen which actions to imitate, if there are few enough possible partner intentions. Still, even if its average performance over time is very similar to that of the Imitator (in this case where the intention uncertainty is only between two goals), the Learner never performs worse than the baselines, always converges faster to its optimal performance, and in most cases converges to value with a considerably smaller variance in its cooperation reward.

The Imitator and plain ME-IRL agent show this variance because they lack a good model to



(a) $r^0 = -0.25$, deterministic human (b) $r^0 = -0.50$, deterministic human (c) $r^0 = -0.25$, 20% passive human

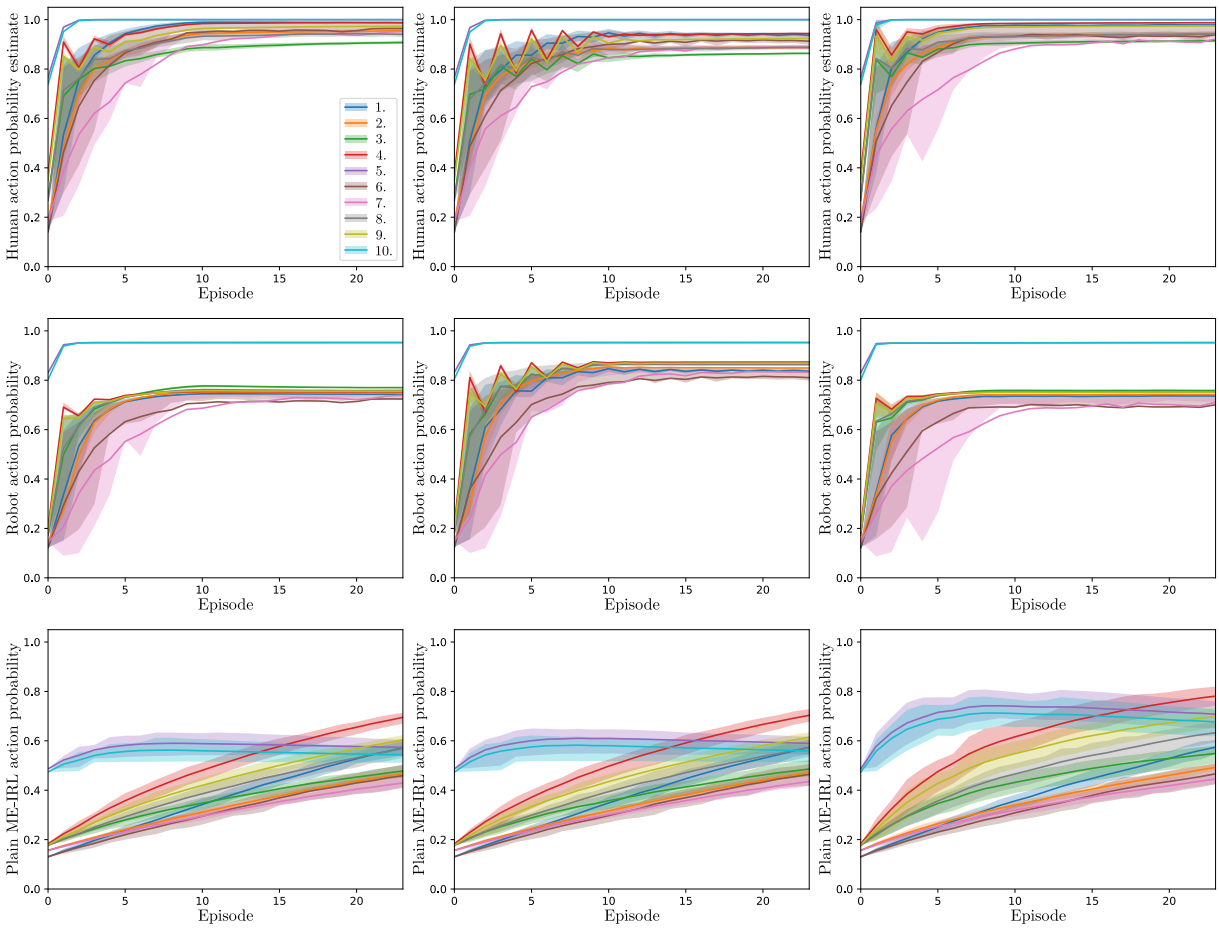
Fig. 10 Learning behavior compared, mean and interquartile regions comparing the Learner with the Imitator, Passive agent, and plain ME-IRL agent for the two different punishments for not acting (a,b), and (c) with a human who does not act for 20% of the time.

estimate the human intention and the uncertainty over it. Therefore, they cannot capture choice states in which the same action is not always right and where it may sometimes be better to not initiate an action. Furthermore, as is, only the Learner is able to cope in case the human changes intention. The Learner copes naturally even with the extreme case where the human changes intention to put the object back at the start support.

The Learner is also naturally able to cope well with cases where the human might start to rely on the robot once it has learned the preference to steer the large part of the trajectory and the human can follow passively. The Imitator could be programmed not to update its action table when the human is passive, but this would add another prior. The beauty of the proposed learning algorithm is that it does not need any prior and learns very fast nevertheless.

In Fig. 10, the only prior is the nominal passive policy which we used as a working baseline, but we obtained similar results without it, or when we initialize with a different preference. The plain ME-IRL agent is performing considerably worse than the Learner, as it does not consider the possibility for the human to take different actions than the robot. Furthermore, the intention estimate is very simplistic. Adding a more accurate human model and a more clever intention estimate will improve these results, but basically that is what the presented Learner does.

Figures 11 and 12 show the human policy estimate and the robot policy in terms of action probabilities that should be dominant in the states along the preferred trajectory, to each of the cases in Fig. 10, as well as the learned agent policy of the plain ME-IRL agent. For the Learner, we see that in every case, the human policy estimate converges to the same almost equally fast, even when the partner is partially passive. The plain ME-IRL agent's policies converge slower, and especially in case of the "letting go" actions to a much lower value. Partially, the lower learning rate can



(a) $r^0 = -0.25$, deterministic human (b) $r^0 = -0.50$, deterministic human (c) $r^0 = -0.25$, 20% passive human

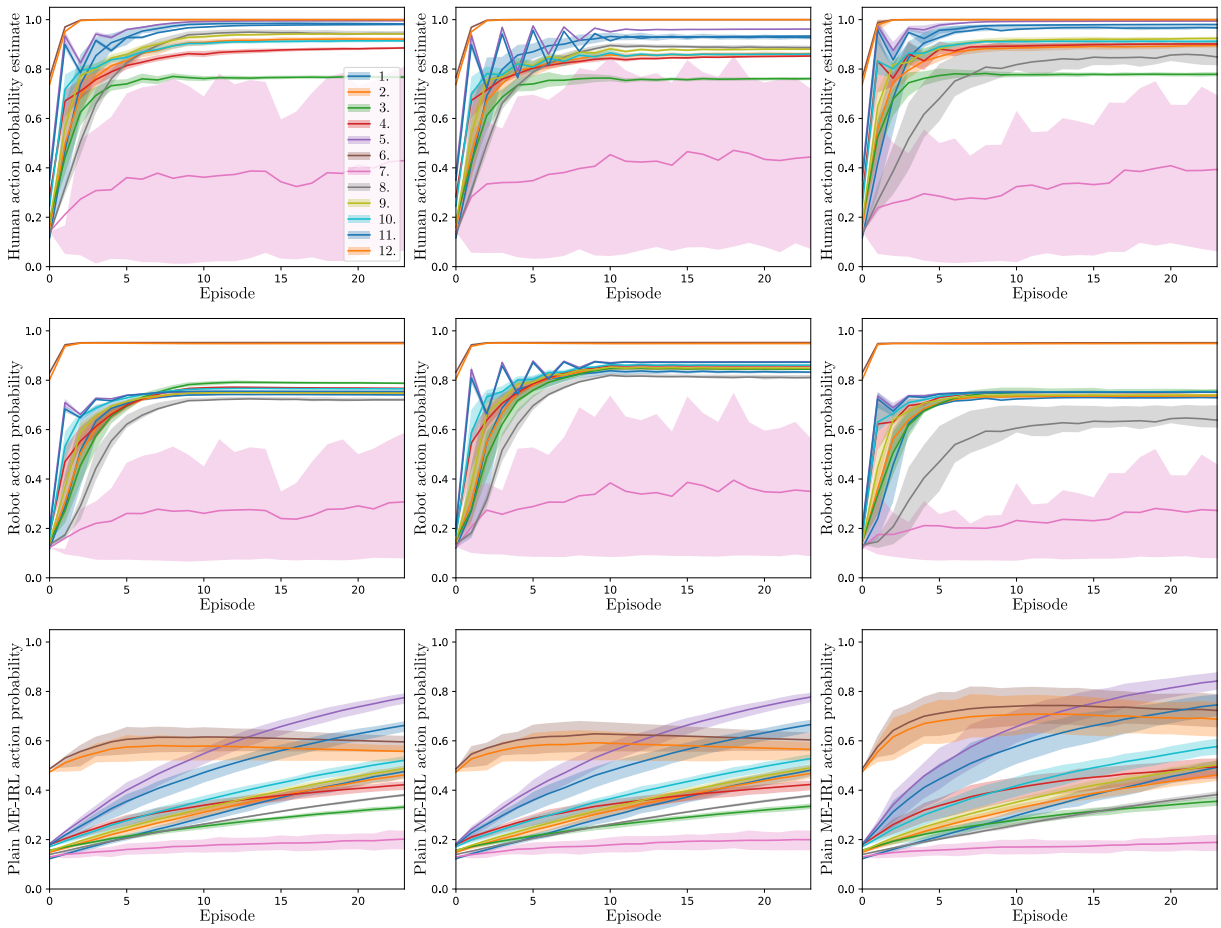
Fig. 11 Learned human policy estimate (top) and resulting robot policy (bottom) of Preference 1 (straight) for the two different punishments for not acting (a,b), and (c) with a human who does not act for 20% of the time. The colored lines, with interquartile bounds, correspond to the following state-action pairs, for the intention to go to rack A (Fig. 1b): 1. s = horizontal right above a stand (B or C), a = move over to rack; 2. s = horizontal low next to rack, a = move up to final height; 3. s = horizontal high next to rack, a = rotate; 4. s = vertical high next to rack, a = put on rack; 5. s = on rack, a = let go of wheel; for the intention to go to a stand (B or C): 6. s = vertical high right next to rack, a = move over to intended stand; 7. s = vertical high above stand, a = move down to just above stand; 8. s = vertical right above stand, a = rotate; 9. s = horizontal right above stand, a = put on stand; 10. s = on stand, a = let go of wheel.

be held accountable, but a higher learning rate destabilized the learning.

In Fig. 12, we see our model has trouble capturing one specific human action. This explains why our learner struggles more to learn the presented detour case. In that state, the robot remains unsure about which corresponding action to take, resulting in an extra passive action most of the times it passes through that state. As the plain ME-IRL agent has trouble learning the same action, we can conclude that not being able to capture this preferences is caused by the features making up the reward function no being able to capture this particular preference.

8 Conclusion

This paper presents a novel method for learning a human preference model for intention-aware cooperation from collaborative episodes. This enables our robot system to learn a personalized model of its human partner for improved collaboration. Our main contribution is a concept for learning human preferences as an explicit function of intention, exploiting two-level Theory of Mind reasoning.



(a) $r^0 = -0.25$, deterministic human (b) $r^0 = -0.50$, deterministic human (c) $r^0 = -0.25$, 20% passive human

Fig. 12 Learned human policy estimate (top) and resulting robot policy (bottom) of Preference 2 (detour) for the two different punishments for not acting (a,b), and (c) with a human who does not act for 20% of the time. The colored lines, with interquartile bounds, correspond to the following state-action pairs, for the intention to go to rack A (Fig. 1b): 1. s = horizontal right above a stand (B or C), a = move to comfort height; 2. s = horizontal low next to stand, a = move over to rack; 3. s = horizontal low next to rack, a = move up to final height; 4. s = horizontal high next to rack, a = rotate; 5. s = vertical high next to rack, a = put on rack; 6. s = on rack, a = let go of wheel; for the intention to go to a stand (B or C): 7. s = vertical high right next to rack, a = rotate; 8. s = horizontal high right next to rack, a = move down to comfort height; 9. s = horizontal low next to rack, a = move over to intended stand; 10. s = horizontal low next to stand, a = move up to just above stand; 11. s = horizontal right above stand, a = put on stand; 12. s = on stand, a = let go of wheel.

The acquired model captures preferences of how to collaboratively move objects, as well as how to infer the human's intention from the collaborative actions. We could show that our model allows the robot to take proactive actions that match both its partner's preferences and intention, with fewer mistakes than an imitation learner would make, or a plain ME-IRL learner without a human model.

A user study revealed that participants using our learning algorithm feel significantly more understood and supported in their preferences and intentions compared to an agent that just imitates their actions. Furthermore, the users felt that the task was much easier to perform with our agent, and felt it improved their performance. The fact that this was observed during only nine episodes, with seven different combinations of start position and intention, demonstrates how the generalizing capabilities of our method make our agent learn really fast.

The proposed concepts come with some limitations and assumptions. Firstly, we overestimate the knowledge of the human of the robot's policy, by giving the model access to the actual most

recent robot policy. Secondly, preference learning and intention estimation were restricted to prescribed motions between a small set of predefined waypoints. Future work will focus on relaxing this assumption. Thirdly, the large set of hand-designed features used in the Inverse Reinforcement Learning limits the scalability of the method. Future work should explore and integrate learning of a minimal set of optimal intention-parameterized features, e.g., following Bobu et al (2022). Despite these assumptions, our methods enable a robot system to learn the user's preferences as well as to estimate their intentions from only a very few interactive episodes. This allows robots to quickly learn how to provide people with personalized proactive support, improving human-robot interaction and physical cooperation.

Declarations

Funding

This research received funding from the Honda Research Institute Europe.

Ethics approval

The experimental protocols were approved by the Human Research Ethics Committee at the Delft University of Technology on the 19th of November 2021.

Availability of data and materials

Can be requested from the corresponding author.

Code availability

The code to this research is available in the following GitHub repository: https://github.com/LindavdSpaa/learning_collaborative_preferences

Author contributions

L.v.d.S. developed the concept and methods, which were reviewed by J.K. and M.G.. The experiments were designed by L.v.d.S., J.K. and M.G.. The programming and data analysis were done by L.v.d.S., who wrote the first draft of the paper. L.v.d.S., J.K. and M.G. revised the paper. All authors read and approved the submitted version.

References

Avaei A, van der Spaa L, Peternel L, et al (2023) An incremental inverse reinforcement learning approach for motion planning with human preferences. Robotics 12(2):61

Bai H, Cai S, Ye N, et al (2015) Intention-aware online POMDP planning for autonomous driving in a crowd. In: 2015 IEEE Int. Conf. on robotics and automation (ICRA), IEEE, pp 454–460

Baker CL, Tenenbaum JB (2014) Modeling human plan recognition using bayesian theory of mind. Plan, activity, and intent recognition: Theory and practice pp 177–204

Belardinelli A, Kondapally AR, Ruiken D, et al (2022) Intention estimation from gaze and motion features for human-robot shared-control object manipulation. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp 9806–9813

Bobu A, Wiggert M, Tomlin C, et al (2022) Inducing structure in reward learning by learning features. The International Journal of Robotics Research 41(5):497–518

Boularias A, Kober J, Peters J (2011) Relative entropy inverse reinforcement learning. In: Proceedings of the Fourteenth Int. Conf. on Artificial Intelligence and Statistics, pp 182–189

Buehler MC, Weisswange TH (2018) Online inference of human belief for cooperative robots. In: 2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), IEEE, pp 409–415

Buşoniu L, Babuška R, De Schutter B (2010) Multi-agent Reinforcement Learning: An Overview, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 183–221

Choudhury R, Swamy G, Hadfield-Menell D, et al (2019) On the utility of model learning in hri. In: 2019 14th ACM/IEEE Int. Conf. on Human-Robot Interaction (HRI), IEEE, pp 317–325

Duchaine V, Gosselin CM (2007) General model of human-robot cooperation using a novel velocity based variable impedance control. In: Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC'07), IEEE, pp 446–451

Fitter NT, Mohan M, Kuchenbecker KJ, et al (2020) Exercising with baxter: preliminary support for assistive social-physical human-robot interaction. Journal of neuroengineering and rehabilitation 17:1–22

Franceschi P, Maccarini M, Piga D, et al (2023) Human preferences' optimization in phri collaborative tasks. In: 2023 20th International Conference on Ubiquitous Robots (UR), IEEE, pp 693–699

Gienger M, Ruiken D, Bates T, et al (2018) Human-robot cooperative object manipulation with contact changes. In: 2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), IEEE

Hadfield-Menell D, Russell SJ, Abbeel P, et al (2016) Cooperative inverse reinforcement learning. In: Advances in neural information processing systems, pp 3909–3917

Hafs A, Verdel D, Burdet E, et al (2024) A finite-horizon inverse differential game approach for optimal trajectory-tracking assistance with a wrist exoskeleton. hal-04443499

Haninger K, Hegeler C, Peternel L (2022) Model predictive control with gaussian processes for flexible multi-modal physical human robot interaction. In: 2022 Int. Conf. on Robotics and Automation (ICRA), IEEE, pp 6948–6955

Hanna A, Larsson S, Götvall PL, et al (2022) Deliberative safety for industrial intelligent human–robot collaboration: Regulatory challenges and solutions for taking the next step towards industry 4.0. Robotics and Computer-Integrated Manufacturing 78:102,386

Hawkins KP, Bansal S, Vo NN, et al (2014) Anticipating human actions for collaboration in the presence of task and sensor uncertainty. In: 2014 IEEE Int. Conf. on Robotics and Automation (ICRA), IEEE, pp 2215–2222

Jara-Ettinger J (2019) Theory of mind as inverse reinforcement learning. Current Opinion in Behavioral Sciences 29:105–110

Jin Zj, Qian H, Chen Sy, et al (2011) Convergence analysis of an incremental approach to online inverse reinforcement learning. Journal of Zhejiang University SCIENCE C 12(1):17–24

Karami AB, Jeanpierre L, Mouaddib AI (2009) Partially observable markov decision process for managing robot collaboration with human. In: 2009 21st IEEE Int. Conf. on Tools with Artificial Intelligence, IEEE, pp 518–521

Koert D, Pajarinen J, Schotschneider A, et al (2019) Learning intention aware online adaptation of movement primitives. IEEE Robotics and Automation Letters 4(4):3719–3726

Koppula HS, Jain A, Saxena A (2016) Anticipatory planning for human-robot teams. In: Experimental Robotics, Springer, pp 453–470

Lai Y, Paul G, Cui Y, et al (2022) User intent estimation during robot learning using physical human robot interaction primitives. Autonomous Robots 46(2):421–436

Losey DP, Bajcsy A, O'Malley MK, et al (2022) Physical interaction as communication: Learning robot objectives online from human corrections. The International Journal of Robotics Research 41(1):20–44

Malik D, Palaniappan M, Fisac JF, et al (2018) An efficient, generalized bellman update for cooperative inverse reinforcement learning. arXiv preprint arXiv:180603820

Mehr N, Wang M, Bhatt M, et al (2023) Maximum-entropy multi-agent dynamic games: Forward and inverse solutions. IEEE Transactions on Robotics

Musić S, Hirche S (2020) Haptic shared control for human-robot collaboration: A game-theoretical approach. IFAC-PapersOnLine 53(2):10,216–10,222

Nikolaidis S, Hsu D, Srinivasa S (2017a) Human-robot mutual adaptation in collaborative tasks: Models and experiments. The Int Journal of Robotics Research 36(5-7):618–634

Nikolaidis S, Nath S, Procaccia AD, et al (2017b) Game-theoretic modeling of human adaptation in human-robot collaboration. In: Proceedings of the 2017 ACM/IEEE international conference on human-robot interaction, pp 323–331

Ong SC, Png SW, Hsu D, et al (2009) Pomdps for robotic tasks with mixed observability. In: Robotics: Science and systems

Parekh S, Habibian S, Losey DP (2022) Rili: Robustly influencing latent intent. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp 01–08

Park JS, Park C, Manocha D (2019) I-planner: Intention-aware motion planning using learning-based human motion prediction. The Int Journal of Robotics Research 38(1):23–39

Peters L, Rubies-Royo V, Tomlin CJ, et al (2023) Online and offline learning of player objectives from partial observations in dynamic games. The International Journal of Robotics Research p 02783649231182453

Ranatunga I, Cremer S, Popa DO, et al (2015) Intent aware adaptive admittance control for physical human-robot interaction. In: 2015 IEEE Int. Conf. on Robotics and Automation (ICRA), IEEE, pp 5635–5640

Rhinehart N, Kitani K (2018) First-person activity forecasting from video with online inverse reinforcement learning. IEEE transactions on pattern analysis and machine intelligence

Sadigh D, Sastry S, Seshia SA, et al (2016) Planning for autonomous cars that leverage effects on human actions. In: Robotics: Science and Systems, Ann Arbor, MI, USA

Schmerling E, Leung K, Vollprecht W, et al (2018) Multimodal probabilistic model-based planning for human-robot interaction. In: 2018 IEEE Int. Conf. on Robotics and Automation (ICRA), IEEE, pp 1–9

Schwarting W, Pierson A, Alonso-Mora J, et al (2019) Social behavior for autonomous vehicles. Proceedings of the National Academy of Sciences 116(50):24,972–24,978

Schweitzer PJ, Seidmann A (1985) Generalized polynomial approximations in markovian decision processes. Journal of mathematical analysis and applications 110(2):568–582

Sendhoff B, Wersing H (2020) Cooperative intelligence-a humane perspective. In: 2020 IEEE Int. Conf. on Human-Machine Systems (ICHMS), IEEE, pp 1–6

Shih A, Ermon S, Sadigh D (2022) Conditional imitation learning for multi-agent games. In: 2022 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI), IEEE, pp 166–175

Van der Spaa L, Franzese G, Kober J, et al (2022) Disagreement-aware variable impedance control for online learning of physical human-robot cooperation tasks. In: ICRA 2022 full day workshop - Shared Autonomy in Physical Human-Robot Interaction: Adaptability and Trust

Stouraitis T, Chatzinikolaidis I, Gienger M, et al (2020) Online hybrid motion planning for dyadic collaborative manipulation via bilevel optimization. IEEE Transactions on Robotics 36(5):1452–1471

Tian R, Tomizuka M, Dragan AD, et al (2023) Towards modeling and influencing the dynamics of human learning. In: Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction, pp 350–358

Tijsma AD, Drugan MM, Wiering MA (2016) Comparing exploration strategies for q-learning in random stochastic mazes. In: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, pp 1–8

Wang WZ, Shih A, Xie A, et al (2022) Influencing towards stable multi-agent interactions. In: Conference on robot learning, PMLR, pp

1132–1143

Williams CK, Rasmussen CE (2006) Gaussian Processes for Machine Learning, vol 2. MIT Press Cambridge, MA

Xie A, Losey D, Tolsma R, et al (2021) Learning latent representations to influence multi-agent interaction. In: Conference on robot learning, PMLR, pp 575–588

Zhifei S, Joo EM (2012) A review of inverse reinforcement learning theory and recent advances. In: Evolutionary Computation (CEC), 2012 IEEE Congress on, IEEE, pp 1–8

Ziebart BD, Maas AL, Bagnell JA, et al (2008) Maximum entropy inverse reinforcement learning. In: Aaai, Chicago, IL, USA, pp 1433–1438

Zyner A, Worrall S, Nebot E (2019) Naturalistic driver intention and path prediction using recurrent neural networks. IEEE Transactions on Intelligent Transportation Systems