

Rethinking the Implications of Human Feedback for Preference Learning in Human-Robot Collaboration

Qiping Zhang, Kate Candon, Debasmita Ghose, Marynel Vázquez

Department of Computer Science
Yale University, New Haven, CT, United States
qiping.zhang@yale.edu

Abstract: In Human-Robot Interaction, the standard approach to learn a reward model that represents human preferences for robot behavior consists of three steps. First, the robot collects limited *direct* evidence from human feedback (e.g., positive or negative binary feedback). Then, the robot utilizes the direct evidence to derive accepted or rejected labels to feasible but unchosen actions using fixed implication rules. Finally, the robot updates the reward model with both the direct and derived evidence. Unfortunately, the fixed rule can hinder preference learning: in a user study with two collaborative simulation environments, human-provided implication labels often differed from the standard fixed rule, and using the human labels substantially improved reward learning with the Preference Learning from Implicit and Explicit Feedback (PIE) algorithm. Consequently, we propose IMPLIED, an implication modeling method that treats fixed-rule implications as an initial guide while learning to infer and revise accepted and rejected action labels over time. Across evaluations on recorded human-robot interaction trajectories and a physical robot pizza-making study, IMPLIED predicts human implications more accurately than the fixed rule approach and LLM baselines, approaching the performance of a human-label oracle. In turn, IMPLIED reduces preference-estimation error and leads to robot actions that are more often rational with respect to a combined reward (which includes the true preference reward and a task-specific reward) compared to baselines. By learning to reason about the implications of human feedback, this work enables more faithful and efficient robot behavior adaptation during human-robot collaboration.

Keywords: Explicit and Implicit Human Feedback, Interactive Learning

1 Introduction

There are many valid ways to complete everyday tasks, and people have different preferences about aspects such as action ordering, object placement, or division of labor. Because preferences are often personal and difficult to specify in advance [1], collaborative robots must not only learn how to help a person achieve a task goal, but also how the person prefers the task to be done.

Prior work in Human-Robot Interaction (HRI) typically assumes that robots obtain *direct* evidence about a person’s preferences from two sources. A person can give *explicit feedback* about robot actions to indicate if actions matched their preferences via modalities like evaluative feedback [2, 3, 4, 5, 6, 7, 8], demonstrations [9, 10], or corrections [11, 12, 13, 14]. People can also provide *implicit feedback* through other communicative signals in response to a particular action [15, 16, 17, 18, 19, 20, 21, 22, 23, 24]. The direct evidence indicates if a particular action matches the user’s preference and should be treated as *accepted*, or did not and should be treated as *rejected*. This evidence is useful, but limited. It can be difficult to know what to infer about the other feasible actions that were

*Project website: <https://implied.interactive-machines.com>

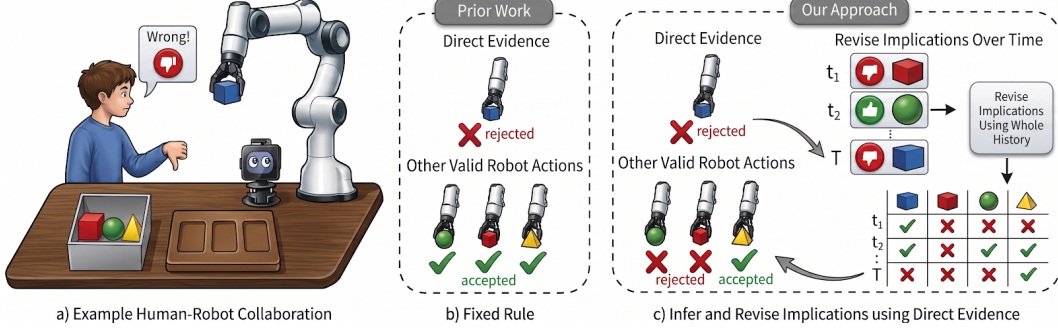


Figure 1: (a) In a collaboration, person provides direct evidence, rejecting the robot’s executed action. (b) Typical implication applies fixed-rule to label other feasible actions given the direct evidence. (c) Our approach, IMPLIED, uses direct evidence overtime to revise implication labels for unchosen actions.

not taken, which we call *unchosen actions*. Following prior work [2, 16, 25], we call the accept and reject labels for all actions, both chosen and unchosen, *implications*.

Converting limited direct evidence into broader accepted and rejected feedback sets via fixed implication rules is a long-standing practice. For example, this practice is common for inverse reinforcement learning [26], modeling human choices for robot learning [16], interactive querying [2], and collaborative preference learning [25]. Intuitively, negative feedback for a robot action (Fig. 1a) could indicate that the executed action is rejected and other feasible actions are accepted (Fig. 1b). But if this assumption is wrong, it could be difficult to learn correct preferences from noisy sets of accepted and rejected behavior. To investigate this possible shortcoming, we conducted a user study where people collaborated with a robot to perform two tasks in simulation. The tasks involved high-level collaborative decision-making, where at each step the human and the robot each had to select from a discrete, enumerable set of high-level actions. Participants provided direct evidence for preference learning via explicit and implicit feedback, and also labeled unchosen human and robot actions as accepted or rejected. We found evidence that the naive implication rules commonly used in prior work were inconsistent with human labels, suggesting that robots could better learn preferences if they reasoned more carefully about the unchosen actions.

To this end, we propose IMPLIED, Introspective Modeling for Preference LearnIng with EviDence. IMPLIED uses fixed-rule implications as a prior for generating implications for human feedback when direct evidence from the person is limited. As the interaction progresses, the robot relies more on a Gaussian Process classifier [27] trained on the direct evidence. Because the implications are re-computed whenever new human feedback is provided, the robot can revisit its prior decisions for which unchosen actions should be treated as accepted or rejected (see Fig. 1c). In this sense, IMPLIED is introspective [28, 29, 30] because it uses new evidence to revise the derived implications of past unchosen actions.

We evaluated IMPLIED across recorded human-robot interaction trajectories from our initial user study with simulation environments and in a second physical robot pizza-making study. We found that IMPLIED predicted human implications more accurately than the traditional fixed-rule approach and several LLM baselines, approaching the performance of a human-label oracle. The improved implications reduced preference-estimation error and led to robot actions that were more often rational in relation to a reward that considered the human preferences and the task’s objective goal. In summary, our contributions are threefold:

1. We formulate implication modeling as a learnable aspect of preference learning from feedback for high-level collaborative decision-making, where robots infer accepted/rejected labels for actions that are available but not taken.
2. We conduct a user study to collect human implication labels for unchosen actions, showing that fixed implications often diverge from human judgments and affect preference learning.

3. We propose IMPLIED, the first approach that, to our knowledge, revises implication labels for unchosen actions from accumulated direct evidence. We evaluate IMPLIED against state-of-the-art baselines showing that it can facilitate preference learning.

2 Preliminaries: Learning Preferences Over a Human-Robot Collaboration

Consider a collaboration between a human (H) and a robot (R) on a task, where each has fixed roles, e.g., the robot may pass pizza ingredients to the human, and the human may use them to assemble a pizza. As in Candon et al. [25], we study the problem of learning a reward function that captures the human’s preferences over the collaboration. This learning setup can be seen as a generalization of learning preferences over robot behavior only (e.g., as in [2, 16]), as it considers both (explicit) human feedback about the robot’s behavior and (implicit) feedback about the human’s behavior.

Collaborative Setup. We view the interaction as a Markov Decision Process (MDP) $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, R \rangle$, where $\mathbf{s} \in \mathcal{S}$ is the state space, $\mathcal{A} = \mathcal{A}_H \times \mathcal{A}_R$ is the action space with $a_H \in \mathcal{A}_H$ and $a_R \in \mathcal{A}_R$ the simultaneous high-level actions taken by the human and robot, $\mathcal{T}(\mathbf{s}'|\mathbf{s}, a_H, a_R)$ is a deterministic transition function from $\mathbf{s}$ to $\mathbf{s}'$ given the actions, and $R(\mathbf{s}, a_H, a_R)$ is a shared, total reward that the team seeks to maximize. Building off of Cooperative Inverse Reinforcement Learning [31], $R(\mathbf{s}, a_R, a_H) = R^{\text{goal}}(\mathbf{s}, a_R, a_H) + \gamma R^{\text{pref}}(\mathbf{s}, a_R, a_H; \mathbf{w})$, where γ controls the relative importance of a task-specific goal reward (R^{goal}) and a reward that describes the human’s preferences for the collaboration (R^{pref}), which is parameterized by $\mathbf{w} \in \mathbb{R}^d$. Because of the fixed-role assignments, the goal and preference reward are assumed to be decomposable into individual rewards [25], such that the human acts to maximize $R_H(\mathbf{s}, a_H) = R_H^{\text{goal}}(\mathbf{s}, a_H) + \gamma R_H^{\text{pref}}(\mathbf{s}, a_H; \mathbf{w})$, and the robot should maximize $R_R(\mathbf{s}, a_R) = R_R^{\text{goal}}(\mathbf{s}, a_R) + \gamma R_R^{\text{pref}}(\mathbf{s}, a_R; \mathbf{w})$. Importantly, the task goal is public information, so both H and R know R_H^{goal} and R_R^{goal} . Further, the human and robot also know which features ϕ matter to describe the preference reward, but only the human knows the preference parameters $\mathbf{w}$ for the collaboration. During the interaction, the robot has to learn $R^{\text{pref}}(\mathbf{s}, a_R, a_H; \mathbf{w}) = R_R^{\text{pref}}(\mathbf{s}, a_R; \mathbf{w}) + R_H^{\text{pref}}(\mathbf{s}, a_H; \mathbf{w}) = \mathbf{w}^\top (\phi_R(\mathbf{s}, a_R) + \phi_H(\mathbf{s}, a_H))$ by estimating $\mathbf{w}$, where ϕ_R and ϕ_H map the state and individual actions to the preference feature space.

Preference Learning from Implicit and Explicit Feedback. Candon et al. [25] proposed the PIE algorithm for estimating the preferences considering: 1) *explicit* human feedback about the robot’s actions, provided as binary evaluative feedback (although other types of explicit feedback could be possible [2, 16]); and 2) *implicit* human feedback, provided via the human’s actions in the interaction. The feedback generates what we term *direct evidence* for chosen actions by the collaborators (R or H): it leads to assigning the chosen actions in a given state $\mathbf{s}$ to an accepted ($f_{\mathbf{s},m}^+$) or rejected ($f_{\mathbf{s},m}^-$) feedback set based on the feedback modality, either explicit ($m = \text{exp}$) or implicit ($m = \text{imp}$). Because direct evidence is limited, preference learning algorithms, including PIE, also *derive evidence* for unchosen actions. They assign other feasible actions that could have been taken in the state when the feedback was provided to the accepted or rejected sets. The process of classifying state-action observations into the feedback sets is often referred to as the “implication” of human feedback [2], and is not considered a learnable aspect of preference learning in prior work.

As is common in the literature (e.g., [2, 16, 26]), PIE uses a fixed set of rules to implement the implication of human feedback. Consider a timestep in the collaboration that leads to a state $\mathbf{s}$ where the robot takes action a_R and the human takes a_H . For explicit feedback, if the human indicates that a_R is acceptable via a positive button press, then PIE adds that state-action observation $(\mathbf{s}, a_R)$ to the accepted set ($f_{\mathbf{s},\text{exp}}^+$). Also, all other possible state-action observations $(\mathbf{s}, a)$ for unchosen but feasible robot actions in that state, $a \in \mathcal{A}_R(\mathbf{s}) \setminus \{a_R\}$, are added to the rejected set ($f_{\mathbf{s},\text{exp}}^-$). Conversely, if the human indicates unacceptable robot behavior via a negative button press, $(\mathbf{s}, a_R)$ is added to the rejected set ($f_{\mathbf{s},\text{exp}}^-$), and the state-action observations for other possible robot actions are added to the accepted set ($f_{\mathbf{s},\text{exp}}^+$). For implicit feedback, the assumption is that the human acts optimally relative to R_H . The $(\mathbf{s}, a_H)$ observation gets added to the accepted set ($f_{\mathbf{s},\text{imp}}^+$). Meanwhile, the rejected set ($f_{\mathbf{s},\text{imp}}^-$) consists of the observations $(\mathbf{s}, a)$ for all other possible human actions, $a \in \mathcal{A}_H(\mathbf{s}_t) \setminus \{a_H\}$.

Finally, PIE estimates the weights $\mathbf{w}$ using a non-parametric belief distribution computed via gradient ascent, optimizing for the log-likelihood ($\log \mathcal{L}(\mathbf{w})$) of the accepted behavior:

$$\mathcal{L}(\mathbf{w}) = \prod_{(f_{s,m}^+, f_{s,m}^-) \in \mathbf{F}} P(f_{s,m}^+ | \mathbf{w}) = \prod_{(f_{s,m}^+, f_{s,m}^-) \in \mathbf{F}} \frac{\sum_{(s,a) \in f_{s,m}^+} B_m(s, a)}{\sum_{(s,a) \in f_{s,m}^+ \cup f_{s,m}^-} B_m(s, a)} \quad (1)$$

where $B_m(s, a) = \exp(\hat{\beta}_m(R_c^{\text{goal}}(s, a) + \gamma R_c^{\text{pref}}(s, a)))$ is the exponential component of the Boltzmann rationality model, the collaborator c is either H (for $m = \text{imp}$ feedback) or R (for $m = \text{exp}$ feedback), and the action a is the corresponding human or robot action in state s .

3 The Implications of Real Human Feedback in H-R Collaborations

With approval from our Institutional Review Board (IRB), we recruited 18 participants to collaborate with a robot in two MuJoCo environments adapted from RoCo simulation components [32]. Half of the participants were assigned to a Fixed-Rule condition, where the robot estimated preferences using the standard implicature rule (as in Sec. 2) after each timestep. The other half were assigned to the Human Implication condition, where the robot estimated preferences using the same learning method, but with participant-provided implication labels. Thus, the condition affected the feedback sets used by the robot in the interaction, which in turn shaped its preference updates and behavior.

Study procedure. Participants completed sandwich-making and block-sorting tasks, each for three episodes with different initializations. These tasks involved high-level decision-making, a setting that is common in human-robot collaboration [17, 25, 31], where interaction data is scarce and actions must be coarse enough for the partners to communicate and reason about them.

In sandwich-making, the objective was to assemble a target sandwich; in block-sorting, the objective was to move all blocks into bins. Both tasks require sequential collaboration where task progress and latent preferences jointly determine desirable behavior, while differing in object semantics and action structure. Preferences could concern, for example, ingredient order, workspace occupancy, or color/shape-to-bin mappings. As commonly assumed in HRI (e.g., [2, 25]), we assumed that each participant had stable preferences.

Before each task, participants selected a true preference (such as placing cheese before bread or sorting blue blocks into the red bin) and used it consistently when acting and evaluating the robot. The robot knew the task goal reward but not the weights that captured the participant’s true preference. After each feedback step, the robot updated its preference estimate using gradient ascent on Eq. (1) and selected later actions based on its latest preference estimate.

At each timestep, the participant and robot each selected one feasible action. Then, after both actions were executed, the participant evaluated the robot’s action as *accepted* or *rejected*, considering both task progress and alignment with their preference. The participant then labeled each of the robot’s unchosen feasible actions as well, and finally labeled their own unchosen feasible actions using the same criterion. These labels served as human *ground-truth* implications, indicating how direct evidence at that timestep should extend to feasible actions that were not taken by the agents. See Appx. A for further details on the study and task design.

Implication inconsistency. We compared the fixed-rule and human-provided implication labels with two metrics: *timestep inconsistency*, which is the percentage of timesteps with at least one label mismatch for unchosen actions; and the *label inconsistency*, which is the percentage of individual unchosen-action labels that do not match across the interaction history. Both were computed per participant and reported as mean $\pm$ standard error. Considering implications from both explicit feedback on robot actions and implicit feedback based on human actions, the sandwich-making task had 0.27 ± 0.03 timestep inconsistencies and 0.21 ± 0.02 label inconsistencies; meanwhile, block-sorting had 0.41 ± 0.02 and 0.22 ± 0.01 , respectively. These results show that human judgments about unchosen actions differ substantially from the fixed implication rule.

Human implications improve preference learning. We tested whether differences in implications affected preference learning. We compared the two conditions using the $L2$ error between the normalized preference weight estimates and the ground-truth preferences. We also computed the rollout *conflict proportion*, i.e., the fraction of rollout robot actions that would not maximize the combined robot reward under the participant’s true preferences. Lower values in Fig. 2 indicate more accurate preferences and more aligned robot behavior.

At the earliest task-completion step across participants (vertical dashed line in Fig. 2) in the sandwich-making task, linear mixed-effects analyses showed that Human Implication significantly reduced sandwich-making $L2$ error relative to Fixed-Rule (0.65 ± 0.03 vs. 0.97 ± 0.10 , $p < 0.01$). The same was observed for conflict proportion (0.10 ± 0.01 vs. 0.25 ± 0.05 , $p < 0.05$). In block-sorting, Human Implication also reduced $L2$ error (0.39 ± 0.08 vs. 0.60 ± 0.12) and conflict proportion (0.11 ± 0.05 vs. 0.21 ± 0.07), though these differences were not statistically significant. Together, these findings motivate considering the implications of human feedback as a learnable component of preference learning.

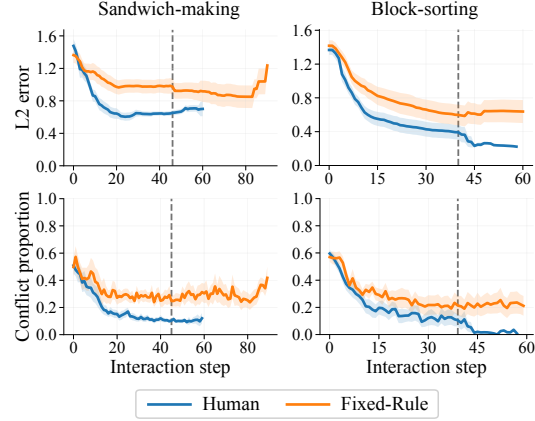


Figure 2: Preference-learning performance, with mean and std. error across participants. Human implications improved learning performance. The dashed line indicates the earliest task-completion step per task.

4 The IMPLIED Method for Inferring and Revising Implications

In prior work [2, 16, 25], the feedback sets ($f_{s,m}^+$, $f_{s,m}^-$) used for preference learning via Eq. (1) are generated by a fixed implication rule; however, Sec. 3 showed that these fixed-rule implications can differ from human judgments. Thus, we propose to instead treat the implications of unchosen actions as a learnable aspect of preference learning. Our insight is that *the more direct evidence the robot receives during the collaboration, the better it should be at inferring accepted or rejected labels for unchosen actions*. To explain how this inference process works, we re-define the format of the elements in the accepted and rejected feedback sets: they now contain state-action pairs, as in Sec. 2, along with metadata useful for reasoning about the implications of human feedback:

$$\mathbf{o} = (\underbrace{\mathbf{s}, a}_{\text{observation}}, \underbrace{c, m, t, e}_{\text{metadata}}) \quad (2)$$

The metadata stores information about which collaborator $c \in \{H, R\}$ took or could take the action a , which feedback modality $m \in \{exp, imp\}$ led to the state-action observation, the timestep t when the observation was generated, and the type of evidence $e \in \{direct, derived\}$ that was used to classify $\mathbf{o}$ into the feedback sets. Observations from chosen actions by the collaborators correspond to $e = direct$ evidence; those for unchosen actions are $e = derived$ evidence.

Our method, Introspective Modeling for Preference LearnIng with EviDence (IMPLIED), labels observations $\mathbf{o}$ of chosen actions as accepted or rejected per the direct evidence. For example, for explicit feedback, if a human indicates a robot’s action is acceptable, then the observation is added to the accepted feedback set. However, for observations of actions that were not chosen by the collaborators (i.e., the derived evidence), IMPLIED estimates the probability that any such observation $\mathbf{o}$ belongs in the accepted set ($f_{s,m}^+$) given *all* prior direct evidence $\mathcal{D}$, $P(\mathbf{o} \in f_{s,m}^+ | \mathcal{D})$. Importantly, the set $\mathcal{D}$ includes both observations and their corresponding labels, indicating whether the observations belong to the accepted or rejected feedback set. For example, if the interaction has reached the T -th timestep, then $\mathcal{D}$ is:

$$\mathcal{D} = \{(\mathbf{o}', y') \mid \mathbf{o}' = (\mathbf{s}', a', c', m', t', e') \in f_{s',m'}^+ \cup f_{s',m'}^- \wedge 1 \leq t' \leq T \wedge e' = direct\} \quad (3)$$

where $y' \in \{+, -\}$ is the observation label for $\mathbf{o}'$ per the direct evidence

Only if the probability $P(\mathbf{o} \in f_{\mathbf{s},m}^+|\mathcal{D})$ is greater than a nominal threshold ($\tau = 0.5$ in our experiments), IMPLIED assigns the observation $\mathbf{o}$ for an unchosen action to the accepted set; otherwise, it is assigned to the rejected set ($f_{\mathbf{s},m}^-$). This classification process happens every time new feedback is received, so the derived evidence can change over time as the interaction progresses.

Classifying observations of unchosen actions. IMPLIED defines $P(\mathbf{o} \in f_{\mathbf{s},m}^+|\mathcal{D})$ for an observation $\mathbf{o}$ corresponding to an unchosen action as a mixture distribution. This mixture uses the standard fixed-rule implication [2, 16, 25] as a structured prior. Also, it takes into consideration the probability that the observation belongs into the accepted set according to a learnable classifier trained on all the available direct evidence $\mathcal{D}$ that the robot has thus far in the collaboration. The mixture is:

$$P(\mathbf{o} \in f_{\mathbf{s},m}^+|\mathcal{D}) = (1 - \lambda(\mathbf{o})) P_{\text{Fixed}}(\mathbf{o} \in f_{\mathbf{s},m}^+|\mathcal{D}) + \lambda(\mathbf{o}) P_{\text{Learned}}(\mathbf{o} \in f_{\mathbf{s},m}^+|\mathcal{D}) \quad (4)$$

where $\lambda(\mathbf{o})$ is an observation-specific weight that balances the contributions of P_{Fixed} and P_{Learned} .

We implement P_{Fixed} in Eq. (4) via delta functions $\mathbb{I}$ whose output depends on the direct evidence obtained when the observation $\mathbf{o} = (\mathbf{s}, a, c, m, t, e)$ for an unchosen action a was generated. First, we search for the element $(\mathbf{o}', y')$ in $\mathcal{D}$ (as in Eq. (3)) that has $\mathbf{o}' = (\mathbf{s}', a', c', m', t', e')$ and the same timestep $t' = t$, state $\mathbf{s}' = \mathbf{s}$, feedback modality $m' = m$, and collaborator $c' = c$ as the observation $\mathbf{o}$. Then, if $a' = a$, we assign the label y' to $\mathbf{o}$, $P_{\text{Fixed}} = \mathbb{I}(y' = +)$. Otherwise, if $a' \neq a$, we assign the opposite label, $P_{\text{Fixed}} = \mathbb{I}(y' \neq +)$.

The probability P_{Learned} in Eq. (4) is computed with a Gaussian Process (GP) classifier [27, 33, 34] trained using data derived from $\mathcal{D}$. We chose a GP because it can learn quickly during interactions and provides a principled mechanism to measure epistemic uncertainty, i.e., uncertainty due to a lack of classifier knowledge [35]. We use the uncertainty to adapt the weight λ in Eq. (4), such that when the GP uncertainty is high, IMPLIED favors the fixed prior rather than the learned probability.

Specifically, the GP represents P_{Learned} via a function $h_\theta : \mathcal{Z} \rightarrow [0, 1]$ that maps features $z(\mathbf{o})$ of an observation to the posterior probability that $\mathbf{o}$ should be treated as accepted, i.e., included in $f_{\mathbf{s},m}^+$. For the features, we use the goal reward derived from the state-action pair in $\mathbf{o}$ and the corresponding collaborator preference features ϕ_c , $z(\mathbf{o}) = z((\mathbf{s}, a, c, m, t, e)) = [R_c^{\text{goal}}(\mathbf{s}, a), \phi_c(\mathbf{s}, a)]^T$. We made this choice because both the goal reward and ϕ matter for the MDP’s combined reward R (see Sec. 2). From an implementation perspective, the GP’s function $h_\theta = \sigma \circ f$ is a composition of a sigmoid likelihood function, $\sigma(u) = 1/(1 + \exp(-u))$, and a latent function $f \sim \mathcal{N}(\mathbf{0}, K)$ with a linear kernel K with an additive bias component. For inference, because the sigmoid likelihood is non-Gaussian, we perform Monte Carlo integration with M samples. We use their mean $\mu(\mathbf{o}) = \frac{1}{M} \sum_{r=1}^M h_\theta(z(\mathbf{o}))$ as P_{Learned} , and their variance $v(\mathbf{o}) = \frac{1}{M} \sum_{r=1}^M (h_\theta(z(\mathbf{o})) - \mu(\mathbf{o}))^2$ as the classifier’s confidence. Finally, we set λ in Eq. (4) such that it is inversely proportional to the classifier’s uncertainty (modeled by the variance v) and bounded to $[0, 1]$, $\lambda(\mathbf{o}) = 1 - 4v(\mathbf{o})$. See Appx. B for the derivation of the equation for λ and more implementation details for the GP.

5 Experiments and Results

5.1 Evaluation on Human-in-the-Loop Simulation Data

We first evaluate how well IMPLIED predicts the implications of human feedback and whether it improves downstream preference learning using the interaction trajectories from the user study from Sec. 3. This evaluation considers both implicit and explicit human feedback, as described previously. However, unlike during data collection, we hold the observed states, actions, and direct evidence fixed, and compare how different implication methods (IMPLIED and the baselines described below) reconstruct accepted and rejected feedback sets for preference learning compared to a human’s ground-truth implication labels.

Note that IMPLIED only changes how the feedback sets are constructed, so it is not tied to PIE: it applies to any preference-learning algorithm that derives implications for feasible but unchosen actions. See Appx. C for the results of IMPLIED when preference learning considers only implicit human feedback, or only explicit feedback. Furthermore, it is possible to use a different classifier

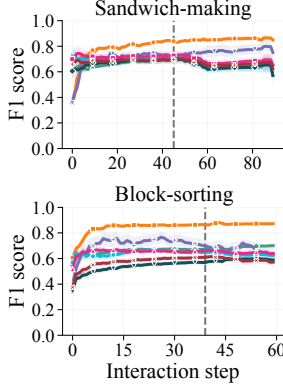


Figure 3: F1 score of classifying unchosen actions over time.

Table 1: Downstream preference-learning performance using different implications. Values are mean $\pm$ standard error at the earliest task-completion timestep across participants. **Human** is the oracle condition using ground-truth implication labels, and **IMPLIED** highlights our proposed method.

Implication	Sandwich-making		Block-sorting	
	L2 Error $\downarrow$	Conflict Prop. $\downarrow$	L2 Error $\downarrow$	Conflict Prop. $\downarrow$
Human	0.69 ± 0.05	0.11 ± 0.01	0.40 ± 0.06	0.10 ± 0.03
IMPLIED	0.71 ± 0.06	0.13 ± 0.01	0.42 ± 0.07	0.12 ± 0.03
Fixed-Rule	0.86 ± 0.07	0.20 ± 0.03	0.55 ± 0.07	0.20 ± 0.04
No-Prior	0.93 ± 0.06	0.20 ± 0.03	0.63 ± 0.09	0.19 ± 0.05
LLM	0.87 ± 0.08	0.19 ± 0.04	0.59 ± 0.07	0.20 ± 0.04
LLM-H	0.88 ± 0.08	0.24 ± 0.06	0.61 ± 0.08	0.21 ± 0.04
LLM-B	0.88 ± 0.08	0.20 ± 0.03	0.58 ± 0.07	0.17 ± 0.04
LLM-HB	0.85 ± 0.08	0.21 ± 0.05	0.56 ± 0.08	0.20 ± 0.05

than a GP for revising implications. For example, Appx. B.4 provides results with a K-Nearest Neighbors (KNN) classifier. However, we have got the best results with a GP, as described in Sec. 4; thus, our evaluation focuses on comparing IMPLIED using a GP classifier against the baselines.

Baselines. We compare IMPLIED against several baselines for inferring the implications of human feedback. The first is the Fixed-Rule implication approach from prior work (e.g., from PIE [25], as described in Sec. 2), which tests how implication prediction and downstream preference learning perform with a fixed heuristic for assigning observations to feedback sets. The second baseline is No-Prior. Since IMPLIED combines a structured fixed-rule prior with a GP-based prediction in Eq. (4), this ablation removes the structured prior and uses only the GP classifier to predict implications for unchosen actions. Further, we compare IMPLIED against four LLM baselines (implemented using Gemini 3.0 Flash) that copy the accepted/rejected labels from the direct evidence for chosen actions, but predict with the large model the labels for unchosen actions. These baselines serve to evaluate if a general-purpose LLM can use task context and broader world knowledge about human preferences to generate derived evidence. The LLM baselines receive different amounts of information when classifying unchosen actions: LLM receives only the current world state s , the executed a_H and a_R actions, and explicit feedback on the robot action at that timestep; the LLM-B baseline additionally receives a description of the preference-belief, feature by feature; LLM-H receives the interaction history, including previous states, actions, and feedback, on top of the information provided in the LLM baseline; and LLM-HB receives both the prior interaction history and a description of the preference-belief. See Appx. D for the prompts used for each LLM baseline. Finally, we also compare IMPLIED against using Human oracle implications for preference learning. The Human implications were collected directly from study participants (per Sec. 3).

Evaluation metrics. Since participants complete the task in different numbers of interaction steps, we evaluate all implication-prediction methods at the earliest task-completion timestep across participants, i.e., the largest timestep for which interaction data is available for every participant. We consider three metrics: 1) average *F1 classification score* across the study participants, considering all the labels predicted for unchosen actions up to the earliest time-completion step; 2) the average *L2 error* between the ground-truth preference weights w and the normalized, average preference weight from the preference-belief; and 3) the average *rollout conflict proportion*, defined as the fraction of rollout robot actions that do not maximize the combined robot reward R_R under the true preference. Higher values are better for F1, and lower values are better for the latter two metrics.

How well do methods classify state-action observations (with unchosen actions) as accepted or rejected? Fig. 3 shows the F1 classification scores over time—the Figure excludes results for the Human implications as they involve no inference. IMPLIED (orange line) generally led to higher F1 score faster than the non-human baselines. Also, at the earliest task-completion step across participants (vertical line in Fig. 3), IMPLIED achieved the highest F1-score in both tasks. In sandwich-making, linear mixed-effects analysis showed that IMPLIED (0.84 ± 0.01) had significantly higher

F1-score at the earliest task-completion step ($p < 0.01$). Similarly, IMPLIED (0.87 ± 0.01) significantly outperformed the non-human baselines in block-sorting at that step ($p < 0.001$). Thus, our method inferred human judgments for unchosen actions more accurately than non-human baselines.

How well do methods predict the human’s preferences? Table 1 shows that preference learning (per Eq. (1)) with IMPLIED closely approaches learning with ground-truth human implications. A linear mixed-effects analysis for the sandwich-making task ($p < 0.001$) showed that IMPLIED led to significantly lower L2 error than all non-human baselines, except for LLM-HB. Also, IMPLIED led to significantly lower conflict proportion than all non-human baselines ($p < 0.05$). In block-sorting, IMPLIED significantly outperformed all other non-human baselines on L2 error ($p < 0.001$) and conflict proportion ($p < 0.05$). These results suggest that implication revision improves learning.

5.2 Evaluation in Real-World Human-Robot Collaboration

With approval from our IRB, we evaluated learning during physical collaborations with 20 participants (Fig. 4). We used a pizza-making setup validated in prior work [25], where a human collaborates with a Franka Panda arm to assemble toy pizzas in a lab. Different from [25], the robot updated its preference estimate during the interaction, rather than doing so offline. The study had a within-subjects design, where each participant experienced robot learning per Eq. (1) with the common Fixed-Rule implication as well as with IMPLIED. The conditions were counterbalanced, and the robot’s memory was cleared between them to avoid information leakage. For each condition, the team made two pizzas with different recipes—but the same were used for Fixed-Rule and IMPLIED. Participants decided on their own when to provide explicit feedback. Implicit feedback was received on every step based on the human’s actions. See Appx. E for more details.

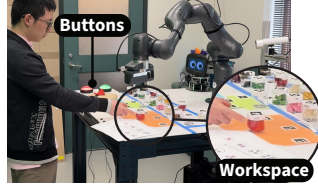


Figure 4: Collaboration setup.

Table 2 reports L2 preference error and conflict proportion at the earliest shared task-completion step. Linear mixed-effects analyses showed that online learning with IMPLIED achieved significantly lower L2 error ($p < 0.01$) and conflict proportion ($p < 0.05$) than learning with Fixed-Rule. We also measured two objective interaction costs: the number of explicit feedback instances a participant provided, and the number of interaction steps taken to complete the task. Online learning with IMPLIED required significantly fewer instances of explicit feedback than with Fixed-Rule (34.10 ± 0.88 vs. 43.45 ± 1.05 , $p < 0.0001$) and significantly fewer interaction steps to completion (40.05 ± 0.83 vs. 45.35 ± 1.15 , $p < 0.05$).

Table 2: Online learning performance for the physical human-robot collaboration study.

Implication	L2 Error ↓	Conflict Prop. ↓
IMPLIED	0.86 ± 0.07	0.16 ± 0.03
Fixed-Rule	1.14 ± 0.03	0.25 ± 0.03

Further, at the end of the study, participants answered four comparison questions for the conditions. Two-tailed binomial tests showed that significantly more participants reported that IMPLIED made the robot learn faster (17/20, $p < 0.01$), required less teaching effort (16/20, $p < 0.05$), led the robot to make fewer mistakes (17/20, $p < 0.01$), and was the method they would prefer the robot to use in the future (18/20, $p < 0.001$). These results suggest that users perceived IMPLIED as more preferable than the Fixed-Rule approach.

6 Related Work

There is a long history of work in preference learning. Early work (e.g., [36, 37]) focused on Inverse Reinforcement Learning (IRL), based on the observation that in “natural learning,” a reward function is often not explicitly specified and instead must be learned. In HRI, learning human preferences typically involves a robot learning from human demonstrations of the task they want the robot to complete (e.g., [9, 38, 39]) or from evaluative feedback provided by a human observer (e.g., [3, 40]). More related to our work, recent approaches have addressed how to combine different types of feedback: contributing a theoretical framework describing how different types of feedback can be mapped to a choice from a set of choices [16]; integrating demonstrations together with

preference queries [41]; combining demonstrations, corrections, preferences, and binary feedback [2]; reasoning about evaluative binary feedback alongside implicit feedback from human task actions [25]; or combining instructions and demonstrations [42].

Beyond combining feedback types, several methods derive labels beyond direct human input. For instance, Zhan et al. [43] train an adversarial network to take over the human’s role in labeling trajectory pairs, SURF [44] assigns confidence-based pseudo-labels to unlabeled segment pairs, and in text-to-image alignment, Semi-DPO [45] self-trains on a consensus-filtered subset to relabel conflicting preference pairs. Furthermore, D-REX [46] and SSRR [47] instead synthesize preference rankings by injecting noise into a cloned policy. Closer to our setting, Loftin et al. [19] model the human’s feedback strategy to interpret what the absence of feedback implies. Our work adds to this literature by showing that the implications of human feedback can be learned during preference learning, revising the labels of feasible unchosen actions from accumulated interaction evidence.

7 Limitations and Future Work

We proposed IMPLIED, a method that learns and revises implication labels from accumulated direct evidence. Our results suggest that IMPLIED improves implication prediction, preference estimation, and robot behavior in comparison to using a common fixed-rule implication or using large models to derive evidence for preference learning.

However, our work is not without limitations, which point to interesting avenues for future work. First, we assumed that the relevant preference-feature dimensions are known, so learning focuses on estimating their weights ($\mathbf{w}$ in Eq. (1)). This follows a long tradition of using linear reward functions in HRI, often due to sample efficiency concerns, but limits the method to tasks where designers can specify what humans may care about. Future work could investigate ways to jointly infer both the preference representation [20] and the weights.

Second, we focused on learning from two types of feedback modalities: binary explicit feedback, and implicit feedback provided by the human’s actions. In the future, it would be interesting to explore using more modalities, like demonstrations [2] or language [16] for which unified learning formalisms exist.

Third, our experiments used structured tasks with known goal rewards and relatively stable preferences. Extending implication modeling beyond these assumptions to more open-ended collaboration settings, where human preferences may shift over time [17, 48, 49, 50], will require richer action representations and robustness to noisy or inconsistent feedback. Relatedly, IMPLIED assumes a non-adversarial collaborator. A user who gives systematically misleading feedback could induce an overconfident and wrong GP that dominates inference. Detecting such inconsistency and falling back to the prior is an open problem.

Finally, we assumed an enumerable set of discrete high-level actions, which is also common in prior HRI work [17, 25, 49, 51]. Future work could study implication modeling in continuous action spaces and with vision-language-action (VLA) policies, which would first require deciding the unit of feedback. Specifically, Eq. (1) reasons about one high-level action at a time, so implications would instead have to be defined over trajectories. INQUIRE [2] does so, but assumes fixed, pre-segmented trajectories, whereas during a collaboration trajectory boundaries are not obvious. Non-collaborative VLA settings, where trajectory lengths can be pre-determined, may thus be a more tractable entry point for this future work direction.

Acknowledgments

This work was partially supported by the National Science Foundation (Grant No. IIS-2143109, IIS-2106690, and OIA-2535195). Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF). The authors are also thankful to Sasha Lew for his assistance with the robot system that was used for the real-world evaluation presented in Sec. 5.2.

References

- [1] P. Agrawal. The task specification problem. In *Conference on Robot Learning*, pages 1745–1751. PMLR, 2022.
- [2] T. Fitzgerald, P. Koppol, P. Callaghan, R. Q. J. H. Wong, R. Simmons, O. Kroemer, and H. Admoni. Inquire: Interactive querying for user-aware informative reasoning. In *6th Annual Conference on Robot Learning*, 2022.
- [3] W. B. Knox and P. Stone. Interactively shaping agents via human reinforcement: the tamer framework. In *Proceedings of the Fifth International Conference on Knowledge Capture, K-CAP '09*, pages 9–16, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605586588. doi:10.1145/1597735.1597738. URL <https://doi.org/10.1145/1597735.1597738>.
- [4] M. L. Schrum, E. Hedlund-Botti, N. Moorman, and M. C. Gombolay. Mind meld: Personalized meta-learning for robot-centric imitation learning. In *2022 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 157–165, 2022. doi:10.1109/HRI53351.2022.9889616.
- [5] R. Akrou, M. Schoenauer, and M. Sebag. Preference-based policy learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 12–27. Springer, 2011.
- [6] N. Wilde, A. Blidaru, S. L. Smith, and D. Kulić. Improving user specifications for robot behavior through active preference learning: Framework and evaluation. *The International Journal of Robotics Research*, 39(6):651–667, 2020.
- [7] X. Wang, K. Lee, K. Hakhamaneshi, P. Abbeel, and M. Laskin. Skill preferences: Learning to extract and execute robotic skills from human feedback. In *Conference on robot learning*, pages 1259–1268. PMLR, 2022.
- [8] D. J. Hejna III and D. Sadigh. Few-shot preference learning for human-in-the-loop rl. In *Conference on Robot Learning*, pages 2014–2025. PMLR, 2023.
- [9] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.
- [10] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.
- [11] D. P. Losey and M. K. O’Malley. Including uncertainty when learning from human corrections. In *Conference on Robot Learning*, pages 123–132. PMLR, 2018.
- [12] S. Wang, A. Wang, S. Goncharova, B. Scassellati, and T. Fitzgerald. Effects of robot competency and motion legibility on human correction feedback. In *2025 20th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 789–799. IEEE, 2025.
- [13] A. Bajcsy, D. P. Losey, M. K. O’Malley, and A. D. Dragan. Learning from physical human corrections, one feature at a time. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 141–149, 2018.
- [14] A. Wang, S. Wang, and T. Fitzgerald. Enhancing goal inference via correction timing. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems*, pages 2744–2753, 2026.
- [15] D. Ghose, M. A. Lewkowicz, K. Gezahegn, J. Lee, T. Adamson, M. Vázquez, and B. Scassellati. Tailoring visual object representations to human requirements: A case study with a recycling robot. In *Conference on Robot Learning*, pages 583–593. PMLR, 2023.

- [16] H. J. Jeon, S. Milli, and A. Dragan. Reward-rational (implicit) choice: A unifying formalism for reward learning. *Advances in Neural Information Processing Systems*, 33:4415–4426, 2020.
- [17] D. Ghose, O. Gitelson, R. Jin, G. Abawe, M. Vázquez, and B. Scassellati. I’ve changed my mind: Robots adapting to changing human goals during collaboration. *IEEE Robotics and Automation Letters*, 11(2):1490–1497, 2025.
- [18] Y. Cui, Q. Zhang, B. Knox, A. Allievi, P. Stone, and S. Niekum. The empathic framework for task learning from implicit human feedback. In *Conference on robot learning*, pages 604–626. PMLR, 2021.
- [19] R. Loftin, B. Peng, J. MacGlashan, M. L. Littman, M. E. Taylor, J. Huang, and D. L. Roberts. Learning behaviors via human-delivered discrete feedback: modeling implicit feedback strategies to speed up learning. *Autonomous agents and multi-agent systems*, 30(1):30–59, 2016.
- [20] N. Dennler, S. Nikolaidis, and M. Mataríć. Contrastive learning from exploratory actions: Leveraging natural interactions for preference elicitation. In *2025 20th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 778–788, 2025.
- [21] D. Ghose, O. Gitelson, M. Lewkowicz, J. Brawer, A. Roncone, M. Vazquez, and B. Scassellati. Robots influencing humans to reveal their goals during collaboration and competition. *Autonomous Robots*, 50(3):38, 2026.
- [22] Q. Zhang, A. Narcomey, K. Candon, and M. Vázquez. Self-annotation methods for aligning implicit and explicit human feedback in human-robot interaction. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 398–407, 2023.
- [23] K. Candon, N. C. Georgiou, H. Zhou, S. Richardson, Q. Zhang, B. Scassellati, and M. Vázquez. React: Two datasets for analyzing both human reactions and evaluative feedback to robots over time. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 885–889, 2024.
- [24] Q. Zhang, N. Tsoi, M. Nagib, B. Choi, J. Tan, H.-T. L. Chiang, and M. Vázquez. Predicting human perceptions of robot performance during navigation tasks. *ACM Transactions on Human-Robot Interaction*, 14(3):1–27, 2025.
- [25] K. Candon, Q. Zhang, A. Lew, H. Claire, L. Qian, A. Quarles, C. Sarkar, and M. Vázquez. Learning human preferences over a human-robot collaboration based on explicit and implicit human feedback. In *Proceedings of the 21st ACM/IEEE International Conference on Human-Robot Interaction*, pages 1040–1049, 2026.
- [26] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.
- [27] C. K. Williams and C. E. Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.
- [28] M. Fox, M. Ghallab, G. Infantes, and D. Long. Robot introspection through learned hidden markov models. *Artificial Intelligence*, 170(2):59–113, 2006.
- [29] K. Liang, Z. Zhang, and J. F. Fisac. Introspective planning: Aligning robots’ uncertainty with inherent task ambiguity. *Advances in Neural Information Processing Systems*, 37:71998–72031, 2024.
- [30] S. Daftry, S. Zeng, J. A. Bagnell, and M. Hebert. Introspective perception: Learning to predict failures in vision systems. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 1743–1750. IEEE, 2016.

- [31] D. Hadfield-Menell, S. J. Russell, P. Abbeel, and A. Dragan. Cooperative inverse reinforcement learning. In *Advances in neural information processing systems*, volume 29, 2016.
- [32] Z. Mandi, S. Jain, and S. Song. Roco: Dialectic multi-robot collaboration with large language models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 286–299. IEEE, 2024.
- [33] D. J. MacKay et al. Introduction to gaussian processes. *NATO ASI series F computer and systems sciences*, 168:133–166, 1998.
- [34] A. G. d. G. Matthews, M. Van Der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrà, Z. Ghahramani, and J. Hensman. Gpflow: A gaussian process library using tensorflow. *Journal of Machine Learning Research*, 18(40):1–6, 2017.
- [35] E. Hüllermeier and W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine learning*, 110(3):457–506, 2021.
- [36] S. Russell. Learning agents for uncertain environments. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 101–103, 1998.
- [37] A. Y. Ng, S. Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.
- [38] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.
- [39] A. G. Billard, S. Calinon, and R. Dillmann. Learning from humans. *Springer handbook of robotics*, pages 1995–2014, 2016.
- [40] J. MacGlashan, M. K. Ho, R. Loftin, B. Peng, G. Wang, D. L. Roberts, M. E. Taylor, and M. L. Littman. Interactive learning from policy-dependent human feedback. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2285–2294. PMLR, 06–11 Aug 2017.
- [41] E. Bıyık, D. P. Losey, M. Palan, N. C. Landolfi, G. Shevchuk, and D. Sadigh. Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences. *The International Journal of Robotics Research*, 41(1):45–67, 2022.
- [42] F. G. Zhu, J. Oh, and R. Simmons. Interpret: Interactive policy restructuring and training enable effective imitation learning from laypersons. In *Proceedings of the 21st ACM/IEEE International Conference on Human-Robot Interaction*, pages 864–873, 2026.
- [43] H. Zhan, F. Tao, and Y. Cao. Human-guided robot behavior learning: A gan-assisted preference-based reinforcement learning approach. *IEEE Robotics and Automation Letters*, 6(2):3545–3552, 2021.
- [44] J. Park, Y. Seo, J. Shin, H. Lee, P. Abbeel, and K. Lee. Surf: Semi-supervised reward learning with data augmentation for feedback-efficient preference-based reinforcement learning. *arXiv preprint arXiv:2203.10050*, 2022.
- [45] X. Liu, M. Li, Z. Lyu, Y. Shang, and C. Chen. Learning from noisy preferences: A semi-supervised learning approach to direct preference optimization. *arXiv preprint arXiv:2604.24952*, 2026.
- [46] D. S. Brown, W. Goo, and S. Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*, pages 330–359. PMLR, 2020.

- [47] L. Chen, R. Paleja, and M. Gombolay. Learning from suboptimal demonstration via self-supervised reward regression. In *Conference on robot learning*, pages 1262–1277. PMLR, 2021.
- [48] R. Ma, J. Qu, A. Bobu, and D. Hadfield-Menell. Goal inference from open-ended dialog. *arXiv preprint arXiv:2410.13957*, 2024.
- [49] D. Ghose, O. Gitelson, M. Vazquez, and B. Scassellati. Open-ended goal inference through actions and language for human-robot collaboration. In *In Proceedings of the 2026 ACM/IEEE International Conference on Human-Robot Interaction (HRI '26)*, 2026.
- [50] S. Jain and B. Argall. Probabilistic human intent recognition for shared autonomy in assistive robotics. *ACM Transactions on Human-Robot Interaction (THRI)*, 9(1):1–23, 2019.
- [51] J. Brawer, D. Ghose, K. Candon, M. Qin, A. Roncone, M. Vázquez, and B. Scassellati. Interactive policy shaping for human-robot collaboration with transparent matrix overlays. In *Proceedings of HRI*, 2023.
- [52] M. Görner, R. Haschke, H. Ritter, and J. Zhang. Moveit! task constructor for task-level motion planning. In *2019 International conference on robotics and automation (ICRA)*, pages 190–196. IEEE, 2019.
- [53] S. Thompson, A. Narcomey, A. Lew, and M. Vázquez. Shutter: A low-cost and flexible social robot platform for in-the-wild deployments. In *Companion of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '24, pages 94–96, 2024.
- [54] M. Colledanchise and P. Ögren. *Behavior trees in robotics and AI: An introduction*. CRC Press, 2018.

A Details of the User Study in MuJoCo Simulation

In this section, we describe in more detail the two simulation environments used for our initial user study (Sec. 3), which gathered evidence that the fixed-rule implication does not always match human judgments for unchosen actions. Fig. 5 shows a screenshot of each of the simulated tasks. For both, the robot and the human have fixed roles.

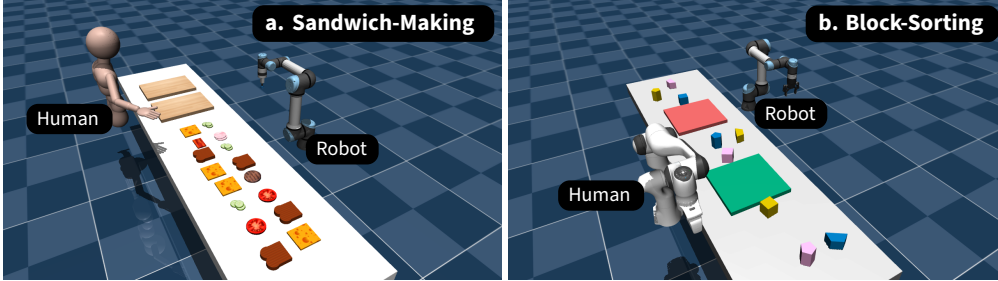


Figure 5: MuJoCo tasks. For both tasks, a human controls an agent (shown on the left side of both images). The other agent (the robot) is autonomous. It estimates preferences and acts according to them during our user study. These MuJoCo tasks were implemented by adapting and extending RoCo simulation components [32].

A.1 Study Protocol

We collected human feedback and implication labels in two MuJoCo human-robot collaboration environments: sandwich-making (Fig. 5(a)) and block-sorting (Fig. 5(b)). The study was approved by our local Institutional Review Board. In both environments, the human and robot acted in a shared task state. Before each task, participants selected a true preference from the available options and were instructed to use this preference consistently when choosing their own actions and evaluating the robot. The robot knew the task goal reward but did not know the weights corresponding to the participant’s true preference. Further details on the available task initializations and preference options are provided in Appx. A.2 and A.3.

At each timestep, the participant first selected their own action from the available human actions through the graphical user interface shown in Fig. 6(a). The robot also selected one action from its available action set according to the current policy $\pi_R(a_R|\mathbf{s}) = P(a_R|\mathbf{s}) \propto \exp(\beta_R R_R(\mathbf{s}, a_R))$, where the combined robot reward R_R is determined by the latest preference estimate $\mathbf{w}$, and β_R controls how rational the robot’s actions are. The joint human-robot actions were then rendered in the simulation.

After the transition in simulation, the participant used the interface to give explicit binary feedback on the robot’s action taken (Fig. 6(b)). The participant was instructed to judge whether the robot’s action helped complete the task while respecting how they preferred the task to be performed. The interface then asked the participant to label the implications of the robot’s unchosen actions as positive or negative (Fig. 6(c)). Finally, the participant labeled the implications of their own unchosen actions (Fig. 6(d)).

Half of the participants were assigned to a Fixed-Rule condition, in which the robot estimated preferences using the standard implicature rule described in Sec. 2 after each timestep. The other half were assigned to the Human Implication condition, in which the robot used the same preference-learning method but with the participant-provided implication labels. Thus, the condition determined which feedback sets were used by the robot during the interaction. After each feedback step, the robot updated its preference estimate using gradient ascent on Eq. (1) and selected later actions based on its latest preference estimate.

Each participant completed both tasks in the simulation, and the order of the two tasks was counterbalanced between participants. For each task, the participant completed three episodes. The three episodes used different initializations, randomly sampled from the available task initialization options. Each episode continued until the task was completed. Across episodes, the participant interacted with the robot, observed the rendered simulation state, and provided the action selections (implicit feedback), explicit feedback, and implication labels described above.

Select Actions
✕

Select your action

Human

- ☐ WAIT
- ☒ PICK bacon workspace (meat)
- ☐ PICK tomato2 workspace (veggie)
- ☐ PICK ham workspace (meat)

Provide Feedback
✕

The robot performed "PICK cheese3 storage (cheese)". What is your feedback on the robot's action?

- ☐ Positive
- ☒ Negative

(b) Explicit human feedback on robot action.

(a) Human action selection.

Action Implications
✕

The robot performed "PICK cheese3 storage (cheese)". What is your opinion for the unchosen actions of Robot?

Action	Positive	Negative
WAIT	☐	☒
PICK bacon workspace (meat)	☐	☒
PICK ham workspace (meat)	☐	☒
PICK beef_patty storage (meat)	☒	☐
PICK bread_slice storage (bread)	☒	☐
PICK cheese storage (cheese)	☐	☒
PICK tomato1 storage (veggie)	☒	☐
PICK tomato2 workspace (veggie)	☐	☒
PICK cucumber storage (veggie)	☒	☐

(c) Implications of unchosen robot actions.

Action Implications
✕

You performed "PICK bacon workspace (meat)". What is your opinion for your unchosen actions?

Action	Positive	Negative
WAIT	☐	☒
PICK ham workspace (meat)	☒	☐
PICK tomato2 workspace (veggie)	☐	☒

(d) Implications of unchosen human actions.

Figure 6: Graphical user interface used in the MuJoCo simulation study. At each timestep, participants (a) selected their own action, (b) provided explicit feedback on the robot's action, (c) labeled implications for the robot's unchosen actions, and (d) labeled implications for their own unchosen actions.

A total of 18 participants completed the study. Each study session typically lasted 45 min, and each participant was compensated \$15. Participants were recruited through flyers, online postings, and word of mouth, and were required to be at least 18 years old, fluent in English, and have normal or corrected-to-normal hearing and vision.

A.2 Sandwich-Making Task

Domain basics. Fig. 5(a) shows the sandwich-making task, in which a robot and a human assemble a target sandwich from 16 ingredients:

- Bread (4): bread_slice1, bread_slice2, bread_slice3, bread_slice4
- Cheese (4): cheese1, cheese2, cheese3, cheese4
- Meat (3): bacon, ham, beef_patty
- Veggie (5): tomato1, tomato2, cucumber1, cucumber2, cucumber3

The task locations are storage, workspace, and sandwich, with additional robot- and human-hand states. Initially, all ingredients are in storage, and the workspace, sandwich, and both hands are empty. The workspace can contain at most four ingredients. The robot can move ingredients between storage and workspace, while the human can move ingredients between workspace and sandwich. Each episode specifies a target recipe, and completion is evaluated using required counts over goal ingredient classes (bread, cheese, bacon, ham, beef, tomato, and cucumber).

15

State space. The task state is represented by the allocation of ingredients across the five relevant locations: storage, workspace, sandwich, the robot’s hand, and the human’s hand. Together with the target recipe, this allocation determines the relevant task progress: which required goal-class counts have already been satisfied on the sandwich, which ingredients are available to each agent, whether either agent is holding an ingredient, and whether the workspace capacity constraint is active.

Action space and transitions. The robot’s action space has a total of 19 actions: 16 PICK(ingredient) actions, PUT(ingredient, storage), PUT(ingredient, workspace), and WAIT. If the robot is not holding an ingredient, it may pick an ingredient from storage or workspace; if it is holding an ingredient, it may put it in storage or, when the workspace is not full, put it in the workspace. The human’s action space also has a total of 19 actions: 16 PICK(ingredient) actions, PUT(ingredient, workspace), PUT(ingredient, sandwich), and WAIT. The human may pick only from the workspace; if holding an ingredient, the human may put it on the sandwich or return it to the workspace when space is available. WAIT is always available for both agents. At each step, the robot and human choose simultaneous actions; a pick action transfers an ingredient to the acting agent’s hand, and a put action transfers a held ingredient to the chosen target location.

Preference space. Human preferences are represented by a 10-dimensional weight vector,

$$\mathbf{w} = [\text{cheese_before_bread}, \text{cheese_before_meat}, \\ \text{cheese_before_veggie}, \text{bread_before_meat}, \\ \text{bread_before_veggie}, \text{meat_before_veggie}, \\ \text{workspace_size_le0}, \text{workspace_size_le1}, \\ \text{workspace_size_le2}, \text{workspace_size_le3}]$$

The first six dimensions encode pairwise ordering preferences over the four broad ingredient types cheese, bread, meat, and veggie. For example, a positive value of `cheese_before_bread` means that all target cheese should be added before any target bread; a negative value represents the reverse preference, and a value near zero represents no preference between the two types. The last four dimensions encode preferences over workspace occupancy: dimension k is positive when the post-action workspace size is at most k and negative otherwise. Weights are normalized to unit Euclidean norm, and sampled preferences are constrained to have acyclic ordering preferences and consistent workspace-size preferences.

True preference options and environment initializations. Before the sandwich-making task, participants selected one of six preference choices and used it consistently across all sandwich-making trials. The true preference options were:

- *Preference A:* Bread before cheese before veggie before meat.
- *Preference B:* Bread before meat before veggie.
- *Preference C:* Cheese before meat before bread and veggie.
- *Preference D:* Veggie before meat.
- *Preference E:* Prefer more than two ingredients in the workspace.
- *Preference F:* Prefer at most two ingredients in the workspace.

Each environment initialization specified one target recipe. The recipe options were:

- *Initial state 1:* 2 bread, 1 cheese, 1 ham, 1 bacon, 1 tomato, 1 cucumber.
- *Initial state 2:* 2 bread, 2 cheese, 1 beef, 1 tomato, 1 cucumber.
- *Initial state 3:* 2 bread, 2 cheese, 1 ham, 1 bacon, 1 beef.
- *Initial state 4:* 2 bread, 2 cheese, 1 ham, 1 bacon, 1 tomato.
- *Initial state 5:* 2 bread, 1 cheese, 2 tomato, 1 cucumber.
- *Initial state 6:* 1 bread, 1 cheese, 1 ham, 1 tomato.

Reward function. The shared reward is decomposed into goal and preference terms,

$$R(s, a_R, a_H) = R_{\text{goal}}(s, a_R, a_H) + \gamma R_{\text{pref}}(s, a_R, a_H),$$

where γ controls the relative importance of the preference reward and goal reward.

For the goal reward, each agent receives a step cost of -0.3 , and completing the sandwich gives a $+10$ bonus. The robot is rewarded for moving still-needed target ingredients toward the human: picking a needed target from storage gives $+2$, and placing a needed target from storage into the workspace gives $+3$. To incentivize efficient collaboration, regressive or redundant robot actions are penalized: picking a needed ingredient that is already in the workspace gives -2 (or -0.1 if the workspace is full), returning a needed target to storage gives -1 , cycling a workspace item back to the workspace gives -4 , handling already-satisfied target ingredients gives penalties of -3 or -5 , and handling non-target ingredients gives penalties of -2 or -10 depending on whether the action is moving them away from or toward the main task area. Waiting while holding an ingredient, or while a needed target remains unavailable to the human, gives -5 . The human receives $+2$ for picking a still-needed target from the workspace and $+3$ for placing it on the sandwich. Returning a needed target to the workspace gives -1 , acting on an already-satisfied target gives -3 or -5 , acting on a non-target ingredient gives -2 or -10 , and attempting to place without holding an ingredient gives -1 for the workspace or -2 for the sandwich.

The preference reward is linear in interpretable feature vectors:

$$R_{\text{pref}}(s, a_R, a_H) = \mathbf{w}^\top \phi_R(s, a_R) + \mathbf{w}^\top \phi_H(s, a_H) = \mathbf{w}^\top (\phi_R(s, a_R) + \phi_H(s, a_H)).$$

The feature vectors ϕ_R and ϕ_H have the same 10 dimensions as $\mathbf{w}$. Ordering features are $+1$ when an action supports the preferred ingredient ordering, -1 when it violates that ordering, and 0 when the action is irrelevant. Workspace features evaluate the workspace size induced by the action, with features set to zero for size thresholds that are no longer relevant once too few target ingredients remain.

A.3 Block-Sorting Task

Domain basics. Fig. 5(b) shows the block-sorting task, in which a robot and a human sort nine blocks into two target bins. The blocks are the Cartesian product of three colors and three shapes:

- Blue: blue_square, blue_polygon, blue_trapezoid
- Pink: pink_square, pink_polygon, pink_trapezoid
- Yellow: yellow_square, yellow_polygon, yellow_trapezoid

From left to right, the five table regions are left, red, mid, green, and right. The red and green regions are the target bins, while left, mid, and right are staging regions. Initially, each block is placed in one of the staging regions. The robot can reach left, red, and mid; the human can reach mid, green, and right. The table regions do not impose capacity limits. The task is complete when every block is in either the red or green bin and neither agent is holding a block.

State space. The task state is represented by the allocation of blocks across the five table regions left, red, mid, green, and right, plus the two hand locations of the robot and human. This allocation determines whether each block has already reached a target bin, which blocks are reachable by each agent, whether either agent is holding a block, and whether the task is complete.

Action space and transitions. The action space of each agent has 15 possible actions: nine PICK(block) actions, five PLACE(block, location) actions for left, red, mid, green, and right, and WAIT. An agent can take only actions involving regions it can reach. Thus, when the robot is empty-handed, it may pick a block from left, red, or mid; when holding a block, it may place it in one of those same regions. The human follows the analogous rule over mid, green, and right. The middle region is reachable by both agents and enables handoff between the robot side and the human side. At each step, simultaneous actions move picked blocks into the acting agent’s hand or move held blocks to the selected reachable region. WAIT is always available.

Preference space. Human preferences specify which bin each block type should ultimately be sorted into. The preference weight vector has nine dimensions,

$$\mathbf{w} = [\text{blue_square}, \text{blue_trapezoid}, \text{blue_polygon}, \\ \text{pink_square}, \text{pink_trapezoid}, \text{pink_polygon}, \\ \text{yellow_square}, \text{yellow_trapezoid}, \text{yellow_polygon}]$$

A positive weight means that the corresponding block type is preferred in the red bin, a negative weight means that it is preferred in the green bin, and a value near zero means either bin is acceptable. Weights are normalized to unit Euclidean norm; sampled preferences are constrained so that not all block-bin dimensions have the same sign.

True preference options and environment initializations. Before the block-sorting task, participants selected one of six preference choices and used it consistently across all block-sorting trials. The true preference options were:

- *Preference A:* Blue and yellow blocks to red; pink blocks to green.
- *Preference B:* Blue and yellow blocks to green; pink blocks to red.
- *Preference C:* Squares and polygons to red; trapezoids to green.
- *Preference D:* Squares and polygons to green; trapezoids to red.
- *Preference E:* Blue square to red; pink polygon and yellow trapezoid to green; all other blocks either.
- *Preference F:* Blue polygon to green; pink trapezoid and yellow square to red; all other blocks either.

Each environment initialization specified one initial table layout, i.e., the blocks initially placed in the left, middle, and right staging regions:

- *Initial state 1:* Left: blue square, pink trapezoid, yellow polygon; Mid: pink square, blue polygon, yellow trapezoid; Right: yellow square, blue trapezoid, pink polygon.
- *Initial state 2:* Left: blue square, pink square; Mid: blue polygon, pink polygon, yellow polygon, yellow square, pink trapezoid; Right: blue trapezoid, yellow trapezoid.
- *Initial state 3:* Left: blue square, pink square, yellow square, blue polygon; Mid: pink polygon; Right: yellow polygon, pink trapezoid, blue trapezoid, yellow trapezoid.
- *Initial state 4:* Left: blue square, pink trapezoid; Mid: yellow square, pink square, blue polygon; Right: yellow polygon, blue trapezoid, pink polygon, yellow trapezoid.
- *Initial state 5:* Left: blue square, pink square, yellow square; Mid: blue polygon, pink polygon, yellow polygon; Right: blue trapezoid, pink trapezoid, yellow trapezoid.
- *Initial state 6:* Left: blue square, blue polygon, blue trapezoid; Mid: pink square, pink polygon, pink trapezoid; Right: yellow square, yellow polygon, yellow trapezoid.

Reward function. As in the sandwich task, the shared reward decomposes into goal and preference terms:

$$R(s, a_R, a_H) = R_{\text{goal}}(s, a_R, a_H) + \gamma R_{\text{pref}}(s, a_R, a_H), \text{ with } R_{\text{pref}} = \mathbf{w}^\top (\phi_R(s, a_R) + \phi_H(s, a_H))$$

The goal reward uses the same cases for the robot and the human. Each agent receives a step cost of -0.3 , and completing the sort gives a $+10$ bonus. Goal progress is encouraged by rewarding picks from staging regions ($+0.8$) and placements into either target bin ($+1.5$). Placing a block into the shared middle region receives a smaller reward ($+0.3$) because it can support handoff between agents. Actions that undo or delay progress are penalized: picking from a target bin gives -0.6 , placing into a side staging region gives -0.5 , attempting to place without holding a block gives -1.0 , and waiting while holding a block gives -1.0 . Waiting without holding a block has no additional goal reward.

The feature vector $\phi_{\text{agent}} \in \{-1, 0, 1\}^9$, where agent is either the human or robot, marks the block type affected by the action. Placing a block in the red bin, or removing it from the green bin, gives a $+1$ feature for that block; placing it in the green bin, or removing it from the red bin, gives a -1 feature. Actions involving the shared middle region are signed according to which side can next make progress: robot actions that take a block from mid or human actions that place a block in mid contribute $+1$, while human actions that take a block from mid or robot actions that place a block in mid contribute -1 . Otherwise, the feature will be 0 .

A.4 Additional Preference-Learning Parameters

Table 3 summarizes the shared preference-learning parameters used for our study. Following the learning setup used by Candon et al. [25], we used one shared set of preference-learning parameters across different collaboration tasks and implication sources. This choice keeps the preference-learning procedure general and makes the comparisons focus on the source of implication labels.

Table 3: Preference-learning parameters for user study in MuJoCo simulation.

Parameter	Value	Description
learning_rate	1×10^{-4}	Step size for preference-weight updates
weight_samples	10	Number of samples representing the belief over weight
max_iteration	1000	Maximum iterations for each preference update
convergence_threshold	1×10^{-3}	Threshold for preference update convergence
β_R	1.0	Robot action rationality coefficient
β_{exp}	1.0	Explicit-feedback rationality coefficient
β_{imp}	1.0	Implicit-feedback rationality coefficient
γ	5.0	Relative weight on preference reward vs. goal reward

A.5 Demographic Information

Participants completed a demographic survey at the beginning of the study. The statistics for the 18 participants are summarized below:

- Gender: 12 participants were male and 6 were female.
- Age: The participants’ mean age was 24 years old (SD = 7.2), with ages ranging from 18 to 50.
- Familiarity with robotics: Participants rated their familiarity with robotics on a 7-point Likert scale, where 1 indicated “not familiar at all” and 7 indicated “very familiar.” The mean rating was 4.78 (SD = 1.40).
- Frequency of interacting with robots: 6 participants reported interacting with robots less than once a month, 6 reported once a month, 1 reported once a week, 3 reported 2–3 times a week, 1 reported 4–6 times a week, and 1 reported daily.

B Details of the GP Model for IMPLIED

B.1 Derivation of λ in Eq. (4)

We use the variance $v(\mathbf{o})$ to derive $\lambda(\mathbf{o})$ in Eq. (4). The function $\lambda(\mathbf{o})$ returns an observation-specific weight that balances the contributions of the fixed-rule prior and the learned GP. Since implication labels are binary, the learned GP outputs a probability for the positive implication label, which can be interpreted as the parameter of a Bernoulli distribution over accepted/rejected labels. The most uncertain case occurs when this probability is $p = 0.5$, where the two labels are equally likely. The corresponding Bernoulli variance is therefore maximized at $v_{\max} = p(1 - p) = 0.25$. We use this maximum variance to normalize the GP uncertainty and let:

$$\lambda(\mathbf{o}) = 1 - v(\mathbf{o})/v_{\max} = 1 - 4v(\mathbf{o}) \in [0, 1] \quad (5)$$

This choice gives a bounded weight in $[0, 1]$ and makes the confidence in the learned GP higher when the model is more certain, i.e., when the variance is lower. This confidence also reflects distance from the decision boundary, which we set to 0.5 probability in our experiments. If the probability samples behave like Bernoulli draws with mean $\mu(\mathbf{o})$, then:

$$\begin{aligned}
 v(\mathbf{o}) &\approx \mu(\mathbf{o})(1 - \mu(\mathbf{o})) \\
 \lambda(\mathbf{o}) &= 1 - 4v(\mathbf{o}) \\
 &\approx 1 - 4\mu(\mathbf{o})(1 - \mu(\mathbf{o})) \\
 &= 1 - 4\mu(\mathbf{o}) + 4\mu(\mathbf{o})^2 \\
 &= 4(\mu(\mathbf{o}) - 0.5)^2.
 \end{aligned} \quad (7)$$

Thus, $\lambda(\mathbf{o})$ is small when $\mu(\mathbf{o})$ is near 0.5 and large when $\mu(\mathbf{o})$ is near 0 or 1.

B.2 GP Hyperparameter Search

For the GP classifier in Sec. 4, the covariance matrix K is computed from Z-score normalized observation features. Let $\tilde{z}(\mathbf{o})$ denote the standardized version of $z(\mathbf{o})$. For two observations $\mathbf{o}_i$ and $\mathbf{o}_j$, we use $K_{ij} = k_\theta(\tilde{z}(\mathbf{o}_i), \tilde{z}(\mathbf{o}_j))$, with $k_\theta(\tilde{z}, \tilde{z}') = \sigma_\ell^2 \tilde{z}^\top \tilde{z}' + \sigma_b^2$. Here, the parameters $\theta = (\sigma_\ell^2, \sigma_b^2)$, where σ_ℓ^2 controls the scale of the linear dependence on the feature inputs, and σ_b^2 is the additive bias variance, allowing the latent acceptance function to include a global offset. Thus, the latent function values used by $h_\theta = \sigma \circ f$ follow the GP prior induced by this kernel, $f \sim \mathcal{N}(\mathbf{0}, K)$.

For experiments with human-in-the-loop simulation data, we used leave-one-participant-out cross-validation, as explained in Appx. C. For the GP, the train/val split served to select the GP kernel hyperparameters $\theta = (\sigma_\ell^2, \sigma_b^2)$ separately within each leave-one-participant-out fold. We selected θ from a finite grid formed by the Cartesian product of $\sigma_\ell^2 \in \{0.25, 0.5, 1.0, 2.0, 4.0\}$ and $\sigma_b^2 \in \{0.05, 0.1, 0.2, 0.5, 1.0\}$. Specifically, in each fold, one participant was held out for testing, and the remaining participants were split into 80% training and 20% validation participants. For each candidate θ , we fit the GP classifier using cumulative direct evidence from the training participants and evaluate implication-prediction performance on the validation participants. The parameter setting with the best validation performance is then fixed and used for making predictions on the held-out test participant. Implication labels from the held-out participant are never used for hyperparameter selection.

For experiments with real-world human-robot collaboration, we use a single parameter setting selected from the search performed for the simulation data. Specifically, we chose the parameters most frequently selected as optimal across the simulation folds, yielding $\sigma_\ell^2 = 1.0$ and $\sigma_b^2 = 0.2$. These values were used for all participants in the real-world human-robot collaboration (Sec. 5.2).

B.3 Sensitivity to the Threshold τ

We set $\tau = 0.5$ in Sec. 4 because it is the standard decision boundary for binary classification. To verify that the choice is reasonable, we conducted a sensitivity analysis by re-running the evaluation of Sec. 5.1 on data from Sec. 3 with $\tau \in \{0.25, 0.5, 0.75\}$. Table 4 reports the results. In sandwich-making, $\tau = 0.5$ significantly outperformed both alternatives on all three metrics ($p < 0.05$). In block-sorting, $\tau = 0.5$ significantly outperformed $\tau = 0.25$ and $\tau = 0.75$ on F1 ($p < 0.01$), and outperformed $\tau = 0.25$ on L2 error ($p < 0.05$). We further swept τ in increments of 0.05 for both tasks and found no significant difference from $\tau = 0.5$ for $\tau \in [0.40, 0.60]$.

Table 4: Sensitivity analysis for the threshold τ .

Task	τ	F1 $\uparrow$	L2 Error $\downarrow$	Conflict Prop. $\downarrow$
Sandwich-making	0.25	0.77 $\pm$ 0.03	0.84 $\pm$ 0.06	0.18 $\pm$ 0.04
	0.50	0.84 $\pm$ 0.01	0.71 $\pm$ 0.06	0.13 $\pm$ 0.01
	0.75	0.76 $\pm$ 0.02	0.80 $\pm$ 0.07	0.17 $\pm$ 0.03
Block-sorting	0.25	0.72 $\pm$ 0.04	0.56 $\pm$ 0.08	0.16 $\pm$ 0.05
	0.50	0.87 $\pm$ 0.01	0.42 $\pm$ 0.07	0.12 $\pm$ 0.03
	0.75	0.74 $\pm$ 0.01	0.49 $\pm$ 0.07	0.16 $\pm$ 0.04

B.4 Comparison with a Simpler Learned Classifier

We tested whether a simpler learner for IMPLIED could lead to performance close to that of the GP. We swapped P_{learned} for a K-Nearest Neighbors (KNN) classifier, keeping the same set of features $z(\mathbf{o})$, the Z-score normalization, and the mixture in Eq. (4). We selected $k \in \{3, 5, 7, 9\}$ using the leave-one-participant-out cross validation mentioned in Appx. B.2. We set $\mu(\mathbf{o})$ to the fraction of neighbors voting for the accepted label and used the corresponding Bernoulli variance $v(\mathbf{o}) = \mu(\mathbf{o})(1 - \mu(\mathbf{o}))$ to compute $\lambda(\mathbf{o}) = 1 - 4v(\mathbf{o})$.

On the recorded interaction data from Sec. 3, GP outperformed KNN at the earliest shared task-completion step. In sandwich-making, GP led to significantly higher F1 (0.84 $\pm$ 0.01 vs. 0.75 $\pm$ 0.03, $p < 0.01$), lower L2 error (0.71 $\pm$ 0.06 vs. 0.89 $\pm$ 0.07, $p < 0.001$), and lower conflict proportion

(0.13 ± 0.01 vs. 0.19 ± 0.03 , $p < 0.05$) than KNN. In block-sorting, GP also led to significantly higher F1 (0.87 ± 0.01 vs. 0.71 ± 0.03 , $p < 0.001$), lower L2 error (0.42 ± 0.07 vs. 0.60 ± 0.08 , $p < 0.001$), and lower conflict proportion (0.12 ± 0.03 vs. 0.20 ± 0.05 , $p < 0.05$) than KNN.

C Learning with Human-in-the-Loop Simulation Data

In Sec. 5.1, we evaluate implication prediction and preference learning using both implicit feedback from human actions and explicit feedback on robot actions. For these experiments, we use leave-one-participant-out cross-validation to measure generalization performance. In each fold, one participant was held out for testing, and the remaining participants were split into 80% training and 20% validation participants. The training and validation data served to fit hyperparameters, e.g., for the GP in IMPLIED and the No-Prior baseline, as mentioned in Appx. B.

For the Large Language Model baselines, we used Gemini 3.0 Flash, with a temperature of 0 (making the output the most deterministic possible) and a high thinking level. We chose this particular model because it is faster than other frontier “Pro” models, like Gemini 3.0 Pro, allowing for learning *during* collaborations as opposed to afterwards.

The next sections provide additional analyses that isolate each feedback modality. The evaluation protocol is the same as described before and in Sec. 5.1: we hold the recorded MuJoCo interaction

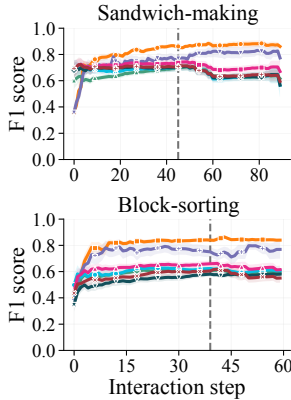


Figure 7: F1 score for classifying unchosen robot actions over time with **explicit feedback only**. See first column of Tab. 5 for legend.

Table 5: Downstream preference-learning performance using **explicit feedback only**. Learning is achieved per Eq. (1) with gradient ascent, but the actions of the human are not considered as evidence for learning. Each row in the table shows results for different implication approaches. Values are mean $\pm$ standard error at the earliest task-completion timestep across participants. **Human** uses ground-truth implication labels for explicit feedback, and **IMPLIED** highlights our proposed method.

Implication	Sandwich-making		Block-sorting	
	L2 Error $\downarrow$	Conflict Prop. $\downarrow$	L2 Error $\downarrow$	Conflict Prop. $\downarrow$
Human	0.83 ± 0.05	0.15 ± 0.02	0.62 ± 0.06	0.15 ± 0.03
IMPLIED	0.87 ± 0.06	0.16 ± 0.02	0.66 ± 0.07	0.14 ± 0.02
Fixed-Rule	1.03 ± 0.06	0.25 ± 0.04	0.79 ± 0.06	0.26 ± 0.04
No-Prior	1.01 ± 0.06	0.26 ± 0.03	0.91 ± 0.08	0.30 ± 0.04
LLM	1.09 ± 0.07	0.29 ± 0.06	0.85 ± 0.05	0.33 ± 0.04
LLM-H	1.07 ± 0.08	0.27 ± 0.04	0.78 ± 0.06	0.26 ± 0.04
LLM-B	1.08 ± 0.07	0.30 ± 0.05	0.87 ± 0.06	0.31 ± 0.05
LLM-HB	1.05 ± 0.08	0.24 ± 0.05	0.84 ± 0.06	0.34 ± 0.04

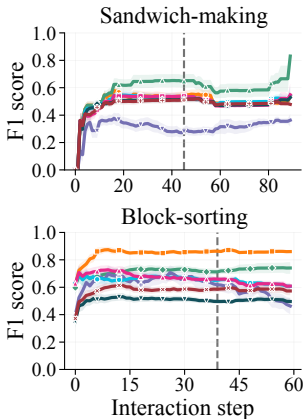


Figure 8: F1 score for classifying unchosen human actions over time using **implicit feedback only**. See first column of Tab. 6 for legend.

Table 6: Downstream preference-learning performance using **implicit feedback only**. Learning is achieved per Eq. (1) with gradient ascent, but no feedback about the robot actions is considered as evidence for learning. Each row in the table shows results for different implication approaches. Values are mean $\pm$ standard error at the earliest task-completion timestep across participants. **Human** uses ground-truth implication labels for implicit feedback, and **IMPLIED** highlights our proposed method.

Implication	Sandwich-making		Block-sorting	
	L2 Error $\downarrow$	Conflict Prop. $\downarrow$	L2 Error $\downarrow$	Conflict Prop. $\downarrow$
Human	0.81 ± 0.05	0.25 ± 0.03	0.53 ± 0.08	0.16 ± 0.04
IMPLIED	0.84 ± 0.05	0.25 ± 0.03	0.56 ± 0.08	0.18 ± 0.05
Fixed-Rule	0.86 ± 0.06	0.24 ± 0.03	0.66 ± 0.07	0.28 ± 0.06
No-Prior	1.01 ± 0.07	0.30 ± 0.04	0.74 ± 0.07	0.31 ± 0.05
LLM	0.82 ± 0.06	0.27 ± 0.03	0.67 ± 0.07	0.26 ± 0.05
LLM-H	0.85 ± 0.06	0.29 ± 0.04	0.71 ± 0.08	0.26 ± 0.05
LLM-B	0.77 ± 0.06	0.26 ± 0.03	0.68 ± 0.07	0.25 ± 0.06
LLM-HB	0.83 ± 0.06	0.26 ± 0.03	0.65 ± 0.07	0.27 ± 0.05

trajectories fixed, apply each implication method to construct accepted and rejected feedback sets, and evaluate both implication-label prediction and downstream preference learning.

Explicit feedback only. Fig. 7 and Table 5 show results when preference learning uses only explicit feedback on the robot’s executed actions and the corresponding implications for unchosen robot actions. The overall trend is consistent with the combined-feedback results: IMPLIED led to a higher F1 score faster than the non-human baselines. Also, at the earliest task-completion step across participants (vertical dashed line in Fig. 7), IMPLIED achieved the highest F1-score in both tasks. In sandwich-making, linear mixed-effects analyses showed that IMPLIED (0.86 ± 0.02) achieved significantly higher F1-score than the non-human baselines at the earliest task-completion step ($p < 0.05$). Likewise, in block-sorting, IMPLIED (0.84 ± 0.01) got significantly higher F1 at that step ($p < 0.05$). In downstream learning (Table 5), a linear mixed-effects analysis for the sandwich-making task showed that IMPLIED achieved significantly lower L2 error ($p < 0.01$) than all non-human baselines, except for No-Prior, and significantly lower conflict proportion ($p < 0.05$) than all non-human baselines. For block-sorting, IMPLIED also significantly outperformed all other non-human baselines on L2 error ($p < 0.05$) and conflict proportion ($p < 0.01$). These results suggest that, even when only explicit robot-action feedback is available, revising implications from accumulated direct evidence improves the feedback sets used for preference learning.

Implicit feedback only. Fig. 8 and Table 6 show results when preference learning uses only implicit feedback from the human’s executed task actions and the corresponding implications for unchosen human actions. In block-sorting, the results again follow the main trend: IMPLIED led to a higher F1 score faster than the non-human baselines. A linear mixed-effects analysis showed that IMPLIED (0.86 ± 0.01) achieved significantly higher F1-score than the non-human baselines at the earliest task-completion step (vertical dashed line in Fig. 8), with $p < 0.0001$. In downstream learning, a linear mixed-effects analysis showed that IMPLIED achieved significantly lower L2 error ($p < 0.05$) and conflict proportion ($p < 0.05$) than all non-human baselines in block sorting. This indicates that the method can also revise implications derived from human action choices, not only explicit evaluations of robot actions.

When learning in the sandwich-making task with implicit feedback only, we found different results than all of our prior results: IMPLIED underperformed the Fixed-Rule baseline in terms of F1 classification score (Fig. 8). We attribute this result to the design of the sandwich-making task. In this task, the implicit-feedback action space available to the human was more constrained than for the block-sorting task, and human implication labels for implicit feedback had very low inconsistency against the fixed rule (0.04 ± 0.01 timestep inconsistencies and 0.03 ± 0.01 label inconsistencies). Despite this, IMPLIED led to 0.84 ± 0.05 L2 Error on preference learning versus 0.86 ± 0.06 at the earliest task-completion step (which were not significantly different, $p = 0.3945$). Also, IMPLIED led to very similar Conflict Proportion performance (0.25 ± 0.03 vs. 0.24 ± 0.03), suggesting that IMPLIED does not hinder preference learning in a meaningful way when the fixed-rule prior is close to the optimal implications for a given task.

D Prompts for LLM Baselines

Sec. 5.1 compares IMPLIED with four LLM baselines that differ in whether the model receives prior interaction history and/or the current preference-belief summary. Below, we show representative prompts from the most informative condition, LLM-HB, which includes both the interaction history (H) and the preference-belief description (B). The first prompt corresponds to the sandwich-making task; the second one to the block-sorting task. The other LLM baselines use the same prompt structure but remove the corresponding information parts: LLM removes both history and belief, LLM-H removes only the belief part, and LLM-B removes only the history part. Each prompt asks the LLM to label the current timestep’s unchosen robot and human actions as *accepted* or *rejected* and to return strict JSON.

Example LLM-HB prompt for the sandwich-making task.

You are a robot collaborating with a human to make a sandwich together.
In the environment, there are three areas: the storage, the workspace, and the sandwich area.

The storage initially contains ingredient instances from four ingredient types: cheese, bread, meat, and veggies.

These ingredients are selected and used to make the sandwich.

The workspace starts empty. You (Robot) can:

- PICK ingredients from storage to your hand,
- PUT ingredients from your hand to workspace,
- PICK ingredients from workspace to your hand,
- PUT ingredients from your hand back to storage,
- WAIT.

The workspace can contain at most 4 ingredients.

The human can:

- PICK ingredients from workspace,
- PUT ingredients from hand to sandwich,
- PUT ingredients from hand back to workspace,
- WAIT.

You can only reach storage/workspace. The human can only reach workspace/sandwich.

In each step, you (Robot) and the human act simultaneously.

The human also gives explicit binary feedback (positive or negative) on your executed action.

The objective of the collaboration is to achieve the task goal efficiently while also ensuring that robot and human actions align with the human's preference for how the task should be performed.

In the collaboration, the human has a fixed latent preference that defines how the goal should be achieved, and the human consistently follows this preference when selecting actions and providing feedback.

The human's actions and explicit binary feedback to you reflect both task-goal progress and this preference.

The goal of each sandwich-making task is to complete a specific target sandwich in as few steps as possible, and the task goal may change when a new task begins.

The human's preference over the collaboration specifies how the target sandwich should be made, and this preference does not change across tasks.

In particular, the human cares about the following aspects of the collaboration:

1) The order in which ingredient types are used (6 dimensions):

- cheese before bread
- cheese before meat
- cheese before veggie
- bread before meat
- bread before veggie
- meat before veggie

Here, ingredient types are cheese, bread, meat, and veggie. Each dimension reflects whether one ingredient type should generally be completed before another.

Example: 'cheese before bread' means required cheese ingredients should be placed before required bread ingredients.

2) How full the workspace should be during collaboration (4 dimensions):

- whether the human prefers the workspace to contain at most 0 ingredients,
- at most 1 ingredient,
- at most 2 ingredients,
- or at most 3 ingredients, whenever possible.

In each task, you know the target sandwich recipe goal, but you do not know the human's latent preference. Infer it from interaction history and feedback.

Below is the history of the collaboration tasks between you and the human:

Task 1:

Sandwich recipe goal: 1 cheese, 2 breads, 1 bacon, 1 ham, 1 tomato, 1 cucumber

Step 0

State:

- human_hand: (empty)
- robot_hand: (empty)

```

- sandwich: (empty)
- storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3,
bread_slice4, cheese1, cheese2, cheese3, cheese4, cucumber1, cucumber2,
cucumber3, ham, tomato1, tomato2
- workspace: (empty)
Executed actions:
- You (Robot): PICK cucumber2 storage (veggie)
- Human: WAIT
Human explicit feedback for your action: negative
Step 1
State:
- human_hand: (empty)
- robot_hand: cucumber2
- sandwich: (empty)
- storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3,
bread_slice4, cheese1, cheese2, cheese3, cheese4, cucumber1, cucumber3, ham,
tomato1, tomato2
- workspace: (empty)
Executed actions:
- You (Robot): PUT cucumber2 workspace (veggie)
- Human: WAIT
Human explicit feedback for your action: negative
Step 2
State:
- human_hand: (empty)
- robot_hand: (empty)
- sandwich: (empty)
- storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3,
bread_slice4, cheese1, cheese2, cheese3, cheese4, cucumber1, cucumber3, ham,
tomato1, tomato2
- workspace: cucumber2
Executed actions:
- You (Robot): PICK ham storage (meat)
- Human: WAIT
Human explicit feedback for your action: negative
Step 3
State:
- human_hand: (empty)
- robot_hand: ham
- sandwich: (empty)
- storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3,
bread_slice4, cheese1, cheese2, cheese3, cheese4, cucumber1, cucumber3,
tomato1, tomato2
- workspace: cucumber2
Executed actions:
- You (Robot): PUT ham workspace (meat)
- Human: WAIT
Human explicit feedback for your action: negative
Step 4
State:
- human_hand: (empty)
- robot_hand: (empty)
- sandwich: (empty)
- storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3,
bread_slice4, cheese1, cheese2, cheese3, cheese4, cucumber1, cucumber3,
tomato1, tomato2
- workspace: cucumber2, ham
Executed actions:
- You (Robot): PICK cheese3 storage (cheese)
- Human: WAIT
Human explicit feedback for your action: positive
Step 5
State:
- human_hand: (empty)
- robot_hand: cheese3

```



```

- sandwich: (empty)
- storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3,
bread_slice4, cheese1, cheese2, cheese4, cucumber1, cucumber3, tomato1,
tomato2
- workspace: cucumber2, ham
Executed actions:
- You (Robot): PUT cheese3 workspace (cheese)
- Human: WAIT
Human explicit feedback for your action: positive
Step 6
State:
- human_hand: (empty)
- robot_hand: (empty)
- sandwich: (empty)
- storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3,
bread_slice4, cheese1, cheese2, cheese4, cucumber1, cucumber3, tomato1,
tomato2
- workspace: cheese3, cucumber2, ham
Executed actions:
- You (Robot): PICK bread_slice1 storage (bread)
- Human: PICK cheese3 workspace (cheese)
Human explicit feedback for your action: negative
Step 7
State:
- human_hand: cheese3
- robot_hand: bread_slice1
- sandwich: (empty)
- storage: bacon, beef_patty, bread_slice2, bread_slice3, bread_slice4,
cheese1, cheese2, cheese4, cucumber1, cucumber3, tomato1, tomato2
- workspace: cucumber2, ham
Executed actions:
- You (Robot): PUT bread_slice1 workspace (bread)
- Human: PUT cheese3 sandwich (cheese)
Human explicit feedback for your action: negative
Step 8
State:
- human_hand: (empty)
- robot_hand: (empty)
- sandwich: cheese3
- storage: bacon, beef_patty, bread_slice2, bread_slice3, bread_slice4,
cheese1, cheese2, cheese4, cucumber1, cucumber3, tomato1, tomato2
- workspace: bread_slice1, cucumber2, ham
Executed actions:
- You (Robot): PICK bacon storage (meat)
- Human: PICK ham workspace (meat)
Human explicit feedback for your action: positive
Step 9
State:
- human_hand: ham
- robot_hand: bacon
- sandwich: cheese3
- storage: beef_patty, bread_slice2, bread_slice3, bread_slice4, cheese1,
cheese2, cheese4, cucumber1, cucumber3, tomato1, tomato2
- workspace: bread_slice1, cucumber2
Executed actions:
- You (Robot): PUT bacon workspace (meat)
- Human: PUT ham sandwich (meat)
Human explicit feedback for your action: positive
Step 10
State:
- human_hand: (empty)
- robot_hand: (empty)
- sandwich: cheese3, ham
- storage: beef_patty, bread_slice2, bread_slice3, bread_slice4, cheese1,
cheese2, cheese4, cucumber1, cucumber3, tomato1, tomato2

```

```

- workspace: bacon, bread.slice1, cucumber2
Executed actions:
- You (Robot): PICK bread.slice2 storage (bread)
- Human: PICK bacon workspace (meat)
Human explicit feedback for your action: positive
Step 11
State:
- human.hand: bacon
- robot.hand: bread.slice2
- sandwich: cheese3, ham
- storage: beef_patty, bread.slice3, bread.slice4, cheese1, cheese2,
cheese4, cucumber1, cucumber3, tomato1, tomato2
- workspace: bread.slice1, cucumber2
Executed actions:
- You (Robot): PUT bread.slice2 workspace (bread)
- Human: PUT bacon sandwich (meat)
Human explicit feedback for your action: positive
Step 12
State:
- human.hand: (empty)
- robot.hand: (empty)
- sandwich: cheese3, ham, bacon
- storage: beef_patty, bread.slice3, bread.slice4, cheese1, cheese2,
cheese4, cucumber1, cucumber3, tomato1, tomato2
- workspace: bread.slice1, bread.slice2, cucumber2
Executed actions:
- You (Robot): PICK tomato2 storage (veggie)
- Human: PICK bread.slice1 workspace (bread)
Human explicit feedback for your action: positive
Step 13
State:
- human.hand: bread.slice1
- robot.hand: tomato2
- sandwich: cheese3, ham, bacon
- storage: beef_patty, bread.slice3, bread.slice4, cheese1, cheese2,
cheese4, cucumber1, cucumber3, tomato1
- workspace: bread.slice2, cucumber2
Executed actions:
- You (Robot): PUT tomato2 workspace (veggie)
- Human: PUT bread.slice1 sandwich (bread)
Human explicit feedback for your action: positive
Step 14
State:
- human.hand: (empty)
- robot.hand: (empty)
- sandwich: cheese3, ham, bacon, bread.slice1
- storage: beef_patty, bread.slice3, bread.slice4, cheese1, cheese2,
cheese4, cucumber1, cucumber3, tomato1
- workspace: bread.slice2, cucumber2, tomato2
Executed actions:
- You (Robot): WAIT
- Human: PICK tomato2 workspace (veggie)
Human explicit feedback for your action: positive
Step 15
State:
- human.hand: tomato2
- robot.hand: (empty)
- sandwich: cheese3, ham, bacon, bread.slice1
- storage: beef_patty, bread.slice3, bread.slice4, cheese1, cheese2,
cheese4, cucumber1, cucumber3, tomato1
- workspace: bread.slice2, cucumber2
Executed actions:
- You (Robot): WAIT
- Human: PUT tomato2 sandwich (veggie)
Human explicit feedback for your action: positive

```

Step 16

State:

- human.hand: (empty)
- robot.hand: (empty)
- sandwich: cheese3, ham, bacon, bread.slice1, tomato2
- storage: beef_patty, bread.slice3, bread.slice4, cheese1, cheese2, cheese4, cucumber1, cucumber3, tomato1
- workspace: bread.slice2, cucumber2

Executed actions:

- You (Robot): PICK bread.slice2 workspace (bread)
- Human: PICK cucumber2 workspace (veggie)

Human explicit feedback for your action: negative

Step 17

State:

- human.hand: cucumber2
- robot.hand: bread.slice2
- sandwich: cheese3, ham, bacon, bread.slice1, tomato2
- storage: beef_patty, bread.slice3, bread.slice4, cheese1, cheese2, cheese4, cucumber1, cucumber3, tomato1
- workspace: (empty)

Executed actions:

- You (Robot): PUT bread.slice2 workspace (bread)
- Human: PUT cucumber2 sandwich (veggie)

Human explicit feedback for your action: positive

Step 18

State:

- human.hand: (empty)
- robot.hand: (empty)
- sandwich: cheese3, ham, bacon, bread.slice1, tomato2, cucumber2
- storage: beef_patty, bread.slice3, bread.slice4, cheese1, cheese2, cheese4, cucumber1, cucumber3, tomato1
- workspace: bread.slice2

Executed actions:

- You (Robot): WAIT
- Human: PICK bread.slice2 workspace (bread)

Human explicit feedback for your action: positive

Step 19

State:

- human.hand: bread.slice2
- robot.hand: (empty)
- sandwich: cheese3, ham, bacon, bread.slice1, tomato2, cucumber2
- storage: beef_patty, bread.slice3, bread.slice4, cheese1, cheese2, cheese4, cucumber1, cucumber3, tomato1
- workspace: (empty)

Executed actions:

- You (Robot): WAIT
- Human: PUT bread.slice2 sandwich (bread)

Human explicit feedback for your action: positive

Task 2:

Sandwich recipe goal: 2 cheeses, 2 breads, 1 beef patty, 1 tomato, 1 cucumber

Step 0

State:

- human.hand: (empty)
- robot.hand: (empty)
- sandwich: (empty)
- storage: bacon, beef_patty, bread.slice1, bread.slice2, bread.slice3, bread.slice4, cheese1, cheese2, cheese3, cheese4, cucumber1, cucumber2, cucumber3, ham, tomato1, tomato2
- workspace: (empty)

Executed actions:

- You (Robot): PICK cheese2 storage (cheese)
- Human: WAIT

Human explicit feedback for your action: positive

Step 1

State:
 - human_hand: (empty)
 - robot_hand: cheese2
 - sandwich: (empty)
 - storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3, bread_slice4, cheese1, cheese3, cheese4, cucumber1, cucumber2, cucumber3, ham, tomato1, tomato2
 - workspace: (empty)
 Executed actions:
 - You (Robot): PUT cheese2 workspace (cheese)
 - Human: WAIT
 Human explicit feedback for your action: positive

Step 2

State:
 - human_hand: (empty)
 - robot_hand: (empty)
 - sandwich: (empty)
 - storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3, bread_slice4, cheese1, cheese3, cheese4, cucumber1, cucumber2, cucumber3, ham, tomato1, tomato2
 - workspace: cheese2
 Executed actions:
 - You (Robot): PICK cheese3 storage (cheese)
 - Human: PICK cheese2 workspace (cheese)
 Human explicit feedback for your action: positive

Step 3

State:
 - human_hand: cheese2
 - robot_hand: cheese3
 - sandwich: (empty)
 - storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3, bread_slice4, cheese1, cheese4, cucumber1, cucumber2, cucumber3, ham, tomato1, tomato2
 - workspace: (empty)
 Executed actions:
 - You (Robot): PUT cheese3 workspace (cheese)
 - Human: PUT cheese2 sandwich (cheese)
 Human explicit feedback for your action: positive

Current step (Task 2, Step 4):

Sandwich recipe goal: 2 cheeses, 2 breads, 1 beef patty, 1 tomato, 1 cucumber

- State:
 - human_hand: (empty)
 - robot_hand: (empty)
 - sandwich: cheese2
 - storage: bacon, beef_patty, bread_slice1, bread_slice2, bread_slice3, bread_slice4, cheese1, cheese4, cucumber1, cucumber2, cucumber3, ham, tomato1, tomato2
 - workspace: cheese3
 - Executed actions:
 - You (Robot): WAIT
 - Human: PICK cheese3 workspace (cheese)
 - Human explicit feedback for your action: negative

Latest belief over preference before incorporating the CURRENT step:

- cheese before bread: 10/10 strong accept, 0/10 weak accept, 0/10 weak reject, 0/10 strong reject
 - cheese before meat: 3/10 strong accept, 7/10 weak accept, 0/10 weak reject, 0/10 strong reject
 - cheese before veggie: 0/10 strong accept, 9/10 weak accept, 1/10 weak reject, 0/10 strong reject
 - bread before meat: 0/10 strong accept, 0/10 weak accept, 10/10 weak reject, 0/10 strong reject
 - bread before veggie: 0/10 strong accept, 1/10 weak accept, 9/10 weak reject, 0/10 strong reject

- meat before veggie: 3/10 strong accept, 7/10 weak accept, 0/10 weak reject, 0/10 strong reject
- workspace at most 0 ingredients: 0/10 strong accept, 3/10 weak accept, 7/10 weak reject, 0/10 strong reject
- workspace at most 1 ingredient: 0/10 strong accept, 0/10 weak accept, 10/10 weak reject, 0/10 strong reject
- workspace at most 2 ingredients: 0/10 strong accept, 0/10 weak accept, 10/10 weak reject, 0/10 strong reject
- workspace at most 3 ingredients: 0/10 strong accept, 3/10 weak accept, 7/10 weak reject, 0/10 strong reject

- The unchosen actions:

Your unchosen actions: PICK bacon storage (meat), PICK ham storage (meat), PICK beef_patty storage (meat), PICK bread.slice storage (bread), PICK cheese storage (cheese), PICK cheese3 workspace (cheese), PICK tomato storage (veggie), PICK cucumber storage (veggie)

Human unchosen actions: WAIT

Now, based on the collaboration history (states, actions, human feedback), and the latest belief over the human's preference before the CURRENT step, predict the human feedback for each action that you and the human did not choose, in the CURRENT step only, considering both the task goal and the human's preference.

Label each unchosen action as:

- accepted: accepted behavior with respect to both task-goal progress and inferred human preference.
- rejected: rejected behavior with respect to task-goal progress and/or inferred human preference.

Rules:

- 1) Only include unique action keys from the provided unchosen actions of you ("Robot") and the human ("Human").
- 2) Every label must be exactly 'accepted' or 'rejected'.
- 3) Return JSON only.

Return STRICT JSON only with this shape:

```
{
  "Robot": {
    "PICK bacon storage (meat)": "accepted|rejected",
    "PICK ham storage (meat)": "accepted|rejected",
    "PICK beef_patty storage (meat)": "accepted|rejected",
    "PICK bread.slice storage (bread)": "accepted|rejected",
    "PICK cheese storage (cheese)": "accepted|rejected",
    "PICK cheese3 workspace (cheese)": "accepted|rejected",
    "PICK tomato storage (veggie)": "accepted|rejected",
    "PICK cucumber storage (veggie)": "accepted|rejected"
  },
  "Human": {
    "WAIT": "accepted|rejected"
  }
}
```

Example LLM-HB prompt for the block-sorting task.

You are a robot collaborating with a human to sort the blocks on the table together.

In the environment, there are nine unique blocks:

- blue square
- blue polygon
- blue trapezoid
- pink square
- pink polygon
- pink trapezoid
- yellow square
- yellow polygon
- yellow trapezoid

From left to right, there are five areas: left panel, red bin, middle panel, green bin, right panel.

The blocks are initially placed in the left panel, the middle panel, or the right panel.

You can only reach the left panel, the middle panel, and the red bin.

The human can only reach the middle panel, the right panel, and the green bin.

The middle panel can be reached by both you and the human.

Both you and the human can pick blocks from and place blocks into reachable areas. All areas can contain any number of blocks.

In each step, you (Robot) and the human act simultaneously.

The human also gives explicit binary feedback (positive or negative) on your executed action.

The objective of the collaboration is to achieve the task goal efficiently while also ensuring that robot and human actions align with the human's preference for how the task should be performed.

In the collaboration, the human has a fixed latent preference that defines how the goal should be achieved, and the human consistently follows this preference when selecting actions and providing feedback.

The human's actions and explicit binary feedback to you reflect both task-goal progress and this preference.

The goal of each block-sorting task is to place all blocks into either the red bin or the green bin in as few steps as possible.

The human's preference over the collaboration specifies how the blocks should be sorted, and this preference does not change across tasks.

In particular, the human cares about which bin each block should be sorted into.

There are 9 preference dimensions, one for each unique block type:

- blue square
- blue trapezoid
- blue polygon
- pink square
- pink trapezoid
- pink polygon
- yellow square
- yellow trapezoid
- yellow polygon

For each block type, a positive preference means the human prefers that block in the red bin, while a negative preference means the human prefers that block in the green bin.

In each task, you know the sorting goal, but you do not know the human's latent preference. Infer it from interaction history and feedback.

Below is the history of the collaboration tasks between you and the human:

Task 1:

Block-sorting goal: place all blocks into either the red bin or the green bin

Step 0

State:

- human_hand: (empty)
- robot_hand: (empty)
- left: blue_square, pink_trapezoid
- red: (empty)
- mid: blue_polygon, pink_square, yellow_square
- green: (empty)
- right: blue_trapezoid, pink_polygon, yellow_polygon, yellow_trapezoid

Executed actions:

- You (Robot): PICK pink_square mid
- Human: PICK yellow_trapezoid right

Human explicit feedback for your action: negative

Step 1

State:

- human_hand: yellow_trapezoid
- robot_hand: pink_square
- left: blue_square, pink_trapezoid

```

- red: (empty)
- mid: blue_polygon, yellow_square
- green: (empty)
- right: blue_trapezoid, pink_polygon, yellow_polygon
Executed actions:
- You (Robot): PLACE pink_square red
- Human: PLACE yellow_trapezoid mid
Human explicit feedback for your action: negative
Step 2
State:
- human_hand: (empty)
- robot_hand: (empty)
- left: blue_square, pink_trapezoid
- red: pink_square
- mid: blue_polygon, yellow_square, yellow_trapezoid
- green: (empty)
- right: blue_trapezoid, pink_polygon, yellow_polygon
Executed actions:
- You (Robot): PICK yellow_trapezoid mid
- Human: PICK blue_trapezoid right
Human explicit feedback for your action: positive
Step 3
State:
- human_hand: blue_trapezoid
- robot_hand: yellow_trapezoid
- left: blue_square, pink_trapezoid
- red: pink_square
- mid: blue_polygon, yellow_square
- green: (empty)
- right: pink_polygon, yellow_polygon
Executed actions:
- You (Robot): PLACE yellow_trapezoid red
- Human: PLACE blue_trapezoid mid
Human explicit feedback for your action: positive
Step 4
State:
- human_hand: (empty)
- robot_hand: (empty)
- left: blue_square, pink_trapezoid
- red: pink_square, yellow_trapezoid
- mid: blue_polygon, blue_trapezoid, yellow_square
- green: (empty)
- right: pink_polygon, yellow_polygon
Executed actions:
- You (Robot): PICK pink_square red
- Human: PICK blue_polygon mid
Human explicit feedback for your action: positive
Step 5
State:
- human_hand: blue_polygon
- robot_hand: pink_square
- left: blue_square, pink_trapezoid
- red: yellow_trapezoid
- mid: blue_trapezoid, yellow_square
- green: (empty)
- right: pink_polygon, yellow_polygon
Executed actions:
- You (Robot): PLACE pink_square mid
- Human: PLACE blue_polygon green
Human explicit feedback for your action: positive
Step 6
State:
- human_hand: (empty)
- robot_hand: (empty)
- left: blue_square, pink_trapezoid

```



```

- red: yellow_trapezoid
- mid: blue_trapezoid, pink_square, yellow_square
- green: blue_polygon
- right: pink_polygon, yellow_polygon
Executed actions:
- You (Robot): PICK blue_trapezoid mid
- Human: PICK pink_square mid
Human explicit feedback for your action: positive

Step 7
State:
- human_hand: pink_square
- robot_hand: blue_trapezoid
- left: blue_square, pink_trapezoid
- red: yellow_trapezoid
- mid: yellow_square
- green: blue_polygon
- right: pink_polygon, yellow_polygon
Executed actions:
- You (Robot): PLACE blue_trapezoid red
- Human: PLACE pink_square green
Human explicit feedback for your action: positive

Step 8
State:
- human_hand: (empty)
- robot_hand: (empty)
- left: blue_square, pink_trapezoid
- red: blue_trapezoid, yellow_trapezoid
- mid: yellow_square
- green: blue_polygon, pink_square
- right: pink_polygon, yellow_polygon
Executed actions:
- You (Robot): WAIT
- Human: PICK pink_polygon right
Human explicit feedback for your action: negative

Step 9
State:
- human_hand: pink_polygon
- robot_hand: (empty)
- left: blue_square, pink_trapezoid
- red: blue_trapezoid, yellow_trapezoid
- mid: yellow_square
- green: blue_polygon, pink_square
- right: yellow_polygon
Executed actions:
- You (Robot): PICK blue_square left
- Human: PLACE pink_polygon green
Human explicit feedback for your action: positive

Step 10
State:
- human_hand: (empty)
- robot_hand: blue_square
- left: pink_trapezoid
- red: blue_trapezoid, yellow_trapezoid
- mid: yellow_square
- green: blue_polygon, pink_polygon, pink_square
- right: yellow_polygon
Executed actions:
- You (Robot): PLACE blue_square red
- Human: PICK yellow_polygon right
Human explicit feedback for your action: negative

Step 11
State:
- human_hand: yellow_polygon
- robot_hand: (empty)
- left: pink_trapezoid

```

```

- red: blue_square, blue_trapezoid, yellow_trapezoid
- mid: yellow_square
- green: blue_polygon, pink_polygon, pink_square
- right: (empty)
Executed actions:
- You (Robot): PICK pink_trapezoid left
- Human: PLACE yellow_polygon green
Human explicit feedback for your action: positive

Step 12
State:
- human_hand: (empty)
- robot_hand: pink_trapezoid
- left: (empty)
- red: blue_square, blue_trapezoid, yellow_trapezoid
- mid: yellow_square
- green: blue_polygon, pink_polygon, pink_square, yellow_polygon
- right: (empty)
Executed actions:
- You (Robot): PLACE pink_trapezoid mid
- Human: PICK yellow_square mid
Human explicit feedback for your action: negative

Step 13
State:
- human_hand: yellow_square
- robot_hand: (empty)
- left: (empty)
- red: blue_square, blue_trapezoid, yellow_trapezoid
- mid: pink_trapezoid
- green: blue_polygon, pink_polygon, pink_square, yellow_polygon
- right: (empty)
Executed actions:
- You (Robot): PICK pink_trapezoid mid
- Human: PLACE yellow_square green
Human explicit feedback for your action: positive

Step 14
State:
- human_hand: (empty)
- robot_hand: pink_trapezoid
- left: (empty)
- red: blue_square, blue_trapezoid, yellow_trapezoid
- mid: (empty)
- green: blue_polygon, pink_polygon, pink_square, yellow_polygon,
yellow_square
- right: (empty)
Executed actions:
- You (Robot): PLACE pink_trapezoid red
- Human: WAIT
Human explicit feedback for your action: positive

Task 2:
Block-sorting goal: place all blocks into either the red bin or the green bin

Step 0
State:
- human_hand: (empty)
- robot_hand: (empty)
- left: blue_polygon, blue_square, pink_square, yellow_square
- red: (empty)
- mid: pink_polygon
- green: (empty)
- right: blue_trapezoid, pink_trapezoid, yellow_polygon, yellow_trapezoid
Executed actions:
- You (Robot): PICK pink_square left
- Human: PICK blue_trapezoid right
Human explicit feedback for your action: positive

Step 1

```

```

State:
- human.hand: blue.trapezoid
- robot.hand: pink.square
- left: blue.polygon, blue.square, yellow.square
- red: (empty)
- mid: pink.polygon
- green: (empty)
- right: pink.trapezoid, yellow.polygon, yellow.trapezoid
Executed actions:
- You (Robot): WAIT
- Human: PLACE blue.trapezoid mid
Human explicit feedback for your action: negative

Step 2
State:
- human.hand: (empty)
- robot.hand: pink.square
- left: blue.polygon, blue.square, yellow.square
- red: (empty)
- mid: blue.trapezoid, pink.polygon
- green: (empty)
- right: pink.trapezoid, yellow.polygon, yellow.trapezoid
Executed actions:
- You (Robot): PLACE pink.square mid
- Human: PICK pink.trapezoid right
Human explicit feedback for your action: positive

Step 3
State:
- human.hand: pink.trapezoid
- robot.hand: (empty)
- left: blue.polygon, blue.square, yellow.square
- red: (empty)
- mid: blue.trapezoid, pink.polygon, pink.square
- green: (empty)
- right: yellow.polygon, yellow.trapezoid
Executed actions:
- You (Robot): PICK blue.polygon left
- Human: PLACE pink.trapezoid mid
Human explicit feedback for your action: positive

Current step (Task 2, Step 4):
Block-sorting goal: place all blocks into either the red bin or the green bin
- State:
- human.hand: (empty)
- robot.hand: blue.polygon
- left: blue.square, yellow.square
- red: (empty)
- mid: blue.trapezoid, pink.polygon, pink.square, pink.trapezoid
- green: (empty)
- right: yellow.polygon, yellow.trapezoid
- Executed actions:
- You (Robot): PLACE blue.polygon mid
- Human: PICK pink.polygon mid
- Human explicit feedback for your action: positive

Latest belief over preference before incorporating the CURRENT step:
- blue square to red bin: 3/10 strong accept, 3/10 weak accept, 3/10 weak reject, 1/10 strong reject
- blue trapezoid to red bin: 2/10 strong accept, 8/10 weak accept, 0/10 weak reject, 0/10 strong reject
- blue polygon to red bin: 0/10 strong accept, 0/10 weak accept, 9/10 weak reject, 1/10 strong reject
- pink square to red bin: 0/10 strong accept, 0/10 weak accept, 5/10 weak reject, 5/10 strong reject
- pink trapezoid to red bin: 3/10 strong accept, 7/10 weak accept, 0/10 weak reject, 0/10 strong reject

```

- pink polygon to red bin: 0/10 strong accept, 3/10 weak accept, 7/10 weak reject, 0/10 strong reject
- yellow square to red bin: 0/10 strong accept, 0/10 weak accept, 10/10 weak reject, 0/10 strong reject
- yellow trapezoid to red bin: 1/10 strong accept, 9/10 weak accept, 0/10 weak reject, 0/10 strong reject
- yellow polygon to red bin: 0/10 strong accept, 2/10 weak accept, 8/10 weak reject, 0/10 strong reject

- The unchosen actions:
Your unchosen actions: PLACE blue_polygon left, PLACE blue_polygon red, WAIT
Human unchosen actions: PICK pink_square mid, PICK yellow_polygon right, PICK blue_trapezoid mid, PICK yellow_trapezoid right, PICK pink_trapezoid mid, WAIT

Now, based on the collaboration history (states, actions, human feedback), and the latest belief over the human's preference before the CURRENT step, predict the human feedback for each action that you and the human did not choose, in the CURRENT step only, considering both the task goal and the human's preference.

Label each unchosen action as:

- accepted: accepted behavior with respect to both task-goal progress and inferred human preference.
- rejected: rejected behavior with respect to task-goal progress and/or inferred human preference.

Rules:

- 1) Only include unique action keys from the provided unchosen actions of you ("Robot") and the human ("Human").
- 2) Every label must be exactly 'accepted' or 'rejected'.
- 3) Return JSON only.

Return STRICT JSON only with this shape:

```
{
  "Robot": {
    "PLACE blue_polygon left": "accepted|rejected",
    "PLACE blue_polygon red": "accepted|rejected",
    "WAIT": "accepted|rejected"
  },
  "Human": {
    "PICK pink_square mid": "accepted|rejected",
    "PICK yellow_polygon right": "accepted|rejected",
    "PICK blue_trapezoid mid": "accepted|rejected",
    "PICK yellow_trapezoid right": "accepted|rejected",
    "PICK pink_trapezoid mid": "accepted|rejected",
    "WAIT": "accepted|rejected"
  }
}
```

E Details of the Real-World Human-Robot Collaboration Study

Prior to starting data collection, the study protocol was submitted to our local Institutional Review Board, and was exempt from standard, full-board review because it involved a benign behavioral intervention with adult subjects, who had to consent to taking part in the research.

E.1 Study Setup and Procedure

Participants collaborated with a robot system to assemble target pizzas at a tabletop station (Fig. 9). The station contained three task areas: storage, workspace, and on-pizza zone. Storage contained all candidate ingredients; the workspace served as the handoff area between the robot and participant; and the on-pizza zone recorded ingredients that the participant had already used. The robot's role in the collaboration was to choose ingredients from storage and place them in the workspace. The participant's role was to decide which workspace ingredients to use for the target pizza and when to use them. The workspace and on-pizza zone shared a capacity of four items, so when these areas were full the robot first had to clear an item back to storage before it could pass another ingredient.

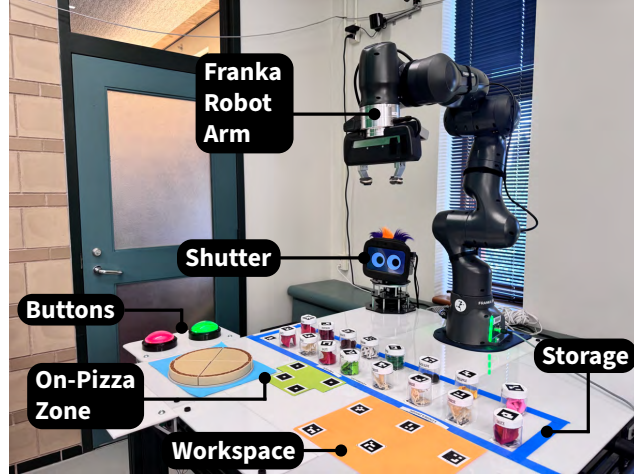


Figure 9: Layout of the tabletop pizza-making station in the physical human-robot collaboration study.

The robot system included two physical robots with distinct roles. The Franka Emika Panda arm handled manipulation of the ingredient containers, with pick-and-place motions generated through the MoveIt Task Constructor framework [52]. Shutter [53], a tabletop social robot, provided speech and gaze cues during the interaction. Behavior trees [54] coordinated the robots’ task-level and communicative actions.

The explicit feedback interface consisted of a green button for positive feedback and a red button for negative feedback. Participants were instructed to base their feedback on both parts of the task: whether the robot’s behavior helped complete the target recipe and whether it respected their selected preference over how the pizza should be made. Participants were free to provide feedback whenever they considered it useful; they were not required to press a button at every step.

A study session began after participants consented to take part in the study and to be audio- and video-recorded. Participants then completed a practice pizza-making task. The practice pizza contained two cheeses and two sauces, and the practice preference was to use cheese before sauce. The practice task served to familiarize participants with the physical task, the handoff protocol, the capacity constraint of the shared workspace, and the feedback buttons. The practice was also used to fit participant-specific rationality coefficients for inferring preferences from implicit and explicit human feedback, $\hat{\beta}_{\text{imp}}$ and $\hat{\beta}_{\text{exp}}$ in Eq. (1), which were then used in the following pizza-making tasks for that participant.

After practice, participants experienced two experimental conditions: preference learning per Eq. (1) with IMPLIED, and preference learning per Eq. (1) with the Fixed-Rule implication. The order of the conditions was counterbalanced between participants.

For each condition, the participants completed two pizza-making tasks (i.e., two trials) sequentially. The same pizza recipes were used across both conditions, but the robot’s learned preference estimate was reset between the conditions. Participants were not told which implication for preference learning was being used by the robot in a given condition; they only knew that the robot was trying to learn their preferences during the collaboration using two learning methods. After completing both conditions, participants answered comparison questions about the two learning methods.

Each lab session lasted about 45 minutes, and participants were compensated US\$15. Eligibility required participants to be adults, fluent in English, and able to perceive the robot’s speech and visual cues with normal or corrected-to-normal hearing and vision.

E.2 Pizza-Making Task Specification

Domain basics. The physical pizza-making task uses the same high-level collaboration structure as the pizza domain described by Candon et al. [25]. There are 16 ingredients, divided into four broad ingredient types:

- Sauce (4): sauce1, sauce2, sauce3, sauce4
- Cheese (4): cheese1, cheese2, cheese3, cheese4
- Meat (3): ham, pepperoni, sausage
- Veggie (5): broccoli, greenpepper, blackolive, mushroom, pineapple

The robot can access storage and the workspace, while the human can access the workspace and on-pizza zone. A target recipe specifies how many ingredients of each goal class are needed on the pizza. Cheese and sauce are counted by class, while the meat and veggie ingredients are counted by their specific types (e.g., ham, pepperoni, pineapple). In each trial corresponding to making a given pizza, the robot was given the target recipe but did not know the participant’s true preference. The participant knew both.

State space. The task state is represented by the allocation of ingredients across storage, the workspace, the on-pizza zone, and the two agents’ hands. Together with the target recipe, this allocation determines which goal-class counts are satisfied, which ingredients are available to the robot or human, whether either agent is holding an ingredient, and whether the shared workspace/on-pizza capacity is active.

Action space and transitions. The robot’s action space has 19 actions in total: 16 PICK(ingredient) actions, PUT(ingredient, storage), PUT(ingredient, workspace), and WAIT. If empty-handed, the robot could pick an ingredient from storage or the workspace; if holding an ingredient, it could put the ingredient in storage or, when space was available, in the workspace. The human’s action space has 33 actions in total: 16 PICK(ingredient) actions from the workspace, 16 PLACE(ingredient) actions that mark a held ingredient as added to the pizza, and WAIT. At each step, the robot and human act simultaneously. A pick action transfers an ingredient to the acting agent’s hand; a robot put action transfers a held ingredient to storage or the workspace; and a human place action marks a held ingredient as used on the pizza.

Preference space. Human preferences are represented by an 8-dimensional weight vector,

$$\mathbf{w} = [\text{cheese_before_sauce}, \text{cheese_before_meat}, \text{cheese_before_veggie}, \text{sauce_before_meat}, \text{sauce_before_veggie}, \text{meat_before_veggie}, \text{workspace_size_le1}, \text{workspace_size_le2}]$$

The first six dimensions encode pairwise ordering preferences over cheese, sauce, meat, and veggie. For example, a positive value of `cheese_before_sauce` means that all target cheese should be used before any target sauce, while a negative value represents the reverse ordering. The last two dimensions encode preferences over the number of ingredients in the workspace/on-pizza area. Weights are normalized to unit Euclidean norm.

True preference options and environment initializations. Before the main trials, participants selected one of six true preference options and used it consistently throughout the pizza-making task:

- *Preference A:* Sauce before cheese before veggie before meat.
- *Preference B:* Sauce before meat before veggie.
- *Preference C:* Cheese before meat before sauce and veggie.
- *Preference D:* Veggie before meat.
- *Preference E:* Prefer more than two ingredients in the workspace/on-pizza area.
- *Preference F:* Prefer at most two ingredients in the workspace/on-pizza area.

The main trials used target recipes drawn from the following set:

- *Combo Pizza:* 1 cheese, 1 sauce, 1 pepperoni, 1 ham, 1 green pepper, 1 black olive.
- *Hawaiian Pizza:* 1 pineapple, 1 green pepper, 1 ham, 1 pepperoni, 1 cheese, 2 sauce.
- *Deluxe Pizza:* 1 broccoli, 1 black olive, 1 sausage, 1 pepperoni, 2 cheese, 1 sauce.
- *Sausage Pizza:* 2 cheese, 2 sauce, 1 sausage, 1 green pepper, 1 mushroom.

- *Meat Pizza*: 1 sausage, 1 ham, 1 pepperoni, 2 cheese, 2 sauce.

Reward function. The shared reward is decomposed into goal and preference terms:

$$R(s, a_R, a_H) = R_{\text{goal}}(s, a_R, a_H) + \gamma R_{\text{pref}}(s, a_R, a_H),$$

where γ controls the relative importance of the preference reward and goal reward. The goal term is the sum of agent-specific rewards. Each agent receives a step cost of -0.3 , and completing the pizza gives a $+10$ bonus. The robot is rewarded for making target ingredients available to the human: picking a still-needed target ingredient from storage gives $+2$, and placing such an ingredient from storage into the workspace gives $+3$. The robot is penalized for actions that do not advance the recipe: picking a needed target that is already in the workspace or on the human side gives -2 (or $+0.5$ when the shared capacity is full), picking from storage when the shared capacity is full gives -5 , moving a needed target back to storage gives -1 , and cycling a target from the workspace or human side back to the workspace gives -4 . Acting on an already-satisfied target gives -3 or -5 depending on the action, except that clearing an already-used item from a full workspace gives $+1$. Acting on a non-target ingredient gives -2 or -10 . Waiting while holding an ingredient, or while a needed target remains unavailable to the human, gives -5 .

The human receives $+2$ for picking a still-needed target ingredient from the workspace and $+3$ for placing it on the pizza. Picking or placing an already-satisfied target gives -5 , acting on a non-target ingredient gives -10 , and attempting to place without holding an ingredient gives -2 . These goal rewards encourage the robot to stage useful ingredients for the human and encourage the human to add only ingredients that still contribute to the target recipe.

The preference reward is linear in interpretable feature vectors:

$$R_{\text{pref}}(s, a_R, a_H) = \mathbf{w}^\top \phi_R(s, a_R) + \mathbf{w}^\top \phi_H(s, a_H) = \mathbf{w}^\top (\phi_R(s, a_R) + \phi_H(s, a_H)).$$

The feature vectors ϕ_R and ϕ_H use the same eight dimensions as $\mathbf{w}$. Ordering features are positive when an action supports the preferred ingredient ordering, negative when it violates that ordering, and zero when the action is irrelevant. Workspace-size features evaluate whether the resulting workspace/on-pizza occupancy is consistent with the selected occupancy preference.

E.3 Demographic Information

Participants completed a demographic survey at the beginning of the study. The statistics for the 20 participants are summarized below:

- Gender: 16 participants were male, 3 were female, and 1 was non-binary/third gender.
- Age: The participants' mean age was 26 years old ($SD = 2.1$), with ages ranging from 21 to 31.
- Familiarity with robotics: Participants rated their familiarity with robotics on a 7-point Likert scale, where 1 indicated "not familiar at all" and 7 indicated "very familiar." The mean rating was 4.6 ($SD = 2.01$).
- Frequency of interacting with robots: 7 participants reported interacting with robots less than once a month, 4 reported once a month, 1 reported once a week, 2 reported 2–3 times a week, 3 reported 4–6 times a week, and 3 reported daily.