Optimal Interactive Learning on the Job via Facility Location Planning

Shivam Vats[†*] Michelle Zhao[‡*] Patrick Callaghan[‡] Mingxi Jia[†]
Maxim Likhachev[‡] Oliver Kroemer[‡] George Konidaris[†]
[†]Brown University [‡]Carnegie Mellon University [*]Equal contribution

***Abstract*—Collaborative robots must continually adapt to novel tasks and user preferences without overburdening the user. While prior interactive robot learning methods aim to reduce human effort, they are typically limited to single-task scenarios and are not well-suited for sustained, multi-task collaboration. We propose COIL (Cost-Optimal Interactive Learning)—a multi-task interaction planner that minimizes human effort across a sequence of tasks by strategically selecting among three query types (skill, preference, and help). When user preferences are known, we formulate COIL as an uncapacitated facility location (UFL) problem, which enables bounded-suboptimal planning in polynomial time using off-the-shelf approximation algorithms. We extend our formulation to handle uncertainty in user preferences by incorporating one-step belief space planning, which uses these approximation algorithms as subroutines to maintain polynomial-time performance. Simulated and physical experiments on manipulation tasks show that our framework significantly reduces the amount of work allocated to the human while maintaining successful task completion.**

I. Introduction

Collaborative robots hold the promise of reducing human workload, increasing productivity, and improving quality of life by assisting with tedious and labor-intensive tasks. For human-robot collaboration to be truly effective, robots must safely complete assigned tasks according to individual user preferences. Although robots can be equipped with prior knowledge about the user and a library of common skills, they will inevitably face novel situations in practical long-term deployments. How can they learn and adapt on the job when faced with tasks that are beyond their capabilities or when the user preference is unclear? We explore these questions in the context of multi-task domains, such as factories and households, that require collaboration on a variety of tasks during deployment. The central challenge in this setting is to enable robots to learn interactively from humans while ensuring that all tasks are completed with minimum human effort.

Prior interactive learning approaches leverage active querying strategies to proactively identify queries that result in maximum uncertainty reduction [6], volume removal [29] or information gain [4]. Such active learning strategies have been shown to reduce the number of queries. Furthermore, the use of multiple query types, such as demonstrations and preference queries, has been shown to alleviate the teaching burden by accounting for the cost associated with each query type [10].

Correspondence to shivam_vats@brown.edu.

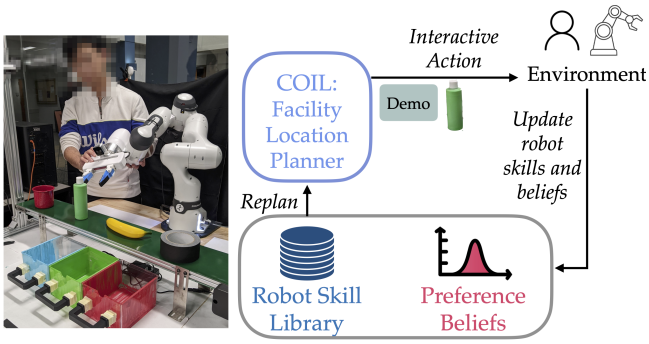

Fig. 1: A human-robot team working together to pick objects off a conveyor and pack them in bins. The robot queries the human online to learn motor skills and user preference about how each object should be grasped and where it should be placed. We propose a planning algorithm COIL that optimizes how the robot allocates tasks and uses these different types of queries to minimize human effort during the course of its deployment.

However, these approaches focus on single tasks, and therefore do not guarantee minimization of human effort across multiple tasks in extended collaboration. The latter challenges the robot to reason proactively about the expected utility of its queries so that it does not overburden the user by seeking to learn all the tasks.

We propose a novel multi-task interaction planner **COIL** (Cost-Optimal Interactive Learning) that uses three types of queries— *skill*, *preference*, and *human help*—to complete a given sequence of tasks, simultaneously satisfying hidden user preferences and minimizing user burden. Our key theoretical contribution is a **novel formulation** of multi-task interactive robot learning as an instance of the uncapacitated facility location (UFL) problem. Different from prior works, our formulation is cost-optimal and jointly minimizes human burden during learning and deployment. Our formulation has significant theoretical and practical implications. *Theoretically*, it shows that interactive robot learning in our setting can be approximated, i.e, a bounded sub-optimal solution can be computed in polynomial time. *Practically*, it enables the use strong off-the-shelf UFL solvers for planning. We believe that our UFL formulation opens exciting avenues for future research by leveraging rich theory and algorithms developed by the operations research community, without the need to develop solutions from scratch. Furthermore, we extend our UFL formulation to incorporate **multiple query types** by

proposing a simple but effective one-step belief space planning algorithm that can request user preferences when uncertain. Notably, even solving this one-step belief space planning problem is NP-hard. Our UFL formulation enables us to leverage fast approximation algorithms as sub-routines to plan efficiently. We extensively evaluate and analyze the behavior of our algorithm in simulated and physical manipulation domains. We find that COIL minimizes user burden significantly better than existing approaches, can adapt to failures in teaching, and efficiently scales to long-duration collaboration. COIL results in interactions with $12\% - 20\%$ and 23% lower cost than the best performing baseline in simulated manipulation experiments and physical conveyor experiments respectively. Additional details and videos of our experiments are available at https://sites.google.com/view/optimal-interactive-learning.

II. Related Work

Learning through multiple interaction modalities is an active area of robotics research [14, 23]. Bıyık et al. [4] first passively collect demonstrations to initialize a belief about the human's reward function and then actively probe the user with preference queries to zero-in on their true reward. Fitzgerald et al. [10] also aim to learn a human's reward function, but they do so by actively selecting the query from among Corrections, Demonstrations, Preference Queries, and Binary Rewards expected to be most informative to the learner. In a similar vein, Mehta and Losey [22] use multiple modalities of physical interaction queries to first learn a reward model online using a neural network representation and then apply constrained optimization to identify an optimal trajectory. Like our work, these works aim to learn from multiple interaction types. However, the aforementioned methods apply to singular tasks, and while they actively query the human for feedback that will be useful to the robot (and can do so while considering the cost of querying), they do not plan for the future utility of learning those tasks in the way our method does.

Human preference is another wide-spread topic of interest in robotics [5, 24, 25, 26, 32]. Jeon et al. [17] present a unifying formalism for preference-based learning algorithms through multiple types of interaction with the understanding that a "preference" is implicit in the feedback provided by the human and the skill being taught. Moreover, these preferences must be learned through multiple interaction instances. In contrast, our work encodes preferences as discrete state or action parameters by which a skill must abide and are learned through just one interaction instance. Hadfield-Menell et al. [12] introduce cooperative inverse reinforcement learning to learn a human's reward function (i.e., preference). While their POMDP-based approach could be applied in our paradigm, our use of a HiP-MDP [9] makes the problem tractable and enables us to design an efficient, effective planner. Work like Bajcsy et al. [2] isolates a user's preferences by learning features one-at-a-time until converging to the complete reward model, but they do so solely through demonstration queries and without considering future utility. Finally, reinforcement learning from human feedback (RLHF) methods have achieved state of the art results in sample complexity and reward learning for complex tasks [1, 8, 13, 28]. Even so, these successes come with a substantial cost of requiring hundreds or even thousands of preference queries and human-hours, rendering them infeasible for on-the-job learning from people.

On-the-job collaborative methods automate task allocation between machine and human either through modeled or learned approaches [3, 15, 30, 37, 36]. For instance, Liu et al. [21] propose a learning and deployment framework in which the human monitors the robot's performance to intervene and provide corrections. Data from deployment is then used in subsequent rounds to improve the robot's policy. Vats et al. [33] use a mixed integer program to decide when to learn new skills and when to delegate tasks to the user. While their method accounts for the future utility of learning a task, they do so without also considering the human's preferences. Racca et al. [27] argue that active learning methods can increase the cognitive effort of human teachers by asking difficult questions that require frequent context change. This point suggests that sample-efficient learning cannot be the sole end goal; the difficulty of answering each question should be taken into account as well. While some of the aforementioned methods do model different costs associated with querying, our method is the first to do so across a stream of incoming tasks while considering human preferences as well.

III. Problem Statement

A robot is asked to collaborate with a user to jointly complete a sequence of physical tasks $(\tau_i)_{i=1}^n = \tau_i, \ldots, \tau_n$, e.g., picking objects from a conveyor and sorting them into bins according to hidden user preferences. Each task is described by a vector $\tau_i \in \mathcal{T}$ and has an associated reward function:

$$\begin{aligned} r^i_{\text{task}} &= r^i_{\text{safe}} + r^i_{\text{pref}} \\ &= -c_{\text{skill-fail}} \cdot \mathbb{I}[\text{safety violated}] + r^i_{\text{pref}} \end{aligned} \tag{1}$$

The *safety reward* r_{safe} models user-agnostic safety constraints that the robot must not violate (e.g., avoid collisions, do not drop objects). The *preference reward* r_{pref} measures how closely robot actions match user preferences (e.g., placing a plate on the right shelf, not placing fingers inside a mug). Initially, r_{pref} is unknown to the robot but can be learned by querying the human. In addition, the robot starts with a base set of skills $\mathcal{L}_0 := \{\pi_0\}$ which may not cover all tasks. Hence, the robot should either request the user to handle such tasks or ask the user to teach it new skills such that it can then complete them autonomously. Our goal is to complete all the tasks while minimizing the burden placed on the user during the interaction. The human effort required to respond to each type of robot query is quantified using costs: c_{hum} for assigning a task to the human, c_{skill} for requesting the human to teach a motor skill, and c_{pref} for requesting their preference. Hence, the overall objective is to maximize J:

$$J = \sum_{i=1}^{n} r^i_{\text{task}} - c^i_{\text{query}} - c^i_{\text{rob}}, \tag{2}$$

where $c^i_{\text{query}} = c^i_{\text{hum}} + c^i_{\text{skill}} + c^i_{\text{pref}}$ measures the cost of querying the human, and c^i_{rob} is a fixed cost incurred if the robot undertakes the task. Note that $\max J \equiv \min -J$. We will use this fact later to convert this into a minimization problem.

IV. Approach

We model each task as a hidden-parameter MDP (HiP-MDP), where some parameters of the reward function that capture user preferences are uncertain. Each task HiP-MDP is $\mathcal{M} := (S, A, \Theta, r_{\text{task}}(s, a; z), T, P_\Theta)$, where Θ is a discrete set of possible reward parameters and P_Θ is the prior over these parameters. The robot updates its posterior b^θ over the parameters based on its interaction with the human. z is a hidden distribution over reward parameters in Θ and represents the user's preferences for the given task. The reward for each task $r_{\text{task}}(s, a; z)$ is thus a function of the hidden preference distribution. We model the choice of acting autonomously and the three types of queries that the robot can make as four distinct actions that comprise the robot's action space A:

1) *Execute Skill.* The robot executes a skill to complete the task. This is modeled by a *rob* action which has a fixed cost c_{rob} and a variable task reward r_{task} that captures how well the robot completed the task.
2) *Query type 1: Request Skill.* Ask the user to teach a new motor skill for a task via demonstrations. The user's effort to perform this query is captured by the cost c_{skill}. If the robot fails to learn from the human, then the query is deemed a *teaching failure*.
3) *Query type 2: Request Human.* Ask the user to complete the task. The user effort required to fulfill this request is captured by the cost c_{hum}.
4) *Query type 3: Request Preference.* Ask the user for their preference about how a task should be done. The user effort is captured by the cost c_{pref}. Some prior work define "preference query" as a comparative choice between two possible options. In our case, the user chooses a preferred option from all available options. We will refer to it as a "preference request" to avoid confusion.

Robot Skills. The robot learns a library $\mathcal{L}$ of task-specific skills $\pi(s, \theta)$ that are parameterized by task and user preference parameters. Define the expected return for executing skill π with a specific preference parameter θ to complete task τ_i as

$$R^i(\pi, \theta) \leftarrow \sum_{a \sim \pi(s,\theta)} r^i_{\text{task}}(s, a|\theta).$$

To complete τ_i, the robot must not only select which skill to use, but also which preference parameter to execute said skill with. Let the robot have skills $\mathcal{L} := \{\pi_1, \ldots, \pi_k\}$. Then, the robot must choose a combination of skill and preference parameter π^*, θ^* with the highest return:

$$\pi^*, \theta^* = \arg\max_{\pi \in \mathcal{L}, \theta \in \Theta} R^i(\pi, \theta). \tag{3}$$

Skill Return Model. The true reward function is unknown to the robot at planning time. Hence, the robot must use a reward model to predict the expected return from skill execution before learning it. Let the robot's belief about the user's preference for the current task be b^Θ. We provide the robot with a domain-specific function ρ^{safe}_π to predict the probability of safe execution. $\rho^{\text{safe}}_\pi(\tau', \theta')$ predicts the probability of safe execution of a skill π when deployed on a task τ' with parameter θ'. This function is defined based on similarity between (τ', θ') and the conditions (τ, θ) that π will be learned for: $\rho^{\text{safe}}_\pi(\tau', \theta') = f(\|\tau - \tau'\| + \|\theta - \theta'\|)$. See Appendix E for our instantiation in experiments. Intuitively, the probability of skill success is high if both tasks and preferences are similar and low otherwise. The preference belief b^Θ models the probability of preference satisfaction. The skill return model is then:

$$\begin{aligned}\hat{R}^\tau(\pi, \theta') = -[&c_{\text{skill-fail}} \cdot (1 - \lambda_{\text{teach}} \cdot \rho^{\text{safe}}_\pi(\tau', \theta')) + \\ &c_{\text{pref-fail}} \cdot (1 - b^\Theta(\theta'))].\end{aligned} \tag{4}$$

The first term predicts the expected penalty for safety violation and the second terms predicts the penalty for preference violation. λ_{teach} estimates the probability that the robot will successfully learn a skill from the human. We model the success of teaching as a Bernoulli process and use Bayesian inference to compute the posterior distribution with a Beta distribution $\text{Beta}(\alpha, \beta)$ as the prior. We use the mean of the distribution in the return model as the estimated probability of successful teaching:

$$\lambda_{\text{teach}} = \mathbb{E}[\text{Beta}(\alpha, \beta)] = \frac{\alpha}{\alpha + \beta}.$$

This enables the robot to identify difficult-to-learn tasks and *adapt* its plan online if teaching fails so that it does not waste human time. Our experiments will show that non-adaptive methods expend significant human effort trying to learn such tasks even after repeated teaching failures whereas COIL assigns them to the human and focuses on learning feasible tasks.

A. Skill Learning Under Known Preferences: Facility Location Formulation

We first consider the case in which the robot cannot query the human about their preference online and must plan with its belief about the preferences. Previous work by Vats et al. [33] proposed a mixed integer programming formulation ADL to plan in a similar setting. Unfortunately, solving MIP optimally is NP-hard which makes it difficult to scale ADL to larger problems. We propose a novel uncapacitated facility location (UFL) formulation that has tight polynomial-time approximation algorithms which can efficiently compute high-quality solutions even for large problems. Intuitively, the challenge of identifying the minimal cost set of interactive actions that cover the full task sequence maps nicely onto the UFL problem which seeks to service demands (tasks) by allocating facility resources (interactive actions) while minimizing costs.

Facility Location Problem. We first briefly introduce the uncapacitated facility location (UFL) problem. Please refer to Williamson and Shmoys [35, chapter 4] for a more detailed

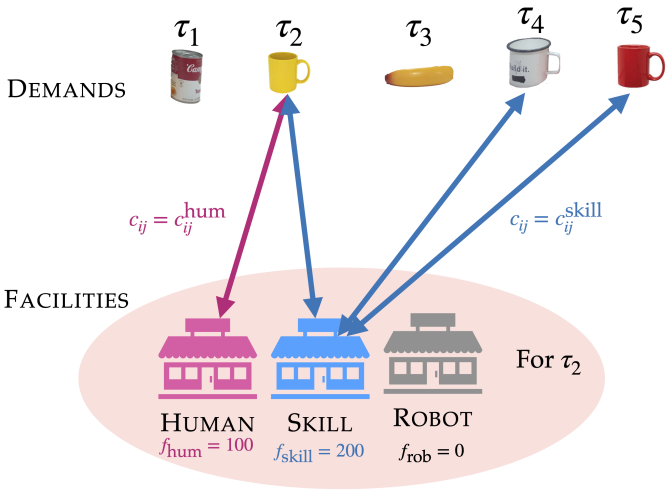

Fig. 2: Facility location formulation. Tasks $\tau_1, \ldots, \tau_5$ are demands to be satisfied. Facilities correspond to interactive actions available for every task. We highlight facilities for τ_2: *Human* facility can only service τ_2. *Skill* facility can service similar future tasks τ_2, τ_4, τ_5. *Robot* facility cannot service any task as the robot hasn't learned a skill yet. Furthermore, none of the facilities can service past tasks.

treatment. The facility location problem has a set of demands $D = \{1, \ldots, m\}$ and a set of facilities $F = \{1, \ldots, n\}$. There is *facility cost* f_i associated with opening each facility $i \in F$ and an *assignment* or *service cost* c_{ij} of serving demand j by facility i. The goal is to serve all the demands by opening a subset of facilities $F' \subseteq F$ such that the overall cost of opening the facilities in F' and the cost of assigning each demand $j \in D$ to the nearest facility $i \in F'$ is minimized:

$$\min \underbrace{\sum_{i \in F'} f_i}_{\text{facility cost}} + \underbrace{\sum_{j \in D} \min_{i \in F'} c_{ij}}_{\text{service cost}}. \tag{5}$$

We formulate the interaction planning problem as a facility location problem by defining demands, facilities, facility costs and service costs $\langle D, F, f_i, c_{ij} \rangle$.

Tasks as Demands. Define the set of demands as the set of tasks $F = \{\tau_1, \ldots, \tau_N\}$. A solution to UFL has to satisfy all the demands. Hence, this guarantees that it will complete all the tasks.

Robot Actions as Facilities. In the following, we define facilities corresponding to all the actions for every task τ_i.

1) *Human Facilities.* Define one *human* facility i_{hum} with a facility cost of c_{hum} that captures the cost of asking the user to complete the task. The service cost to serve τ_i is 0 as human effort is modeled by c_{hum} and ∞ for all the other tasks as this action cannot not complete them.
2) *Skill Facilities.* Define one *skill* facility i_{skill} with a facility cost of c_{skill} that corresponds to requesting the user to teach the robot a new skill π_i for τ_i. The service cost for completing a task τ_j consists of a fixed robot execution cost c_{rob} and the task reward

$$c_{ij}^{\text{skill}} = c_{\text{rob}} - \max_{\theta \in \Theta} \hat{R}^j(\pi_i, \theta).$$

The robot executes π_i with preference parameters θ^* that maximize the expected reward under its preference belief distribution.

3) *Robot Facilities.* Define one *robot* facility i_{rob} with a facility cost of 0 that corresponds to the robot using a previously learned skill π_i. The service cost for completing a task τ_j consists of a fixed robot execution cost c_{rob} and the task reward as above.

Algorithm 1 Facility location formulation for COIL.

procedure COIL-KNOWNPREFS($(b_i)_{i=1}^N$)
  $D \leftarrow \{\tau_1, \ldots, \tau_N\}, F \leftarrow \emptyset$ ▷ demands and facilities
  for $i \in 1, \cdots, N$ **do**
    F.insert(i_{hum}), $f_i^{\text{hum}} \leftarrow c_{\text{hum}}$ ▷ human facility
    $c_{ij}^{\text{hum}} \leftarrow 0$ if $i = j$ else ∞
    F.insert(i_{skill}), $f_i^{\text{skill}} \leftarrow c_{\text{skill}}$ ▷ skill facility
    for $j \in i, \ldots, N$ **do**
      $c_{ij}^{\text{skill}} = c_{\text{rob}} - \max_{\theta \in \Theta} \hat{R}^j(\pi_i, \theta)$
  for $i \in \mathcal{L}$ **do** ▷ skills already learned
    F.insert(i_{robot}), $f_i^{\text{robot}} \leftarrow 0$ ▷ robot facility
    for $j \in 1, \ldots, N$ **do**
      $c_{ij}^{\text{robot}} = c_{\text{rob}} - \max_{\theta \in \Theta} \hat{R}^j(\pi_i, \theta)$
  $(a_i)_{i=1}^N, J \leftarrow$ SOLVEUFL(D, F, f, c)
  return $(a_i)_{i=1}^N, J$
procedure COIL($(b_i)_{i=1}^N$)
  $(a_i)_{i=1}^N, J \leftarrow$ COIL-KNOWNPREFS($(b_i)_{i=1}^N$)
  $\bar{J} \leftarrow 0$
  for $\theta \in \Theta$ **do** ▷ possible query response
    $(b'_i)_{i=1}^N \leftarrow$ UPDATEPREFBELIEFS(θ)
    $J' \leftarrow$ COIL-KNOWNPREFS($(b'_i)_{i=1}^N$)
    $\bar{J} = \bar{J} + b_1(\theta) \cdot J'$
  if $c_{\text{pref}} + \frac{\bar{J}}{\sum b_1} \leq J$ **then**
    $a_1 \leftarrow a_{\text{pref}}$
  return $(a_i)_{i=1}^N$
procedure INTERACTION($(\tau_i)_{i=1}^N$)
  $(b_i)_{i=1}^N \leftarrow$ INITPREFBELIEF()
  $k \leftarrow 1$ ▷ current task index
  while all tasks are not done **do**
    plan $\leftarrow$ COIL$((b_i)_{i=k}^N)$ ▷ replan
    execute the first action and update k
    update λ_{teach}
return

Approximation Algorithms. We implement the approximation algorithm proposed by Jain and Vazirani [16]. In the worst case, this algorithm is 3-suboptimal when the facility location problem is metric and $\log(n)$-suboptimal otherwise. This algorithm has a run-time of $O(n^2 \log(n))$, where n is the number of demands. In practice, we found it to be near-optimal for our problems. Let X be the set of facilities opened so far, and let U be the set of demands that are not served by open facilities. Then, the algorithm iteratively picks a demand $i \in F$ and $Y \subseteq U$ that minimizes the ratio $\frac{f_i + \sum_{j \in Y} c_{ij}}{|Y|}$ and sets f_i to 0. Intuitively, it opens a facility that has the minimum cost

per demand for some subset of demands and assigns those demands to the opened facility.

B. Planning for Preference Requests

Equipped with a provably bounded-suboptimal plan for learning under known preferences as a reference point, COIL then determines when preference requests are needed to clarify user preferences before execution. For each task, COIL evaluates the expected change, under its current beliefs b^θ, in overall plan cost should it request the user's preference for the current task. If the expected plan cost plus preference request cost is lower than the current plan (line 22), COIL elects to first reduce uncertainty in the preference parameters for the current task. In this way, COIL augments its bounded-suboptimal reference plan with uncertainty-guided preference requests when necessary, towards improving the plan to handle uncertainty in preference parameters. During the interaction, we replan after every interactive action execution to take into account the latest information. This is enabled by the fast runtime of COIL which is polynomial in the number of tasks and preference parameters.

Theorem 1. *The worst-case runtime of COIL is polynomial in n and k, where n is the number of tasks and $k = |\Theta|$ is the number of possible preference parameters.*

Proof: COIL solves the facility location problem $k = |\Theta| + 1$ times (see line 18) in total by calling the function COIL-KNOWNPREFS. The complexity of COIL-KNOWNPREFS is polynomial when SOLVEUFL is an approximation algorithm. In particular, we use an approximation algorithm with runtime $O(n^2 \log(n))$, where n is the number of tasks. Hence, the overall complexity of COIL is $O(k(n^2 \log(n))$, which is polynomial in n and k. ■

V. EXPERIMENTAL SETUP

To evaluate the efficacy of our proposed approach, we run a series of quantitative experiments to study how **COIL** affects the costs of learning on the job during task execution compared with baselines. We further investigate quantitatively the nature of queries throughout the interaction to understand how **COIL** and other approaches induces different interactive behaviors.

Domains. We study the combined learning and execution costs incurred by our approach through experiments in three controlled environments: an object pickup and dropoff **Gridworld** environment implemented using MiniGrid [7], a simulated **7DoF Manipulation** environment implemented using robosuite [38], and a real-world **Conveyor** manipulation setting. Brief descriptions of each domain are below; please see the appendix B for details.

1) **Gridworld** is a discrete, 17x17 grid comprised of a sequence of 15 total objects of nine varieties distinguished by object *type*, *color*, and *position*. Each object defines a task the agent must execute, and each task is to navigate to an object, pick it up, and transport it to the user's preferred location for that object. There are three possible goal locations.
2) **7DoF Manipulation** is a simulated bin-packing task where a Franka robot manipulator must pick up and put 30 total objects of seven different varieties (e.g., milk carton, mug) into one of four different bins. Each object defines a task the agent must execute.
3) **Conveyor** is a real-world instantiation of a factory setting in which a Franka tabletop manipulator and a human employee work side by side to sort objects arriving on a conveyor belt into three containers. Some of the objects, such as the mug, have two possible grasps: handle and rim. The robot must pick each object using the preferred grasp and place it in the preferred bin. We randomly sample 5 task sequences consisting of 20 tasks each. We pretrain GEM [18] using human demonstrations for every task and use it to provide demonstrations in our experiments.

Baselines. We compare **COIL** with state-of-the-art approaches for interactive robot learning and task allocation. Implementation details for each baseline can be found in the Appendix D.

1) *Confidence-based ADL (**C-ADL**).* ADL [33] addresses the similar problem of planning for skill learning and task allocation and thus makes for a strong baseline. However, ADL does not aim to learn human preferences and thus lacks the ability to reason jointly over preference and skill learning. To ensure a fair comparison, the **C-ADL** baseline makes preference queries when the robot's confidence over the human's preference is below some predefined threshold.
2) *Information Gain (**IG**).* Because **COIL** plans over multiple query types (skill, preference) and multiple human contribution types (human, robot), it makes sense to design a baseline inspired by interactive learning methods that incorporate multiple types of human feedback (e.g., demonstrations, preferences, etc.). In these methods, a widely-used approach is to select the query which provides the greatest expected information gain [10]. While these methods typically learn reward functions for a single task and consider neither human contributions nor the future utility of learning particular skills, we design an information gain objective which does both as the **IG** baseline in our interaction paradigm.
3) *Confidence-based Autonomy (**CBA**).* Inspired by Chernova and Veloso [6], **CBA** requests skill and preference teaching if the robot is uncertain about user preferences or skills and otherwise assigns the task either to itself or the human.

Human Cost Profiles. The costs a human might associate with teaching, executing tasks, or responding to preference requests depend on domain-specific variables such as the cost of labor and the difficulty of teaching. We study the performance of our approach under multiple simulated users with three different cost profiles. In our evaluations, we assign a cost of $c_{\text{rob}} = 10$ to each robot execution, $c_{\text{hum}} = 80$ to human task execution, and $c_{\text{pref}} = 20$ to each preference request. The robot incurs a

Cost	Algo	#teach	#human	#pref	#robot	Cost
Low	COIL	4.43 (0.88)	2.43 (1.26)	4.43 (0.88)	12.57 (1.26)	**630.67** (50.26)
	C-ADL	6.87 (0.72)	0.0 (0.0)	6.87 (0.72)	15.0 (0.0)	630.67 (50.26)
	IG	6.87 (0.72)	0.0 (0.0)	6.87 (0.72)	15.0 (0.0)	630.67 (50.26)
	CBA	6.87 (0.72)	0.0 (0.0)	6.87 (0.72)	15.0 (0.0)	630.67 (50.26)
Med	COIL	4.13 (0.85)	2.77 (1.26)	4.13 (0.85)	12.23 (1.26)	**839.67** (53.7)
	C-ADL	4.17 (0.86)	2.7 (1.29)	6.87 (0.72)	12.3 (1.29)	893.0 (60.84)
	IG	6.87 (0.72)	0.0 (0.0)	6.87 (0.72)	15.0 (0.0)	974.0 (86.16)
	CBA	6.87 (0.72)	0.0 (0.0)	6.87 (0.72)	15.0 (0.0)	974.0 (86.16)
High	COIL	0.37 (0.66)	13.2 (3.12)	0.53 (0.67)	1.8 (3.12)	**1158.0** (85.42)
	C-ADL	2.33 (0.7)	6.37 (1.72)	6.87 (0.72)	8.63 (1.72)	1199.67 (78.42)
	IG	4.17 (0.86)	2.7 (1.29)	6.87 (0.72))	12.3 (1.29)	1309.67 (108.27)
	CBA	6.87 (0.72)	0.0 (0.0)	6.87 (0.72)	15.0 (0.0)	1660.67 (157.96)

TABLE I: In the Gridworld domain, we find that COIL makes fewer preference queries than the confidence-based baselines because COIL only asks for human preferences if it believes that this information will be useful later. Format is mean(standard deviation).

penalty cost of $c_{\text{skill-fail}} = 100$ if it fails to successfully execute any skill, and a penalty of $c_{\text{pref-fail}} = 100$ if it successfully executes a skill by placing the object in a goal undesired by the user. To compare the profiles with minimal cost tuning, we examine profiles across the cost of human skill teaching.

1) *Low-Cost Teaching* ($c_{\text{skill}} = 50$): This profile simulates an experienced teacher for whom providing demonstrations of robot skills is not burdensome.
2) *Med-Cost Teaching* ($c_{\text{skill}} = 100$): This profile simulates a teacher for whom teaching is moderately more burdensome than performing the task themselves.
3) *High-Cost Teaching* ($c_{\text{skill}} = 200$): This profile simulates a novice teacher for whom providing demonstrations of robot skills is highly burdensome.

Preference Belief Estimation. The robot maintains a belief estimate b^{Θ} of the user's hidden personal preferences for every task. The probability that each preference parameter θ satisfies user preference, i.e., $b^{\Theta}(\theta)$, is modeled as a Bernoulli distribution. These belief estimates are updated based on feedback from the user using a Bayesian filter (described in Appendix section A).

VI. Experimental Results

We summarize our results through four major takeaways derived from five distinct experimental focuses: (A) cost comparisons under different human cost profiles when all task skills are learnable by the robot, (B) scalability and computational costs of **COIL** under long task sequences, (C) online-adaptiveness of **COIL** when challenging-to-learn skills necessitate online replanning, and (D) a real-world instantiation of **COIL** that enables the human-robot team to teach and learn as they manipulate objects which arrive on a conveyor.

A. *COIL outperforms myopic interactive learning.*

COIL significantly outperforms myopic interactive learning methods that do not consider plan over the future utility of all possible plans (Figure 3). In the Gridworld domain instantiated such that all objects are learnable by the robot, we evaluate the incurred plan costs over 30 task sequences and randomized human object arrangement preferences. Under the low-cost

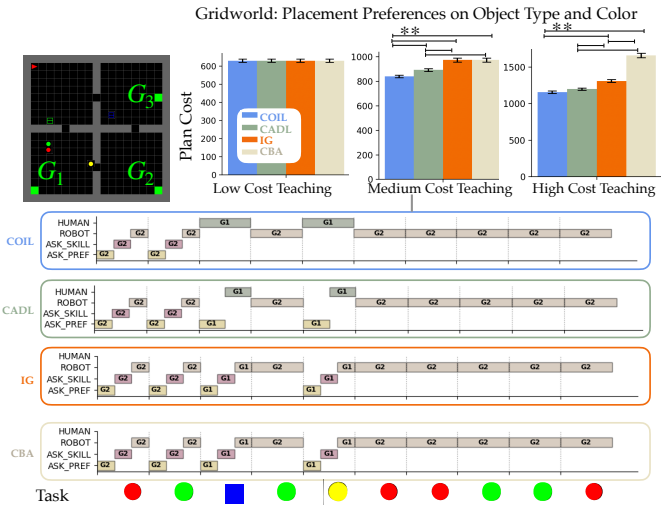

Fig. 3: Under Med- and High-Cost teaching cost profiles, **COIL** consistently chooses the lowest cost plan compared with baselines. We highlight the qualitative behavior of **COIL** compared to baselines: **COIL** under the medium cost teaching profile assigns singleton tasks to the human when the cost of learning is high. Error bars indicate standard error over 30 randomized task sequences and true human preferences in the Gridworld domain. ** represents $p < 0.01$ significance.

Cost	Algo	#teach	#human	#pref	#robot	Cost
Low	COIL	6.6 (1.50)	4.0 (3.58)	5.6 (1.02)	19.4 (4.29)	**1519.0** (351.86)
	COIL-NoAd	8.3 (2.9)	0.8 (0.98)	5.6 (1.02)	20.9 (2.84)	1630.0 (402.34)
	C-ADL	8.8 (3.76)	0.0 (0.0)	6.4 (0.8)	21.2 (3.76)	1705.0 (509.91)
	IG	30.0 (0.0)	0.0 (0.0)	6.4 (0.8)	0.0 (0.0)	2403.0 (279.11)
Med	COIL	6.0 (1.095)	5.1 (3.91)	5.6 (1.0198)	18.9 (4.4821)	**1722.0** (406.98)
	COIL-NoAd	7.6 (2.61)	1.5 (0.92)	5.6 (1.02)	20.9 (2.84)	1940.0 (474.15)
	C-ADL	7.4 (2.97)	1.4 (1.2)	6.4 (0.8)	21.2 (3.76)	1990.0 (638.12)
	IG	30.0 (0.0)	0.0 (0.0)	6.4 (0.8)	0.0 (0.0)	3839.0 (274.02)
High	COIL	4.1 (0.3)	8.0 (4.54)	4.0 (0.0)	17.9 (4.72)	**2099.0** (391.80)
	COIL-NoAd	5.9 (2.34)	5.3 (2.53)	4.0 (0.0)	18.8 (3.79)	2495.0 (785.94)
	C-ADL	6.3 (2.93)	3.0 (1.84)	6.4 (0.8)	20.7 (3.80)	2644.0 (864.56)
	IG	0.0 (0.0)	30.0 (0.0)	6.4 (0.8)	0.0 (0.0)	2528.0 (16.0)

TABLE II: Results on the manipulation domain. On average, **COIL** plans interactions that result in 7% to 18% reduction in cost compared to the best performing baseline. The improvement over baselines is particularly marked when the cost of teaching is more expensive than assigning the task to the human, i.e, medium and high cost profiles. The reported statistics are averaged over 10 interactions with 30 randomly sampled tasks each.

profile, **COIL** and all baselines achieve similar plan costs (no significance). As the cost of human demonstrations increases (med-cost and high-cost), **COIL** consistently achieves the lowest-cost plans of all the methods. We compared statistical differences using a one-way ANOVA [31] between costs achieved for each algorithm, and evaluated pairwise differences using pairwise t-tests [19] with Bonferroni correction [34] if significant main effects were present. For the Med-Cost human cost profile in the Gridworld domain, we found significant effects of planner ($F = 23.48, p < 0.01$), and found significant ($p < 0.01$) pairwise differences between **COIL** and all baseline algorithms. For the High-Cost human cost profile in the Gridworld domain, we found significant effects of planner ($F = 120.58, p < 0.01$), and found significant ($p < 0.01$) pairwise differences between **COIL** versus **IG**

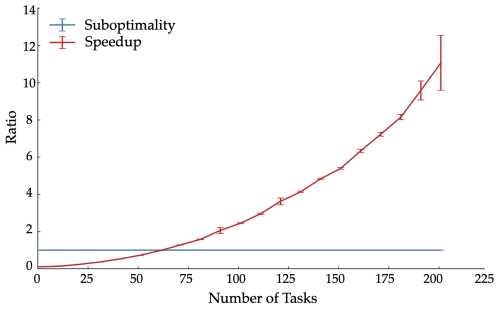

Fig. 4: COIL computes near-optimal solutions significantly faster than optimal mixed integer programming.

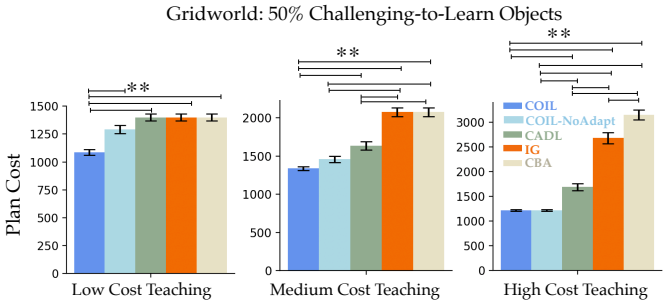

Fig. 5: In the Gridworld environment, when half of the objects are challenging-to-learn, **COIL** achieves the lowest plans, compared to **COIL-NoAdapt** and other baselines across teaching profiles. In high cost teaching, **COIL** and **COIL-NoAdapt** often assign tasks to the human off the bat, reducing the impact of adaptivity. Error bars indicate standard error over 30 randomized task sequences and true human preferences. ** represents $p < 0.01$ significance.

and **CBA**. Though we did not find a pairwise significant difference, we observe that **COIL** achieves a slightly lower cost on average compared with **C-ADL** (Figure 3). We provide all statistical testing values in Appendix E-D.

To understand the differences between the behaviors induced by each algorithm, the bottom four rows of Figure 3 highlight a task sequence of 10 objects when each method collaborates with the med-cost human profile. **COIL** identifies the lowest-cost plan by requesting to be taught the skills and preferences associated with repeated instances of the same object and delegating singleton tasks to the human partner. **C-ADL** identifies a plan similar to **COIL**, but it asks for preference queries for singleton tasks that are eventually assigned to the human, incurring additional preference cost. Under Med-Cost teaching, **IG** and **CBA** opt to learn every preference and skill (Table I). On the other hand, under Low-Cost teaching, all approaches choose to learn all preferences and skills, given that the total cost of doing so is not expensive. These results highlight a particular strength of **COIL** to balance learning, requests for human contribution, and execution especially when the right course of action under nuanced costs is not easily known.

B. COIL efficiently scales to long task sequences without compromising plan quality.

We compare the approximation algorithm used in COIL with an optimal mixed-integer programming (MIP) approach.The MIP formulation is implemented using Gurobi [11], a state-of-the-art MIP solver, while our algorithm is implemented in Python, leaving room for further improving run-time. COIL efficiently computes near-optimal solutions significantly faster than MIP. The speedup is defined as the ratio of MIP runtime to COIL runtime, while sub-optimality is measured as the ratio of the cost of plans generated by COIL to those produced by MIP. In Figure 4, we observe that the computational advantage of COIL over MIP gets more pronounced for longer task sequences without compromising on solution quality.

C. Adapting to teaching failures online reduces human burden.

Thus far, our assumption that the robot can reliably learn skills for all tasks enabled us to evaluate and compare the plan costs incurred by **COIL** and baselines. While accurately modeling the feasibility of skill learning is not the focus of this work, we investigate how **COIL** can adapt to failures in learning with online replanning. Importantly, these "challenging" skills are comparatively difficult to learn and are revealed as such only after the robot tries to acquire the skill via human demonstrations. We demonstrate these results in both the Gridworld and Simulated 7DoF Manipulation domains.

1) COIL Adaptivity in Gridworld: In the simulated Gridworld domain, we control for the presence of challenging skills by varying the number of skills that are challenging-to-learn (i.e., can never be learned), and examine the relationship between number of challenging skills and the benefit of estimating λ_{teach} online in the **COIL** framework. As the proportion of challenging skills increases (from 10% to 50% (Fig 5) to 90%), the added benefit of replanning based on the observed feasibility of teaching enables **COIL** to identify lower-cost plans than **COIL-NoAdapt** which optimistically assumes that all skills are learnable. Baselines that do not adapt to teaching failures also incur higher costs than **COIL**. (Fig 5). Refer to Appendix F for reported statistical analyses and detailed analysis of results on 10% and 90% challenging-to-objects.

2) COIL Adaptivity in Simulated 7DoF Manipulation: We next study the adaptivity of **COIL** in the simulation manipulation domain. We consider task sequences of 30 objects, sampled from 7 unique varieties. The mug object is too wide for the robot gripper which results in a teaching failure when the human tries to teach the robot. **COIL** takes this failure into account by updating λ_{teach} for all mugs and adapts its plan to assign mugs to the human. This results in statistically significantly lower costs than baselines (figure 6, table II). In this experiment, we compare with our 3 strongest baselines, derived from our earlier results, removing **CBA** from our analysis. Statistical analyses are detailed in Appendix G.

D. COIL succeeds in real-world operations.

We evaluate **COIL** on a physical conveyor domain designed to emulate collaboration in factories. We ran the experiments with a medium cost profile with $c_{\text{hum}} = 50, c_{\text{skill}} = 100$, 5 different task sequences, each with 20 objects randomly

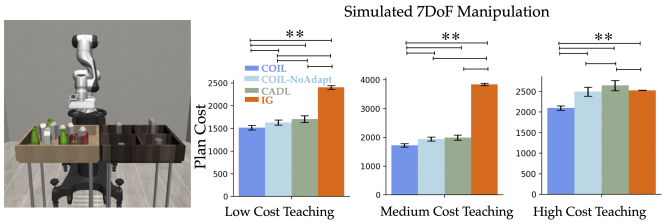

Fig. 6: In the Simulated 7DoF Manipulation domain, **COIL** achieves significantly lower plan costs than baseline methods. Error bars represent standard error over 10 randomized initialization of the 30-object task sequence.

Algorithm	#teach	#human	#pref	#robot	Cost
COIL	1 (1)	12.25 (4.65)	1.5 (1)	6.25 (3.77)	**870** (94.52)
CADL	2 (1.15)	9.75 (3.86)	14 (1.63)	8.25 (2.75)	1075 (59.72)
COIL (teach fail)	1	19	1	0	**1280**
CADL (teach fail)	6	10	16	4	2320

TABLE III: Results on a physical conveyor. We ran experiments with 5 different task sequences, each with 20 objects, with COIL and CADL. We observed teaching failure on the white mug (possibly because its shiny surface made camera-based pose estimation difficult). Hence, we report the run with teaching failure separately from the other 4 runs. COIL was able to achieve significantly lower cost than the baseline in both situations. CADL especially struggled in the case of teaching failure as it repeatedly requested to be taught the mug skill.

sampled from a set of 12 objects. Each sequence consisted of one high-frequency object which was five times more likely to be sampled than the other objects. The objects appeared one-by-one in front of the robot, at which point it needed to decide whether to autonomously pick up the object using an appropriate grasp and place it in the correct bin, or request help from the human. Some objects—such as mugs and bottles—had two feasible grasps, of which only one was preferred by the human. Similarly, the human had hidden preferences about which bin each object belonged to. The robot queried the human to understand their preferred grasp and target bin for each object.

We report the results from our experiment in table III. Compared to C-ADL, **COIL** made fewer skill and preference requests. C-ADL has a tendency to overburden the user with preference requests while **COIL** learns user preference for a task only when it intends to complete the task autonomously. A priori, it is desirable to learn how to manipulate the high-frequency object in each experiment. However, we observed that the robot was unable to learn a skill for the white mug, possibly because its shiny surface made our camera-based pose estimation difficult. Because of its adaptive nature, **COIL** changed its plan and assigned all mugs to the human. By contrast, C-ADL repeatedly requested the human to teach it how to manipulate the mug and expended significant human effort in the process.

Limitations. To plan for a set of tasks, COIL must know the composition of that set beforehand. This prior knowledge will be difficult to come by reliably in all real-world settings. Additionally, the human's task preferences are assumed to

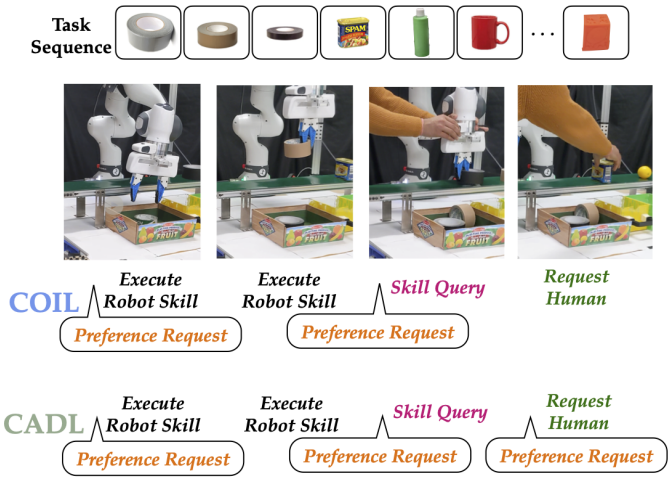

Fig. 7: In the real-world conveyor task sequence, **COIL** minimizes costs relative to C-ADL, requesting user preferences when uncertain, executing known skills, and requesting demonstrations to learn new skills. We highlight three tasks along the plan to learn generated by **COIL**. C-ADL requests unneeded preferences before assigning tasks to the human.

be discrete and stationary, but prior work reveals that a human's preferences take a variety of form and are subject to change over time. Furthermore, we use prior knowledge about similarities between tasks to predict the generalization of skills to future tasks. In many complex domains, such a prior may be inaccurate. For example, while the robot may be confident that its learned skill can be executed again on the next instance of the same skill, perturbations in the environment (e.g., slight differences in object orientation), unobserved environment variables (e.g., lighting), and other factors may cause the robot to fail when rolling out its learned skill.

VII. Conclusion

Teams are best positioned to succeed when each member accounts for their teammate's abilities and preferences. Robot teammates will need to do the same if human-robot teams are to succeed, and COIL is our proposed means of enabling this capability. By planning over multiple interaction types, contributions from both human and robot, and the human's associated task preferences and contribution costs, COIL identifies plans which minimize the burden placed upon the human on its way to achieving task success. COIL does so by formulating the interaction as a facility location problem which identifies the optimal sequence of robot actions and human contributions and then deducing if that optimal plan could be improved with more certain beliefs about the human teammate's preferences. COIL identifies plans which cost less as compared to baselines and efficiently scales to long task sequences. Moreover, COIL can learn human preferences for both task completion and teammate task contributions, and it can re-plan when skills prove too difficult to learn. Finally, COIL demonstrates its effectiveness in a real-world conveyor

domain, inspired by its potential application in collaborative factories of the future. In our future work, we are interested in extending our planner to multi-step tasks. One possibility would be to decompose a multi-step tasks into a sequence of sub-tasks which can then be handled by our planner. Another direction of interest is to plan for unordered sets of tasks which introduces an additional challenge of scheduling.

Acknowledgments

We thank Prof. Reid Simmons and Prof. Henny Admoni for their valuable feedback, Prof. Anupam Gupta for insightful discussions and Lakshita Dodeja for her help with our real-world robot experiments. This work was supported by the Office of Naval Research (ONR) under REPRISM MURI N000142412603 and ONR grant #N00014-22-1-2592, as well as by the National Science Foundation (NSF) via grant #1955361. Partial funding was also provided by the Robotics and AI Institute.

References

[1] Nichola Abdo, Cyrill Stachniss, Luciano Spinello, and Wolfram Burgard. Robot, organize my shelves! Tidying up objects by predicting user preferences. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1557–1564. doi: 10.1109/ICRA.2015.7139396. URL https://ieeexplore.ieee.org/abstract/document/7139396.

[2] Andrea Bajcsy, Dylan P. Losey, Marcia K. O'Malley, and Anca D. Dragan. Learning from Physical Human Corrections, One Feature at a Time. In *2018 13th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 141–149. URL https://ieeexplore.ieee.org/document/9473611/?arnumber=9473611.

[3] Connor Basich, Justin Svegliato, Kyle Hollins Wray, Stefan Witwicki, Joy-Deep Biswas, and Shlomo Zilberstein. Learning to optimize autonomy in competence-aware systems. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, volume 2020-May, pages 123–131. ISBN 978-1-4503-7518-4. URL www.ifaamas.org.

[4] Erdem Bıyık, Dylan P Losey, Malayandi Palan, Nicholas C Landolfi, Gleb Shevchuk, and Dorsa 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.

[5] Erdem Biyik, Nima Anari, and Dorsa Sadigh. Batch active learning of reward functions from human preferences. *ACM Transactions on Human-Robot Interaction*, 13(2):1–27, 2024.

[6] Sonia Chernova and Manuela Veloso. Confidence-based policy learning from demonstration using Gaussian mixture models. 5:1315–1322. doi: 10.1145/1329125.1329407.

[7] Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo de Lazcano, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *CoRR*, abs/2306.13831, 2023.

[8] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[9] Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *IJCAI: proceedings of the conference*, volume 2016, page 1432, 2016.

[10] Tesca Fitzgerald, Pallavi Koppol, Patrick Callaghan, Russell Q Wong, Reid Simmons, Oliver Kroemer, and Henny Admoni. INQUIRE: INteractive Querying for User-aware Informative REasoning.

[11] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024. URL https://www.gurobi.com.

[12] Dylan Hadfield-Menell, Anca Dragan, Pieter Abbeel, and Stuart Russell. Cooperative Inverse Reinforcement Learning. URL http://arxiv.org/abs/1606.03137.

[13] Joey Hejna, Rafael Rafailov, Harshit Sikchi, Chelsea Finn, Scott Niekum, W Bradley Knox, and Dorsa Sadigh. Contrastive prefence learning: Learning from human feedback without rl. *arXiv preprint arXiv:2310.13639*, 2023.

[14] Minyoung Hwang, Luca Weihs, Chanwoo Park, Kimin Lee, Aniruddha Kembhavi, and Kiana Ehsani. Prompt-able behaviors: Personalizing multi-objective rewards from human preferences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16216–16226, June 2024.

[15] Toshiyuki Inagaki. Adaptive Automation: Sharing and Trading of Control. 2001.10:79–84. doi: 10.1299/jsmetld.2001.10.79.

[16] Kamal Jain and Vijay V Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. In *40th annual symposium on foundations of computer science (Cat. No. 99CB37039)*, pages 2–13. IEEE, 1999.

[17] Hong Jun Jeon, Smitha Milli, and Anca Dragan. Reward-rational (implicit) choice: A unifying formalism for reward learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 4415–4426. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/2f10c1578a0706e06b6d7db6f0b4a6af-Abstract.html.

[18] Mingxi Jia, Haojie Huang, Zhewen Zhang, Chenghao Wang, Linfeng Zhao, Dian Wang, Jason Xinyu Liu, Robin Walters, Robert Platt, and Stefanie Tellex. Open-vocabulary pick and place via patch-level semantic maps. *arXiv preprint arXiv:2406.15677*, 2024.

[19] Tae Kyun Kim. T test as a parametric statistic. *Korean journal of anesthesiology*, 68(6):540–546, 2015.

[20] Pallavi Koppol, Henny Admoni, and Reid G Simmons.

Interaction considerations in learning from humans. In *IJCAI*, pages 283–291, 2021.

[21] Huihan Liu, Soroush Nasiriany, Lance Zhang, Zhiyao Bao, and Yuke Zhu. Robot Learning on the Job: Human-in-the-Loop Autonomy and Learning During Deployment. URL http://arxiv.org/abs/2211.08416.

[22] Shaunak A. Mehta and Dylan P. Losey. Unified Learning from Demonstrations, Corrections, and Preferences during Physical Human-Robot Interaction, January 2024. URL http://arxiv.org/abs/2207.03395. arXiv:2207.03395 [cs].

[23] Yannick Metz, David Lindner, Raphaël Baur, and Mennatallah El-Assady. Mapping out the space of human feedback for reinforcement learning: A conceptual framework, 2024. URL https://arxiv.org/abs/2411.11761.

[24] Austin Narcomey, Nathan Tsoi, Ruta Desai, and Marynel Vázquez. Learning human preferences over robot behavior as soft planning constraints. *arXiv preprint arXiv:2403.19795*, 2024.

[25] Heramb Nemlekar, Robert Ramirez Sanchez, and Dylan P. Losey. Pecan: Personalizing robot behaviors through a learned canonical space, 2024. URL https://arxiv.org/abs/2407.16081.

[26] Andi Peng, Yuying Sun, Tianmin Shu, and David Abel. Pragmatic feature preferences: Learning reward-relevant preferences from human input. *arXiv preprint arXiv:2405.14769*, 2024.

[27] Mattia Racca, Antti Oulasvirta, and Ville Kyrki. Teacher-aware active robot learning. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 335–343. IEEE, 2019.

[28] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

[29] Dorsa Sadigh, Anca Dragan, Shankar Sastry, and Sanjit Seshia. Active Preference-Based Learning of Reward Functions. In *Robotics: Science and Systems XIII*. ISBN 978-0-9923747-3-0. doi: 10.15607/RSS.2017.XIII.053.

[30] Christopher J Shannon, David C Horney, Kimberly F Jackson, and Jonathan P How. Human-autonomy teaming using flexible human performance models: An initial pilot study. *Advances in Human Factors in Robots and Unmanned Systems*, page 211, 2017.

[31] Lars St, Svante Wold, et al. Analysis of variance (anova). *Chemometrics and intelligent laboratory systems*, 6(4): 259–272, 1989.

[32] Linda van der Spaa, Jens Kober, and Michael Gienger. Simultaneously learning intentions and preferences during physical human-robot cooperation. *Autonomous Robots*, 48(4):11, 2024.

[33] Shivam Vats, Oliver Kroemer, and Maxim Likhachev. Synergistic scheduling of learning and allocation of tasks in human-robot teams. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2789–2795. IEEE, 2022.

[34] Eric W Weisstein. Bonferroni correction. *https://mathworld. wolfram. com/*, 2004.

[35] David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.

[36] Michelle Zhao, Reid Simmons, Henny Admoni, Aaditya Ramdas, and Andrea Bajcsy. Conformalized interactive imitation learning: Handling expert shift and intermittent feedback, 2024. URL https://arxiv.org/abs/2410.08852.

[37] Michelle D Zhao, Reid Simmons, and Henny Admoni. Learning human contribution preferences in collaborative human-robot tasks. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 3597–3618. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/zhao23b.html.

[38] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, Yifeng Zhu, and Kevin Lin. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

APPENDIX A
BAYESIAN PREFERENCE BELIEF ESTIMATOR

We model the human's response x_i to the robot's request for user preferences on task τ_i using a Bayesian belief estimator. Let $(b_i)_{i=1}^{N}$ be the set of beliefs for the sequence of tasks at timestep t, with b^i the current beliefs for task τ_i. The updated beliefs are

$$b^{'i}(\theta') := b^i(\theta'|x_i) \propto b^i(\theta')\mathbb{P}[x_i|\theta']$$

which serve as updated beliefs for τ_i after receiving the human's response. The domain-specific likelihood $\mathbb{P}[x|\theta']$ is defined in Section E. The robot posits similar objects receive similar treatment and updates its beliefs for future similar tasks using the similarity function between tasks. That is, $\forall j \in (i, N)$, if $f(\|\tau_i - \tau_j\|) < \epsilon$, we update $b^{'j}(\theta') := b^j(\theta'|x) \propto b^j(\theta')\mathbb{P}[x_i|\theta']$.

APPENDIX B
ENVIRONMENT DETAILS

Fig. 8: Manipulation domain. Each interaction involves picking 50 objects and placing them in their respective bins.

In this section, we describe in detail the environments used in our experimental evaluation.

1) **Gridworld** is a discrete, 17x17 grid comprised of a sequence of 15 total objects of nine varieties distinguished by object *type*, *color*, and *position*. Each object defines a task the agent must execute, and each task is to navigate to an object, pick it up, and transport it to the user's preferred location for that object. There are three possible goal locations. The 15 objects are randomly drawn from a set of 9 unique objects, whose distribution may vary. The robot is initialized with an empty skill library $\mathcal{L}_0$. We report results for 30 randomly initialized task sequences.
2) **7DoF Manipulation** is a simulated decluttering task where a Franka robot manipulator must pick up and put away 30 total objects of seven different varieties (e.g., milk carton, mug) into one of four different shelves. The frequency and order of all the objects are randomly sampled to generate a task sequence. Each object defines a task the agent must execute. The robot is initialized with an empty skill library $\mathcal{L}_0$.
3) **Conveyor** is a real-world environment where a Franka tabletop manipulator needs to pick up the object on the conveyor and place it into one of the three colored boxes to which it belongs. The robot is initialized with an empty skill library $\mathcal{L}_0$. We randomly sample five task sequences consisting of 20 tasks each. Since many objects are common across these experiments, we use a pretrained policy as the expert demonstrator to avoid redundant effort in the experiments. We pretrain GEM [18] using human demonstrations for every task. GEM provides language-conditioned few-shot learning ability via $SE(2)$-equivariance and open-vocabulary relevancy maps.

APPENDIX C
HUMAN COST PROFILES

In our evaluations, we examined the behavior of **COIL** compared with baselines under a suite of human teaching cost profiles. We draw from prior literature on teaching with different feedback types [10] to assign reasonable costs to each. We intuit robot execution is minimally burdensome and refer to prior works which find that demonstrations are more costly than preference queries [20]. The question is for different users, how much more costly is a demonstration than a preference request, and how do those costs compare to the cost of the human executing the task themselves. As these relative costs may differ across users, we investigate the performance of each algorithm under various demonstration (skill teaching) cost profiles. We assign a cost of $c_{rob} = 10$ to each robot execution, $c_{hum} = 80$ to human task execution, and $c_{pref} = 20$ to each preference query. The robot incurs a penalty cost of $c_{fail} = 100$ if it fails to successfully execute any skill, and a penalty of $c_{pref-fail} = 100$ if it successfully executes a skill by placing the object in a goal undesired by the user. We examine profiles across the cost of human skill teaching.

1) *Low-Cost Teaching* ($c_{skill} = 50$): This profile simulates an experienced teacher for whom providing demonstrations of robot skills is not burdensome.
2) *Med-Cost Teaching* ($c_{skill} = 100$): This profile models teaching to be moderately more burdensome than performing the task themselves.
3) *High-Cost Teaching* ($c_{skill} = 200$): This profile simulates a novice teacher for whom providing demonstrations of robot skills is highly burdensome.

APPENDIX D
BASELINE DETAILS

In this section, we describe in detail the baselines used in our experimental evaluation.

A. Information Gain (IG).

At timestep t, with the current task τ_i, **IG** reasons over actions in set $A = \{pref, skill, rob, hum\}$.

IG takes action a^{τ_i} at task τ_i where:

$$a^{\tau^i} = \arg\max_{a \in A} \left[IG^{\tau_i}(a)\mathbb{1}_{pref} + SG^{\tau_i}(a)\mathbb{1}_{skill} \right] - \beta c(a) \tag{6}$$

where $\mathbb{1}_{pref}$ is a indicator variable equal to 1 if $a = pref$, and so on. Effectively, this objective reasons about (1) the information gain in preference space, IG, of requesting a preference, and (2) the information gain in skill space, or skill gain SG of requesting a skill demonstration. For each potential robot action, the objective compares the potential gain relative to the cost of the query, where cost is scaled by factor β to enable composition. We tune the β hyperparameter, with details in Section D-D.

Let $(b_i)_{i=1}^N$ be the set of beliefs for the sequence of tasks at timestep t. Using a Bayesian belief update, let $(b_i^g)_{i=1}^N$ be the set of updated beliefs for the sequence of tasks at timestep t when response g is given to query a on task τ_i. In turn, the optimistic entropy reduction when the best potential choice g is made given query a can be written as $IG^{\tau_i}(a) = \max_g \sum_{j=i}^N H(b_j^g) - H(b_i)$.

Skill gain is computed similarly and can be reasoned about as information gain over skills. Let $\mathcal{L}_t$ be the current skill library. Let $\mathcal{L}_t^g = \mathcal{L}_t \cup \tau_i$ be the updated skill library should a demonstration to goal g be requested on τ_i. Skill gain, then, can be represented as $SG^{\tau_i}(a) = \max_g \sum_{j=i}^N \mathbb{E}_{b_j^\theta}\left[\rho^{\text{safe}}_{\pi,\mathcal{L}_t^g}(\tau_j,\theta) - \rho^{\text{safe}}_{\pi,\mathcal{L}_t}(\tau_j,\theta)\right]$, where $\rho^{\text{safe}}_{\pi,\mathcal{L}_t}(\tau_j,\theta')$ is the likelihood of safe execution on τ_j given skill library $\mathcal{L}_t$ (computed by a lookup into $\mathcal{L}_t$).

Lastly, β represents a scale factor needed to scale the costs in order to make them comparable with information gain and skill gain values. Let $\hat{\theta}_j := \arg\max_{\theta'} b_j^\theta$ be the maximum a posteriori estimate for the robot's belief on the preference for task τ_j. The costs for each query follow from the costs used by COIL:

1) $c(pref) = c^i_{\text{pref}}$
2) $c(rob) = c^i_{\text{rob}} + c_{\text{skill-fail}} \cdot (1 - \rho^{\text{safe}}_{\pi,\mathcal{L}_t}(\tau_j, \hat{\theta}_j)) + c_{\text{pref-fail}} \cdot (1 - b_j^\theta(\hat{\theta}_j))$
3) $c(skill) = c^i_{\text{skill}} + c^i_{\text{rob}} + c_{\text{skill-fail}} \cdot (1 - \rho^{\text{safe}}_{\pi,\mathcal{L}_t}(\tau_j, \hat{\theta}_j))$
4) $c(hum) = c^i_{hum}$

B. Confidence-based ADL (C-ADL)

Vats et al. [33] use a mixed integer program to decide when to learn new skills and when to delegate tasks to the user. While their method accounts for the future utility of learning a task, they do so without also considering the human's preferences. In order to ensure a fair comparison in our paradigm, we enable **C-ADL** to request preference queries for each task with a confidence-based threshold. We use a confidence threshold of $\alpha = 0.8$. If the robot isn't sufficiently confident about the user preference, the robot requests a preference. Once received, the robot acts according to its ADL plan with $\hat{\theta}_j := \arg\max_{\theta'} b_j^\theta$ the maximum a posteriori estimate for the robot's belief on the preference for task τ_j.

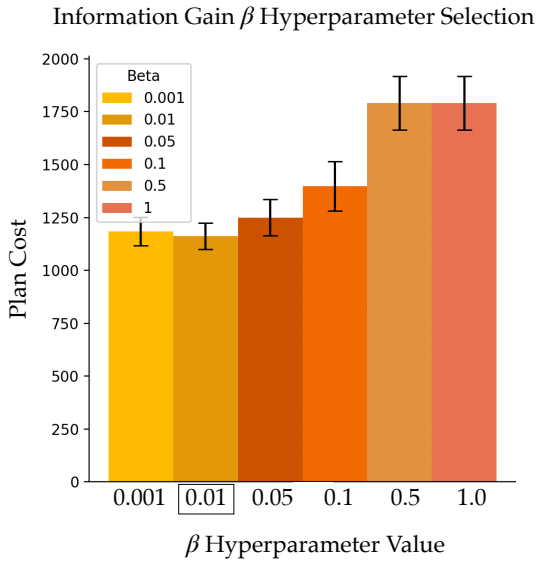

Fig. 9: We plot the average plan cost for 10 seeds per all of the three human cost profiles. For **IG** in our comparisons, we choose the scale factor which achieves the lowest plan cost, $\beta = 0.01$.

C. Confidence-based Autonomy (CBA).

Inspired by Chernova and Veloso [6], **CBA** requests skill and preference teaching if the robot is uncertain about either. Otherwise, it assigns the task to itself or the human. We use a confidence threshold of $\alpha = 0.8$. If the robot isn't sufficiently confident about the user preference, the robot requests the user's preference. Let $\hat{\theta}_j := \arg\max_{\theta'} b_j^\theta$ be the maximum a posteriori estimate for the robot's belief on the preference for task τ_j. Next, the robot requests a skill query if it isn't sufficiently confident on the execution of the skill. Given current skill library at timestep t, recall $\mathcal{L}_t$, $\rho^{\text{safe}}_{\pi,\mathcal{L}_t}(\tau_j, \hat{\theta}_j)$ is the likelihood of task success on τ_j given skill library $\mathcal{L}_t$, which can be computed by a lookup into $\mathcal{L}_t$. If both are confident ($\rho^{\text{safe}}_{\pi,\mathcal{L}_t}(\tau_j, \hat{\theta}_j) > \alpha$), then the robot executes its skill with the most likely preference. Important to note, this is a learning-prioritizing baseline, as it reasons through all learning options, and doesn't reason over delegation to the human.

D. Information Gain Hyperparameter Sensitivity Analysis

The **IG** baseline relies on a scale factor β which scales the human teaching costs to be evaluated jointly with the information gain. The choice of this β scale factor influences the performance of the algorithm, as a factor too high will focus only on picking the action with the minimal cost, disregarding information gain. Prior to running our comparisons, we first running a sensitivity analysis on **IG** for different values of $\beta \in [0.001, 0.01, 0.05, 0.1, 0.5, 1.0]$. In Figure 9, we plot the average plan cost for 10 seeds per each of the three human cost profiles. For **IG** in our comparisons, we choose the scale factor which achieves the lowest plan cost, $\beta = 0.01$.

APPENDIX E
DOMAIN-SPECIFIC INSTANTIATIONS OF SKILL RETURN MODEL AND PREFERENCE BELIEFS

A. Gridworld

We provide the robot with a domain-specific function ρ_π^{safe} to predict the probability of safe execution. $\rho_\pi^{\text{safe}}(\tau', \theta')$ predicts the probability of safe execution of a skill π when deployed on a task τ' with parameter θ'. This function is defined based on similarity between (τ', θ') and the tasks currently in the robot's skill library $\mathcal{L}$. For the Gridworld domain, we define $\rho_\pi^{\text{safe}}(\tau', \theta') = 1$ if $\|\tau - \tau'\| < \epsilon$ for any $\tau \in \mathcal{L}$, where $\|\tau - \tau'\| = 0$ if the object at τ and τ' are of the same type and color, and 0 otherwise. ϵ is a small constant 0.01.

The set of beliefs for the sequence of tasks is initialized $(b_i)_{i=1}^N$ is initialized as a uniform distribution over Bernoulli random variables indicating whether each goal position is suitable for the task object. Upon receiving a human response for where the object should be placed, the likelihood function for Bernoulli random variable associated with the goal is given a probability of 1, with the others 0, which is used in the Bayes update to update the robot's preference beliefs.

B. Simulated 7DoF Manipulation

The manipulation domain has 7 different types of objects: milk carton, bread loaf, cereal box, can, bottle, lemon and cup. Each object is described by its type, color, dimensions and pose. We define $\rho_\pi^{\text{safe}}(\tau', \theta') = 1$ if $\|\tau - \tau'\| < \epsilon$ and θ is equal to θ', where τ and θ are the task and preference respectively that π was trained for. $\|\tau - \tau'\| = 0$ if the object at τ and τ' are of the same type (e.g. milk carton), and 0 otherwise. ϵ is a small constant 0.01. This is based on the assumption that the skills taught to the robot can generalize to different colors and object poses. We use the same similarity function in the preference belief estimator to update the beliefs of similar future objects.

C. Conveyor

The conveyor domain has 12 different types of objects: pink bottle, green bottle, white mug, red mug, brown tape, black tape, orange block, blue block, banana, lemon, can and spam. Each object is described by its type, category $\in \{\text{office}, \text{kitchen}, \text{toys}\}$, color, dimensions and pose. We define $\rho_\pi^{\text{safe}}(\tau', \theta') = 1$ if $\|\tau - \tau'\| < \epsilon$ and θ is equal to θ', where τ and θ are the task and preference respectively that π was trained for. $\|\tau - \tau'\| = 0$ if the object at τ and τ' of the same type (e.g. tape), and 0 otherwise. ϵ is a small constant 0.01. This is based on the assumption that the skills taught to the robot can generalize to variations in color, dimensions and pose. The preference belief estimator uses the object category to update the beliefs of other objects. This is based on the assumption that the human is likely to have similar preferences for objects of the same category.

D. Gridworld Under All Teachable Objects: Statistical Analyses

Our experiments under varied human cost profiles, we evaluated our null hypothesis of no significance in the sample (N=30) of total costs between the planning approaches. We compared statistical differences using a one-way ANOVA [31] between costs achieved for each algorithm, and evaluated pairwise differences using pairwise t-tests [19] with Bonferroni correction [34] if significant main effects were present. For the Low-Cost human profile, we did not find significant differences between the approaches.

For the Med-Cost human cost profile in the Gridworld domain, we found significant effects of planner ($F = 23.48, p < 0.001$), and found significant ($p < 0.01$) pairwise differences between **COIL** and all baseline algorithms. **COIL** achieved significantly lower cost plans than **C-ADL** ($F = -3.54, p < 0.001$), **IG** ($F = -7.13, p < 0.001$), and **CBA** ($F = -7.12, p < 0.001$). **C-ADL** also achieved significantly lower plan costs than **IG** ($F = -4.14, p < 0.001$) and **CBA** ($F = -4.14, p < 0.001$).

For the High-Cost human cost profile in the Gridworld domain, we found significant effects of planner ($F = 120.58, p < 0.001$), and found significant ($p < 0.01$) pairwise differences between **COIL** versus **IG** ($F = -5.92, p < 0.001$) and **CBA** ($F = -15.07, p < 0.001$). **C-ADL** also achieved significantly lower plan costs than **IG** ($F = -4.43, p < 0.001$) and **CBA** ($F = -14.07, p < 0.001$).

APPENDIX F
ADDITIONAL RESULTS ON ADAPTING TO TEACHING FAILURES IN THE GRIDWORLD DOMAIN

We similarly compared statistical differences using a one-way ANOVA between costs achieved for each algorithm, and evaluated pairwise differences using pairwise t-tests with Bonferroni correction if significant main effects were present.

A. Adapting when 10% of Objects are Challenging-to-Learn

We did not find significant effects of planner in the Low-Cost profile. Under Med-Cost in the Gridworld domain, we found significant effects of planner ($F = 14.54, p < 0.001$), and found significant ($p < 0.01$) pairwise differences between **COIL** versus **IG** and **CBA**. We did not find statistical significance between **COIL** and **COIL-NoAdapt** plan costs, but observe a slight decrease in plan cost in Figure 10. **COIL** achieved significantly lower cost plans than **IG** ($F = -5.73, p < 0.001$), and **CBA** ($F = -5.73, p < 0.001$). **COIL-NoAdapt** also achieved significantly lower plan costs than **IG** ($F = -1.25, p < 0.001$) and **CBA** ($F = -4.90, p < 0.001$). **C-ADL** outperformed **IG** ($F = -3.70, p < 0.001$) and **CBA** ($F = -3.70, p < 0.001$).

Under High-Cost in the Gridworld domain, we found significant effects of planner ($F = 222.61, p < 0.001$), and found significant pairwise differences between **COIL** versus **C-ADL**, **IG** and **CBA**. We did not find statistical significance between **COIL** and **COIL-NoAdapt** plan costs. **COIL** achieved significantly lower cost plans than **C-ADL** ($F = -6.55, p < 0.001$),

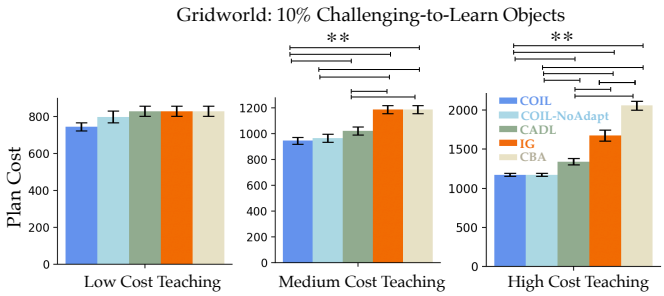

Fig. 10: In the Gridworld environment, when 10% of the objects are challenging-to-learn, **COIL** achieves the lowest plans, compared to **COIL-NoAdapt** and other baselines in all teaching profiles. In high-cost and med-cost teaching, **COIL** and **COIL-NoAdapt** often assign tasks to the human off the bat, reducing the impact of adaptivity. Error bars represent standard error over N=30.

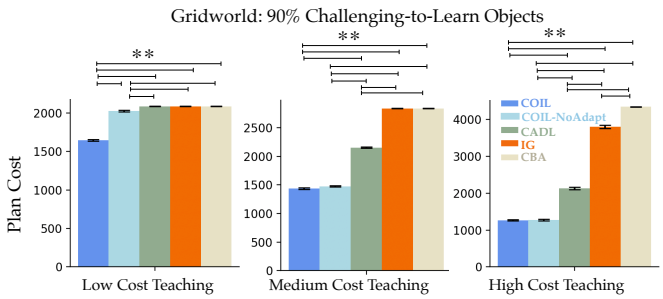

Fig. 11: In the Gridworld environment, when 90% of the objects are challenging-to-learn, **COIL** achieves the lowest plans, compared to **COIL-NoAdapt** and other baselines in all teaching profiles. In high-cost and med-cost teaching, **COIL** and **COIL-NoAdapt** often assign tasks to the human off the bat, reducing the impact of adaptivity. Error bars represent standard error over N=30.

IG ($F = -20.32, p < 0.001$), and **CBA** ($F = -20.25, p < 0.001$). **COIL-NoAdapt** also achieved significantly lower plan costs than **C-ADL** ($F = -6.47, p < 0.001$), **IG** ($F = 20.24, p < 0.001$) and **CBA** ($F = -20.24, p < 0.001$). **C-ADL** outperformed **IG** ($F = -13.08, p < 0.001$) and **C-ADL** ($F = - - 13.35, p < 0.001$), and **IG** outperformed **CBA** ($F = -0.55, p < 0.001$).

B. Adapting when 50% of Objects are Challenging-to-Learn

Under Low-Cost in the Gridworld domain, we found significant effects of planner ($F = 18.57, p < 0.001$), and found significant pairwise differences between **COIL** compared to all baselines. We found statistical significance between **COIL** and **COIL-NoAdapt** plan costs, with **COIL** achieving a lower plan cost than **COIL-NoAdapt** ($F = -4.57, p < 0.001$). **COIL** achieved significantly lower cost plans than **C-ADL** ($F = -7.72, p < 0.001$), **IG** ($F = -7.72, p < 0.001$), and **CBA** ($F = -7.72, p < 0.001$). We did not find additional pairwise significances between the other planners.

Under Med-Cost in the Gridworld domain when half of the unique objects are challenging-to-learn, we found significant effects of planner ($F = 50.03, p < 0.001$), and found significant pairwise differences between **COIL** versus **C-ADL**, **IG** and **CBA**. We did not find statistical significance between **COIL** and **COIL-NoAdapt** plan costs. **COIL** achieved significantly lower cost plans than **C-ADL** ($F = -4.91, p < 0.001$), **IG** ($F = -11.74, p < 0.001$), and **CBA** ($F = -11.74, p < 0.001$). **COIL-NoAdapt** also achieved significantly lower plan costs than **IG** ($F = -8.73, p < 0.001$) and **CBA** ($F = -8.73, p < 0.001$), but had no difference with **C-ADL**. **C-ADL** outperformed **IG** ($F = -5.63, p < 0.001$) and **CBA** ($F = -5.63, p < 0.001$).

Under High-Cost, we found significant effects of planner ($F = 127.82, p < 0.001$), and found significant pairwise differences between **COIL** versus **C-ADL**, **IG** and **CBA**. We did not find statistical significance between **COIL** and **COIL-NoAdapt** plan costs. **COIL** achieved significantly lower cost plans than **C-ADL** ($F = -6.20, p < 0.001$), **IG** ($F = -12.15, p < 0.001$), and **CBA** ($F = -18.80, p < 0.001$). **COIL-NoAdapt** also achieved significantly lower plan costs than **C-ADL** ($F = -6.20, p < 0.001$), **IG** ($F = -12.15, p < 0.001$) and **CBA** ($F = -18.80, p < 0.001$). **C-ADL** outperformed **IG** ($F = -7.11, p < 0.001$) and **CBA** ($F = -11.74, p < 0.001$), and **IG** outperformed **CBA** ($F = -3.02, p < 0.001$).

C. Adapting when 90% of Objects are Challenging-to-Learn

We lastly increased the proportion of challenging-to-learn objects to nearly all object varieties to evaluate how each planner influenced the interaction (Figure 11). Under Low-Cost, we found significant effects of planner ($F = 918.38, p < 0.001$), and found significant pairwise differences between **COIL** versus all baselines. **COIL** achieved significantly lower cost plans than **COIL-NoAdapt** ($F = 28.35, p < 0.001$), **C-ADL** ($F = -40.92, p < 0.001$), **IG** ($F = -40.92, p < 0.001$), and **CBA** ($F = -40.92, p < 0.001$). **COIL-NoAdapt** also achieved significantly lower plan costs than **C-ADL** ($F = -6.90, p < 0.001$), **IG** ($F = -6.90, p < 0.001$) and **CBA** ($F = -6.90, p < 0.001$). We did not find additional pairwise significances between the other planners.

Under Med-Cost, we found significant effects of planner ($F = 4513.26, p < 0.001$), and found significant pairwise differences between **COIL** versus **C-ADL**, **IG** and **CBA**. We did not find statistical significance between **COIL** and **COIL-NoAdapt** plan costs. **COIL** achieved significantly lower cost plans than **C-ADL** ($F = -41.10, p < 0.001$), **IG** ($F = -100.08, p < 0.001$), and **CBA** ($F = -100.08, p < 0.001$). **COIL-NoAdapt** also achieved significantly lower plan costs than **C-ADL** ($F = -37.36, p < 0.001$), **IG** ($F = -92.01, p < 0.001$) and **CBA** ($F = -92.01, p < 0.001$). **C-ADL** outperformed **IG** ($F = -62.45, p < 0.001$) and **CBA** ($F = -62.45, p < 0.001$).

Under High-Cost, we found significant effects of planner ($F = 2596.18, p < 0.001$), and found significant pairwise differences between **COIL** versus **C-ADL**, **IG** and **CBA**. We did not find statistical significance between **COIL** and **COIL-NoAdapt** plan costs. **COIL** achieved significantly lower cost plans than **C-ADL** ($F = -23.25, p < 0.001$), **IG** ($F = -51.29, p < 0.001$), and **CBA** ($F = -169.35, p <$

0.001). **COIL-NoAdapt** also achieved significantly lower plan costs than **C-ADL** ($F = -22.16, p < 0.001$), **IG** ($F = -49.97, p < 0.001$) and **CBA** ($F = -145.20, p < 0.001$). **C-ADL** outperformed **IG** ($F = -29.50, p < 0.001$) and **CBA** ($F = -67.20, p < 0.001$), and **IG** outperformed **CBA** ($F = -11.71, p < 0.001$).

APPENDIX G
SIMULATED 7DOF MANIPULATION: STATISTICAL ANALYSES

We similarly compared statistical differences using a one-way ANOVA between costs achieved for each algorithm, and evaluated pairwise differences using pairwise t-tests with Bonferroni correction if significant main effects were present. In this experiment, we compare with our 3 strongest baselines, derived from our earlier results, removing **CBA** from our analysis.

Under Low-Cost, we found significant effects of planner ($F = 996.49, p < 0.001$), and found significant pairwise differences between **COIL** versus all baselines. **COIL** achieved significantly lower cost plans than **COIL-NoAdapt** ($F = -6.99, p < 0.001$), **C-ADL** ($F = -11.10, p < 0.001$), **IG** ($F = -60.68, p < 0.001$). **COIL-NoAdapt** also achieved significantly lower plan costs than **C-ADL** ($F = -3.41, p < 0.001$), and **IG** ($F = -49.19, p < 0.001$). **C-ADL** outperformed **IG** ($F = -36.55, p < 0.001$).

Under Med-Cost, we found significant effects of planner ($F = 4445.39, p < 0.001$), and found significant pairwise differences between **COIL** versus all baselines. **COIL** achieved significantly lower cost plans than **COIL-NoAdapt** ($F = -11.99, p < 0.001$), **C-ADL** ($F = -11.16, p < 0.001$), **IG** ($F = -138.80, p < 0.001$). **COIL-NoAdapt** also achieved significantly lower plan costs than **C-ADL** ($F = -2.35, p < 0.001$), and **IG** ($F = -111.16, p < 0.001$). **C-ADL** outperformed **IG** ($F = -87.31, p < 0.001$).

Under High-Cost, we found significant effects of planner ($F = 166.95, p < 0.001$), and found significant pairwise differences between **COIL** versus all baselines. **COIL** achieved significantly lower cost plans than **COIL-NoAdapt** ($F = -15.71, p < 0.001$), **C-ADL** ($F = -19.03, p < 0.001$), **IG** ($F = -36.22, p < 0.001$). **COIL-NoAdapt** also achieved significantly lower plan costs than **C-ADL** ($F = -4.60, p < 0.001$), but had no difference when compared to **IG**. **C-ADL** underperformed **IG** ($F = 4.29, p < 0.001$).