

# Learning to Share Autonomy Across Repeated Interaction

Ananth Jonnavittula and Dylan P. Losey

**Abstract**— Wheelchair-mounted robotic arms (and other assistive robots) should help their users perform everyday tasks. One way robots can provide this assistance is *shared autonomy*. Within shared autonomy, both the human and robot maintain control over the robot’s motion: as the robot becomes confident it understands what the human wants, it increasingly intervenes to automate the task. But how does the robot know what tasks the human may want to perform in the first place? Today’s shared autonomy approaches often rely on *prior knowledge*: for example, the robot must know the set of possible human goals *a priori*. In the long-term, however, this prior knowledge will inevitably break down — sooner or later the human will reach for a goal that the robot did not expect. In this paper we propose a learning approach to shared autonomy that takes advantage of repeated interactions. Learning to assist humans would be impossible if they performed completely different tasks at every interaction: but our insight is that users living with physical disabilities *repeat* important tasks on a daily basis (e.g., opening the fridge, making coffee, and having dinner). We introduce an algorithm that exploits these repeated interactions to *recognize* the human’s task, *replicate* similar demonstrations, and *return* control when unsure. As the human repeatedly works with this robot, our approach continually learns to assist tasks that were never specified beforehand: these tasks include both discrete goals (e.g., reaching a cup) and continuous skills (e.g., opening a drawer). Across simulations and an in-person user study, we demonstrate that robots leveraging our approach match existing shared autonomy methods for known goals, and outperform imitation learning baselines on new tasks. See videos here: <https://youtu.be/P1h4t3wQeIA>

## I. INTRODUCTION

Imagine teleoperating a wheelchair-mounted robot arm to open your refrigerator door (see Figure 1). The first few times you need to open the fridge, you must carefully guide the robot throughout the entire process of reaching, pulling, and opening the door. But after you’ve interacted with this robot for several weeks — and opened your fridge *many* times — the intelligent robot should learn to *assist* you. The next time you start teleoperating the arm towards your fridge, this robot should recognize what you want, and autonomously take over to help pull open the door.

Today’s assistive robot arms often leverage *shared autonomy* to help their users perform complex tasks [1]–[7]. Here the robot is given a discrete set of tasks the human may want to complete *a priori*. For instance, the human may want to reach for their cup or move their plate. Based on the human’s inputs so far, the robot infers which task the human is trying to perform, and arbitrates control to automate that task.

This approach to shared autonomy makes sense when the robot knows *all* your potential tasks — but what happens

The authors are members of the Collaborative Robotics Lab ([Collab](#)), Dept. of Mechanical Engineering, Virginia Tech, Blacksburg, VA 24061. e-mail: {ananth, losey}@vt.edu

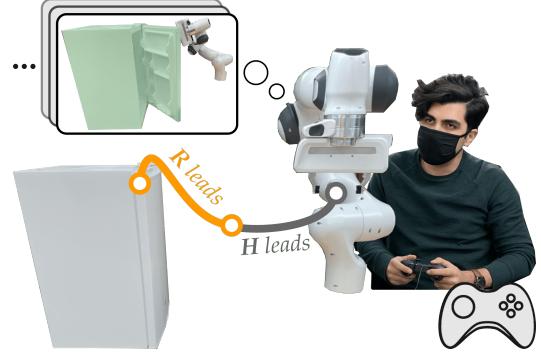


Fig. 1. User teleoperating an assistive robot arm to open their fridge door. The robot does not have any pre-programmed knowledge about this task; however, the human and robot have completed similar tasks many times before. We enable robots to learn to assist humans from scratch by exploiting the *repeated* nature of everyday tasks.

when some tasks are inevitably *left out*? Going back to our example, let’s say the robot does not know you might want to open the fridge. As you guide the arm towards the door, the robot gets confused: this robot is unsure about what task you are trying to do, and cannot assist you in this unexpected task. Even worse, the robot remains confused no matter how many times you repeat the process of opening the fridge.

For assistive robot arms to be helpful in practice, these robots must be capable of *learning* a spectrum of new tasks over time. This would be extremely challenging if every task was a unique one-off the robot had never seen before. But our insight is that, over the many weeks, months, and years a user living with disabilities works with their assistive robot:

*Humans constantly repeat tasks  
that are important in their everyday life.*

For example, you open your refrigerator door on a daily basis. Applying our insight enables assistive robot arms to learn to share autonomy by exploiting the *repeated interaction* inherent in assistive applications. Here the robot remembers how you controlled the arm to open the fridge in the past, recognizes that you are providing similar inputs during the current interaction, and assists by autonomously mimicking the behavior you previously demonstrated. Across repeated interactions, these robot arms should learn to assist not only *discrete goals* (e.g., reaching a cup) but also *continuous skills* (e.g., opening a door).

Overall, we make the following contributions:

**Capturing Latent Intent.** We formalize shared autonomy with repeated interaction. During each interaction the human has in mind some desired task: we introduce an autoencoder approach that learns to recognize the human’s current task and replicate similar past interactions.

**Returning Control when Uncertain.** Our approach should assist during previously seen tasks without interfering whenever the human tries to do a new task. We introduce a discriminator to measure the confidence of our learned shared autonomy, so that the robot returns control to the human whenever it is unsure about what the human really wants.

**Conducting a User Study.** We assess our resulting algorithm in scenarios where the human is trying to reach discrete goals and perform continuous skills. We baseline against safe imitation learning and shared autonomy approaches, and show that our algorithm enables the robot to learn to assist for initially unknown tasks across repeated interactions.

## II. RELATED WORK

**Application – Assistive Robot Arms.** Over 13% of American adults living with physical disabilities have difficulty with at least one activity of daily living (ADL) [8]. Assistive robots — such as wheelchair-mounted robot arms [9] — have the potential to help users perform these everyday tasks without relying on caregivers. Recent work on assistive robot arms has focused on automating ADLs such as eating dinner [10]–[12], getting dressed [13], and reaching common objects [14]. Our research takes inspiration from the fact that people living with disabilities need *assistance* performing tasks that they *repeat* on a daily basis.

**Shared Autonomy.** How should we provide this assistance? Rather than relying on the human to constantly teleoperate the robot arm, often it makes sense for the robot to automate parts of the task it understands. In shared autonomy both the human and robot arbitrate control over the robot’s motion. We divide related work on shared autonomy into two groups: *inference* and *optimization*.

Within inference works such as [1]–[7], the robot is given a discrete set of possible goals the human may want to reach *a priori*. Based on the human’s inputs so far, the robot infers which goal is most likely, and autonomously moves towards that target. The level of robotic assistance is proportional to the robot’s confidence in the human’s goal.

By contrast, within optimization works like [15]–[18] the robot is given an underlying reward function that the human wants to optimize (e.g., maximize distance from obstacles). At timesteps when the user’s input would lead to low reward, the robot intervenes, and corrects the human’s action.

Both approaches to shared autonomy require *prior knowledge* about what the human wants: the robot must observe the human’s potential goals or underlying reward function. But in the long-term these priors will inevitably fail — sooner or later the human will reach for a goal that the robot did not expect. Accordingly, here we remove this key assumption by exploiting *repetition* in assistive human-robot interaction.

**Repeated Interaction.** Prior work enables robots to adapt to humans over the course of repeated interaction. For example, robots can recognize emerging conventions in human inputs [19], or imitate the human’s behavior [20]. Most relevant to our approach is existing research on *latent representations* [21]–[24]. Here the robot learns to embed complex behavior

into a compact, high-level representation (e.g., skills, strategies, or plans). We similarly apply latent representations to learn the human’s desired *tasks* across repeated interaction.

## III. FORMALIZING SHARED AUTONOMY WITH REPEATED INTERACTION

Let us return to our motivating example where the user is teleoperating their assistive robot arm. Each time the human interacts with the robot, they have in mind a *task* they want the robot to perform: some of these tasks are new (e.g., moving a coffee cup), while others the robot may have seen before (e.g., opening the fridge). We represent the human’s current task as  $z \in \mathcal{Z}$ , so that during interaction  $i$ , the human wants to complete task  $z^i$ . Within this paper tasks include both discrete goals and continuous skills: i.e., a task  $z$  could be reaching the cup or opening a drawer. We test both types of tasks  $z$  in our experiments. The assistive robot’s goal is to help the human complete their current task. However, the robot does not know (a) *which* task the human currently has in mind or (b) *how* to correctly perform that task.

**Dynamics.** The robot arm is in state  $s$  and takes action  $a$ . Within our experiments,  $s$  is the robot’s joint position,  $a$  is the robot’s joint velocity, and the robot has dynamics:

$$s^{t+1} = s^t + \Delta t \cdot a^t \quad (1)$$

The human uses a joystick to tell the robot what action to take. Let  $a_{\mathcal{H}}$  be the human’s *commanded action* — i.e., the joint velocity corresponding to the human’s joystick input. The robot assists the human with an *autonomous action*  $a_{\mathcal{R}}$ , so that the overall action  $a$  is a linear blend of the human’s joystick input and the robot’s assistive guidance [1], [2]:

$$a = \beta \cdot a_{\mathcal{R}} + (1 - \beta) \cdot a_{\mathcal{H}} \quad (2)$$

Here  $\beta \in [0, 1]$  arbitrates control between human and robot. When  $\beta \rightarrow 0$ , the human always controls the robot, and when  $\beta \rightarrow 1$ , the robot acts autonomously.

**Human.** So how does the human choose inputs  $a_{\mathcal{H}}$ ? During interaction  $i$  we assume the human has in mind a desired task  $z^i$ . We know that this task guides the human’s commanded actions; similar to prior work [3], we accordingly write the *human’s policy* as  $\pi_{\mathcal{H}}(a_{\mathcal{H}} | s, z^i)$ . This policy is the gold standard, because if we knew  $\pi_h$  we would know exactly how the human likes to perform each task  $z \in \mathcal{Z}$ . It’s important to recognize that this policy is highly *personalized*. Imagine that the current task is to reach a coffee cup at state  $s^*$ : one human might prefer to move directly towards the cup with actions  $a_{\mathcal{H}} \propto (s^* - s)$ , while another user takes a circuitous route to stay farther away from obstacles.

**Repeated Interaction.** In practice the assistive robot cannot directly observe either  $z^i$  or  $\pi_h$ . Instead, the robot observes the states that it visits and the commands that the human provides. Let  $\tau = \{(s^1, a_{\mathcal{H}}^1), \dots, (s^T, a_{\mathcal{H}}^T)\}$  be the entire *sequence* of robot states and human commands that the robot observed over the course of an interaction. As the human and robot repeatedly collaborate and interact, the robot collects a *dataset* of these sequences:  $\mathcal{D} = \{\tau^1, \tau^2, \dots, \tau^{i-1}\}$ . Notice

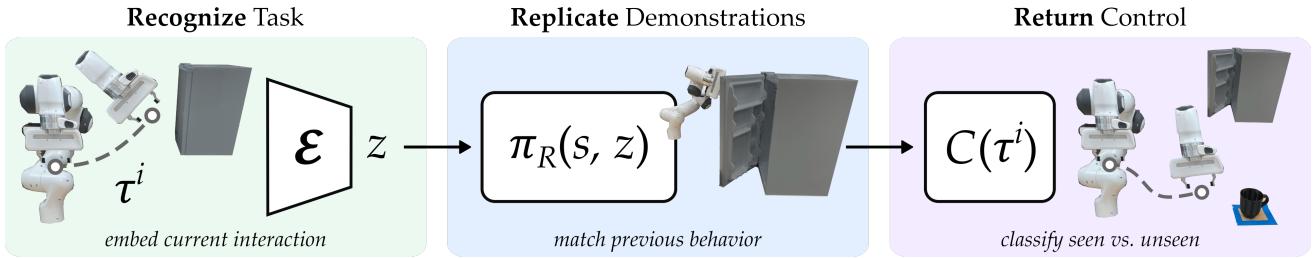


Fig. 2. Our proposed approach for learning to share autonomy across repeated interaction. (Left) The robot embeds the human’s behavior  $\tau^i$  during the current interaction to a distribution over latent tasks  $z$ . (Middle) The robot then chooses assistive actions  $a_{\mathcal{R}}$  conditioned on its state  $s$  and latent task  $z$ . Policy  $\pi_{\mathcal{R}}$  is trained to match the user’s behavior from previous interactions. (Right) To decide whether or not to trust this assistive action, the robot turns to a discriminator  $C$ . The discriminator assesses whether the current interaction  $\tau^i$  is similar to any previously seen interaction: if so, the robot increases autonomy. In this example the robot remembers how the human has opened the fridge in the past, and assists for that task. But when the human does something new (reaching for the cup) the robot realizes that it does not know how to help, and arbitrates control back to the human.

that here we distinguish the current interaction  $\tau^i$ . Because the robot only knows the states and human inputs up to the present time,  $\tau^i = \{(s^1, a_{\mathcal{H}}^1), \dots, (s^{t-1}, a_{\mathcal{H}}^{t-1})\}$ .

**Robot.** In settings where an assistive robot arm repeatedly interacts with a human, the robot has access to three pieces of information. The robot knows its state  $s$ , the human’s current behavior  $\tau^i$ , and the events of previous interactions  $\mathcal{D}$ . Given  $(s, \tau^i, \mathcal{D})$ , the robot needs to decide: (a) what assistance  $a_{\mathcal{R}}$  to provide and (b) how to arbitrate control with  $\beta$ . We emphasize that here the robot makes no assumptions about either the human’s underlying tasks or how to complete them — instead, the robot must extract this information from previous interactions.

#### IV. LEARNING TO RECOGNIZE TASKS, REPLICATE INTERACTIONS, AND RETURN CONTROL

Our proposed approach is guided by the intuition that — if the robot recognizes that the human’s behavior is similar to a previous interaction — the robot can assist by mimicking that previous interaction. Take our motivating example of opening the fridge door: the next time the human starts guiding the robot towards this door, the robot should infer which task the human is trying to perform, and then autonomously open the door just like the human did. There are three key challenges to this problem. First, the robot must **recognize** the human’s task  $z^i$  during the current interaction. Next, the robot should **replicate** any previous interactions that are similar to this task. Finally, the robot must know when it is unsure, and **return** control to the human if the task is new or unexpected. In this section we outline how our approach tackles these three main challenges (see Figure 2).

##### A. Recognize: Embedding Interactions to a Latent Space

Our first step is to extract the user’s high-level task  $z^i$  from the robot’s low-level observations. Recall that the human’s low-level behavior during the current interaction (i.e., their commanded actions at each robot state) is captured by  $\tau^i$ . This behavior is guided by the human’s desired task: when the human wants to open the fridge, they provide commands  $a_{\mathcal{H}}$  that move the robot towards that door, and when the human wants to pick up a coffee cup, they provide a different set of commands to reach that cup. Accordingly, we leverage

$\tau^i$  to recognize the underlying task  $z^i$ . More formally, we introduce an encoder:

$$z \sim \mathcal{E}(\cdot | \tau^i) \quad (3)$$

This encoder *embeds* the human’s behavior to a probability distribution over the latent space  $\mathcal{Z} \subseteq \mathbb{R}^d$ . We learn the encoder network from previous human interactions as described in the following subsection.

Our encoder  $\mathcal{E}$  is analogous to *goal prediction* from prior work on shared autonomy [1]–[3]. In these prior works, the robot records the human’s current behavior  $\tau^i$ , and then applies Bayesian inference to predict the human’s goal  $z^i$ . Our encoder  $\mathcal{E}(z | \tau^i)$  practically accomplishes the same thing: it provides us with a distribution over tasks the human may want to complete. The difference is that — when using Bayesian inference — the robot needs to know the human’s possible tasks *a priori*. When training the encoder we make no such assumption. Moreover, because the sequence  $\tau^i$  only considers the human’s actions  $a_{\mathcal{H}}$ , we avoid feedback loops where the robot unintentionally uses its own actions  $a_{\mathcal{R}}$  to convince itself of the human’s goal [3].

##### B. Replicate: Matching the Demonstrated Behavior

As the human uses their joystick to teleoperate the robot towards the fridge door, we leverage our encoder to recognize the human’s task. But what does the robot do once it knows that task? And how do we train the encoder in the first place? We address both issues by introducing a decoder that maps our task predictions into assistive robot actions:

$$a_{\mathcal{R}} = \pi_{\mathcal{R}}(s, z) \quad (4)$$

The decoder  $\pi_{\mathcal{R}}$  determines how the robot assists the human. We want this decoder to replicate previous demonstrations, so that if the human’s current behavior is similar to another interaction  $\tau \in \mathcal{D}$ , the robot will mimic the human’s actions from that previous interaction.

We accomplish this by training the encoder and decoder models using the dataset of interactions  $\mathcal{D}$ . More specifically, we take snippets of the human’s behavior during previous interactions, embed those snippets to a task prediction, and then reconstruct the human’s demonstrated behavior. For some past interaction  $\tau \in \mathcal{D}$ , let  $\xi = \{(s^1, a_{\mathcal{H}}^1), \dots, (s^{k-1}, a_{\mathcal{H}}^{k-1})\}$

be the human's behavior up to timestep  $k$ , and let  $(s^k, a_{\mathcal{H}}^k)$  be the human's behavior at timestep  $k$ . We train the encoder and decoder to minimize the loss function:

$$\mathcal{L} = \mathbb{E}_{z \sim \mathcal{E}(\cdot | \xi)} \|a_{\mathcal{H}}^k - \pi_{\mathcal{R}}(s^k, z)\|^2 \quad (5)$$

across the dataset  $\mathcal{D}$ . In other words, once we train  $\mathcal{E}$  and  $\pi_{\mathcal{R}}$ , we should be able to take a snippet of the human's past behavior and correctly predict the next action the human took. Equation (5) encourages the robot to mimic the human, so that when we encounter a familiar task, the robot will behave like the human did.

We contrast our decoder to *trajectory prediction* from prior works on shared autonomy [1], [2]. Within today's shared autonomy approaches, often the robot assumes it knows the right way to perform each task *a priori*. For instance, if the human wants to reach a cup, the robot assists the human move in a straight line towards that goal. But not all humans complete tasks in the same way. Accordingly, here we *learn* how the user likes to perform their tasks by replicating their personalized demonstrations.

### C. Return: Knowing What We Don't Know

If the human repeats a task that the robot has seen many times before (e.g., opening the fridge), we can rely on our model to assist the human. But what happens if the human tries to perform a new or rarely seen task? Here we *do not trust* the robot's assistive actions since this task is out of the robot's training distribution. In general, deciding where to arbitrate control requires a trade-off: we want the robot to take as many autonomous actions as possible (reducing the human's burden), but we don't want the assistive robot to over-commit to erroneous behavior, and prevent the human from doing what they really intended.

To solve this problem we take inspiration from recent work on safe imitation learning [25]–[27]. Our goal is to determine when the robot should trust the collective output of Equations (3) and (4). Intuitively, if the human's behavior  $\tau^i$  is unlike any seen behavior  $\tau \in \mathcal{D}$  we should return control to the human. We therefore train a discriminator  $\mathcal{C}$  that distinguishes *seen* behavior from *unseen* behavior. Unseen behavior is cheap to produce: we can generate this behavior by applying noisy deformations to the observed interactions  $\tau \in \mathcal{D}$ . At run time, our discriminator outputs a scalar confidence over the human's current behavior, which we then utilize to arbitrate control between human and robot:

$$\beta \propto \mathcal{C}(\tau^i) \quad (6)$$

Recall that  $\beta$  from Equation (2) blends  $a_{\mathcal{R}}$  and  $a_{\mathcal{H}}$ . If  $\tau^i$  deviates from previously seen input patterns,  $\beta \rightarrow 0$ , and the human intervenes to complete this new task. By contrast, if the discriminator recognizes  $\tau^i$  as similar to previous experience,  $\beta \rightarrow 1$  and the robot provides assistance.

**Continual Learning.** During an interaction the robot applies Equations (3), (4), and (6) to assist the human. But what about *between* interactions? Let's say we train our encoder, decoder, and discriminator after the human has collaborated

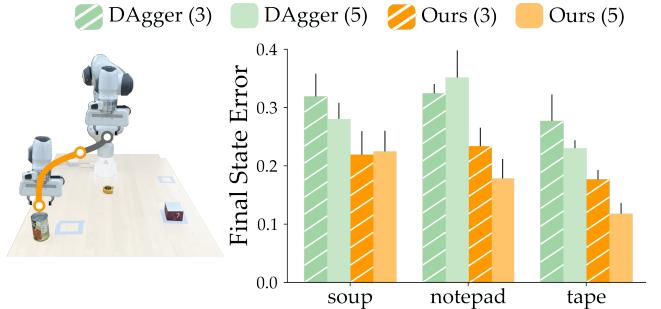


Fig. 3. Comparison to **DAgger** [20], an imitation learning baseline that does not use latent embeddings. A simulated user controls the robot for the first 0.5 s of the interaction: the robot must recognize the human's task and complete the rest of the reaching motion autonomously. We measure the final state error for each goal after training with 3 or 5 repeated interactions.

with the robot for a single week. Over the next week the human uses their robot, they will inevitably perform new tasks. An intelligent assistive robot should also learn these tasks and *continuously adapt* to the human. At the end of interaction  $i$ , we therefore add  $\tau^i$  to dataset  $\mathcal{D}$ . We then *retrain* our models between interactions, updating  $\mathcal{E}$ ,  $\pi_{\mathcal{R}}$ , and  $\mathcal{C}$ . Intermittent retraining enables the robot to continually learn and refine tasks over repeated interaction.

## V. SIMULATIONS

We have proposed an algorithm that learns to assist users over repeated interaction. Our algorithm breaks down into three parts: recognizing the task, replicating prior demonstrations, and returning control when unsure. But how does our approach compare to other imitation learning baselines that also learn from repeated interaction? For instance, what if we remove our encoder, and simply train a behavior cloning agent conditioned on the current interaction  $\tau^i$ ? In this section we perform an *ablation study* in which we determine how recognizing, replicating, and returning all contribute to overall robot success. We conduct these experiments on a real robot arm with simulated human operators.

**Experimental Setup.** We implement our approach (**Ours**) on a 7-DoF FrankaEmika robot arm. A simulated user controls this robot to both reach *discrete goals* (e.g., grasping a can) and perform *continuous skills* (e.g., opening a drawer). This simulated user is not perfect: the human selects commanded actions  $a_{\mathcal{H}}$  with varying levels of Gaussian white noise.

### A. Do We Need Recognition?

In our first experiment with simulated humans we explore whether we need a separate module for task recognition. Recall that in Section IV-A we introduced an encoder which embeds the current interaction  $\tau^i$  to latent task  $z \in \mathcal{Z}$ . Within Section IV-B, we then decoded  $z$  into an assistive robot action using  $\pi_{\mathcal{R}}(s, z)$ . But it's natural to ask if we need this encoder in the first place: in other words, can we obtain similar performance *without* embedding to latent space  $\mathcal{Z}$ ? Here we consider an alternative to Equation (4) where the robot *directly* learns  $\pi_{\mathcal{R}}(s, \tau^i)$  using behavior cloning. Specifically, we baseline against Dataset Aggregation (**DAgger**) [20].

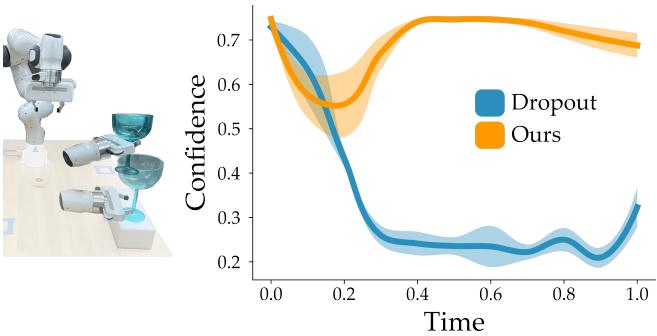


Fig. 4. Comparison to **DropoutDAgger** [26], a safe imitation learning baseline where the robot’s learned policy  $\pi_R$  evaluates its own confidence. Simulated users attempt to lift a glass. Although the robot has seen this continuous skill 5 times before, with **Dropout** the robot is overly sensitive to minor deviations from previous interactions and rarely provides assistance.

The environment here consists of three potential goals: a can of soup, a notepad, or a tape measure. The human teleoperates the robot along 3 or 5 demonstrations to reach each goal. We train **Ours** approach and the **DAgger** baseline from these repeated interactions. At test time, the human guides the robot for the first 0.5 s of the task: based on this input, the robot must recognize which task the human is trying to perform and automate the rest of the reaching motion. We plot the resulting error between the human’s goal and the robot’s final state in Figure 3. As expected, the robot is better able to automate these reaching tasks after more interactions (i.e., compare 3 to 5). But the robot also better identifies the human’s task when using our encoder: as compared to **DAgger**, **Ours** more accurately reaches the human’s goal given the same amount of training data and human input. We conclude that incorporating an encoder for task recognition *does improve* the robot’s assistance.

### B. Do We Need Help Returning?

In our second experiment we consider the opposite end of our pipeline: determining when the robot should provide assistance. Our intuition from Section IV-C is that — when we see an interaction similar to prior interactions — we should trust the output of our learned model. But an alternative approach is to rely on the confidence of our learned model *itself*. Here we turn to prior work on safe imitation learning where the robot samples its encoder-decoder multiple times at the current state, and assesses the similarity of the resulting actions  $a_R$ . If all of these actions are almost identical, the learned model is *confident* it knows what to do; conversely, if the model outputs have high variance, the robot is *unsure*. Specifically, we compare DropoutDAgger [26] (**Dropout**) to **Ours**. Recall that in **Ours** the robot trains a *separate* discriminator to detect whether or not  $\tau^i$  is similar to previous interactions as opposed to relying on the autoencoder model to measure its own confidence.

We compare **Dropout** and **Ours** in a continuous manipulation task where the simulated human is trying to reach and lift a glass. During test time, the human and robot share control throughout the entire interaction using Equation (2). The robot has seen the human perform this task five times before,

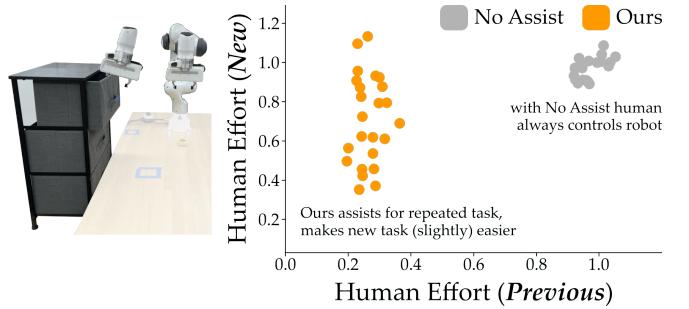


Fig. 5. Simulated users alternate between a previously seen task (opening the drawer) and a new task (reaching a cup). With **No Assist**, both new and previous tasks take about the same amount of human effort. Our approach learns to partially automate the previously seen task without resisting or overriding humans when they try to complete the new task.

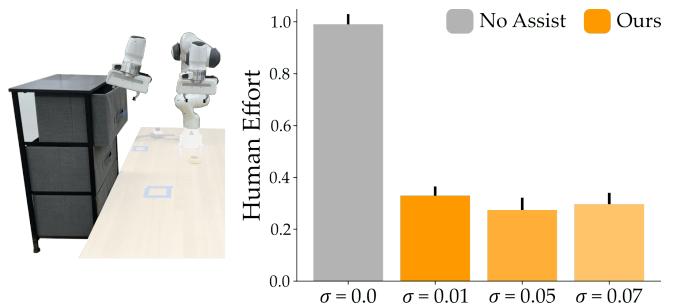


Fig. 6. Simulated humans with increasingly noisy behavior. Here the human is always attempting to open the drawer (the repeated task from Figure 5). We find that our robot correctly recognizes and assists for this task despite noisy and imperfect human teleoperation inputs  $a_H \sim \mathcal{N}(a_H^*, \sigma)$ .

and therefore should be confident in providing assistance. We visualize the robot’s actual confidence  $\beta$  in Figure 4. Interestingly, we find that **Dropout** is overly sensitive to minor deviations from previous interactions, and incorrectly returns control to the human even when the robot can still provide useful help. **Ours** remains confident throughout this known task, suggesting that our separate discriminator better arbitrates control than the learned policy itself.

### C. New Tasks and Noisy Humans

So far our approach recognizes and assists the human in repeated tasks. But what about new tasks the robot has never seen before? Of course, the robot cannot assist on these tasks without any prior experience: but we want to make sure that the robot does not *resist* the human, or *force* them along a previously seen trajectory. This problem becomes particularly challenging when the human is noisy, and the robot must determine whether the imperfect human is trying to repeat a prior task or complete a new task.

We consider an environment that consists of a previously seen skill (opening a drawer) and an unseen goal (reaching a can). **Ours** is trained with five repetitions of the drawer skill: we compare our approach to a **No Assist** baseline, where the human teleoperates the robot without any shared autonomy. At test time, the simulated human alternates between the new and previous tasks with varying levels of Gaussian white noise ( $\sigma$ ). Our results are shown in Figures 5 and 6. Here *Human Effort* is the amount of time the human teleoperates the

robot normalized by the average time required to complete the task. As expected, **Ours** makes it easier for the human to repeatedly open the drawer — but on the new task, **Ours** also correctly returns control to the human. Performing the new task takes no more effort than the **No Assist** baseline; indeed, it often requires less human effort, since the robot leverages what it knows from the drawer skill to automate the start of the reaching motion. This result is robust to human noise. For the previous task, **Ours** consistently reduces human effort across multiple noise levels  $\sigma$ .

## VI. USER STUDY

Our target application is assistive robotic arms: we want to enable these arms to share autonomy with humans on everyday tasks. Motivated by this application, we conducted an in-person user study in which participants teleoperated a 7-DoF robot arm around a simplified household environment (see Figures 1 and 7). We divided the study into *three parts* to explore known and new tasks as well as discrete goals and continuous skills. Participants started by reaching for known goals, then taught the robot new skills, and finally returned to the original tasks. Unlike in our simulations — where we compared our approach to imitation learning alternatives — here we focus on current methods for shared autonomy.

**Independent Variables and Experimental Setup.** Our user study is divided into the three sections described below. Each participant completed every section.

In the first part of the user study participants teleoperated the robot to reach for two discrete goals placed on the table. These potential goals were known *a priori*, and the robot had prior experience reaching for them. Here we compare **Ours** proposed approach to an existing shared autonomy baseline (**Bayes**) [1]. **Bayes** exploits the prior knowledge about the goal set to infer which goal the human is trying to reach and assist them during the task. Similarly, **Ours** learns from the previous, offline demonstrations to recognize the user’s goal and assist the human during the current task.

The shared autonomy baseline is the gold standard when the human wants to complete a task that the robot already knows — but what happens during new tasks? In the second part of the study, participants iteratively performed two new tasks a total of 9 times each. One task was a discrete goal (reaching a cup), while the other was a continuous skill (opening a drawer). Here we compare **Ours** to the **No Assist** baseline, where the human completes these tasks without any robot assistance. We retrained our approach every three trials for both tasks: intuitively, **Ours** should increasingly assist the user as it gets more familiar with these new tasks.

One concern with our approach is that it will specialize in just one or two recent tasks without remembering older tasks. In the last part of the user study, participants take the final learned model from both new tasks and use it to revisit the original reaching tasks. Here we compare three conditions: **No Assist**, where the human acts alone, **Ours (task)**, the robot’s learned assistance with just the user’s data from that specific task, and **Ours (all)**, our approach trained on the user’s full dataset of interactions.

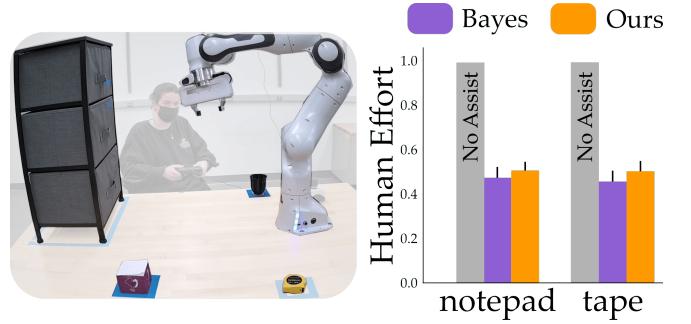


Fig. 7. Experimental setup and objective results from the first part of the user study. Participants teleoperated the robot arm to reach for two goals that were known *a priori*. We find that our approach (**Ours**) learns to offer assistance on par with a shared autonomy baseline (**Bayes**) [1]. Note that we cannot leverage **Bayes** when the human wants to perform new, unexpected tasks (e.g., reaching the cup or opening the drawer), as shown in Figure 10. We include the human effort under **No Assist** as a baseline.

**Dependent Measures – Objective.** Across all three parts of the user study we measured *Human Effort*. Human effort captures the amount of time the human teleoperated the robot during the task. We normalize this by the average time taken to complete the task. Higher values of human effort indicate that the human spent more time guiding the robot’s motion.

**Dependent Measures – Subjective.** We administered a 7-point Likert scale survey after users completed the study (see Figure 9). Questions were organized along five scales: how confident users were that the robot *Recognized* their objective, how helpful the robot’s behavior was (*Replicate*), how trustworthy users thought the robot was (*Return*), whether the robot improved after successive demonstrations (*Improve*), and if they would collaborate with the robot again (*Prefer*).

**Participants and Procedure.** A total of 10 members of the Virginia Tech community participated in our study (3 female, 1 non-binary, average age  $22 \pm 7$  years). All participants provided informed written consent prior to the experiment.

**Hypotheses.** We tested three main hypotheses:

**H1.** *In cases where the robot has prior knowledge about the human’s potential goals, our learning approach will perform similarly to a shared autonomy baseline*

**H2.** *In cases where the human repeatedly performs new tasks, our approach will learn to provide meaningful assistance from scratch*

**H3.** *Our robot will remember how to assist users on previously seen tasks even after learning new ones*

**Results.** The results from each part of our user study are visualized in Figure 7 and Figure 8.

In the first part of the user study participants completed a reaching task with both **Bayes** and **Ours**. We found that the differences in human effort were not statistically significant: users could reach for known, discrete goals just as easily with **Ours** as they could with the shared autonomy baseline. Note that in this task the robot had to recognize which goal the human was reaching for (i.e., either the notepad or tape) and then assist the user while reaching for that target. We emphasize that we cannot apply **Bayes** when the participant

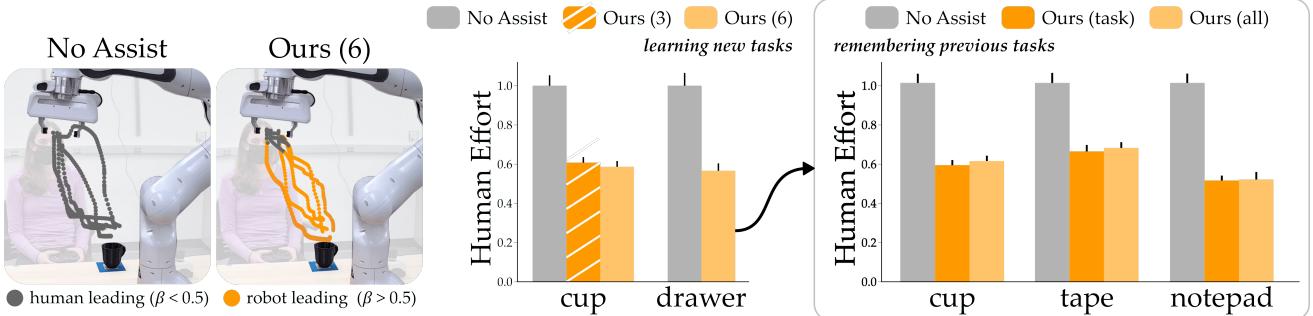


Fig. 8. Objective results from the second and third parts of our user study. (Left) The human teleoperates the robot to reach for a goal it did not know about beforehand. The first few times they interact, the user must lead the robot throughout the entire task (**No Assist**). After training **Ours** on six repeated interactions, the robot recognizes the human’s intent and takes the lead. (Center) Across 3 – 6 repeated interactions the robot learns to provide assistance for a new goal (cup) and skill (drawer). This assistance reduces the human’s effort as compared to completing the task alone. (Right) We take our resulting model trained on all user demonstrations and revisit the original tasks. **Ours (all)** offers similar assistance to **Ours (task)**, a version of our approach trained only with the user’s task-specific data. These results suggest the robot can learn to assist for new tasks without forgetting older ones.

is trying to perform a *new* task (such as opening the drawer), since this baseline requires prior knowledge.

In the second part of our study, the results from Figure 8 demonstrate that **Ours** got better at providing assistance for new tasks over repeated interactions. Humans spent the most effort reaching for the cup or opening the drawer by themselves: but after training our approach on 6 demonstrations, the robot was able to recognize the drawer skill and replicate meaningful assistance. One user commented that “*by the end I didn’t provide any assistance and the robot continued to move in the correct direction.*” We emphasize that — throughout our entire user study — the robot was *never told* what task the participant wanted to do. Instead, the robot had to recognize the participant’s current task based on that user’s joystick inputs.

As a final step we trained our encoder, decoder, and discriminator from the user’s demonstrations across all tasks. We found that this general **Ours (all)** performed similarly to the more specialized **Ours (task)**. For instance, when revisiting the original notepad and tape tasks from the start of the study, **Ours (all)** provided comparable assistance to **Ours (task)**.

Taken together, these results support **H1**, **H2**, and **H3**. Our approach leveraged repeated interactions to learn to share autonomy across new and old tasks with discrete goals and continuous skills. Participants generally perceived the robot’s assistance as helpful. Looking at the subjective results from Figure 9, users thought the robot correctly recognized their intent, made the task easier to complete, and got better at providing assistance over the course of the study.

## VII. CONCLUSION

Today’s shared autonomy algorithms often rely on prior knowledge: e.g., the robot needs to know all the human’s potential tasks *a priori*. Here we remove this assumption by learning to assist humans from scratch. Our approach exploits the *repeated* nature of everyday human-robot interaction to recognize the human’s current task, replicate similar past interactions, and return control when unsure. Our simulations and user study demonstrate that this approach extends to both

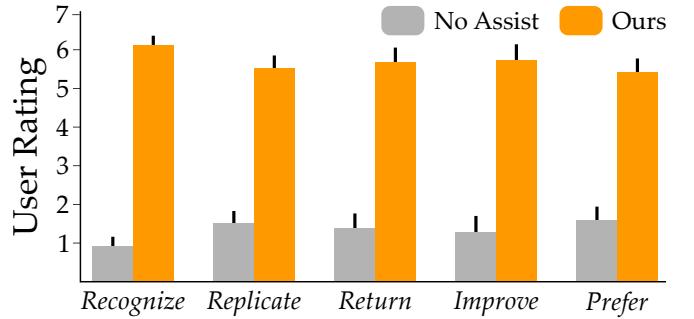


Fig. 9. Subjective results from our in-person user study. Higher ratings indicate user agreement. Overall, participants thought our method provided useful assistance, and they preferred this assistance to trying to complete the tasks by themselves.

discrete goals and continuous skills, and learns online from a practical number of interactions (< 10).

**Limitations.** So far we have focused on how assistive robot arms can adapt to their human users. But as the robot arm gets better at sharing autonomy, the human will also *co-adapt* and modify their own teleoperation strategy. For example, once the human is confident the robot recognizes their current task, the user may stop providing joystick inputs and rely on the robot entirely. A limitation of our approach is that it does not explicitly account for this co-adaptation.

## VIII. APPENDIX

To highlight one shortcoming of state-of-the-art shared autonomy approaches and explain why **Bayes** is not a baseline in the second and third parts of our user study, we illustrate a failure case in Figure 10. Here the user wants to open the drawer, but the robot only has prior knowledge about the notepad and the tape. Recall that under **Bayes** the robot infers which discrete goal the human is trying to reach and then assists towards that goal [1], [2]. But in this scenario the robot *does not know beforehand* that the human may want to open the drawer. As a result, **Bayes** misinterprets the user’s inputs and gradually becomes convinced that the human’s target is actually the notepad *next* to the drawer. This ends in a deadlock: the human teleoperates the robot towards the drawer, while the robot resists and refuses to return control.

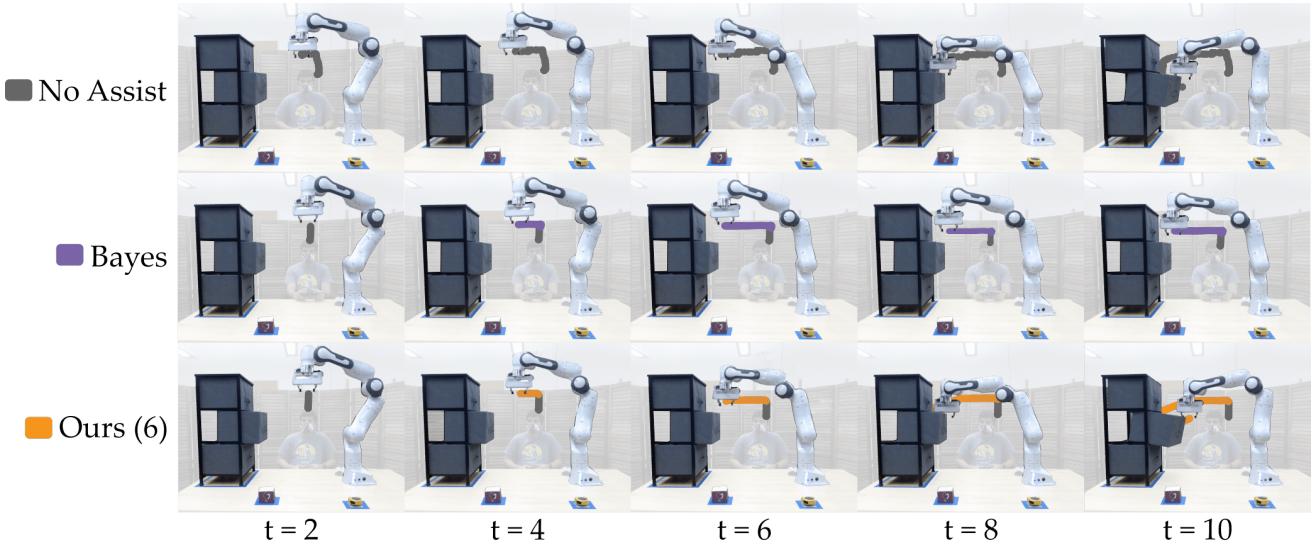


Fig. 10. Representative failure case for an existing shared autonomy approach that relies on prior knowledge (**Bayes** [1]). The user attempts to complete the drawer task with no robot assistance (**No Assist**), with **Bayes**, and with **Ours** proposed method. **Bayes** has prior knowledge only about the notepad and the tape, while **Ours** has been trained on six repeated interactions for the drawer task. The user is able to successfully open the drawer by themselves (top) and with our method (bottom). With **Ours**, we see that the user is initially leading the robot towards the drawer, but once the robot recognizes this task, it takes charge and offers appropriate assistance. By contrast, **Bayes** (middle) mistakes the initial trajectory as towards the notepad, and tries leading the robot towards this known goal. Since both the drawer and the notepad are in front of the robot, the robot is initially able to move in the correct direction. However, after the user's inputs diverge from the notepad and go towards the drawer, the robot gets stuck due to conflicting commands.

Note that the trajectories for **Ours** and **No Assist** are similar — the main difference is that in **Ours** the robot takes the lead and automates the continuous skill.

## REFERENCES

- [1] A. D. Dragan and S. S. Srinivasa, “A policy-blending formalism for shared control,” *The International Journal of Robotics Research*, vol. 32, no. 7, pp. 790–805, 2013.
- [2] S. Jain and B. Argall, “Probabilistic human intent recognition for shared autonomy in assistive robotics,” *ACM Transactions on Human-Robot Interaction*, vol. 9, no. 1, pp. 1–23, 2019.
- [3] S. Javdani, H. Admoni, S. Pellegrinelli, S. S. Srinivasa, and J. A. Bagnell, “Shared autonomy via hindsight optimization for teleoperation and teaming,” *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 717–742, 2018.
- [4] C. Brooks and D. Szafir, “Balanced information gathering and goal-oriented actions in shared autonomy,” in *ACM/IEEE International Conference on Human-Robot Interaction*, 2019, pp. 85–94.
- [5] D. Gopinath, S. Jain, and B. D. Argall, “Human-in-the-loop optimization of shared autonomy in assistive robotics,” *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 247–254, 2016.
- [6] B. A. Newman, R. M. Aronson, S. S. Srinivasa, K. Kitani, and H. Admoni, “Harmonic: A multimodal dataset of assistive human-robot collaboration,” *arXiv preprint arXiv:1807.11154*, 2018.
- [7] S. Nikolaidis, Y. X. Zhu, D. Hsu, and S. Srinivasa, “Human-robot mutual adaptation in shared autonomy,” in *ACM/IEEE International Conference on Human-Robot Interaction*, 2017, pp. 294–302.
- [8] D. M. Taylor, *Americans With Disabilities: 2014*. US Census Bureau, 2018.
- [9] B. D. Argall, “Autonomy in rehabilitation robotics: An intersection,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 441–463, 2018.
- [10] R. Feng, Y. Kim, G. Lee, E. K. Gordon, M. Schmittie, S. Kumar, T. Bhattacharjee, and S. S. Srinivasa, “Robot-assisted feeding: Generalizing skewering strategies across food items on a realistic plate,” in *International Symposium on Robotics Research*, 2019.
- [11] H. J. Jeon, D. P. Losey, and D. Sadigh, “Shared autonomy with learned latent actions,” in *Robotics: Science and Systems*, 2020.
- [12] D. Park, Y. Hoshi, H. P. Mahajan, H. K. Kim, Z. Erickson, W. A. Rogers, and C. C. Kemp, “Active robot-assisted feeding with a general-purpose mobile manipulator: Design, evaluation, and lessons learned,” *Robotics and Autonomous Systems*, vol. 124, p. 103344, 2020.
- [13] Z. Erickson, V. Gangaram, A. Kapusta, C. K. Liu, and C. C. Kemp, “Assistive gym: A physics simulation framework for assistive robotics,” in *IEEE International Conference on Robotics and Automation*, 2020, pp. 10169–10176.
- [14] Y. S. Choi, T. Deyle, T. Chen, J. D. Glass, and C. C. Kemp, “A list of household objects for robotic retrieval prioritized by people with ALS,” in *IEEE International Conference on Rehabilitation Robotics*, 2009, pp. 510–517.
- [15] J. Bragg and E. Brunskill, “Fake it till you make it: Learning-compatible performance support,” in *Uncertainty in Artificial Intelligence*, 2020, pp. 915–924.
- [16] A. Broad, I. Abraham, T. Murphey, and B. Argall, “Data-driven Koopman operators for model-based shared control of human-machine systems,” *The International Journal of Robotics Research*, vol. 39, no. 9, pp. 1178–1195, 2020.
- [17] S. Reddy, A. D. Dragan, and S. Levine, “Shared autonomy via deep reinforcement learning,” in *Robotics: Science and Systems*, 2018.
- [18] C. Schaff and M. R. Walter, “Residual policy learning for shared autonomy,” in *Robotics: Science and Systems*, 2020.
- [19] A. Shih, A. Sawhney, J. Kondic, S. Ermon, and D. Sadigh, “On the critical role of conventions in adaptive human-AI collaboration,” in *International Conference on Learning Representations*, 2020.
- [20] S. Ross, G. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 627–635.
- [21] A. Xie, D. P. Losey, R. Tolsma, C. Finn, and D. Sadigh, “Learning latent representations to influence multi-agent interaction,” in *Conference on Robot Learning*, 2020.
- [22] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, “Learning latent plans from play,” in *Conference on Robot Learning*, 2020, pp. 1113–1132.
- [23] R. Jonschkowski and O. Brock, “Learning state representations with robotic priors,” *Autonomous Robots*, vol. 39, no. 3, pp. 407–428, 2015.
- [24] K. Pertsch, Y. Lee, and J. J. Lim, “Accelerating reinforcement learning with learned skill priors,” in *Conference on Robot Learning*, 2020.
- [25] J. Zhang and K. Cho, “Query-efficient imitation learning for end-to-end autonomous driving,” in *AAAI*, 2017.
- [26] K. Menda, K. Driggs-Campbell, and M. J. Kochenderfer, “Dropoutdagger: A bayesian approach to safe imitation learning,” *arXiv preprint arXiv:1709.06166*, 2017.
- [27] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, “HG-Dagger: Interactive imitation learning with human experts,” in *IEEE International Conference on Robotics and Automation*, 2019, pp. 8077–8083.