SARI: Shared Autonomy across Repeated Interaction

ANANTH JONNAVITTULA, Virginia Tech, Blacksburg, USA
SHAUNAK A. MEHTA, Virginia Tech, Blacksburg, USA
DYLAN P. LOSEY, Virginia Tech, Blacksburg, USA

Assistive robot arms try to 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 intervenes to automate the task. But how does the robot know these tasks in the first place? State-of-the-art approaches to shared autonomy often rely on prior knowledge. For instance, the robot may need to know the human's potential goals beforehand. During long-term interaction these methods will inevitably break down — sooner or later the human will attempt to perform a task that the robot does not expect. Accordingly, in this paper we formulate an alternate approach to shared autonomy that learns assistance from scratch. Our insight is that operators *repeat* important tasks on a daily basis (e.g., opening the fridge, making coffee). Instead of relying on prior knowledge, we therefore take advantage of these repeated interactions to learn assistive policies. We introduce SARI, an algorithm that *recognizes* the human's task, *replicates* similar demonstrations, and *returns* control when unsure. We then combine learning with control to demonstrate that the error of our approach is uniformly ultimately bounded. We perform simulations to support this error bound, compare our approach to imitation learning baselines, and explore its capacity to assist for an increasing number of tasks. Finally, we conduct three user studies with industry-standard methods and shared autonomy baselines, including a pilot test with a disabled user. Our results indicate that learning shared autonomy across repeated interactions matches existing approaches for known tasks and outperforms baselines on new tasks. See videos of our user studies here: https://youtu.be/3vE4omSvLvc

CCS Concepts: • **Human-centered computing** → **Collaborative and social computing**; **Accessibility**.

Additional Key Words and Phrases: Human-Robot Interaction, Shared Autonomy, Imitation Learning

1 INTRODUCTION

Imagine teleoperating a wheelchair-mounted robot arm to open your refrigerator door (see Figure 1). This robot has never interacted with your fridge before: accordingly, for the first few times you open the fridge, you must carefully guide the robot throughout the entire process of reaching, grabbing, and pulling the door. But after you've interacted with this robot for several weeks — and opened your fridge *many* times — an intelligent robot should learn to *assist* you. The next time you start teleoperating the arm towards your fridge, the robot should recognize what you want and partially automate the process of opening the door.

The robot's assistance in this working example is an instance of *shared autonomy*. Shared autonomy for assistive robot arms blends the user's inputs with autonomous actions so that both the human and the robot contribute to the robot's overall motion. When surveyed, disabled adults who operate assistive robots preferred shared autonomy over either fully autonomous or fully human-guided systems [7, 23]. In practice, however, today's shared autonomy approaches rely on *prior information* about the human's tasks. Methods such as [3, 5, 10, 15, 28, 31, 33, 49, 50] require a pre-defined list of goals the human might want to reach: the robot

Authors' addresses: Ananth Jonnavittula, ananth@vt.edu, Virginia Tech, Department of Mechanical Engineering, Blacksburg, VA, USA; Shaunak A. Mehta, mehtashaunak@vt.edu, Virginia Tech, Department of Mechanical Engineering, Blacksburg, VA, USA; Dylan P. Losey, losey@vt.edu, Virginia Tech, Department of Mechanical Engineering, Blacksburg, VA, 24060, USA.
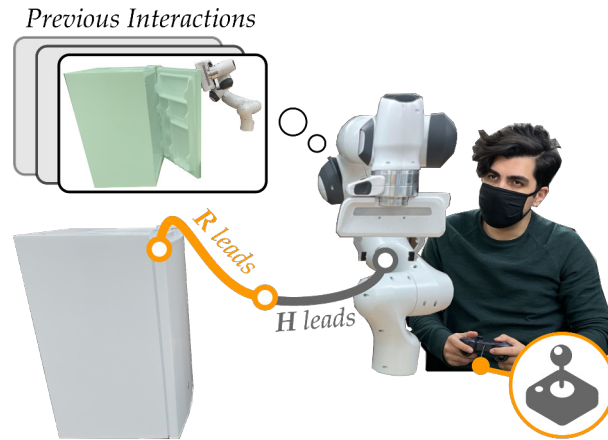
Fig. 1. User teleoperating an assistive robot arm to open their fridge door. The robot does not have any prior knowledge about this task; however, the human and robot have completed similar tasks many times before. Instead of making the human guide the robot through every step of this task, we hypothesize that robot arms can learn to assist humans and share autonomy by exploiting the *repeated* nature of everyday tasks.

infers the human's most likely goal from these discrete options, and autonomously moves towards that goal. Other approaches need demonstrations [32, 39, 45], feedback [55], or constraints [8, 9, 16, 57] that specify which tasks the human might want to perform: the robot then maps the human's inputs to task relevant motions, and overrides or corrects inputs that do not align with the robot's anticipated tasks.

These existing approaches work well when the user wants to perform a task that the robot knows *a priori*. But what happens when the human inevitably wants to complete some new or unexpected task? Going back to our working example, the robot has no prior information about opening the refrigerator. When the user teleoperates the robot towards the fridge door, today's assistive arms assume that the human has made a mistake: instead of helping for the fridge task, shared autonomy guides the robot towards one of its known tasks. Even worse, the robot remains confused — and provides incorrect assistance — no matter how many times the operator tries to repeat the process of opening the fridge [64].

For assistive arms to be practical across long-term interaction, these robots must be capable of *learning* a spectrum of new tasks. This would be extremely challenging if every task was a unique one-off that the robot had never seen before. But our insight is that, over the many weeks, months, and years a human operator works with their assistive robot:

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

We emphasize that these repetitions are never exactly the same. Each time the assistive robot opens the refrigerator it may have a different start position or follow a different trajectory. Hence, we cannot simply record and playback the motions that the human has shown — instead, we need to generalize assistance across similar tasks. Applying our insight enables assistive robot arms to learn to share autonomy by exploiting the *repeated interactions* inherent within assistive applications. Here the robot remembers how the user controlled the arm to open the fridge in the past, recognizes that the user is providing similar inputs during the current interaction, and assists by autonomously mimicking the behavior that user previously demonstrated. Across repeated human-robot interactions these assistive arms should learn to share autonomy for tasks that include not only *discrete goals* (e.g., reaching a cup) but also *continuous skills* (e.g., opening a door).

In this paper we propose, analyze, and evaluate our algorithm **SARI**: **S**hared **A**utonomy across **R**epeated **I**nteraction. Overall, we make the following contributions[1]:

Capturing Latent Intent. We formalize the problem of sharing autonomy across repeated human-robot interaction. During each interaction the human has in mind some desired task: we introduce an end-to-end imitation learning algorithm that learns to recognize the human's current intent and provide autonomous assistance without any pre-defined tasks or prior information.

Returning Control when Uncertain. Our approach should assist during previously seen tasks without overriding the human whenever they try to perform a new task. We introduce a discriminator to measure the confidence of our learned assistance so that the robot automatically returns control to the human when it is unsure about what the human really wants.

Analyzing Stability. We combine learning with control theory to bound the error between the robot's actual state and the human's desired state. We theoretically demonstrate that — even in the worst case — our approach is uniformly ultimately bounded with respect to some radius about the human's goal. We derive this radius as a function of the variance in the human's input commands and the similarity between previously learned task(s) and the human's current task.

Comparing to Baselines. We perform experiments with simulated human operators and real robot arms to demonstrate how each component of our algorithm contributes to its performance. Within these controlled experiments we compare our approach to state-of-the-art imitation learning baselines, and test our method's capacity to learn assistance for an increasing number of tasks.

Conducting User Studies. We assess our resulting algorithm in three separate user studies. The first two studies were performed with non-disabled users, and the final study is a pilot with a disabled user who regularly operates wheelchair-mounted robot arms. In these studies we compare our approach to shared autonomy baselines for tasks that involve discrete goals and continuous skills. Viewed together, our results suggest that SARI enables the robot to learn to assist for known and new tasks, leading to higher objective and subjective performance.

2 RELATED WORK

Our approach learns to share autonomy across repeated human-robot interaction without predefined tasks or offline demonstrations. Our work is motivated by assistive applications where disabled users teleoperate robot arms on a daily basis. Instead of forcing the user to repeatedly guide the robot throughout every step of the motion, we learn to recognize the human's task, imitate their previous interactions, and arbitrate control between the human and robot.

Application – Assistive Robot Arms. Over 13% of American adults living with physical disabilities have difficulty with at least one activity of daily living (ADL) [62]. Assistive robots — such as wheelchair-mounted robot arms [1, 4, 11] — have the potential to help users perform these everyday tasks without relying on caregivers. Recent work on assistive arms has focused on automating ADLs such as eating dinner [6, 18, 32, 52], getting dressed [13, 17, 53], and manipulating household objects [12]. For sufficiently simple daily tasks the disabled adult may not require any assistance from the robot. However, our research takes inspiration from the fact that users need assistance when performing *complex* tasks that are *repeated* on a daily basis (e.g., opening a door). It is mentally tedious and physically burdensome for the human to precisely teleoperate the robot throughout each

---

[1]Parts of this work have been previously published in the IEEE/RSJ International Conference on Intelligent Robots and Systems [34]. Novel contributions include the combination of learning and control (Section 5 and Appendix), experiments on SARI capacity in Section 6.4, and two user studies in Section 7. This added material provides formal guarantees about the performance of our approach, offers a new understanding of how our approach works in practice, and compares our method's performance to state-of-the-art baselines with non-disabled users and one disabled adult.
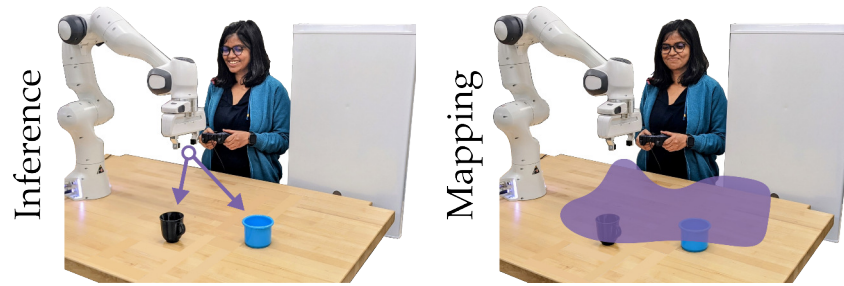
Fig. 2. We separate prior work on shared autonomy for assistive robot arms into two groups. (Left) Some methods are given a discrete set of possible goals, and infer the human's goal from these discrete options. (Right) Other methods learn to map the human's joystick inputs to constrained, task-relevant motions. Although both shared autonomy algorithms help this human reach for the cups, neither can assist the human for a new, unexpected task (like opening the fridge).

step of these everyday tasks [26]. When surveyed, adults who operate assistive arms indicated that they prefer to *share autonomy* with robots [7, 23].

Shared Autonomy. In shared autonomy both the human and robot arbitrate control over the robot's motion. We separate related research on shared autonomy into two classes of algorithms (see Figure 2). First, there are approaches which *infer* the human's goal and then partially automate the arm's motion towards that goal. Second, there are methods which *map* the human's inputs to constrained and task-relevant actions.

Within inference works the robot is given a discrete set of possible goals the human may want to reach *a priori* [3, 5, 10, 15, 21, 28, 31, 35, 43, 49, 50]. Based on the human's teleoperation inputs so far the robot determines which goal(s) are likely, and takes assistive actions to move towards the inferred goal(s). For example, Javdani *et al.* formulate this as a partially observable Markov decision process where the human's goal is the latent state and the human's teleoperation inputs are observations about that latent goal [31]. We emphasize that this class of algorithms *requires prior knowledge* about the human's tasks (i.e., the location of all the goals the human may want to reach). In the long-term these priors will inevitably fall short: sooner or later the human will reach for a goal that the robot did not expect, and the robot will be unable to provide assistance.

Other works on shared autonomy map the human's joystick inputs to motions that are relevant for the current task [8, 9, 16, 32, 39, 45, 55, 57]. When the human provides a suboptimal input (e.g., an input that moves the robot away from its goal) the robot overrides and corrects this human's action. For instance, Reddy *et al.* [55] learn a reward function from the human and then constrain the robot to take actions with high long-term rewards, while Broad *et al.* prevent the human from taking actions that deviate from the robot's expectations. Similarly, Losey *et al.* [39] map the human's joystick inputs to latent, task-relevant robot actions that are learned from offline task demonstrations. Overall, this class of algorithms makes sense when the user wants to perform task(s) that the robot has learned to assist. But if the human attempts to perform a new or unexpected task, then these *constraints become counter-productive*: the robot mistakenly overrides the human and may force them to perform the wrong task.

Beyond these two classes of algorithms we highlight recent shared autonomy work that *learns new tasks* during interaction [54, 64]. Here the robot starts with a discrete set of options and tries to infer the human's current task. If the human's inputs do not match any of these known tasks, shared autonomy stops: the robot returns full control to the human and the human demonstrates their new task to the robot. This task is then added to the discrete library of options and shared autonomy restarts at the next interaction. Like [64] and [54] our approach continually learns to assist for new tasks. However, we do not *separate* our approach into distinct phases for

sharing autonomy or learning tasks. Instead, our robot learns to assist the human each time the user interacts with the robot, regardless of whether they are completing a previously seen task or performing a new skill.

Interactive Imitation Learning. Our technical approach builds on interactive and safe imitation learning [51, 56]. Specifically, we draw connections between shared autonomy and imitation learning techniques where the robot and human periodically switch control. Within these settings the robot attempts to perform the task autonomously; but at times where the human notices that the robot is making a mistake [30, 37, 42, 60], or in states where the robot is uncertain about what it should do [27, 44, 46, 63], the human takes over and guides the robot. Across repeated interactions the robot adds these human corrections to its training data and learns to imitate the human's behavior. Here we leverage a similar approach to learn to share autonomy. More specifically, our technical approach integrates prior work on both interactive imitation learning and representation learning [25, 36, 39, 41]. We develop representation learning to identify the space of possible tasks, and then incorporate imitation learning to mimic how the human previously performed these tasks and provide autonomous assistance.

3 FORMALIZING SHARED AUTONOMY ACROSS REPEATED INTERACTION

Let us return to our motivating example from Figure 1 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 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 is in state $s \in \mathbb{R}^d$ and takes action $a \in \mathbb{R}^d$. Within our experiments, s is the robot's joint position, a is the robot's joint velocity, and the robot has dynamics:

$$s^{t+1} = s^t + \Delta t \cdot a^t \tag{1}$$

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[2]. 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 [15, 28, 49]:

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

Here $\beta \in [0, 1]$ *arbitrates* control between human and robot. When $\beta \to 0$, the human always controls the robot, and when $\beta \to 1$, 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 [31], we accordingly write the *human's policy* as $\pi_{\mathcal{H}}(a_{\mathcal{H}} \mid s, z^i)$. This policy is the gold standard, because if we knew $\pi_{\mathcal{H}}$ 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. Our approach should personalize, and learn the policy that the current user prefers.

---

[2]Although we use joysticks for explanation here, our algorithmic framework is not tied to any teleoperation interface. Users could alternatively control the robot with sip-and-puff devices [28], body-machine interfaces [29], or brain-computer interfaces [48]. All of these interfaces output a commanded action $a_{\mathcal{H}}$. To show our method's ability to generalize, we test using both a joystick and a web-based GUI in our user studies.
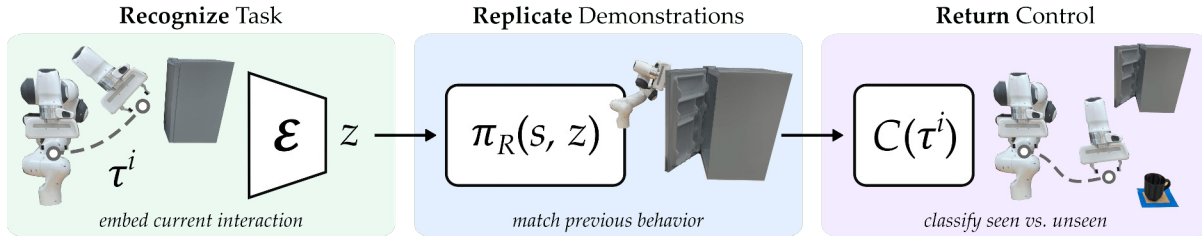
Fig. 3. Outline of SARI, our proposed algorithmic framework for learning to share autonomy across repeated interaction. (Left) The robot embeds the human's behavior τ^i during the current interaction to a distribution over latent tasks z. (Middle) The robot then chooses assistive actions $a_\mathcal{R}$ conditioned on its state s and latent task z. The assistive 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 τ^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.

Repeated Interaction. In practice the assistive robot cannot directly observe either z^i or $\pi_\mathcal{H}$. Instead, the robot observes the states that it visits and the commands that the human provides. Let $\tau = \{(s^1, a_\mathcal{H}^1), \ldots, (s^T, a_\mathcal{H}^T)\}$ be the entire *sequence* of robot states and human commands that the robot observed over the course of an interaction [3]. As the human and robot repeatedly collaborate and interact, the robot collects a *dataset* of these sequences: $\mathcal{D} = \{\tau^1, \tau^2, \ldots, \tau^{i-1}\}$. Notice that here we distinguish the current interaction τ^i. Because the robot only knows the states and human inputs up to the present time, for the current interaction $\tau^i = \{(s^1, a_\mathcal{H}^1), \ldots, (s^{t-1}, a_\mathcal{H}^{t-1})\}$.

Robot. In settings where an assistive robot arm repeatedly interacts with a human the robot has access to four pieces of information. The robot knows its state s, the start and end of current interaction ($\tau_{start}^i, \tau_{end}^i \in \tau^i$), the human's behavior during the current interaction τ^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 between the human and robot through β. We emphasize that under this formulation 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. In practice, the designer may choose to initialize the robot with some tasks before the assistive arm encounters the current user. Under our formulation this prior information takes the form of offline demonstrations: the designer could provide interactions $\mathcal{D}_{\text{offline}}$, so that $\mathcal{D} \rightarrow \mathcal{D} \cup \mathcal{D}_{\text{offline}}$. Moving forward, the robot must leverage the available data $(s, \tau^i, \mathcal{D})$ to learn to share autonomy with the current human operator.

4   LEARNING TO SHARE AUTONOMY ACROSS REPEATED INTERACTION (SARI)

Our proposed approach is guided by the intuition that — if the robot recognizes the human's behavior is similar to a previous interaction — the robot can assist the human by imitating that past 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. 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 introduce an algorithm to tackle these three challenges (see Figure 3). We refer to our method as **SARI**: **S**hared **A**utonomy across **R**epeated **I**nteractions.

---

[3]We emphasize that these interactions can be of any length, and that there is no maximum interaction length specified to the algorithm.

4.1 *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 behavior during the current interaction (i.e., their commanded actions at each robot state) is captured in τ^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 τ^i to recognize the underlying task z^i. More formally, we introduce an encoder:

$$z \sim \mathcal{E}(\,\cdot\mid \tau^i) \tag{3}$$

This encoder *embeds* the human's behavior into a probability distribution over the latent space $\mathcal{Z} \subseteq \mathbb{R}^d$. We will learn the encoder model from previous human interactions as described in the following subsection (4.2).

Our encoder $\mathcal{E}$ is analogous to *goal prediction* from prior works on shared autonomy [3, 5, 10, 15, 28, 31, 49, 50]. In these prior works the robot observes the human's current behavior τ^i, and then applies Bayesian inference to predict the human's goal z^i. Our encoder $\mathcal{E}(z \mid \tau^i)$ accomplishes the same thing: it outputs a distribution over tasks the human may want to complete. The difference is that — when using Bayesian inference — the robot needs to know the set of possible tasks *a priori*. When training the encoder we make no such assumption. Instead, the encoder *learns* a distribution over tasks using only past interactions $\mathcal{D}$. However, one practical concern here is that the robot could convince itself of its own prediction: i.e., because the robot is autonomously moving towards a goal, the robot might think that goal is increasingly likely [31]. We avoid this loop by purposely encoding the sequence τ^i. Since τ^i only includes the human's action $a_{\mathcal{H}}$ (and not the robot's assistance $a_{\mathcal{R}}$), the robot cannot infer a latent task from its own behavior.

4.2 *Replicate*: Imitating 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 robot policy (i.e., a decoder) that maps our task predictions into assistive robot actions:

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

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

We accomplish this by training the encoder and policy models using the dataset of previous 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 $\xi = \{(s^1, a_{\mathcal{H}}^1), \ldots (s^{k-1}, a_{\mathcal{H}}^{k-1})\}$ 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 policy to minimize the loss function:

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

across the dataset $\mathcal{D}$. In other words, we train the encoder from Equation (3) and policy from Equation (4) so that — given a snippet of the human's past behavior — we correctly predict the human's next action. Equation (5) encourages the robot to mimic the human, so that when the robot encounters a familiar task, the arm will take autonomous actions that match the commands which the human previously provided. As a reminder, here the robot is not simply saving and replaying the human's demonstrations: because each interaction is slightly different, the robot is learning a policy model $\pi_{\mathcal{R}}$ to generalize the human's demonstrations to nearby states.

We contrast our policy to *trajectory prediction* [15, 28] or *task constraints* [8, 9, 39, 55, 57] from previous research on shared autonomy. Under these approaches the robot assumes that it knows the correct way to perform each task; e.g., if the human wants to reach for a cup, the robot assumes that it should move in a straight line towards that goal. But we know that tasks are *personalized*, and different users will complete the same task in different ways. Instead of constraining robot assistance to a pre-specified task definition, we therefore learn to imitate how the current user performs each task and train $\pi_{\mathcal{R}}$ to match the current user's behaviors.

4.3 *Return*: Knowing What We Do Not 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 do not want the assistive robot to over-commit to incorrect autonomous actions and prevent the human from doing what they actually intended.

To solve this problem we take inspiration from recent work on interactive and safe imitation learning [27, 37, 46, 63]. Our objective is to determine when the robot should trust the collective output of Equations (3) and (4). Intuitively, if the human's behavior τ^i is unlike any seen behavior $\tau \in \mathcal{D}$ we should return control to the human. We therefore train a discriminator 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 $\tau \in \mathcal{D}$ [40]. 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:

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

In our experiments we implement this as $C(\tau^i) = C(s^t, a_{\mathcal{H}}^t)$, where $(s^t, a_{\mathcal{H}}^t)$ is the most recent state-action pair from τ^i, and the output of C is normalized using a softmax function [22] to obtain the arbitration parameter $\beta \in [0, 1]$. Recall that β from Equation (2) blends the robot and human actions $a_{\mathcal{R}}$ and $a_{\mathcal{H}}$. If τ^i deviates from previously seen input patterns, $\beta \to 0$, and the robot returns control to the human operator. By contrast, if the discriminator recognizes τ^i as similar to previous experience, $\beta \to 1$ and the robot arm partially automates its motion.

Continual Learning. During each interaction the robot applies Equations (3), (4), and (6) to assist the human. But what about *between* interactions, when the human is not providing any inputs to the robot? Imagine we train our encoder, policy, and discriminator after the human has collaborated with the robot for a few minutes. Over the next hour the human 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 τ^i to dataset $\mathcal{D}$. We then *retrain* SARI between interactions, updating $\mathcal{E}$, $\pi_{\mathcal{R}}$, and C. Intermittent retraining enables the robot to continually learn and refine tasks over long-term interaction.

5 ANALYZING STABILITY WITH SARI

The SARI algorithm we introduced in Section 4 learns to recognize tasks, replicate demonstrations, and return control. Here we apply stability theory to this learning approach. Specifically, we explore the performance of SARI when the human attempts to complete a new, previously unseen task. We know that the robot *should* recognize its uncertainty and arbitrate control back to the human. But the robot is also trying to provide assistance and reduce the human's burden — and if the robot mistakenly thinks it knows the human's intent, our system may override the user and autonomously perform the wrong task. For example, in Figure 3 this *false positive* causes the robot to open the fridge (a previously seen skill) instead of reaching for the cup (a new and unexpected goal).

Motivated by this failure case, we bound the error between the robot's final state and the human's desired goal. We first start with a single degree-of-freedom system for the sake of clarity, and then extend our analysis to d-dimensional robot arms. Overall, we prove that the final state error of a SARI robot — i.e., the distance between s and the human's goal — is *uniformly ultimately bounded*. The radius of this bound is a function of the SARI design parameters, the distance between the human's new task and previously seen tasks, and the variance in the human teleoperator's inputs. Throughout this section we conduct experiments with simulated humans and simulated or real robot arms: we find that our theoretical error bounds align with the measured error in these studies. Finally, we extract practical guidelines that designers can leverage to tune the hyperparameters of our SARI algorithm.

5.1 Single Degree-of-Freedom System

To more clearly explain our theoretical analysis we start by considering a 1-DoF assistive robot. Here the state $s \in \mathbb{R}$, the human command $a_{\mathcal{H}} \in \mathbb{R}$, and the robot assistance $a_{\mathcal{R}} \in \mathbb{R}$ are all scalars. The human is teaching this robot to reach for static goals[4]. At every previous interaction the human teleoperated the robot towards a *known goal* g. Now the human changes their mind and attempts to reach a *new goal* g^*. Returning to our motivating example from Figure 3, perhaps the human has repeatedly teleoperated their assistive arm to open the fridge, and now at interaction i the user wants to pick up a cup.

Robot. During past interactions the human guided the robot towards g. We assume these past human actions were sampled from a Gaussian distribution $a_{\mathcal{H}} \sim \mathcal{N}\big((g - s), \sigma_{\mathcal{D}}^2\big)$ that nosily moved from s to g. Applying SARI, the robot collects these state-action pairs into sequences $\tau = \{(s^1, a_{\mathcal{H}}^1), \ldots, (s^T, a_{\mathcal{H}}^T)\}$ and a dataset $\mathcal{D} = \{\tau^1, \ldots, \tau^{i-1}\}$. The robot then learns to recognize and replicate the human's behavior by minimizing Equation (5) across the dataset $\mathcal{D}$. We assume a *best-case* robot that learns to perfectly match the human's past behavior such that the robot's assistive policy is $a_{\mathcal{R}} \sim \mathcal{N}\big((g - s), \sigma_{\mathcal{D}}^2\big)$. Similarly, this best-case robot learns to return control such that the discriminator — i.e., β in Equation (6) — is the robot's learned policy evaluated at the human's current action:

$$\beta(s, a_{\mathcal{H}}) = \frac{1}{\sqrt{2\pi\sigma_{\mathcal{D}}^2}} \exp\left(\frac{-\big(a_{\mathcal{H}} - (g - s)\big)^2}{2\sigma_{\mathcal{D}}^2}\right) \tag{7}$$

We emphasize that $\sigma_{\mathcal{D}}$ captures the precision and consistency of the human's previous interactions. Here $\sigma_{\mathcal{D}} \to 0$ indicates that the human directly guided the robot to the known goal g, while $\sigma_{\mathcal{D}} \to \infty$ indicates that the human's past interactions were noisy and imperfect (i.e., the human may have pressed the joystick in the wrong direction or overshot their goal).

Human. During the current interaction the human reaches for a new, unexpected goal g^*. As before, we assume the human follows a Gaussian distribution $a_{\mathcal{H}} \sim \mathcal{N}\big((g^* - s), \sigma_{\mathcal{H}}^2\big)$. The standard deviation $\sigma_{\mathcal{H}}$ captures the precision of the human's inputs when reaching for this new goal. We recognize it might be easier (or harder) for the human to teleoperate the robot to the new goal, and thus $\sigma_{\mathcal{H}}$ does not necessarily equal $\sigma_{\mathcal{D}}$.

Lyapunov Stability Analysis. The desired equilibrium of the human-robot system is $s = g^*$, i.e., we want the robot to move to the human's new goal. We propose the Lyapunov candidate function:

$$V(t) = \frac{1}{2}e(t)^2, \quad e(t) = g^* - s(t) \tag{8}$$

where $e(t) \in \mathbb{R}$ is the error between the robot's state s and the human's goal g^* during the current interaction. Taking the time derivative of Equation (8), and substituting in the robot's dynamics from Equation (1) and

---

[4]Our analysis can also be extended to continuous skills by assuming that the human's goal is the closest waypoint along the skill's trajectory.

Equation (2), we obtain:

$$\dot{V}(t) = -a_{\mathcal{H}}(g^* - s) + \beta a_{\mathcal{H}}(g^* - s) - \beta a_{\mathcal{R}}(g^* - s) \tag{9}$$

Recall that $a_{\mathcal{H}}$, $a_{\mathcal{R}}$, and β are all probabilistic quantities. We take the expectation of $\dot{V}$ to reach:

$$\mathbb{E}[\dot{V}(t)] = -(g^* - s)^2 + \mathbb{E}[\beta a_{\mathcal{H}}](g^* - s) - \mathbb{E}[\beta](g^* - s)(g - s) \tag{10}$$

Intuitively, we want Equation (10) to be negative so that $V(t)$ *decreases* over time and the human-robot system approaches equilibrium $e(t) = 0$ in expectation.

For our next steps it is critical to understand the role of the arbitration factor β. Recalling that $a_{\mathcal{H}} \sim \mathcal{N}\big((g^* - s), \sigma_{\mathcal{H}}^2)\big)$, we take the expectation of Equation (7) to reach:

$$\mathbb{E}[\beta] = \frac{1}{\sqrt{2\pi(\sigma_{\mathcal{D}}^2 + \sigma_{\mathcal{H}}^2)}} \exp\left(\frac{-(g^* - g)^2}{2(\sigma_{\mathcal{D}}^2 + \sigma_{\mathcal{H}}^2)}\right) \tag{11}$$

This function does not always hold. From our original definition in Equation (8) we remember that $\beta \in [0, 1]$, where $\beta \to 0$ corresponds to full human control and $\beta \to 1$ is fully autonomous behavior. In practice, designers may further limit $\beta \leq \beta_{max}$, $\beta_{max} \in (0, 1]$, so that the human maintains a persistent minimal control over the assistive robot [15, 49, 59]. Accordingly, we here reach *two cases* for our stability analysis: (a) when $\mathbb{E}[\beta] \geq \beta_{max}$ and (b) when $\mathbb{E}[\beta] < \beta_{max}$. Below we derive two separate stability results for both of these cases.

Theorem 1. Consider a 1-DoF robot using SARI. Given the robot's learned policy is $a_{\mathcal{R}} \sim \mathcal{N}\big((g - s), \sigma_{\mathcal{D}}^2)\big)$, the human's current policy is $a_{\mathcal{H}} \sim \mathcal{N}\big((g^* - s), \sigma_{\mathcal{H}}^2)\big)$, and $\mathbb{E}[\beta] \geq \beta_{max}$ in Equation (11), the error is *uniformly ultimately bounded*. The ultimate bound is:

$$|g^* - s| > \beta_{max} \cdot |g^* - g| \tag{12}$$

Proof. Recall that for persistent minimal control we require that $\beta \leq \beta_{max}$. Since we have $\mathbb{E}[\beta] \geq \beta_{max}$ and the system cannot physically exceed β_{max}, we set $\beta = \beta_{max}$. Accordingly, we have that $\mathbb{E}[\beta] = \beta_{max}$ and $\mathbb{E}[\beta a_{\mathcal{H}}] = \beta_{max} \cdot \mathbb{E}[a_{\mathcal{H}}]$ in Equation (10). Rearranging the updated Equation (10), we find that $\mathbb{E}[\dot{V}(t)] < 0$ when Equation (12) is satisfied. It follows that the human-robot system is uniformly ultimately bounded [38, 61]. See the Appendix for more details. □

We can intuitively think of Theorem 1 as a *false-positive* situation: here $\beta = \beta_{max}$ and the robot is fully convinced that the human's current goal is g. Fortunately, our SARI algorithm is designed to prevent false-positives by returning control when the robot is faced with new or previously unseen behaviors. This leads to our second setting where $\mathbb{E}[\beta] < \beta_{max}$.

Theorem 2. Consider a 1-DoF robot using SARI. Given the same conditions as in Theorem 1, but now $\mathbb{E}[\beta] < \beta_{max}$, the error is *uniformly ultimately bounded*. The ultimate bound is:

$$|g^* - s| > \mathbb{E}[\beta] \cdot \frac{\sigma_D^2}{\sigma_{\mathcal{D}}^2 + \sigma_{\mathcal{H}}^2} \cdot |g^* - g| \tag{13}$$

Proof. Since $\mathbb{E}[\beta] < \beta_{max}$ we set $\beta = \beta(s, a_{\mathcal{H}})$. We now have that β depends upon $a_{\mathcal{H}}$: to compute $\mathbb{E}[\beta a_{\mathcal{H}}]$, we turn to the law of the unconscious statistician (LOTUS) [58]:

$$\mathbb{E}[\beta a_{\mathcal{H}}] = \frac{(g - s)\sigma_{\mathcal{H}}^2 + (g^* - s)\sigma_{\mathcal{D}}^2}{\sqrt{2\pi}(\sigma_{\mathcal{D}}^2 + \sigma_{\mathcal{H}}^2)^{3/2}} \exp\left(-\frac{(g^* - g)^2}{2(\sigma_{\mathcal{D}}^2 + \sigma_{\mathcal{H}}^2)}\right) \tag{14}$$

Substituting both Equation (11) and $\mathbb{E}[\beta a_{\mathcal{H}}]$ back into Equation (10), we find that $\mathbb{E}[\dot{V}(t)] < 0$ when Equation (13) is satisfied. From this it follows that the human-robot system is uniformly ultimately bounded [38, 61]. □
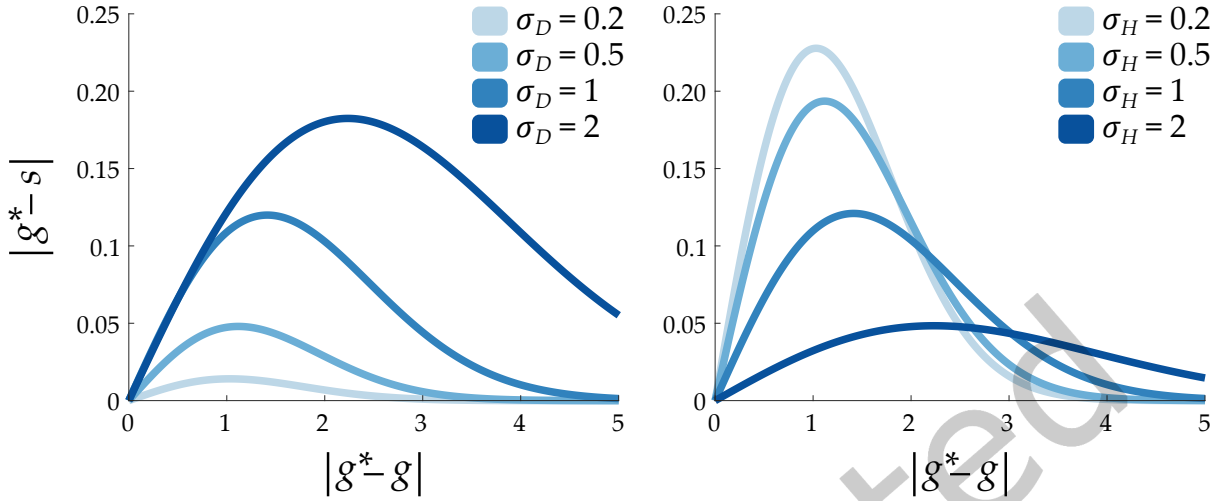
Fig. 4. Error bounds for the 1-DoF system as a function of human noise. All values are in meters. Plots generated using Equation (12) and Equation (13) with $\beta_{max} = 1$. (Left) For a fixed $\sigma_{\mathcal{H}} = 1$ we increase $\sigma_{\mathcal{D}}$. This captures a human that provided increasingly noisy inputs during past interactions when they were reaching for the known goal g. (Right) For a fixed $\sigma_{\mathcal{D}} = 1$ we increase $\sigma_{\mathcal{H}}$. This corresponds to a human that provides increasingly noisy inputs during the current interaction while reaching for the new goal g^*. We conclude that $\sigma_{\mathcal{D}}$ and $\sigma_{\mathcal{H}}$ have opposite effects on the theoretical error bound.

Implications for SARI. We here highlight *three design guidelines* that emerge from the stability analysis of a 1-DoF system. First, looking at Theorem 1, we find that lower values of β_{max} lead to a decreased error $|g^* - s|$. This aligns with our expectations: when $\beta \to 0$ the human always retains control and guides the robot without any autonomous intervention. However, smaller values of β_{max} also limit the maximum assistance the robot can provide, forcing the human to continually teleoperate the robot arm. Hence, choosing β_{max} is a *trade-off* between increased error bounds and increased human effort.

Second, from Theorem 2 the precision of the human's previous interactions ($\sigma_{\mathcal{D}}$) and current interaction ($\sigma_{\mathcal{H}}$) have opposite effects on the error bound (see Figure 4). Humans that accurately moved to goal g will have lower error bounds when reaching for the new goal; i.e., *decreasing* $\sigma_{\mathcal{D}}$ reduces the error $|g^* - s|$. Conversely, after the human starts moving towards the new goal g^*, noisy motions are beneficial: *increasing* $\sigma_{\mathcal{H}}$ reduces the error $|g^* - s|$. We tie both of these trends back to the discriminator in Equation (6). When the human's inputs are easily distinguished from previous interactions — i.e., when the human takes actions the robot has not seen before — SARI returns control and the human can reach their new goal.

Finally, by combining Theorem 1 and Theorem 2 we have that the worst-case error occurs when the human's new goal g^* is *close to* — but not the exact same as — the human's previous goal g. As $|g^* - g|$ increases $\mathbb{E}[\beta] \to 0$ and the error bound in Equation (13) approaches zero. Similarly, when $|g^* - g| \to 0$ we have that $|g^* - s| \to 0$ in both Equation (12) and Equation (13). Our results here are consistent with [20], where Fontaine *et al.* demonstrate that nearby goals are an adversarial setting for existing shared autonomy algorithms.

Experimental Validation. In our analysis we have made two important assumptions about SARI. First, we assumed that the robot's learned policy $a_{\mathcal{H}} \sim \mathcal{N}\big((g - s), \sigma_{\mathcal{D}}^2)\big)$ exactly replicates the human's previous behavior. Second, we assumed that the robot's discriminator learns Equation (7), i.e., the discriminator outputs the likelihood of the human's current action under the robot's learned policy. In Figure 5 we test both of those assumptions by *comparing the theoretical bounds* from Equation (12) and Equation (13) to the *experimental behavior* of our SARI algorithm.
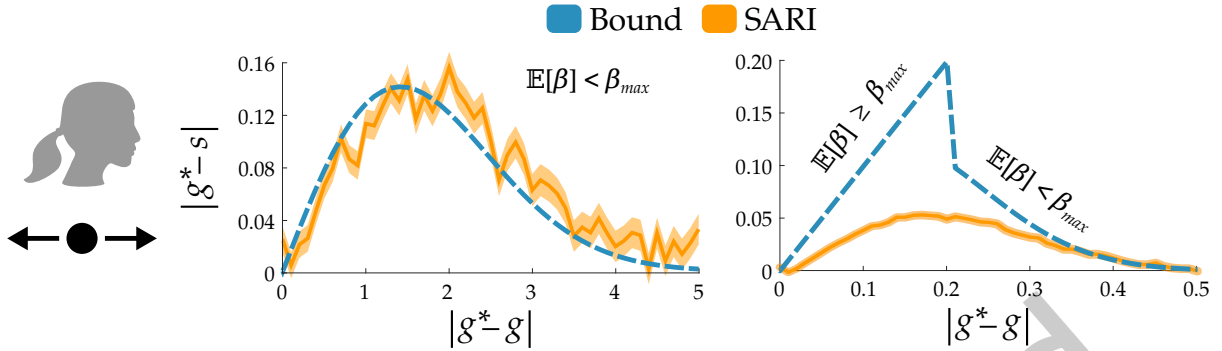
Fig. 5. Error bound and experimental results for a 1-DoF SARI system. All values are in meters. Here a simulated Gaussian human provided 250 demonstrations reaching for their original goal g. These demonstrations were used to train the SARI algorithm; at test time the simulated human reached for a series of new goals g^* with SARI assistance. For each g^* we collected $10,000$ runs — the shaded region is the standard deviation across these runs. (Center) While reaching for the previous goal g and new goal g^* the human had noise $\sigma_\mathcal{D} = \sigma_\mathcal{H} = 1$. For all choices of g^* we have that $\mathbb{E}[\beta] < \beta_{max}$ in Equation (11), and thus the theoretical bound is Equation (13). (Right) We choose $\sigma_\mathcal{D} = \sigma_\mathcal{H} = 0.1$ and had two different theoretical error bounds: When g^* is close to g then $\mathbb{E}[\beta] \geq \beta_{max}$ and Equation (12) applies; but as g^* get farther from g we have that $\mathbb{E}[\beta] < \beta_{max}$, and thus the bound is Equation (13). The bound appears tight when $\mathbb{E}[\beta] < \beta_{max}$ and more conservative when $\mathbb{E}[\beta] \geq \beta_{max}$.

To generate these plots we simulated a 1-DoF point-mass robot and human operator. The simulated human repeatedly guided the robot to a known goal g with actions $a_\mathcal{H} \sim \mathcal{N}\big((g-s), \sigma_\mathcal{D}^2)\big)$, and we followed the procedure from Section 4 to train our SARI algorithm on these interactions. The simulated human then takes actions $a_\mathcal{H} \sim \mathcal{N}\big((g^* - s), \sigma_\mathcal{H}^2)\big)$ to reach a new goal g^* while receiving SARI assistance. For values of g^* where Theorem 2 applies (i.e., when $\mathbb{E}[\beta] < \beta_{max}$) we find a close correspondence between Equation (13) and the robot's measured error $e(t)$. For values of g^* where $\mathbb{E}[\beta] \geq \beta_{max}$ it appears that Theorem 1 becomes overly conservative: the experimental error is consistently lower than Equation (12).

5.2 Multiple Degree-of-Freedom System

So far we have explored the stability and error bounds of a 1-DoF system. We now extend these results to the general case. Here the state $s \in \mathbb{R}^d$, the human command $a_\mathcal{H} \in \mathbb{R}^d$, and the robot assistance $a_\mathcal{R} \in \mathbb{R}^d$ are all d-dimensional vectors. To better distinguish that we are working with vectors we will **bold** these symbols for this section of the paper. Our problem setup is the same as in Section 5.1: during past interactions the human guided the robot towards goal g, and we want to evaluate error during the current interaction when the human is teleoperating the robot to a new, previously unseen goal g^*.

Assumptions. As before, we assume that the human's actions during past interactions were sampled from a multivariate Gaussian distribution $a_\mathcal{H} \sim \mathcal{N}\big((g - s), \Sigma_\mathcal{D}\big)$. During the current interaction the human noisily moves towards their new goal by following the policy $a_\mathcal{H} \sim \mathcal{N}\big((g^* - s), \Sigma_\mathcal{H}\big)$. We make two key assumptions about SARI. (a) Our approach learns to perfectly recognize and replicate the human's past behavior, and provides assistive actions $a_\mathcal{R} \sim \mathcal{N}\big((g - s), \Sigma_\mathcal{D}\big)$. (b) Our discriminator learns to output a scalar β that matches the robot's policy evaluated at the human's action $a_\mathcal{H}$. These assumptions are the same as in Section 5.1. We note that the covariance matrix $\Sigma_\mathcal{D}$ captures the human's noise during past interactions, and $\Sigma_\mathcal{H}$ is the noise during the current interaction.
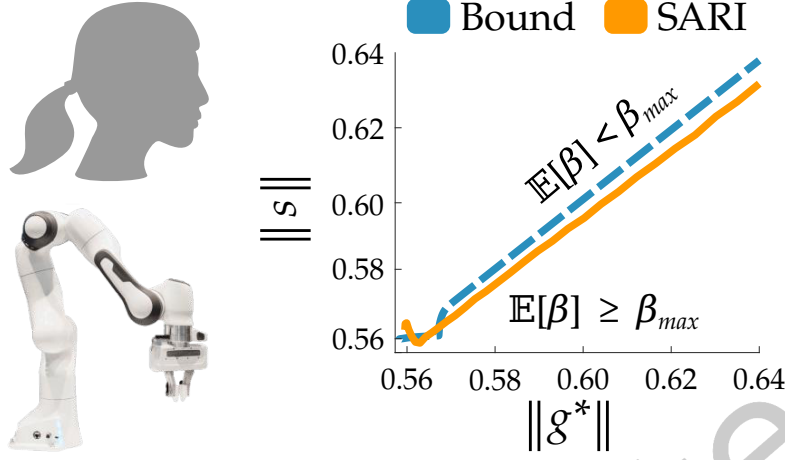
Fig. 6. Error bound and experimental results for a 3-DoF SARI system. All values are in meters, and ideally $\|s\| = \|g^*\|$. A simulated Gaussian human provided 25 demonstrations reaching for their original (x, y, z) goal position on the Franka robot arm. These demonstrations were used to train the SARI algorithm; at test time the simulated human reached a series of new goals g^* with SARI assistance. For each g^* we collected 5 runs. While reaching for the previous goal g and new goal g^* the human had noise $\Sigma_{\mathcal{D}} = \Sigma_{\mathcal{H}} = 1e^{-4} \cdot I$, where I is the identity matrix. For choices of $\|g^*\|$ close to 0.56, we have that $\mathbb{E}[\beta] \geq \beta_{max}$ and the bound is given by Equation (18). As $\|g^*\|$ increases beyond 0.57, we find $\mathbb{E}[\beta] < \beta_{max}$ and the bound is Equation (19).

Lyapunov Stability Analysis. We want SARI to drive the human-robot system towards the equilibrium $s = g^*$. We accordingly propose the Lyapunov candidate function:

$$V(t) = \frac{1}{2}\|e(t)\|^2, \quad e(t) = g^* - s(t) \tag{15}$$

Taking the time derivative of Equation (15), plugging in the robot's dynamics from Equation (1) and Equation (2), and then taking the expectation, we obtain:

$$\mathbb{E}[\dot{V}(t)] = -e^T\left(e - \mathbb{E}[\beta a_{\mathcal{H}}] + \mathbb{E}[\beta](g - s)\right) \tag{16}$$

Our goal here is to find a condition that ensures $\mathbb{E}[\dot{V}(t)] < 0$ so that the human-robot system approaches equilibrium $e(t) = 0$. Importantly, the next steps of our analysis depend on the expected value of β. Given our assumptions about SARI, for a d-dimensional system we find that:

$$\mathbb{E}[\beta] = \frac{1}{\sqrt{(2\pi)^d \det \Sigma}} \exp\left(-\frac{1}{2}\|g^* - g\|^2_{\Sigma^{-1}}\right) \tag{17}$$

where $\Sigma = \Sigma_{\mathcal{D}} + \Sigma_{\mathcal{H}}$ is the sum of the covariance matrices. Recalling that the arbitration factor β must be within $[0, \beta_{max}]$, we again reach two *two cases* for our stability analysis: (a) when $\mathbb{E}[\beta] \geq \beta_{max}$ and (b) when $\mathbb{E}[\beta] < \beta_{max}$. Below we list the general stability results for each case.

Theorem 3. Consider a d-DoF robot using SARI. Given the robot's learned policy is $a_{\mathcal{H}} \sim \mathcal{N}\left((g - s), \Sigma_{\mathcal{D}}\right)$, the human's current policy is $a_{\mathcal{H}} \sim \mathcal{N}\left((g^* - s), \Sigma_{\mathcal{H}}\right)$, and $\mathbb{E}[\beta] \geq \beta_{max}$ in Equation (17), the error is *uniformly ultimately bounded*. The ultimate bound is:

$$\|g^* - s\| > \beta_{max} \cdot \|g^* - g\| \tag{18}$$

Proof. Since $\mathbb{E}[\beta] \geq \beta_{max}$ we set $\beta = \beta_{max}$. Hence $\mathbb{E}[\beta] = \beta_{max}$ and $\mathbb{E}[\beta a_{\mathcal{H}}] = \beta_{max}(g^* - s)$. Substituting this into Equation (16) and applying the Cauchy–Schwarz inequality, $\mathbb{E}[\dot{V}(t)] < 0$ when Equation (18) is satisfied. It follows that the human-robot system is uniformly ultimately bounded [38, 61]. See our Appendix for more details. □

Theorem 4. Given a d-DoF SARI robot under the same conditions as in Theorem 3, but now with $\mathbb{E}[\beta] < \beta_{max}$, the error is *uniformly ultimately bounded*. The ultimate bound is:

$$\|g^* - s\| > \lambda \mathbb{E}[\beta] \cdot \|g^* - g\| \tag{19}$$

where λ is the maximum eigenvalue of $(\Sigma_{\mathcal{D}} + \Sigma_{\mathcal{H}})^{-1}\Sigma_D$.

The full proof for Theorem 4 can be found in the Appendix. We highlight that if $d = 1$ and we have a single DoF robot, then Equation (18) and Equation (19) are equivalent to our univariate results from Equation (12) and Equation (13). Overall, the SARI error bounds are a function of the designer's choice of β_{max}, the amount of noise in the operator's joystick inputs, and the distance between the previous and new goals.

Experimental Validation. To support our stability analysis we compared the theoretical error bounds from Equation (18) and Equation (19) to the actual behavior of our SARI algorithm. We conducted this study on a Franka Emika Robot arm with a simulated human teleoperator.

The results are shown in Figure 6. The simulated human used a Gaussian policy when reaching for g, and we trained SARI using the state-action pairs collected from these interactions. SARI then assisted the simulated human as they reached towards a previously unseen goal g^*. We observe a close correspondence between Equation (19) and the measured error when $\mathbb{E}[\beta] < \beta_{max}$. For goal positions where $\mathbb{E}[\beta] \geq \beta_{max}$ we find that Equation (18) is conservative, and the actual error is less than our theoretical bound. Viewed together, our results from Sections 5.1 and 5.2 support our stability analysis, and suggest that SARI correctly returns control when the human reaches for new and unexpected goals.

6 SIMULATIONS

We have introduced an algorithm that learns to assist users over repeated interaction. Our algorithm (SARI) breaks down into three parts: recognizing the task, replicating prior demonstrations, and returning control when uncertain. In this section we perform simulations to determine how each component of SARI contributes to its overall performance. We recognize that — in practice — human operators will use SARI to assist for multiple everyday tasks. Accordingly, we also test the capacity of our approach, and evaluate how SARI's performance changes as it encounters an increasing number of goals and skills. Throughout this section we compare our approach to state-of-the-art imitation learning baselines that also learn from repeated human-robot interaction. We conduct these experiments on both simulated and real robot arms with simulated human operators.

Experimental Setup. For different simulations we implement SARI on either a 7-DoF Franka Emika robot or a 6-DoF Universal Robots UR10 robot. We test with two different arms to show that our approach is not hardware specific. A simulated user controls the robot to reach *discrete goals* (e.g., grasping a can) and perform *continuous skills* (e.g., opening a drawer). This simulated user is not perfect: the user selects commanded actions $a_{\mathcal{H}}$ with varying levels of Gaussian white noise, similar to the noisy human models from Section 5. Please see the Appendix for additional details on our implementation.

6.1 Do We Need Recognition?

In our first experiment we explore whether we need two separate modules for task recognition and replication. Recall that in Section 4.1 we introduced an encoder which embeds the current interaction τ^i into a latent task prediction $z \in \mathcal{Z}$. Within Section 4.2 we then mapped z to an assistive robot actions using $\pi_{\mathcal{R}}(s, z)$. Here we test whether we need this encoder in the first place: in other words, can we obtain similar performance *without*
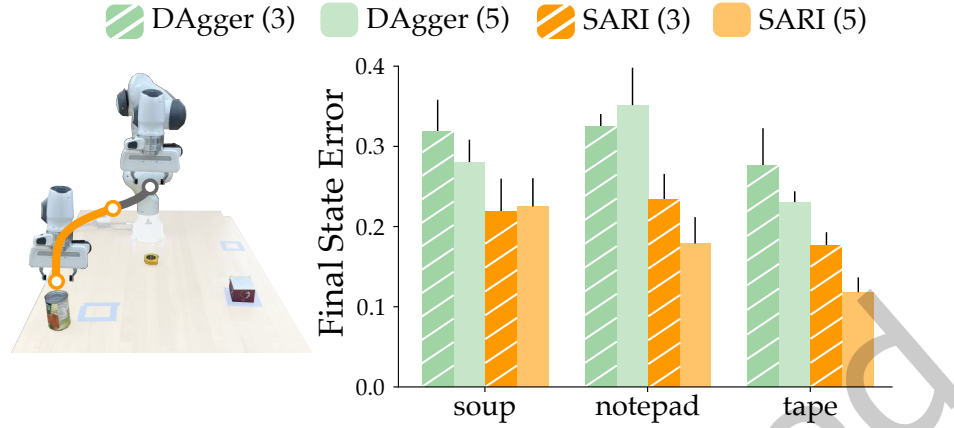
Fig. 7. Comparison to **DAgger** [56], an imitation learning baseline that does not use latent embeddings. A simulated user controls the robot for the first 0.5 seconds of the interaction: the robot must recognize the human's task and complete the rest of the reaching motion autonomously. We measure the final state error for each goal after training with 3 or 5 repeated interactions. Comparing all **DAgger** runs to all **SARI** runs, we find that the final state error is lower with **SARI**: $t(29) = 3.215$, $p < 0.05$.

embedding to latent space $\mathcal{Z}$? We consider an imitation learning baseline that *directly* maps the current interaction τ^i to robot actions $a_\mathcal{R}$ using a learned policy $\pi_\mathcal{R}(s, \tau^i)$. Specifically, we compare **SARI** against **DAgger** [56].

This experiment was performed on the Franka Emika robot arm with a simulated human (see Figure 7). The environment consisted of three potential goals: a can of soup, a notepad, or a tape measure. The human first teleoperated the robot along 3 or 5 demonstrations to reach each goal. We trained **SARI** and **DAgger** from these repeated interactions such that both approaches had access to the same training data. At test time, the human guided the robot for the first 0.5 s of the task: based on this input, the robot had to (a) recognize which task the human was trying to perform and (b) 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 7. Comparing the results when trained with 3 or 5 previous interactions, we find that the robot is better able to provide assistance after additional interactions. Regardless of whether **DAgger** had 3 or 5 demonstrations, however, **SARI** more accurately reached the human's goal given the same simulated human operator. We verify the significance of these results using a paired t-test ($t(29) = 3.215, p < 0.05$) over 5 separate trials with **DAgger** and **SARI**. These results suggest that incorporating a separate encoder for task recognition *improves* the robot's assistance.

6.2 Do We Need Help Returning Control?

In our second experiment we explore the opposite end of our pipeline: determining when the robot should provide assistance. The stability analysis from Section 5 indicates that SARI will return control to the human when the operator is attempting to perform a new task. However, it is equally important for the robot to *retain control* (and provide assistance) when it encounters a *known* task. Here we test if the robot will correctly recognize a previously seen skill. Recall that SARI decides whether or not to provide assistance based on the output of the discriminator from Section 4.3: this discriminator detects if the state-action pairs in τ^i are similar to previous interactions. Instead of training a separate discriminator, one alternative is to rely on the confidence of our learned policy *itself*. Here we turn to prior work on safe imitation learning where the robot samples its learned policy multiple times at the current state, and assesses the similarity of the resulting actions $a_\mathcal{R}$. If all of these
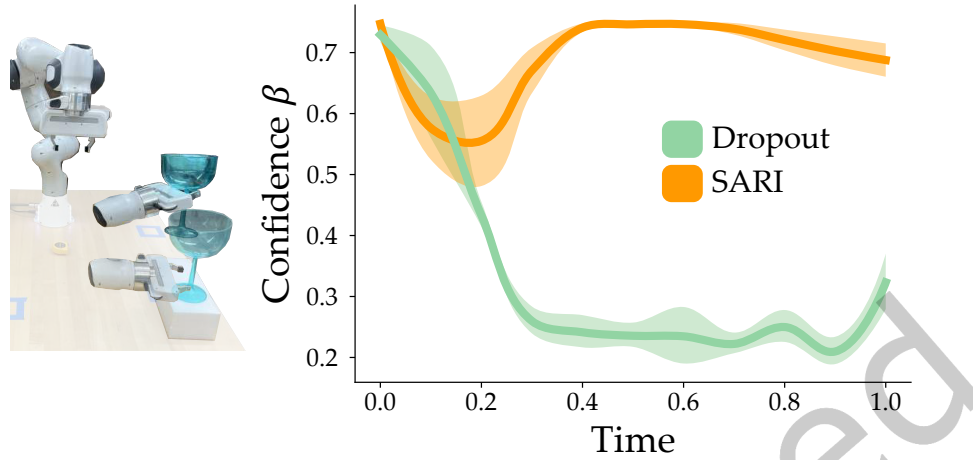
Fig. 8. Comparison to **Dropout**DAgger [46], a safe imitation learning baseline where the robot's learned policy $\pi_{\mathcal{R}}$ evaluates its own confidence. Simulated users attempt to lift a glass. Although the robot has seen this continuous skill 5 times before, with **Dropout** the robot is overly sensitive to minor deviations from previous interactions and rarely provides assistance.

actions are almost identical, the robot is *confident* it knows what to do; conversely, if the model outputs have high variance, the robot is *unsure*. We therefore compare **SARI** to DropoutDAgger [46] (**Dropout**).

This experiment was performed on a Franka Emika robot arm with a simulated human teleoperator (see Figure 8). The simulated human and real robot attempted to complete a continuous manipulation task where the robot must reach and lift a glass. During test time, the human and robot shared control throughout the entire interaction using Equation (2). The robot had seen the human perform this task in five past interactions, and so it should have been confident when providing assistance. To ensure proper comparison, we train **SARI** and **Dropout** on 5 separate trials. We visualize the robot's mean actual confidence β over these trails in Figure 8. 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 assistance. **SARI** remains confident throughout this known task, suggesting that our separate discriminator better arbitrates control than the learned policy itself. This is important in practice: human operators will never perform the same task in the exact same way, and thus the robot must be able to remain confident on known tasks despite some operator variability.

6.3 What if the Operator is Increasingly Noisy?

So far we have focused on the robot's perspective, and have tested each component of our SARI approach. For our third experiment we instead focus on the *human*, and explore how the behavior of the human operator affects SARI. We consider simulated humans with different levels of noise as they attempt to complete previously seen or new tasks. For previously seen tasks, we want to make sure that the robot continues to provide assistance even as the human becomes an increasingly noisy and imperfect operator. For new tasks, we recognize that the robot should not *resist* the human or *force* them along a previously seen trajectory. This problem becomes particularly challenging when the human is noisy, since the robot must determine whether the imperfect human is trying to repeat the known task or complete a new task.

This experiment was performed on a Franka Emika robot arm with a simulated human teleoperator (see Figure 9). The human provided inputs $a_{\mathcal{H}}^*$ to optimally complete the task, and we then injected Gaussian white noise with covariance matrix $\Sigma_{\mathcal{H}} = \text{diag}(\sigma^2, \ldots, \sigma^2)$. The environment contained a previously seen skill (opening
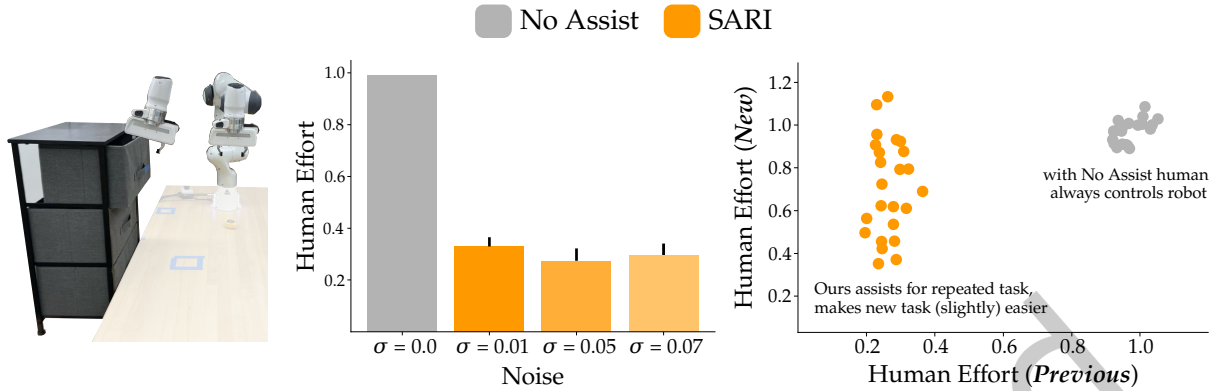
Fig. 9. Simulated humans with increasingly noisy behavior. (Center) The human is always attempting to open the drawer (a previous seen skill). We find that **SARI** correctly recognizes and assists for this task despite noisy and imperfect human teleoperation inputs: $a_{\mathcal{H}} \sim \mathcal{N}\big(a^*_{\mathcal{H}}, \text{diag}(\sigma^2, \ldots, \sigma^2)\big)$. (Right) 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. **SARI** learns to partially automate the previously seen task without resisting humans when they try to complete the new task.

a drawer) and a new goal (reaching a cup). **SARI** was trained with 5 repeated interactions of the drawer skill; however, **SARI** had no prior experience with the cup goal. We compared our approach to a **No Assist** baseline where the human directly teleoperated the robot's end-effector without any shared autonomy. Consistent with prior experiments, we repeat this experiment 5 times and measure the mean *Human Effort* across these trials. We define Human Effort as the amount of time the human teleoperates the robot (i.e., the total time the human is providing joystick inputs) normalized by the average time required to complete the task. Lower values of Human Effort signify that the robot correctly automated the motion, while higher values mean that the human had to teleoperate the robot throughout the task.

We first tested the previously seen drawer skill with increasing levels of human noise σ (see Figure 9, left). Interestingly, we found that **SARI** consistently reduced Human Effort while remaining robust to this range of σ. We then had the simulated human alternate between the new and previous tasks while varying the amount of Gaussian white noise (see Figure 9, right). As expected, **SARI** made it easier for the human to repeatedly open the drawer — but on the new task, **SARI** also correctly returned control back to the human. Performing the new task took no more effort than the **No Assist** baseline; indeed, it often required *less* human effort. To explain this result, we note that the start of the cup task was similar to the start of the drawer skill, and thus **SARI** could automate the beginning of this motion (resulting in less Human Effort).

6.4   How Many Tasks Can We Learn?

For our final experiment with simulated humans we explore **SARI**'s capacity to learn goals and skills. Remember that our motivating application is an assistive robot arm for everyday use: over long-term interaction, this robot will encounter many repeated tasks for which it should provide assistance. More formally, the robot observes an increasing number of interactions τ and aggregates a growing dataset $\mathcal{D} = \{\tau^1, \ldots, \tau^i\}$. Here we test the performance of SARI as it is trained on this iteratively increasing dataset. We separate the experiment into two parts: in the first environment the robot encounters an increasing number of *goals*, and in the second setting the robot must learn to assist for an increasing number of *skills*. Ideally the SARI robot will have the capacity to learn assistance for *all* of these new tasks, without forgetting or failing to assist for previously seen tasks.
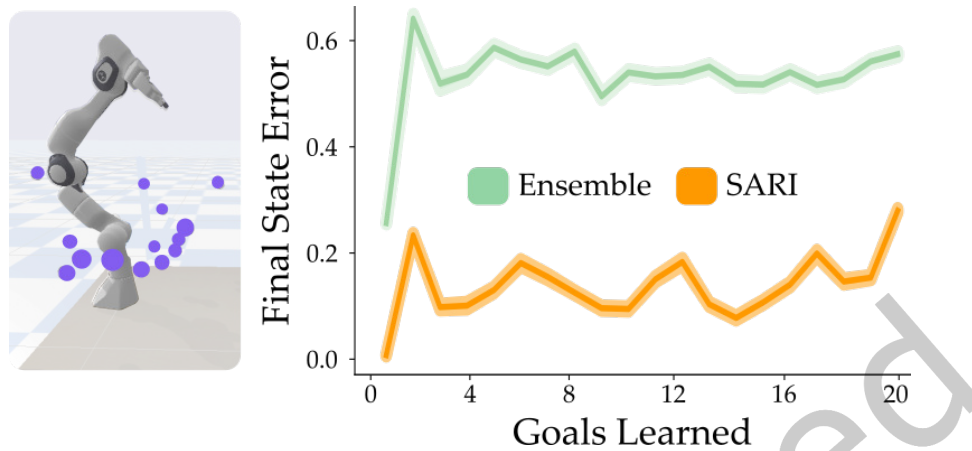
Fig. 10. Capacity to assist for an increasing number of goals. A simulated user repeatedly reached for up to 20 randomly generated goals. The simulated human then teleoperates the robot to reach for all of the goals it has seen so far using two different methods: **Ensemble** [47] and **SARI**. **SARI** has separate modules to recognize, replicate, and return, while **Ensemble** trains multiple replicate modules and returns control when these policies disagree. Final state error is the difference between the robot's final state and the human's intended goal; the shaded region is the standard error about the mean. **SARI** maintains roughly constant performance as the number of goals increases, and consistently has lower error than **Ensemble**.

In both parts of this experiment we compare our proposed approach (**SARI**) to **Ensemble**DAgger [47]. **Ensemble** is an interactive imitation learning approach that trains multiple policies on the human's dataset. Unlike SARI — which has separate models to recognize, replicate, and return — under **Ensemble** the robot only learns policies $\pi_{\mathcal{R}}(s, \tau^i)$ that map the human's behavior directly to robot assistance. The robot trains an ensemble of these policies and compares their outputs: when the actions $a_{\mathcal{R}}$ of each policy agree the robot is confident (i.e., higher β), and when the actions have high variance the robot is uncertain (i.e., lower β).

Goals. To explore our method's capacity to learn goals we simulated a Franka Emika robot arm and human operator in PyBullet [14] (see Figure 10). At the start of the experiment the simulated human repeatedly reached for a single goal, and the robot learned to assist for that goal. Next, the human repeatedly reached for two goals (with new goal positions that were randomly generated), and we tested the robot's ability to assist for *both* goals. Following this pattern, the human iteratively reached for up to 20 goals; during each iteration we tested the robot's ability to assist for *all the goals* it had observed. This procedure ensures that we are capturing the robot's performance on previously seen goals and the new goal after training. To standardize our results, we trained 20 separate **SARI** and **Ensemble** models at every iteration. Each individual **SARI** model assisted the simulated human for a single goal 5 times. For **Ensemble**, we had the ensemble of 20 models assist the human 100 times; put another way, both methods reached for a given goal 100 total times.

To understand how accurately the human-robot system reached goals, we measured the *Final State Error* between the human's actual goal and the robot's final state. Our results are shown in Figure 10. Overall, we observe that both **SARI** and **Ensemble** are *constant* as the number of goals increases: e.g., the error after learning 10 goals is similar to the error after learning all 20 goals. But while both approaches have the capacity to learn multiple goals, we find that **SARI** results in lower error across the board. Our results here are consistent with Section 6.1, and suggest that the recognize and return modules in **SARI** lead to improved performance.

Skills. To explore our method's capacity to learn continuous skills we paired a simulated Gaussian human with a real 6-DoF UR10 arm. Here the simulated human attempted to perform kitchen tasks such as opening a drawer,
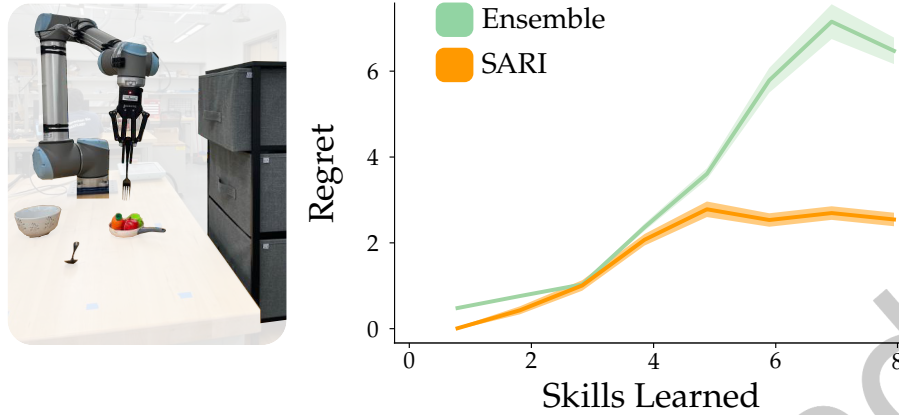
Fig. 11. Capacity to assist for an increasing number of skills. A simulated Gaussian human teleoperated the UR10 robot to perform up to 8 different kitchen skills (e.g., opening a drawer, stabbing a fruit, pushing a bowl). The simulated human then completed these skills when assisted by either **Ensemble** [47] or **SARI**. Regret is the difference between the maximum possible reward and the robot's actual reward: lower values of regret indicate the human-robot system completed the skills correctly. Shaded region indicates standard error about the mean. Unlike Figure 10, we observe that performance decreases as the number of skills increases.

stabbing a piece of fruit, or pushing a bowl (see Figure 11). Similar to **Goals**, we followed an iterative process: first the human and robot repeatedly performed one skill, then two skills, and so on. At each iteration we evaluated the robot's ability to assist for *all the skills* it had seen so far. We had a total of 8 skills, and to remove any ordering bias we repeated the experiment twice: once while observing skills $1 \rightarrow 8$, and once while observing skills $8 \rightarrow 1$. To standardize our results, we trained **SARI** and **Ensemble** 20 separate times at each iteration, and evaluated each model's performance 5 times per skill.

We measured *Regret* to understand if human-robot system performed each skill correctly. Let $R^*(\xi)$ be the maximum reward that the system can achieve on skill ξ; we define *Regret* as $R^*(\xi) - R_{actual}(\xi)$, i.e., the difference between the best-case reward and the robot's actual reward. Our results are plotted in Figure 11. Unlike **Goals**, we find that the system's performance decreases as the number of skills increases. There are two reasons for this: (a) skills are more complicated than goals, and require more assistance than just a straight point-to-point motion and (b) the robot encounters similar states when performing different skills. This could lead to confusion: if the robot observes state s when pushing the bowl and stabbing the fruit, it is unclear which task the human is currently attempting to perform (and what assistance the robot should provide). Despite these challenges, **SARI** maintains consistently lower regret when compared to the **Ensemble** baseline. Overall, our results from both **Goals** and **Skills** suggest that **SARI** has the capacity to learn assistance for multiple tasks. We recognize that this assistance may degrade as the robot continues to aggregate new demonstrations, particularly for skills.

7 USER STUDIES

In Sections 5 and 6 we studied the theoretical and practical performance of SARI with *simulated* human users. In this section we now turn to user studies with *actual* participants. Recall that our target application is assistive robot arms: we want to enable these arms to share autonomy during everyday tasks. Motivated by this application, we conducted two in-person user studies with *non-disabled* participants and one pilot study with a *disabled* adult who regularly operates assistive robot arms. To show that our method is not specific to particular hardware, we include two different robot arms in our experiments. In the first study participants teleoperated a 7-DoF Franka
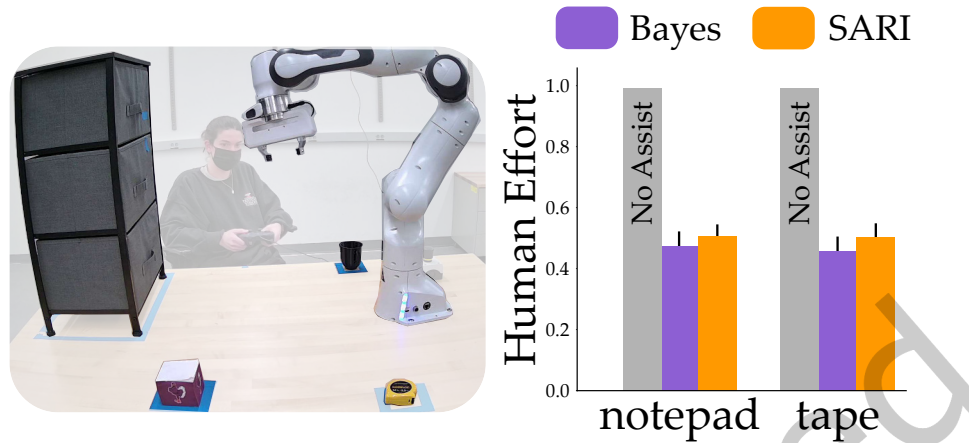
Fig. 12. Experimental setup and objective results from the first part of our first user study. Here participants teleoperated the Franka Emika robot arm to reach for two goals that were known *a priori*. We compared our approach (**SARI**) to a shared autonomy baseline (**Bayes** [15]) and an industry standard mapping for assistive arm teleoperation (**No Assist** [2]). When the robot has prior knowledge about the human's potential tasks, we find that **SARI** learns to offer assistance on par with **Bayes**, and both methods reduce the human's effort when compared to **No Assist**. Note that **Bayes** fails to provide helpful assistance when the human wants to perform new, unexpected tasks (e.g., reaching the cup or opening the drawer), as shown in Figure 13.

Emika robot arm, and in the second and third studies they teleoperated a 6-DoF Universal Robots UR10 robot arm. The non-disabled participants used a handheld joystick interface for teleoperation (see Figures 1 and 12), while the disabled participant used a web-based interface (see Figure 20). Additional implementation details for our user studies can be found in the appendix. Video of our user studies is available here: https://youtu.be/3vE4omSvLvc.

7.1 Learning Discrete Goals and Continuous Skills

We start with a *three part* user study that explores known and new tasks as well as discrete goals and continuous skills. Non-disabled participants started by reaching for known goals, then taught the robot new skills, and finally returned to the original tasks[5]. Figures 12, 13, 14, and 15 correspond to this study. We compared our approach (SARI) to two different state-of-the-art baselines: (a) direct end-effector teleoperation that is used on commercial assistive robot arms [1, 2] and (b) an existing shared autonomy algorithm that infers the human's goal from a discrete set of options [15].

Independent Variables and Experimental Setup. Our first user study was 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 our proposed approach (**SARI**) to an existing shared autonomy baseline (**Bayes**) [15]. For **Bayes** we gave the robot prior information about the location of each goal; during interaction the robot inferred which goal the human wanted and provided assistance towards that goal. For **SARI** we repeatedly teleoperated the robot to both goals during previous, offline interactions (collecting dataset $\mathcal{D}_{\text{offline}}$). We then trained **SARI** on this dataset; during interaction the robot used our approach to recognize the user's goal and assist the human for that task.

---

[5]For video of the first user study, also see: https://youtu.be/Plh4t3wQeIA
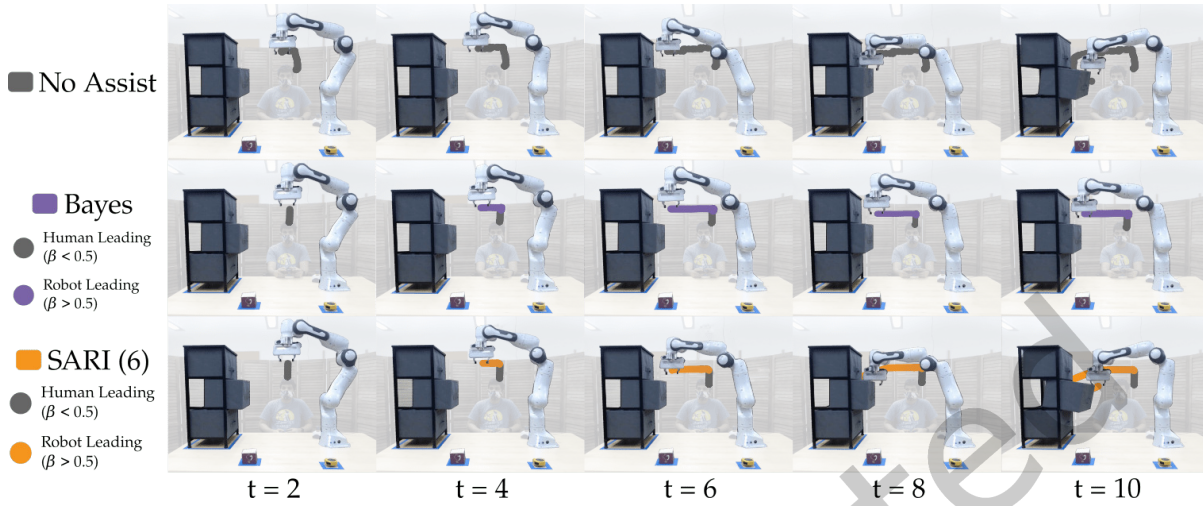
Fig. 13. Representative failure case for an existing shared autonomy approach that relies on prior knowledge (**Bayes** [15]). **Gray** circles indicate the human leading the robot, while **purple** and **orange** indicate the the robot is providing assistance. The user attempts to complete the drawer task with end-effector control (**No Assist**), with **Bayes**, and with **SARI**. None of the methods have prior knowledge about the drawer task; **Bayes** only knows about the notepad and tape goals, and **SARI** has observed 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 **SARI** 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 (**orange** circles). By contrast, **Bayes** (middle) mistakes the initial trajectory as towards the notepad, and continually tries to guide the robot to 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.

The shared autonomy baseline is the gold standard when the human wants to complete a task the robot already knows — but what happens during new tasks? In the second part of our user 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 **SARI** to a **No Assist** baseline. **No Assist** is direct end-effector teleoperation, and is an industry standard approach for assistive robot arms (e.g., pressing right on the joystick causes the robot to move right) [2]. The **No Assist** baseline never learns from interactions; but for **SARI** we retrained our approach every three trials during both tasks. We expect that **SARI** should increasingly assist the user as it gets more familiar with these new tasks.

One concern with our approach is that — as the robot continues to encounter new tasks — it will specialize in just one or two recent tasks without remembering how to share autonomy for older tasks. Accordingly, 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, **SARI (task)**, the robot's learned assistance with just the user's data from that specific task, and **SARI (all)**, our approach trained on the user's full dataset of all interactions.

Dependent Measures – Objective. Across all three parts of the user study we measured *Human Effort*. Human effort is the total time the human teleoperated the robot during the task divided by the average time taken to complete the task. Higher values of human effort indicate that the human had to guide the robot throughout its entire motion, and lower values indicate that the robot partially automated the task.
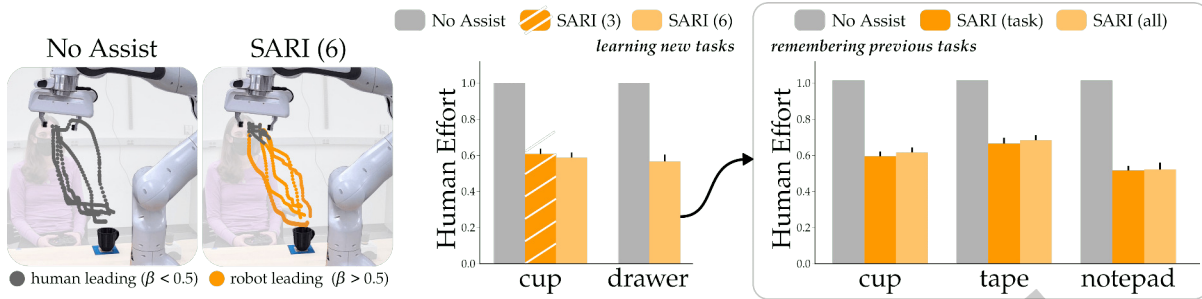
Fig. 14. Objective results from the second and third parts of our user study. (Left) The human teleoperates the robot to reach for a goal it did not know about beforehand. The first few times they interact, the user must lead the robot throughout the entire task. After training **SARI** on six repeated interactions, the robot recognizes the human's intent and automates the rest of the motion; by contrast, with **No Assist** the human always has to teleoperate the robot. (Center) Across $3 - 6$ repeated interactions the robot learns to provide assistance for a new goal (reaching a cup) and skill (opening a 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. **SARI (all)** offers similar assistance to **SARI (task)**, a version of our approach trained only with the user's task-specific data. These results suggest our **SARI** robot has the capacity learn assistance for new tasks without forgetting older ones.

Dependent Measures – Subjective. We administered a 7-point Likert scale survey after users completed the study (see Figure 15). 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 ± 7 years). All participants provided informed written consent prior to the experiment under Virginia Tech IRB #20-755.

Hypotheses. We tested three main hypotheses:

> **H1.** *In cases where the robot has prior knowledge about the human's potential goals, SARI will perform similarly to a shared autonomy baseline*
> **H2.** *In cases where the human repeatedly performs new and previously unseen tasks, SARI will learn to provide meaningful assistance from scratch*
> **H3.** *SARI remembers 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 Figures 12, 13, 14, and 15. In the rest of this subsection we summarize our main findings.

In the first part of the user study participants completed a reaching task with **Bayes** (a shared autonomy baseline) and **SARI** (our proposed approach). Here both methods had prior information about the potential goals: for **Bayes** the robot was given both goal positions, and for **SARI** we recorded offline interactions reaching for each goal. During the user study 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. Our results are shown in Figure 12. To analyze these results we first performed a repeated measures ANOVA, and found that the robot's algorithm had a significant effect on human effort (Notepad: $F(2, 58) = 106$, $p < .001$; Tape: $F(2, 58) = 36.9$, $p < .001$). Post hoc comparisons revealed that both **Bayes** and **SARI** led to less human effort than **No Assist**, but the differences between **Bayes** and **SARI** were not statistically significant (Notepad: $p = .370$; Tape: $p = .203$). These results
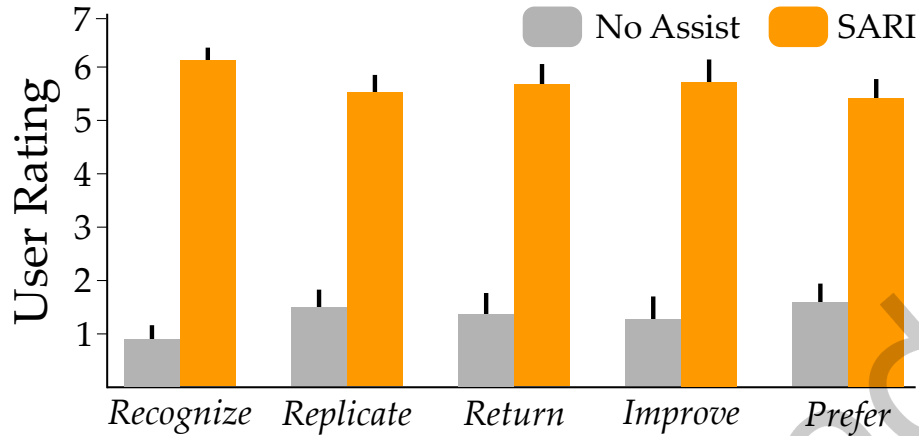
Fig. 15. Subjective results from our in-person user study. Higher ratings indicate user agreement. Overall, participants thought **SARI** provided useful assistance, and they preferred this assistance to trying to complete the tasks with direct end-effector teleoperation (**No Assist**). These scores were provided after participants had completed the entire experiment, including: completing known tasks, assisting for new tasks, and remembering old tasks.

suggest that users could reach for known, discrete goals just as easily with **SARI** as they could with the shared autonomy baseline.

So **SARI** is on par with **Bayes** when the human wants to perform a *known* task — what happens when the human wants to complete a *new*, unexpected task? 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 new task in Figure 13. Here the user attempted to open the drawer, but the robot only had 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 [15, 28, 31]. But in this scenario the robot *does not know beforehand* that the human may want to open the drawer. As a result, **Bayes** misinterpreted the user's inputs and gradually became convinced that the human's target was actually the notepad *next* to the drawer. This interaction ended in a deadlock: the human constantly teleoperated the robot towards the drawer, while the robot continually resisted and refused to return control. Note that the trajectories for **SARI** and **No Assist** are similar — the main difference is that in **SARI** the robot takes the lead and automates the continuous skill. Moving forward we will focus on new tasks, and will compare **SARI** to industry standard teleoperation mappings (**No Assist**).

In the second part of our user study participants repeatedly teleoperated the robot to perform new tasks. This includes the drawer skill in Figure 13 and the cup goal in Figure 14. During the first few interactions **SARI** returned control and the user guided the robot throughout the entire task. But after training **SARI** on 3 and 6 repeated interactions, the robot was able to recognize and partially automate these new tasks. 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. Our results from Figure 14 suggest that **SARI** got better at providing assistance for new tasks over repeated interactions. For example, in the drawer skill the human's effort was significantly less with **SARI** after 6 repeated interactions ($t(29) = 10.5$, $p < .001$).

In the final step of the user study we tested the capacity of our approach. We compared **SARI** trained on all previous interactions to **SARI** trained only on interactions for the given task (see Figure 14). Intuitively, we expected that the more specialized **SARI (task)** would provide the best possible performance: this method has
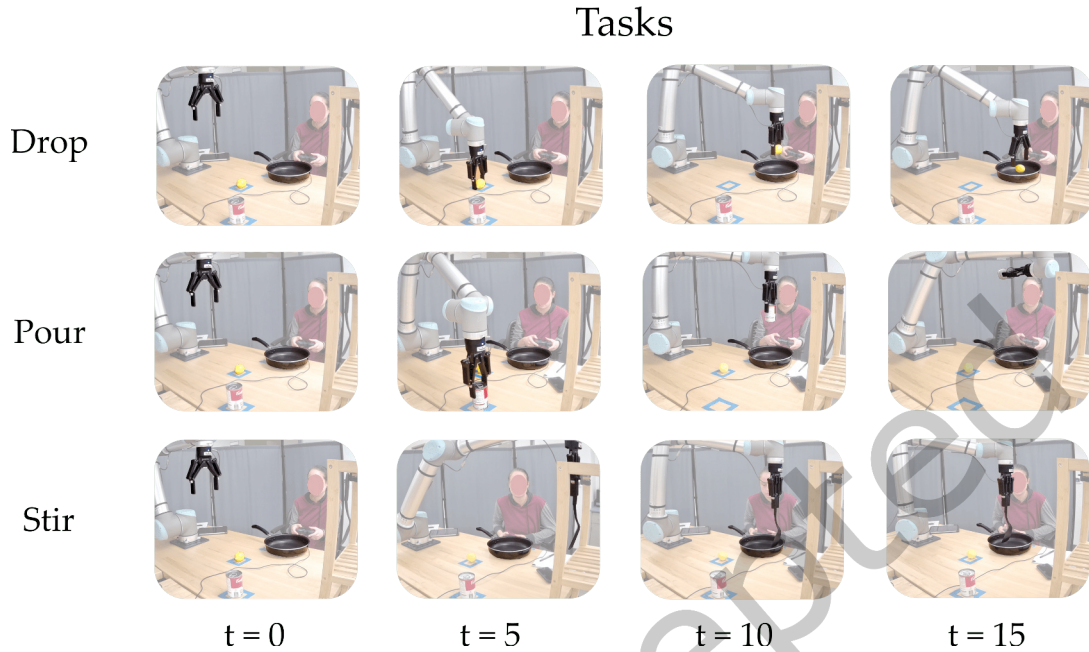
Tasks

Fig. 16. Three tasks used in the second user study. (Top) During **Drop** users picked up a lemon from the table, moved it to the center of the pan, and then dropped it into the pan. (Middle) During **Pour** users picked up a can of soup, moved it to the center of the pan, and then poured the soup in the pan. (Bottom) During **Stir** users fetched a spatula that was placed on a hook, moved towards the top of the pan, and then stirred the pan in a circular motion. All three tasks are continuous skills and were not broken into smaller subtasks. Users completed each task with all three methods (**No Assist**, **CASA**, and **SARI**).

only seen data for the current task and therefore cannot misinterpret the human's inputs. Our results suggest that **SARI** can maintain this performance even when trained with multiple tasks. For three separate goals tasks (cup, tape, and notepad) we first conducted repeated measures ANOVAS, and found that the robot's algorithm had a significant effect on human effort (Cup: $F(2, 38) = 51.9$, $p < .001$; Tape: $F(2, 38) = 47.7$, $p < .001$; Notepad: $F(2, 38) = 74.1$, $p < .001$). Although our approach consistently outperformed **No Assist**, differences between **SARI (task)** and **SARI (all)** were not statistically different (Cup: $p = .416$; Tape: $p = .876$; Notepad: $p = .792$). We therefore conclude that — similar to our simulations in Figure 10 — **SARI** has the capacity to learn assistance for new goals without forgetting how to share autonomy on previously seen tasks.

Taken together, these results support **H1**, **H2**, and **H3**. Our approach leveraged repeated interactions to learn to share autonomy across new and old tasks that included discrete goals and continuous skills. Participants generally perceived the robot's assistance as helpful. Looking at the subjective results from Figure 15, 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.

7.2 Offering Meaningful Assistance and Returning Control

In the previous user study we focused on the robot's ability to learn discrete goals and continuous skills. In our second user study we now focus on the effectiveness of the assistance offered by the robot on known tasks and its ability to *return* control on new tasks. Figures 16, 17, 18, and 19 correspond to this study. We compare our

proposed approach with a previously seen baseline (**No Assist**) and a new state-of-the-art shared autonomy algorithm (**CASA**) [64]. Similar to our method, **CASA** learns to assist users using previous demonstrations.

Experimental Setup. Each participant teleoperated a Universal Robots UR10 robot arm to complete three continuous skills (see Figure 16): picking up a lemon and dropping it in a pan (**Drop**), picking up a can of soup and pouring it in a pan (**Pour**), and picking up a spatula and stirring the pan using a circular motion (**Stir**). We emphasize that these tasks are continuous skills that cannot be easily reduced to a series of goals. To ensure that the human's actions can always steer the robot, we limit the robot's assistive action by restricting the robot's confidence ($\beta_{max} = 0.6$).

The experimental procedure trades-off between teaching new tasks and then repeating known tasks. First, users completed **Drop** with a robot that was pre-trained only on the **Drop** task[6]. Then — using the same assistive models — users completed **Pour**: during this interaction the robot knew **Drop** while **Pour** was an unknown task. Next, users completed **Pour** with a robot that was trained on demonstrations for both **Drop** and **Pour**. Following the same pattern, users completed the new **Stir** task. Here the known tasks are **Drop** and **Pour** and the unknown task is **Stir**. Finally users completed **Stir** with a model that was trained on all three tasks. Users attempted each known and unknown task with every method *three* times. Overall, each individual user provided a total of 27 demonstrations for the three known tasks and 18 demonstrations for the two unknown tasks.

Independent Variables. Participants completed each known and unknown task three times with each method. We then compared the robot's assistance on known tasks and its ability to return control on unknown tasks across three different methods: **No Assist**, **CASA**, and **SARI**. **No Assist** did not offer any type of assistance to the users, while **CASA** and **SARI** learned to offer assistance. More specifically, **CASA** learns cost functions and policies for each new task using Guided Cost Learning [19], and then infers which (if any) of the tasks the human is attempting to complete [64]. Unlike **CASA**, our method (**SARI**) directly matches the human's past behaviors without inferring a cost function or learning a policy to minimize that cost.

Dependent Measures – Objective. For known tasks we measured *Operating Time* and *Opposing Time*, and for unknown tasks we plot *Total Time* and *Mean Confidence*. Here *Operating Time* is the fraction of total time that the human provides input to the robot and *Opposing Time* is the fraction of total time where the dot product between the human's action and the robot's action is negative (i.e., the fraction of time where the robot's assistance is not aligned with the human's commands). *Total Time* is the time required to complete a task in seconds, and *Mean Confidence* measures the robot's average confidence β while the human is performing a task. When the user is performing a *known* task, lower *Operating Time* and *Opposing Time* indicate that the robot is offering meaningful assistance to the user. In the case of *unknown* tasks, lower *Total Time* and *Mean Confidence* indicate that the robot realizes its uncertainly and correctly returns control to the user.

Dependent Measures – Subjective. Similar to the study in Section 7.1 we administered a 7-point Likert scale survey (see Figure 19). Questions were organized along four 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*), and if they would collaborate with the robot again (*Prefer*). Users answered questions related to *Recognize*, *Replicate*, and *Return* after completing each task with each method. At the end of the experiment (after trying each approach) users selected which of the three algorithms they *Preferred*.

Participants and Procedure. We recruited 10 non-disabled members from the Virginia Tech community to participate in our study (5 female, average age 27 ± 4 years). All participants provided informed written consent prior to the experiment under Virginia Tech IRB #20-755. We used a within-subjects design and counterbalanced the order of the learning algorithms between subjects. Before the user study started each participant was given

---

[6]In the interest of time, all models for SARI and CASA were trained using expert demonstrations. SARI required < 10 minutes per model to be fully trained while CASA required over 8 hours per model.
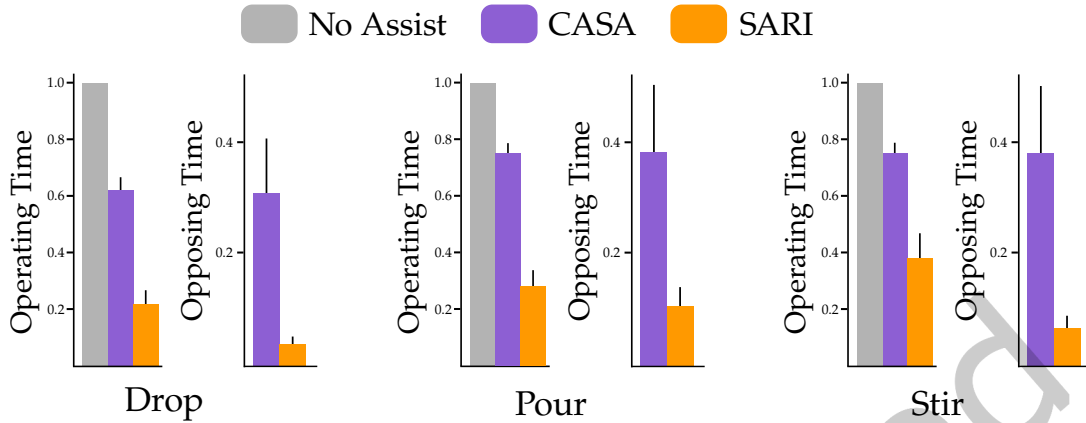
Fig. 17. Objective results for the **known** tasks in our second user study. *Operating Time* is the fraction of the total time where the human provides input to the robot, and *Opposing Time* is the fraction of total time where the robot's actions are antagonistic to the human's input. Note that **No Assist** has no *Opposing Time* since the in this method the human is always in control of the robot. We find that **SARI** offers the most meaningful assistance for all three known tasks by reducing both the *Operating Time* and *Opposing Time* for the human. We performed a one way ANOVA analysis and found that the differences between **CASA** and **SARI** for both *Operating Time* and *Opposing Time* are significant (Drop: $F(2, 87) = 368.6, p < .001$; Pour: $F(2, 87) = 347.4, p < .001$; Stir: $F(2, 87) = 349.6, p < .001$).

$10 - 15$ minutes to teleoperate the robot and familiarize themselves with the controls. The participants in this user study did not take part in the previous experiment from Section 7.1.

Hypotheses. We tested two main hypotheses:

> **H4.** *For previously seen tasks, SARI will better assist than CASA or No Assist*
> **H5.** *For new tasks, SARI will better return control to humans than CASA*

Results – Objective. The objective results from our second user study are shown in Figures 17, 18 and 19. We separate our findings into two categories: performance when the robot has seen the task before (*known*) and performance when the task is new (*unknown*).

Known tasks: Results for the known tasks are shown in Figure 17. We performed a one way ANOVA to analyze these results. Post hoc comparisons revealed that across all three tasks **SARI** reduced the *Operating Time* significantly more than **No Assist** or **CASA** (Drop: $F(2, 87) = 311.0, p < .001$; Pour: $F(2, 87) = 278.7, p < .001$; Stir: $F(2, 87) = 95.3, p < .001$). We also observed that the *Opposing Time* for **SARI** across all three tasks was significantly lower than **CASA** (Drop: $F(2, 87) = 368.6, p < .001$; Pour: $F(2, 87) = 347.4, p < .001$; Stir: $F(2, 87) = 349.6, p < .001$). Note that **No Assist** never opposes since the human is always in control.

Unknown tasks: Results for the unknown tasks are shown in Figure 18. Remember that participants first performed **Pour** while the robot had only seen **Drop**, and then they first performed **Stir** while the robot was trained on **Drop** and **Pour**. Hence, here we do not expect the robot to assist the human: instead, the robot should recognize that the human is completing a new task and return control to the operator. We performed a one way ANOVA to analyze our results and determine whether the robot correctly returned control. We found that **CASA** increased the *Total Time* required to complete both tasks (Pour: $F(2, 87) = 16.6, p < .001$; Stir: $F(2, 87) = 25.437, p < .001$). **CASA** increased the total time because this robot mistakenly thought the human was attempting to perform a known task rather than the unknown task. See the *Mean Confidence*: the maximum allowed mean confidence was 0.6, and **CASA** is close to this maximum confidence for both tasks (i.e., the **CASA** robot was almost entirely convinced it
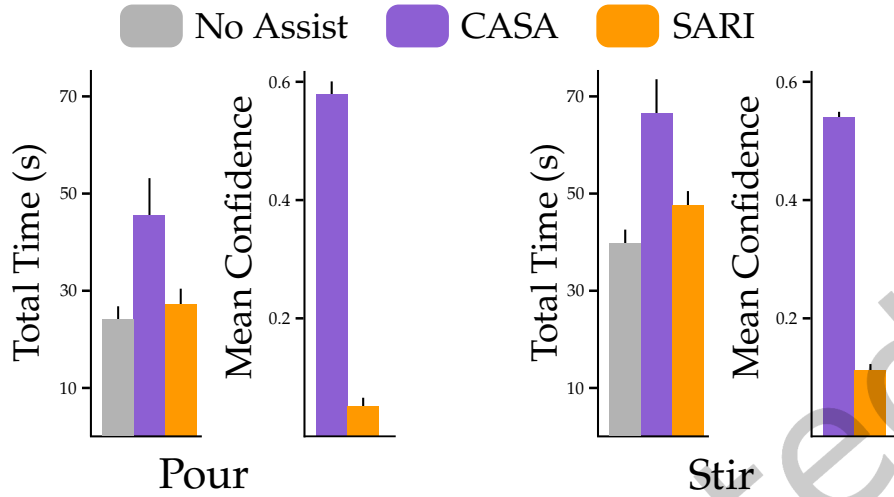
Fig. 18. Objective results for the **unknown** tasks in our second user study. *Mean Confidence* is the robot's average confidence β. Since these tasks are new to the robot, the robot should not be confident: ideally both *Total Time* and *Mean Confidence* will be minimized. **No Assist** always has zero confidence since the human is always in control of this robot. We observe that **CASA** significantly increased the *Total Time* required by the human to complete these tasks (Pour: $F(2, 87) = 16.6, p < .001$; Stir: $F(2, 87) = 25.437, p < .001$). This increase in *Total Time* can be attributed to the high *Mean Confidence* **CASA** maintains (i.e., **CASA** incorrectly thinks it knows the task). By contrast, the **SARI** robot recognizes that it does not know the human's current task and correctly returns control so the human can act without interference.

was performing a known task). Using paired t-tests we found that the mean confidence was significantly lower for **SARI** as compared to **CASA** (Pour: $t(29) = 40.928, p < .001$, Stir: $t(29) = 40.928, p < .001$).

Results – Subjective. We display the user's subjective responses to each algorithm in Figure 19. After confirming that the scales were reliable (Cronbach's $\alpha > 0.7$), we grouped each scale into a single combined score and performed a one-way ANOVA on the results. We observe that across all three tasks users reported that **SARI** best *recognized* their task ($F(2, 177) = 275.6, p < .001$) and *replicated* their demonstrations to offer the most meaningful assistance ($F(2, 177) = 406.8, p < .001$). When it came to *returning* control, **No Assist** was the gold standard; remember that for this method the human was always in control. However, here **SARI** was rated as significantly better than **CASA** ($F(2, 117) = 102.7, p < .001$). Overall, 9 of the 10 participants indicated that **SARI** was their preferred method for sharing autonomy with the robot.

7.3 Assisting Users with Disabilities

In our third and final pilot study we explore how our approach assists a disabled user who operates robot arms on a daily basis. Similar to Section 7.2, the participant interacted with a robot arm that either (a) did not provide any assistance, (b) assisted using the state-of-the-art CASA approach [64], or (c) assisted using our proposed SARI algorithm. This pilot study was conducted remotely: the participant observed the robot in real-time through a live video feed, and remotely teleoperated the robot using a web-based GUI (see Figure 20).

Experimental Setup and Independent Variables. The participant teleoperated the Universal Robots UR10 robot arm to perform two tasks from Section 7.2: **Drop** and **Pour**. The robot had expert demonstrations for both tasks (i.e., both tasks were *known* by the robot). Over a one hour session we collected a total of 6 demonstrations across all tasks and methods from the participant. We compared the effectiveness of the robot's assistance when using **No Assist**, **CASA**, or **SARI**.
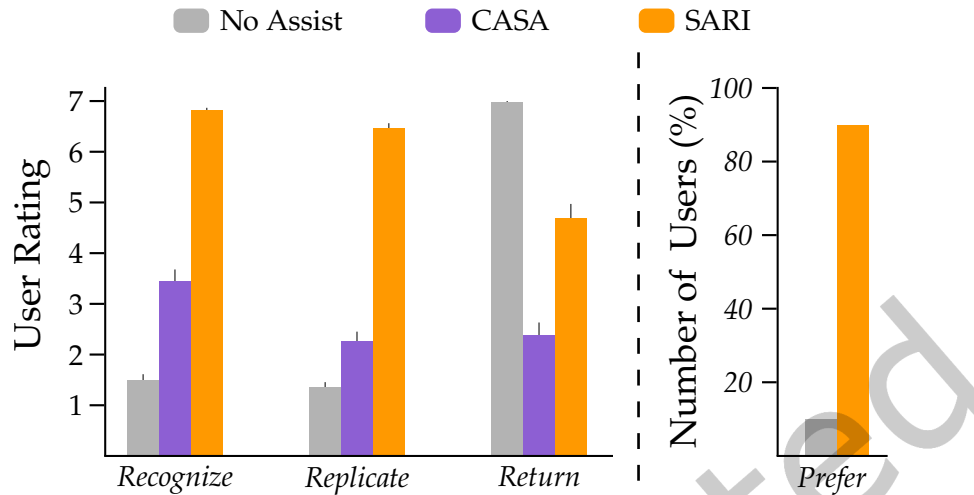
Fig. 19. Subjective results for our second in-person user study. Similar to Figure 15, higher ratings indicate user agreement. Overall, participants found that **SARI** was able to better recognize their task and offer meaningful assistance. Additionally, a majority of users preferred using **SARI** over both **CASA** and **No Assist**. Scores for *Recognize*, *Replicate*, and *Return* were obtained after participants completed each task with each method. For *Prefer* we asked participants to select their favorite algorithm at the end of the experiment: 9 out of 10 users chose **SARI**. All differences shown here are statistically significant ($p < .01$)

Dependent Measures. We measured *Operating Time*, *Opposing Time*, and *Mean Confidence*. Recall that *Operating Time* is the the fraction of total time where the human is providing inputs to the robot, *Opposing Time* is the fraction of total time where the human and robot actions are in opposite directions, and *Mean Confidence* is the robot's average confidence that it should provide assistance (β). Since the robot is performing known tasks, an ideal system will maintain high *Mean Confidence*, partially automate the motion, and reduce *Operating Time* and *Opposing Time*.

Participants and Procedure. We recruited one remote participant for this pilot study. The participant provided informed written consent under Virginia Tech #20-755. Since this user study was conducted online, the participant used a web-based GUI to control the robot (see Figure 20). This GUI includes a table of buttons to control the position and orientation of the end-effector: we designed the GUI to mimic online interfaces for commercial wheelchair-mounted robot arms [2]. The participant moved their cursor and pressed buttons on the GUI using a joystick that was integrated with their wheelchair. For the first 15 minutes of the experiment the participant familiarized themselves with our robot, environment, and controls. During this time we also worked with the participant to find the best camera placements.

Results. Figure 20 presents our results. Remember that the robot is performing a task that it has seen before; ideally the assistive robot will realize this is a known task and help the user. We observed that — similar to the previous user study — both **CASA** and **SARI** reduced the *Operating Time*. However, we also noticed that across both tasks **SARI** provided better assistance by maintaining high *Mean Confidence* and keeping the *Opposing Time* to a minimum. While we only had one participant in this pilot study, the results obtained suggest that **SARI** can offer meaningful assistance to our target population without requiring pre-defined tasks.
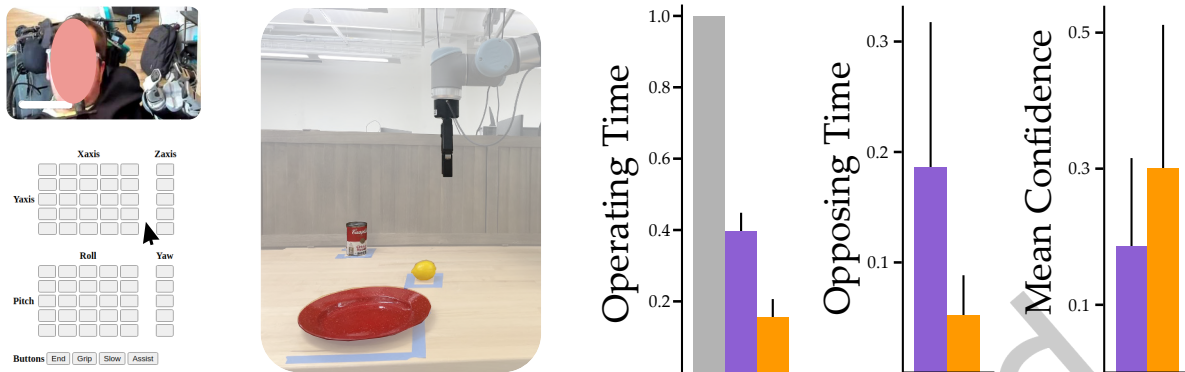
Fig. 20. Experimental setup and objective results for our remote user study with one disabled participant. (Left) The participant used a web-based GUI to teleoperate the robot. The GUI was designed to mimic commercial interfaces [2] and enabled the user to control the robot's end-effector velocity. (Center) We show one of the two camera angles that were streamed in real-time to the remote participant. (Right) Over the course of a one hour session the user completed **Drop** and **Pour** tasks (see Figure 16). Across both tasks **SARI** reduced the *Operating Time* by maintaining a high *Mean Confidence* and low *Opposing Time*. These results suggest that **SARI** can assist adults who regularly use wheelchair-mounted robot arms.

8 CONCLUSION

State-of-the-art shared autonomy algorithms often rely on prior knowledge: e.g., the robot needs to know all of the human's potential tasks *a priori*, or the robot is constrained to actions that assist for previously seen tasks. In this paper we introduce an alternate framework for shared autonomy that leverages the *repeated* nature of everyday human-robot interaction. Our insight is that — if an assistive arm is teleoperated through similar tasks many times — the robot should learn to *partially automate* those tasks. Our approach (SARI) contains separate models that (a) learn to recognize the human's current task, (b) replicate the human's behavior from past task-related interactions, and (c) return control back to the human when the robot is unsure. We leveraged stability analysis that combines learning with control to demonstrate that the error between the human's goal and the SARI robot is uniformly ultimately bounded. We then conducted simulations to support our theoretical error bounds, compare our approach to interactive imitation learning baselines, and explore the capacity of SARI to learn new tasks. Finally, we performed three user studies to demonstrate that SARI assists for both discrete goals and continuous skills, shows greater ability to recognize tasks and return control, and provides meaningful assistance to users with disabilities. Overall, our theoretical and experimental analysis suggests that SARI *personalizes* to the current user, and can learn to share autonomy for the tasks *that user* often performs.

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. One approach to circumvent this issue is by only storing interactions if the human provides corrective actions. However, this co-adaptation is not explicitly accounted for in our approach.

Another potential limitation of SARI may be its capacity. From Section 6.4 we recognize that the robot's assistance decreases as the number of skills increases. More generally, it may not be feasible for a single model to learn assistance for all of the human's everyday tasks. One practical solution is switching models depending on *context*. For example, we could train one instance of SARI to assist for cooking tasks, and another instance of

SARI to assist for dining tasks. During run-time the robot selects which SARI models to use based on the current context (e.g., cooking or eating).

9 APPENDIX

9.1 Details for Error Bounds from Section 5

Below we provide additional details for the error bounds that were presented in Section 5.

Derivation of Equation 11. From Equation (7) we know that the arbitration constant $\beta(s, a_{\mathcal{H}})$ is defined as:

$$\beta(s, a_{\mathcal{H}}) = \frac{1}{\sqrt{2\pi\sigma_{\mathcal{D}}^2}} \exp\left(\frac{-\left(a_{\mathcal{H}} - (g - s)\right)^2}{2\sigma_{\mathcal{D}}^2}\right)$$

Taking into consideration that $a_{\mathcal{H}} \sim \mathcal{N}\left((g^* - s), \sigma_{\mathcal{H}}^2\right)$ and recalling that the law of the unconscious statistician (LOTUS) [58] states that for a function $g(X)$ of a random variable X, the expected value $\mathbb{E}[g(X)] = \int_{-\infty}^{\infty} g(x) f_X(x)\, dx$, where $f_X(x)$ is the probability density function of X. We take the expectation of Equation (7) and substitute the above to obtain:

$$\mathbb{E}[\beta] = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma_{\mathcal{D}}^2}} \exp\left(-\frac{1}{2}\frac{\left(a_{\mathcal{H}} - (g - s)\right)^2}{\sigma_{\mathcal{D}}^2}\right) \cdot \frac{1}{\sqrt{2\pi\sigma_{\mathcal{H}}^2}} \exp\left(-\frac{1}{2}\frac{\left(a_{\mathcal{H}} - (g^* - s)\right)^2}{\sigma_{\mathcal{H}}^2}\right) da_{\mathcal{H}} \quad (20)$$

Upon further simplification we obtain:

$$\mathbb{E}[\beta] = \frac{1}{\sqrt{2\pi(\sigma_{\mathcal{D}}^2 + \sigma_{\mathcal{H}}^2)}} \exp\left(\frac{-(g^* - g)^2}{2(\sigma_{\mathcal{D}}^2 + \sigma_{\mathcal{H}}^2)}\right)$$

Derivation of Equation 14. When taking the expectation of $\mathbb{E}[\beta a_{\mathcal{H}}]$, we utilize LOTUS to obtain:

$$\mathbb{E}[\beta a_{\mathcal{H}}] = \int_{-\infty}^{\infty} \frac{a_{\mathcal{H}}}{\sqrt{2\pi\sigma_{\mathcal{D}}^2}} \exp\left(-\frac{1}{2}\frac{\left(a_{\mathcal{H}} - (g - s)\right)^2}{\sigma_{\mathcal{D}}^2}\right) \cdot \frac{1}{\sqrt{2\pi\sigma_{\mathcal{H}}^2}} \exp\left(-\frac{1}{2}\frac{\left(a_{\mathcal{H}} - (g^* - s)\right)^2}{\sigma_{\mathcal{H}}^2}\right) da_{\mathcal{H}}$$

Upon simplification we obtain:

$$\mathbb{E}[\beta a_{\mathcal{H}}] = \frac{1}{2\pi\sigma_{\mathcal{D}} \cdot \sigma_{\mathcal{H}}} \int_{-\infty}^{\infty} a_{\mathcal{H}} \cdot \exp\left(-\frac{1}{2}\frac{\left(a_{\mathcal{H}} - (g - s)\right)^2}{\sigma_{\mathcal{D}}^2} - \frac{1}{2}\frac{\left(a_{\mathcal{H}} - (g^* - s)\right)^2}{\sigma_{\mathcal{H}}^2}\right) da_{\mathcal{H}} \quad (21)$$

By solving for the integral in Equation (21) we obtain Equation (14).

Derivation of Equation 17. Recall that:

$$\beta(s, \boldsymbol{a_{\mathcal{H}}}) = \frac{1}{(2\pi)^{d/2}|\Sigma_{\mathcal{D}}|^{1/2}} \exp\left(-\frac{1}{2} \cdot (\boldsymbol{a_{\mathcal{H}}} - (\boldsymbol{g^*} - \boldsymbol{s}))^T \Sigma_{\mathcal{D}}^{-1}(\boldsymbol{a_{\mathcal{H}}} - (\boldsymbol{g^*} - \boldsymbol{s}))\right) \quad (22)$$

We rewrite Equation (22) using the canonical parameterization as:

$$\beta(s, \boldsymbol{a_{\mathcal{H}}}) = \exp\left(\xi_{\mathcal{D}} + \boldsymbol{\eta}_{\mathcal{D}}^T \cdot \boldsymbol{a_{\mathcal{H}}} - \frac{1}{2}\boldsymbol{a}_{\mathcal{H}}^T \boldsymbol{\Lambda}_{\mathcal{D}} \boldsymbol{a_{\mathcal{H}}}\right)$$

where $\boldsymbol{\Lambda}_{\mathcal{D}} = \Sigma_{\mathcal{D}}^{-1}$, $\boldsymbol{\eta}_{\mathcal{D}} = \Sigma_{\mathcal{D}}^{-1}(\boldsymbol{g^*} - \boldsymbol{s})$, and $\xi_{\mathcal{D}} = -\frac{1}{2}(d \log 2\pi - \log|\boldsymbol{\Lambda}_{\mathcal{D}}| + \boldsymbol{\eta}_{\mathcal{D}}^T \boldsymbol{\Lambda}_{\mathcal{D}}^{-1} \boldsymbol{\eta}_{\mathcal{D}})$. To compute $\mathbb{E}(\beta)$, we utilize LOTUS:

$$\mathbb{E}[\beta] = \int_{-\infty}^{\infty} \exp\left(\xi_{\mathcal{D}} + \boldsymbol{\eta}_{\mathcal{D}}^T \cdot \boldsymbol{a}_{\mathcal{H}} - \frac{1}{2}\boldsymbol{a}_{\mathcal{H}}^T \Lambda_{\mathcal{D}} \boldsymbol{a}_{\mathcal{H}}\right) \cdot \exp\left(\xi_{\mathcal{H}} + \boldsymbol{\eta}_{\mathcal{H}}^T \cdot \boldsymbol{a}_{\mathcal{H}} - \frac{1}{2}\boldsymbol{a}_{\mathcal{H}}^T \Lambda_{\mathcal{H}} \boldsymbol{a}_{\mathcal{H}}\right)$$

Upon further simplification we obtain:

$$\mathbb{E}[\beta] = \int_{-\infty}^{\infty} \exp\left(\xi_D + \xi_H - \xi + \xi + \boldsymbol{\eta}^T \boldsymbol{a}_h - \frac{1}{2}\boldsymbol{a}_h^T \Lambda \boldsymbol{a}_h\right)$$

where $\Lambda = \Lambda_D + \Lambda_H$, $\boldsymbol{\eta} = \boldsymbol{\eta}_D + \boldsymbol{\eta}_H$, and $\xi = -\frac{1}{2}(d \log 2\pi - \log |\Lambda| + \boldsymbol{\eta}^T \Lambda^{-1} \boldsymbol{\eta})$. With substitution and simplification we obtain:

$$\mathbb{E}[\beta] = \frac{1}{(2\pi)^{d/2}|\Sigma_{\mathcal{D}} + \Sigma_{\mathcal{H}}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{g} - \boldsymbol{g}^*)^T (\Sigma_{\mathcal{D}} + \Sigma_{\mathcal{H}})^{-1}(\boldsymbol{g} - \boldsymbol{g}^*)\right) \tag{23}$$

Equation (23) can be simplified and rewritten as Equation (17).

Proof for Theorem 4. Because we have assumed that $\mathbb{E}[\beta] < \beta_{max}$, we set $\beta = \beta(s, \boldsymbol{a}_{\mathcal{H}})$. Note that β depends upon $\boldsymbol{a}_{\mathcal{H}}$. We transform $\beta(s, \boldsymbol{a}_{\mathcal{H}})$ using the canonical parameterization and apply LOTUS to compute the expectation as:

$$\mathbb{E}[\beta \boldsymbol{a}_{\mathcal{H}}] = \int_{-\infty}^{\infty} \boldsymbol{a}_{\mathcal{H}} \exp\left(\xi_{\mathcal{D}} + \boldsymbol{\eta}_{\mathcal{D}}^T \cdot \boldsymbol{a}_{\mathcal{H}} - \frac{1}{2}\boldsymbol{a}_{\mathcal{H}}^T \Lambda_{\mathcal{D}} \boldsymbol{a}_{\mathcal{H}}\right) \cdot \exp\left(\xi_{\mathcal{H}} + \boldsymbol{\eta}_{\mathcal{H}}^T \cdot \boldsymbol{a}_{\mathcal{H}} - \frac{1}{2}\boldsymbol{a}_{\mathcal{H}}^T \Lambda_{\mathcal{H}} \boldsymbol{a}_{\mathcal{H}}\right) \tag{24}$$

Upon further simplification and reparameterization we obtain:

$$\mathbb{E}[\beta \boldsymbol{a}_{\mathcal{H}}] = \frac{(\Sigma_H + \Sigma_D)^{-1}(\Sigma_D(\boldsymbol{g}^* - \boldsymbol{s}) + \Sigma_H(\boldsymbol{g} - \boldsymbol{s}))}{(2\pi)^{d/2}|\Sigma_D + \Sigma_H|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{g} - \boldsymbol{g}^*)^T (\Sigma_D + \Sigma_H)^{-1}(\boldsymbol{g} - \boldsymbol{g}^*)\right) \tag{25}$$

This can be rewritten as:

$$\mathbb{E}[\beta \boldsymbol{a}_{\mathcal{H}}] = \frac{\Sigma^{-1}(\Sigma_{\mathcal{D}}(\boldsymbol{g}^* - \boldsymbol{s}) + \Sigma_{\mathcal{H}}(\boldsymbol{g} - \boldsymbol{s}))}{\sqrt{(2\pi)^d \det \Sigma}} \cdot \exp\left(-\frac{1}{2}\|\boldsymbol{g}^* - \boldsymbol{g}\|_{\Sigma^{-1}}^2\right) \tag{26}$$

where $\Sigma = \Sigma_{\mathcal{D}} + \Sigma_{\mathcal{H}}$. Substituting Equation (17) and Equation (26) back into Equation (16), we have $\mathbb{E}[\dot{V}(t)] < 0$ when:

$$\|\boldsymbol{g}^* - \boldsymbol{s}\|^2 > \mathbb{E}[\beta] \cdot (\boldsymbol{g}^* - \boldsymbol{s})^T \Sigma^{-1} \Sigma_{\mathcal{D}}(\boldsymbol{g}^* - \boldsymbol{g}) \tag{27}$$

Here we apply the Cauchy–Schwarz inequality to obtain:

$$\|\boldsymbol{g}^* - \boldsymbol{s}\|^2 > \mathbb{E}[\beta]\|\boldsymbol{g}^* - \boldsymbol{s}\| \cdot \|\Sigma^{-1}\Sigma_{\mathcal{D}}(\boldsymbol{g}^* - \boldsymbol{g})\| \tag{28}$$

From the spectral theorem we know that $\|Ax\| \le \lambda_{max}(A)\|x\|$, where $\lambda_{max}(A)$ is the largest eigenvalue of the positive semidefinite matrix A. We substitute this inequality in Equation (26) to obtain a more relaxed constraint. Specifically, we find that $\mathbb{E}[\dot{V}(t)] < 0$ if the following inequality holds:

$$\|\boldsymbol{g}^* - \boldsymbol{s}\|^2 > \|\boldsymbol{g}^* - \boldsymbol{s}\| \cdot \lambda \mathbb{E}[\beta] \cdot \|\boldsymbol{g}^* - \boldsymbol{g}\| \tag{29}$$

where λ is the maximum eigenvalue of $\Sigma^{-1}\Sigma_D$. Rearranging this result yields Equation (19). We conclude that $\mathbb{E}[\dot{V}(t)] < 0$ when Equation (19) is satisfied, and it therefore follows that the human-robot system is uniformly ultimately bounded.

Ultimate Bounds. In our proofs for Theorems 1–4 we have shown that the SARI system is uniformly ultimately bounded, and we have listed the ultimate bounds. However, we have not formally demonstrated *why* Equations (12), (13), (18), and (19) are the ultimate bounds. Here we provide a more rigorous derivation for these results. We focus on Theorem 4, but the same approach applies to each of our Theorems.

Recall from Equation (15) that the Lyapunov candidate function depends on the error e, and remember that $e = g^* - s$. Let α_1 and α_2 be two class κ functions such that:

$$\alpha_1(\|e\|) \leq V(e) \leq \alpha_2(\|e\|) \tag{30}$$

Here the ultimate bound on error e can be taken as [38]:

$$b = \alpha_1^{-1}(\alpha_2(\mu)) \tag{31}$$

where $\mu > 0$ is selected such that $\mathbb{E}[\dot{V}(t)] < 0$ for all $\|e\| > \mu$. Looking back at Equation (19) and the previous proof from the Appendix, we have already identified $\mu = \lambda\mathbb{E}[\beta] \cdot \|g^* - g\|$. We now propose $\alpha_1 = \alpha_2 = \frac{1}{2}\|e\|^2$. These choices are valid because (a) they satisfy Equation (30) and (b) they are class κ functions. Plugging μ, α_1, and α_2 back into Equation (31), the ultimate bound is:

$$b = \lambda\mathbb{E}[\beta] \cdot \|g^* - g\| \tag{32}$$

Intuitively, this result means that the expected error between the human's new goal g^* and the robot's state s will eventually become smaller than b, and will remain smaller than b for the rest of the interaction [61].

9.2 Implementation Details

Our code can be found here: https://github.com/VT-Collab/repeated-shared-autonomy.

Data collection. Across all three user studies we record the robot's and the human's actions. Depending on the task, we collect the robot's joint positions, robot's Cartesian position, state of the gripper, current mode of teleoperation (rotation or translation), and current mode of operation (fast or slow mode). In-person users utilized the joysticks on a Logitech Gamepad F310 controller to provide their inputs. Since the controller is equipped with only two joysticks, users can toggle between translational and rotational inputs using one of the buttons on the controller. Remote users used a web-based GUI that had individual buttons that could be used to command translational and rotational velocities. This GUI also had additional buttons to open and close the gripper, and to speed up or slow down the robot. The inputs from the joystick and the web GUI are converted into a 6-D vector of velocity inputs and stored as the human's current action.

Data augmentation. We augment the data we receive from the demonstrations using Gaussian noise. We create 5 additional samples for each sample we collect by injecting Gaussian noise with zero mean and a small variance. We use this augmented data to train our method as well as all the baselines that require human demonstrations for training.

Deformations for the discriminator. To create samples that represent unseen behavior for our discriminator, we randomly apply a small force to the demonstrations we collected from the human. This force alters the start, end and the shape of the initial trajectory. We use previous work by [40] to generate these deformations. Additional details on our implementation can be found in our code.

Network architecture. We use fully connected networks for the encoder, decoder and the discriminator in our method. While we varied the number of hidden layers and the number of neurons in each layer throughout the project, we found the best results when our encoder consisted of 5 hidden layers, the decoder consisted of 4 hidden layers, and the discriminator consisted of 4 hidden layers. Additional information on our specific implementations can be found in the code repository.

Computational Requirements. Throughout the project we use various versions of Pytorch and Python in our environment. Most of our experiments were run on an Intel PC with an i7-8559U processor and 32GB of RAM. We store the human interaction data as Python pickle files and each interaction requires 72kB of memory. Our Pytorch models were also saved as pickle files and required 42.7kB of memory. No GPU was used for our experiments.

Baselines. For DAgger-based baselines we use fully connected networks similar to our method. For DropoutDagger we use dropout probability of 0.1 and for EnsembleDagger we train and use an ensemble of 20 models. To obtain the network's confidence (β) we scale the variance within the model's action for DropoutDagger and variance between models for EnsembleDagger with a constant value. This constant is obtained by tuning the robot's performance while performing a learned task. For CASA we use a combination of Guided Cost Learning (GCL) and Soft-Actor Critic (SAC) [24]. We implement SAC in-house and modify a pre-existing repository for training using GCL. The original repository can be found here: https://github.com/NinaWie/guided-cost-learning, and our specific implementation can be found in our code repository.

REFERENCES

[1] 2019 (accessed Feb 9, 2023). *Jaco Is a Low-Power Robot Arm That Hooks to Your Wheelchair*. IEEE Spectrum.

[2] 2021 (accessed Feb 9, 2023). *Jaco Assistive Robot User Guide*. https://assistive.kinovarobotics.com/uploads/EN-UG-007-Jaco-user-guide-R05.pdf.

[3] Henny Admoni and Siddhartha Srinivasa. 2016. Predicting user intent through eye gaze for shared autonomy. In *AAAI*.

[4] Brenna D Argall. 2018. Autonomy in rehabilitation robotics: An intersection. *Annual Review of Control, Robotics, and Autonomous Systems* 1 (2018), 441–463.

[5] Reuben M Aronson, Thiago Santini, Thomas C Kübler, Enkelejda Kasneci, Siddhartha Srinivasa, and Henny Admoni. 2018. Eye-hand behavior in human-robot shared manipulation. In *ACM/IEEE International Conference on Human-Robot Interaction*. IEEE, 4–13.

[6] Suneel Belkhale, Ethan K Gordon, Yuxiao Chen, Siddhartha Srinivasa, Tapomayukh Bhattacharjee, and Dorsa Sadigh. 2022. Balancing efficiency and comfort in robot-assisted bite transfer. In *International Conference on Robotics and Automation*. 4757–4763.

[7] Tapomayukh Bhattacharjee, Ethan K Gordon, Rosario Scalise, Maria E Cabrera, Anat Caspi, Maya Cakmak, and Siddhartha S Srinivasa. 2020. Is more autonomy always better? Exploring preferences of users with mobility impairments in robot-assisted feeding. In *ACM/IEEE International Conference on Human-Robot Interaction*. 181–190.

[8] Jonathan Bragg and Emma Brunskill. 2020. Fake it till you make it: Learning-compatible performance support. In *Uncertainty in Artificial Intelligence*. 915–924.

[9] Alexander Broad, Ian Abraham, Todd Murphey, and Brenna Argall. 2020. Data-driven Koopman operators for model-based shared control of human–machine systems. *The International Journal of Robotics Research* 39, 9 (2020), 1178–1195.

[10] Connor Brooks and Daniel Szafir. 2019. Balanced information gathering and goal-oriented actions in shared autonomy. In *ACM/IEEE International Conference on Human-Robot Interaction*. 85–94.

[11] Steven W Brose, Douglas J Weber, Ben A Salatin, Garret G Grindle, Hongwu Wang, Juan J Vazquez, and Rory A Cooper. 2010. The role of assistive robotics in the lives of persons with disability. *American Journal of Physical Medicine & Rehabilitation* 89, 6 (2010), 509–521.

[12] Young Sang Choi, Travis Deyle, Tiffany Chen, Jonathan D Glass, and Charles C Kemp. 2009. A list of household objects for robotic retrieval prioritized by people with ALS. In *IEEE International Conference on Rehabilitation Robotics*. 510–517.

[13] Alexander Clegg, Zackory Erickson, Patrick Grady, Greg Turk, Charles C Kemp, and C Karen Liu. 2020. Learning to collaborate from simulation for robot-assisted dressing. *IEEE Robotics and Automation Letters* 5, 2 (2020), 2746–2753.

[14] Erwin Coumans and Yunfei Bai. 2016–2021. PyBullet, a Python module for physics simulation for games, robotics and machine learning. http://pybullet.org.

[15] Anca D Dragan and Siddhartha S Srinivasa. 2013. A policy-blending formalism for shared control. *The International Journal of Robotics Research* 32, 7 (2013), 790–805.

[16] Yuqing Du, Stas Tiomkin, Emre Kiciman, Daniel Polani, Pieter Abbeel, and Anca Dragan. 2020. AvE: Assistance via empowerment. *Advances in Neural Information Processing Systems* 33 (2020), 4560–4571.

[17] Zackory Erickson, Vamsee Gangaram, Ariel Kapusta, C Karen Liu, and Charles C Kemp. 2020. Assistive gym: A physics simulation framework for assistive robotics. In *IEEE International Conference on Robotics and Automation*. 10169–10176.

[18] Ryan Feng, Youngsun Kim, Gilwoo Lee, Ethan K Gordon, Matt Schmittle, Shivaum Kumar, Tapomayukh Bhattacharjee, and Siddhartha S Srinivasa. 2019. Robot-assisted feeding: Generalizing skewering strategies across food items on a realistic plate. In *International Symposium on Robotics Research*.

[19] Chelsea Finn, Sergey Levine, and Pieter Abbeel. 2016. Guided cost learning: Deep inverse optimal control via policy optimization. In *International Conference on Machine Learning*. 49–58.

[20] Matthew Fontaine and Stefanos Nikolaidis. 2020. A quality diversity approach to automatically generating human-robot interaction scenarios in shared autonomy. In *Robotics: Science and Systems*.

[21] Stefan Fuchs and Anna Belardinelli. 2021. Gaze-based intention estimation for shared autonomy in pick-and-place tasks. *Frontiers in Neurorobotics* 15 (2021), 647930.

[22] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

[23] Deepak Gopinath, Siddarth Jain, and Brenna D Argall. 2016. Human-in-the-loop optimization of shared autonomy in assistive robotics. *IEEE Robotics and Automation Letters* 2, 1 (2016), 247–254.

[24] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*. PMLR, 1861–1870.

[25] Jerry Zhi-Yang He, Aditi Raghunathan, Daniel S Brown, Zackory Erickson, and Anca D Dragan. 2022. Learning Representations that Enable Generalization in Assistive Tasks. *arXiv preprint arXiv:2212.03175* (2022).

[26] Laura V Herlant, Rachel M Holladay, and Siddhartha S Srinivasa. 2016. Assistive teleoperation of robot arms via automatic time-optimal mode switching. In *ACM/IEEE International Conference on Human Robot Interaction*. 35–42.

[27] Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S Brown, and Ken Goldberg. 2022. ThriftyDAgger: Budget-Aware Novelty and Risk Gating for Interactive Imitation Learning. In *Conference on Robot Learning*. 598–608.

[28] Siddarth Jain and Brenna Argall. 2019. Probabilistic human intent recognition for shared autonomy in assistive robotics. *ACM Transactions on Human-Robot Interaction* 9, 1 (2019), 1–23.

[29] Siddarth Jain, Ali Farshchiansadegh, Alexander Broad, Farnaz Abdollahi, Ferdinando Mussa-Ivaldi, and Brenna Argall. 2015. Assistive robotic manipulation through shared autonomy and a body-machine interface. In *IEEE International Conference on Rehabilitation Robotics*. 526–531.

[30] Snehal Jauhri, Carlos Celemin, and Jens Kober. 2021. Interactive Imitation Learning in State-Space. In *Conference on Robot Learning*. 682–692.

[31] Shervin Javdani, Henny Admoni, Stefania Pellegrinelli, Siddhartha S Srinivasa, and J Andrew Bagnell. 2018. Shared autonomy via hindsight optimization for teleoperation and teaming. *The International Journal of Robotics Research* 37, 7 (2018), 717–742.

[32] Hong Jun Jeon, Dylan P Losey, and Dorsa Sadigh. 2020. Shared autonomy with learned latent actions. In *Robotics: Science and Systems*.

[33] Ananth Jonnavittula and Dylan P Losey. 2021. I know what you meant: Learning human objectives by (under) estimating their choice set. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2747–2753.

[34] Ananth Jonnavittula and Dylan P Losey. 2021. Learning to share autonomy across repeated interaction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 1851–1858.

[35] Ananth Jonnavittula and Dylan P Losey. 2022. Communicating robot conventions through shared autonomy. In *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 7423–7429.

[36] Rico Jonschkowski and Oliver Brock. 2015. Learning state representations with robotic priors. *Autonomous Robots* 39, 3 (2015), 407–428.

[37] Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J Kochenderfer. 2019. HG-DAgger: Interactive imitation learning with human experts. In *IEEE International Conference on Robotics and Automation*. 8077–8083.

[38] Hassan K Khalil. 2002. *Nonlinear Systems*. Vol. 3. Prentice-Hall.

[39] Dylan P Losey, Hong Jun Jeon, Mengxi Li, Krishnan Srinivasan, Ajay Mandlekar, Animesh Garg, Jeannette Bohg, and Dorsa Sadigh. 2022. Learning latent actions to control assistive robots. *Autonomous Robots* 46, 1 (2022), 115–147.

[40] Dylan P Losey and Marcia K O'Malley. 2017. Trajectory deformations from physical human–robot interaction. *IEEE Transactions on Robotics* 34, 1 (2017), 126–138.

[41] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. 2020. Learning latent plans from play. In *Conference on Robot Learning*. 1113–1132.

[42] Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. 2020. Human-in-the-loop imitation learning using remote teleoperation. *arXiv preprint arXiv:2012.06733* (2020).

[43] Shaunak A Mehta, Yeunhee Kim, Joshua Hoegerman, Michael D Bartlett, and Dylan P Losey. 2023. RISO: Combining Rigid Grippers with Soft Switchable Adhesives. In *2023 IEEE International Conference on Soft Robotics (RoboSoft)*. 1–8.

[44] Shaunak A Mehta and Dylan P Losey. 2022. Unified learning from demonstrations, corrections, and preferences during physical human-robot interaction. *arXiv preprint arXiv:2207.03395* (2022).

[45] Shaunak A Mehta, Sagar Parekh, and Dylan P Losey. 2022. Learning Latent Actions without Human Demonstrations. In *IEEE International Conference on Robotics and Automation*.

[46] Kunal Menda, Katherine Driggs-Campbell, and Mykel J Kochenderfer. 2017. Dropoutdagger: A Bayesian approach to safe imitation learning. *arXiv preprint arXiv:1709.06166* (2017).

[47] Kunal Menda, Katherine Driggs-Campbell, and Mykel J Kochenderfer. 2019. Ensembledagger: A Bayesian approach to safe imitation learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5041–5048.

[48] Katharina Muelling, Arun Venkatraman, Jean-Sebastien Valois, John Downey, Jeffrey Weiss, Shervin Javdani, Martial Hebert, Andrew B Schwartz, Jennifer L Collinger, and J Andrew Bagnell. 2015. Autonomy infused teleoperation with application to BCI manipulation. In *Robotics: Science and Systems*.

[49] Benjamin A Newman, Reuben M Aronson, Siddartha S Srinivasa, Kris Kitani, and Henny Admoni. 2018. Harmonic: A multimodal dataset of assistive human-robot collaboration. *arXiv preprint arXiv:1807.11154* (2018).

[50] Stefanos Nikolaidis, Yu Xiang Zhu, David Hsu, and Siddhartha Srinivasa. 2017. Human-robot mutual adaptation in shared autonomy. In *ACM/IEEE International Conference on Human-Robot Interaction*. 294–302.

[51] T Osa, J Pajarinen, G Neumann, JA Bagnell, P Abbeel, and J Peters. 2018. An Algorithmic Perspective on Imitation Learning. *Foundations and Trends in Robotics* 7, 1-2 (2018), 1–179.

[52] Daehyung Park, Yuuna Hoshi, Harshal P Mahajan, Ho Keun Kim, Zackory Erickson, Wendy A Rogers, and Charles C Kemp. 2020. Active robot-assisted feeding with a general-purpose mobile manipulator: Design, evaluation, and lessons learned. *Robotics and Autonomous Systems* 124 (2020), 103344.

[53] Kavya Puthuveetil, Charles C. Kemp, and Zackory Erickson. 2022. Bodies Uncovered: Learning to Manipulate Real Blankets Around People via Physics Simulations. *IEEE Robotics and Automation Letters* 7, 2 (2022), 1984–1991.

[54] Calvin Z Qiao, Maram Sakr, Katharina Muelling, and Henny Admoni. 2021. Learning from demonstration for real-time user goal prediction and shared assistive control. In *IEEE International Conference on Robotics and Automation*. IEEE, 3270–3275.

[55] Siddharth Reddy, Anca D Dragan, and Sergey Levine. 2018. Shared autonomy via deep reinforcement learning. In *Robotics: Science and Systems*.

[56] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*. 627–635.

[57] Charles Schaff and Matthew R Walter. 2020. Residual policy learning for shared autonomy. In *Robotics: Science and Systems*.

[58] Mark J Schervish and Morris H DeGroot. 2014. *Probability and Statistics*. Pearson Education.

[59] Mario Selvaggio, Marco Cognetti, Stefanos Nikolaidis, Serena Ivaldi, and Bruno Siciliano. 2021. Autonomy in physical human-robot interaction: A brief survey. *IEEE Robotics and Automation Letters* 6, 4 (2021), 7989–7996.

[60] Jonathan Spencer, Sanjiban Choudhury, Matthew Barnes, Matthew Schmittle, Mung Chiang, Peter Ramadge, and Sidd Srinivasa. 2022. Expert intervention learning. *Autonomous Robots* 46, 1 (2022), 99–113.

[61] Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. 2006. *Robot Modeling and Control*. Vol. 3. Wiley New York.

[62] Danielle M Taylor. 2018. *Americans With Disabilities: 2014*. US Census Bureau.

[63] Jiakai Zhang and Kyunghyun Cho. 2017. Query-efficient imitation learning for end-to-end autonomous driving. In *AAAI*.

[64] Matthew Zurek, Andreea Bobu, Daniel S Brown, and Anca D Dragan. 2021. Situational confidence assistance for lifelong shared autonomy. In *IEEE International Conference on Robotics and Automation*. 2783–2789.