Controlling Assistive Robots with Learned Latent Actions

Dylan P. Losey, Krishnan Srinivasan, Ajay Mandlekar, Animesh Garg, and Dorsa Sadigh

***Abstract*— Assistive robots enable users with disabilities to perform everyday tasks without relying on a caregiver. Unfortunately, the very dexterity that makes these arms useful also makes them challenging to control: the robot has more degrees-of-freedom (DoFs) than the human can directly coordinate with a handheld joystick. Our insight is that we can make assistive robots *easier* for humans to control by learning *latent actions*. Latent actions provide a low-DoF embedding of high-DoF robot behavior: for example, one latent dimension might guide the robot arm along a pouring motion. Because these latent actions are low-dimensional, they can be controlled by the human end-user to fluidly teleoperate the robot. In this paper, we design a teleoperation algorithm for assistive robots that *learns* intuitive latent dimensions from task demonstrations. We formulate the controllability, consistency, and scaling properties that intuitive latent actions should have, and evaluate how different low-dimensional embeddings capture these properties. Finally, we conduct two user studies on a robotic arm to compare our latent action approach to state-of-the-art shared autonomy baselines and direct end-effector teleoperation. Participants completed the assistive feeding and cooking tasks more quickly and efficiently when leveraging our latent actions, and also reported that latent actions made the task easier to perform.**

***Index Terms*— Physically assistive devices, cognitive human-robot interaction, human-centered robotics**

I. Introduction

For the nearly one million American adults that need assistance when eating, taking a bite of food or pouring a glass of water can present a significant challenge [1]. Wheelchair-mounted robotic arms and other physically assistive robots provide highly dexterous tools for performing these tasks without relying on help from a caregiver. In order to be effective, however, these assistive robots must be *easily controllable* by their users.

Consider a person using a robotic arm to pour water into a glass. This person teleoperates the robot using a joystick, and must carefully position the robot's end effector above the cup before changing its orientation to pour the water. The human's input is—by necessity—*low-dimensional*. But the robot arm is *high-dimensional*: it has many degrees-of-freedom (DoFs), and the human needs to precisely coordinate all of these interconnected DoFs to pour the water without spilling. In practice, this can be quite challenging due to the unintuitive mapping from low-dimensional human inputs to high-dimensional robot actions [2], [3].

Current approaches solve this problem when the human's goals are *discrete*: e.g., when the robot should pour water into either glass A or glass B. By contrast, we here propose an approach for controlling the robot in *continuous* spaces using low-DoF actions learned from data. Our insight is that:

High-DoF robot actions can often be embedded *into intuitive, human-controllable, and low-DoF latent actions*

Returning to our pouring example: the human wants the robot arm to (a) carry the cup level with the table and (b) perform a pouring action. Intuitively, this should be reflected in the joystick inputs: one DoF should cause the robot to move the cup forward and backwards in a plane, while the other DoF should make the robot pour more or less water (see Fig. 1).

We explore methods for *learning* these low-DoF latent actions from task-specific training data. We envision settings where the robot has access to demonstrations of related tasks (potentially provided by the caregiver), and the user—in an online setting—wants to control the robot to perform a new task: e.g., now the cup is located in a different place, and the person only wants half a glass of water. In practice, we find that some models result in expressive and intuitive latent actions, and that users can control robots equipped with these models to complete feeding and cooking tasks.

Overall, we make the following contributions:

Formalizing Desirable Properties of Latent Actions. We formally specify a list of properties that user-friendly latent actions must satisfy. This includes *controllability*, i.e., there must be a sequence of latent actions that move the robot to the desired state, and *consistency*, i.e., the robot should have the same behavior under a learned latent action in all states.

Learning Latent Actions through Autoencoders. We learn latent actions using five different autoencoder models, and compare how these models perform with respect to our desired properties. We find that autoencoders conditioned on the robot's current state can accurately reconstruct high-DoF actions from human-controllable, low-DoF inputs.

Evaluating Latent Actions with User Studies. We implement our approach on a Fetch robot arm, and compare to state-of-the-art shared autonomy and teleoperation baselines in two user studies. We find that—during feeding and cooking tasks—using latent actions to control the robot results in improved objective and subjective performance.

Our work demonstrates the role of learning latent actions in improving the user's experience and capability when they are in close collaboration with assistive robots.

II. Related Work

In this paper we leverage learning techniques to identify low-DoF latent actions for assistive robots. Our approach provides an intelligent mapping from the human's joystick

The authors are with the Stanford Intelligent and Interactive Autonomous Systems Group (ILIAD), Department of Computer Science, Stanford University, Stanford, CA 94305. (e-mail: dlosey@stanford.edu)

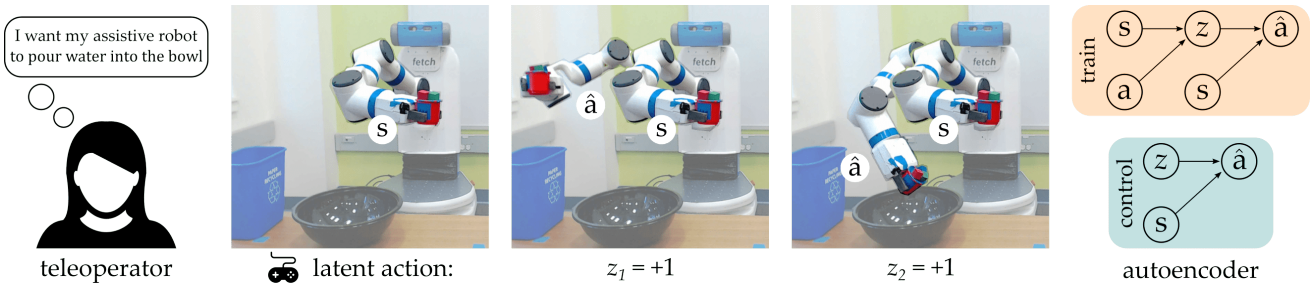

Fig. 1. Human teleoperating a robot arm using latent actions. Because the robot has more DoFs than the human can directly control, we leverage low-DoF embeddings to learn a latent space z for the user to intuitively interact with. Here the robot has been trained on demonstrations of pouring tasks, and learns a 2-DoF latent space. The first latent dimension z_1 moves the cup level with the table, and the second latent dimension z_2 tilts the cup. We explore how conditional autoencoders—such as the one shown on right—can be leveraged to learn these intuitive and human-controllable latent actions. We train an encoder that finds our low-DoF embedding z given the state s and high-DoF action a. The decoder then recovers a high-DoF action $\hat{a}$ based on z and s. During control, the low-DoF human input z is enough to reconstruct their intended high-DoF, continuous robot action $\hat{a}$ conditioned on the current state s.

inputs to the robot's arm motion. Prior works have separately addressed (a) using predefined, naive input mappings and then blending human inputs with corrective robot actions, or (b) learning latent spaces for autonomously robots that are acting without a human in the loop.

Shared Autonomy. Under shared autonomy, the robot combines the user input with autonomous assistance: this method has been applied to improve human control over wheelchairs and robotic arms [3]–[6]. Recent works focus on settings where the human wants their assistive arm to reach a goal (e.g., pick up a cup) [7]–[11]. Within these works, the robot maintains a belief over possible goals, and updates this belief based on the human's inputs [7], [8]. The robot increasingly assists the human as it becomes confident about their goal.

Our paper is most related to shared autonomy research by Reddy *et al.* [12], where the robot learns a mapping between humans inputs and their intended actions using reinforcement learning; however, in [12] the human inputs are the same dimension as the robot action, and thus there is no need to learn a dimensionality reduction mapping. Our work enables a more inuitive control interface for the human by requiring only a few degrees-of-freedom.

Learning Latent Representations. To identify low-DoF embeddings of complex state-space models, we turn to works that learn latent representations from data. Recent research has learned latent dynamics [13], trajectories [14], plans [15], policies [16], and actions for reinforcement learning [17]. These methods typically leverage autoencoder models [18], [19], which learn a latent space without supervision.

Here, we leverage autoencoders to learn a consistent and controllable latent representation for assistive robotics. Previous teleoperation literature has explored principal component analysis (PCA) for reducing the user's input dimension [20], [21]. We will compare our method to this PCA baseline.

III. Problem Statement

We formulate a task as a discrete-time Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma, \rho_0)$. Here $\mathcal{S} \subseteq \mathbb{R}^n$ is the state space, $\mathcal{A} \subseteq \mathbb{R}^m$ is the action space, $\mathcal{T}(s, a)$ is the transition function, $R(s) = \mathbb{1}\{\text{task is solved in } s\}$ is a sparse reward function that indicates task success, $\gamma \in [0, 1)$ is the discount factor, and $\rho_0(\cdot)$ is the initial state distribution. We assume access to a dataset of task demonstrations, and seek to learn the latent action space by leveraging this dataset. Formally, we have a dataset of state-action pairs $\mathcal{D} = \{(s_0, a_0), (s_1, a_1), \ldots\}$, and seek to learn a latent action space $\mathcal{Z} \subset \mathbb{R}^d$ that is of lower dimension than the original action space ($d < m$), along with a function $\phi : \mathcal{Z} \times \mathcal{S} \mapsto \mathcal{A}$ that maps latent actions to robot actions.

Recall our motivating example, where the human is leveraging latent actions to make their assistive robot pour water. There are several *properties* that the human expects latent actions to have: e.g., the human should be able to guide the robot by smoothly changing the joystick direction, and the robot should never abruptly become more sensitive to the human's inputs. In what follows, we *formalize* the properties that make latent actions intuitive. These properties will guide our approach, and provide a principled way of assessing the usefulness of latent actions with humans in the loop.

Latent Controllability. Let $s_i, s_j \in \mathcal{D}$ be two states from the dataset of demonstrations, and let $s_1, s_2, ..., s_K$ be the sequence of states that the robot visits when starting in state $s_0 = s_i$ and taking latent actions $z_1, ..., z_K$. The robot transitions between the visited states using the learned latent space: $s_k = \mathcal{T}(s_{k-1}, \phi(z_{k-1}, s_{k-1}))$. Formally, we say that a latent action space $\mathcal{Z}$ is *controllable* if for every such pairs of states (s_i, s_j) there exists a sequence of latent actions $\{z_k\}_{k=1}^K, z_k \in \mathcal{Z}$ such that $s_j = s_K$. In other words, a latent action space is controllable if it can move the robot between pairs of start and goal states from the dataset.

Latent Consistency. We define a latent action space $\mathcal{Z}$ as *consistent* if the same latent action $z \in \mathcal{Z}$ has a similar effect on how the robot behaves in nearby states. We formulate this similarity via a task-dependent metric d_M: e.g., in pouring tasks d_M could measure the orientation of the robot's end-effector. Applying this metric, consistent latent actions should satisfy: $d_M(\mathcal{T}(s_1, \phi(z, s_1)), \mathcal{T}(s_2, \phi(z, s_2))) < \epsilon$ for $\|s_1 - s_2\| < \delta$ for some $\epsilon, \delta > 0$.

Latent Scaling. Finally, a latent action space $\mathcal{Z}$ is *scalable* if applying larger latent actions leads to larger changes in state. In other words, we would like $\|s - s'\| \to \infty$ as $\|z\| \to \infty$, where $s' = \mathcal{T}(s, \phi(z, s))$.

IV. METHODS

Now that we have formally introduced the properties that a user-friendly latent space should satisfy, we will explore low-DoF embeddings that capture these properties. We are interested in models that balance *expressiveness* with *intuition*: the embedding must reconstruct high-DoF actions while remaining controllable, consistent, and scalable. We assert that only models which reason over the robot's state when decoding the human's inputs can accurately and intuitively interpret the latent action.

A. Models

Reconstructing Actions. Let us return to our pouring example: when the person applies a low-DoF joystick input, the robot completes a high-DoF action. We use *autoencoders* to move between these low- and high-DoF action spaces. Define $\psi : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{Z}$ as an *encoder* that embeds the robot's behavior into a latent space, and define $\phi' : \mathcal{Z} \rightarrow \mathcal{A}$ as a *decoder* that reconstructs a high-DoF robot action $\hat{a}$ from this latent space (see Fig. 1). To encourage models to learn latent actions that accurately reconstruct high-DoF robot behavior, we incorporate the reconstruction error $\|a - \hat{a}\|^2$ into the model's loss function. Both PCA and autoencoder (AE) models minimize this reconstruction error.

Regularizing Latent Actions. When the user slightly tilts the joystick, the robot should not suddenly pour its entire glass of water. To better ensure this consistency and scalability, we incorporate a *normalization* term into the model's loss function. Let us define $\psi : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}^d \times \mathbb{R}^d_+$ as an encoder that outputs the mean μ and covariance σ of the latent action space. We penalize the divergence between this latent action space and a normal distribution: $KL(\mathcal{N}(\mu, \sigma) \parallel \mathcal{N}(0, 1))$. Variational autoencoder (VAE) models trade-off between this normalization term and reconstruction error [18], [19].

Conditioning on State. Importantly, we recognize that the *meaning* of the human's joystick input often depends on what the robot is doing. When the robot is holding a glass, pressing down on the joystick indicates that the robot should pour water; but—when the robot's gripper is empty—it does not make sense for the robot to pour! So that robots can associate meanings with latent actions, we *condition* the interpretation of the latent action on the robot's current state. Define $\phi : \mathcal{Z} \times \mathcal{S} \rightarrow \mathcal{A}$ as a decoder that now makes decisions based on both z and s. We expect that conditional autoencoders (cAE) and conditional variational autoencoders (cVAE) which use ϕ will learn more expressive and controllable actions than their non-state conditioned counterparts.

B. Algorithm

Our approach for training and leveraging these models is shown in Algorithm 1. First, the robot obtains demonstrations of related tasks—these could be provided by a caregiver, or even collected from another end-user. The robot then trains a low-dimensional embedding using this data and one of the models described above. We manually align the learned latent dimensions with the joystick DoF; for example, we rotate z so that up/down on the joystick corresponds to the latent DoF that pours/straightens the glass. When the user interacts with the robot, their inputs are treated as z, and the robot utilizes its decoder ϕ to reconstruct high-DoF actions.

V. SIMULATIONS

To test if the proposed low-DoF embeddings capture our desired user-friendly properties, we perform simulations on robot arms. The simulated robots have more DoF than needed to complete the task, and thus must learn to coordinate their redundant joints when decoding human inputs.

A. Setup

We simulate one- and two-arm planar robots, where each arm has five revolute joints and links of equal length. The state $s \in \mathbb{R}^n$ is the robot's joint position, and the action $a \in \mathbb{R}^n$ is the robot's joint velocity. Hence, the robot transitions according to: $s_{t+1} = s_t + a_t \cdot dt$, where dt is the step size. Demonstrations consist of trajectories of state-action pairs: in each of different simulated tasks, the robot trains with a total of 10000 state-action pairs.

Tasks. We consider four different tasks.

1) *Sine*: a single 5-DoF robot arm moves its end-effector along a sine wave with a 1-DoF latent action
2) *Rotate*: two robot arms are holding a box, and rotate that box about a fixed point using a 1-DoF latent action
3) *Circle*: one robot arm moves along circles of different radii with a 2-DoF latent action
4) *Reach*: a one-arm robot reaches from a start location to a goal region with a 1-DoF latent action

Model Details. We examine models such as PCA, AE, VAE, and state conditioned models such as cAE and cVAE. The encoders and decoders contain between two and four linear layers (depending on the task) with a $tanh(\cdot)$ activation function. The loss function is optimized using Adam with a learning rate of $1e^{-2}$. Within the VAE and cVAE, we set the normalization weight < 1 to avoid posterior collapse.

Dependent Measures. To determine *accuracy*, we measure the mean-squared error between the intended actions a and reconstructed actions $\hat{a}$ on a test set of state-action pairs (s, a) drawn from the same distribution as the training set.

To test model *controllability*, we select pairs of start and goal states (s_i, s_j) from the test set, and solve for the latent actions z that minimize the error between the robot's current state and s_j. We then report this minimum state error.

Algorithm 1 Learning Latent Control for Assistive Robots

1: Collect dataset $\mathcal{D} = \{(s_0, a_0), (s_1, a_1), \ldots\}$ from kinesthetic demonstrations
2: Train autoencoder to minimize loss $L(s, a)$ on $\mathcal{D}$
3: Align the learned latent space
4: **for** $t \leftarrow 1, 2, \ldots, T$ **do**
5: Set latent action z_t as human's joystick input
6: Execute reconstructed action $\hat{a}_t \leftarrow \phi(z_t, s_t)$
7: **end for**

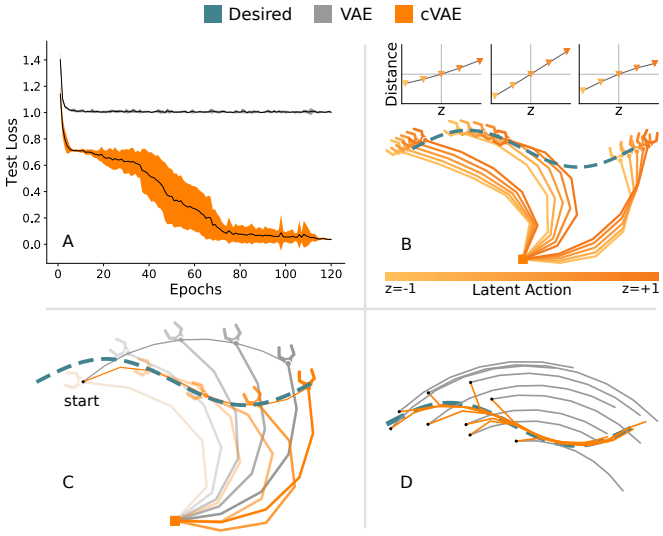

Fig. 2. Results for the *Sine* task. (A) mean-squared error between intended and reconstructed actions normalized by PCA test loss. (B) effect of the latent action z at three states along the sine wave for the cVAE model. Darker colors correspond to $z > 0$ and lighter colors signify $z < 0$. Above we plot the distance that the end effector moves along the sine wave as a function of z at each state. (C) rollout of robot behavior when applying a constant latent input $z = +1$, where both VAE and cVAE start at the same state. (D) end-effector trajectories for multiple rollouts of VAE and cVAE.

We jointly measure *consistency* and *scalability*: to do this, we select 25 states along the task, and apply a fixed grid of latent actions z_i from $[-1, +1]$ at each state. For every (s, z) pair we record the distance and direction that the end-effector travels (e.g., the direction is $+1$ if the end-effector moves right). We then find the best-fit line relating z to distance times direction, and report its R^2 error.

Our results are averaged across 10 trained models of the same type, and are listed in the form $mean \pm SD$.

Hypotheses. We have the following two hypotheses:

H1. *Only models conditioned on the state will accurately reconstruct actions from low-DoF inputs.*

H2. *State conditioned models will learn a latent space that is controllable, consistent, and scalable.*

B. Sine Task

This task and our results are shown in Fig. 2. We find that including state conditioning greatly improves *accuracy* when compared to the PCA baseline: AE and VAE incur $98.0 \pm 0.6\%$ and $100 \pm 0.8\%$ of the PCA loss, while cAE and cVAE obtain $1.37 \pm 1.2\%$ and $3.74 \pm 0.4\%$ of the PCA loss, respectively.

We likewise observe that cAE and cVAE are more *controllable* than their alternatives. When using the learned latent actions to move between 1000 randomly selected start and end states along the sine wave, cAE and cVAE have an average end-effector error of 0.05 ± 0.01 and 0.10 ± 0.01. Models without state conditioning—PCA, AE, and VAE—have average errors 0.90, 0.94 ± 0.01, and 0.95 ± 0.01.

When evaluating *consistency* and *scalability*, we discover that every model's relationship between latent actions and robot behavior can be modeled as approximately linear: PCA has the highest $R^2 = 0.99$, while cAE and cVAE have the lowest $R^2 = 0.94 \pm 0.04$ and $R^2 = 0.95 \pm 0.01$.

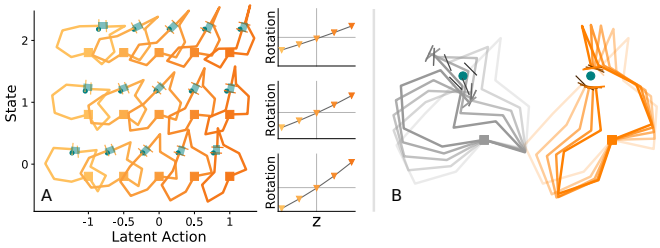

Fig. 3. Results for the *Rotate* task. (A) the robot uses two arms to hold a light blue box, and learns to rotate this box around the fixed point shown in teal. Each state corresponds to a different fixed point, and positive z causes counterclockwise rotation. On right we show how z affects the rotation of the box at each state. (B) rollout of the robot's trajectory when the user applies $z = +1$ for VAE and cVAE models, where both models start in the same state. Unlike the VAE, the cVAE model coordinates its two arms.

C. Rotate Task

We summarize the results for this two-arm task in Fig. 3. Like in the *Sine* task, the models conditioned on the current state are more *accurate* than their non-conditioned counterparts: AE and VAE have $28.7 \pm 4.8\%$ and $38.0 \pm 5.8\%$ of the PCA baseline loss, while cAE and cVAE reduce this to $0.65 \pm 0.05\%$ and $0.84 \pm 0.07\%$. The state conditioned models are also more *controllable*: when using the learned z to rotate the box, AE and VAE have $56.8 \pm 9\%$ and $71.5 \pm 8\%$ as much end-effector error as the PCA baseline, whereas cAE and cVAE achieve $5.4 \pm 0.1\%$ and $5.9 \pm 0.1\%$ error.

When testing for *consistency* and *scalability*, we measure the relationship between the latent action z and the change in orientation for the end-effectors of both arms (i.e., ignoring their location). Each model exhibits a linear relationship between z and orientation: $R^2 = 0.995 \pm 0.004$ for cVAE and $R^2 = 0.996 \pm 0.002$ for cVAE. In other words, there is an approximately linear mapping between z and the orientation of the box that the two arms are holding.

D. Circle Task

Next, consider the one-arm task in Fig. 4 where the robot has a 2-DoF latent action space. We here focus on the learned latent dimensions $z = [z_1, z_2]$, and examine how these latent dimensions correspond to the underlying task. Recall that the training data consists of state-action pairs which translate the robot's end-effector along (and between) circles of different radii. Ideally, the learned latent dimensions correspond to these axes, e.g., z_1 controls tangential motion while z_2 controls orthogonal motion. Interestingly, we found that this intuitive mapping is *only* captured by the state conditioned models. The average angle between the directions that the end-effector moves for z_1 and z_2 is $27 \pm 20°$ and $34 \pm 15°$ for AE and VAE models, but this angle increases to $72 \pm 9°$ and $74 \pm 12°$ for the cAE and cVAE (ideally $90°$). The state conditioned models better *disentangle* their low-dimensional embeddings, supporting our hypotheses and demonstrating how these models produce user-friendly latent spaces.

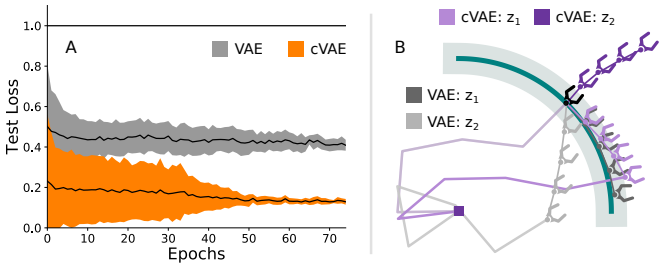

Fig. 4. Results for the *Circle* task. (A) mean-squared error between desired and reconstructed actions normalized by the PCA test loss. (B) 2-DoF latent action space $z = [z_1, z_2]$ for VAE and cVAE models. The current end-effector position is shown in black, and the colored grippers depict how changing z_1 or z_2 affects the robot's state. Under the cVAE model, these latent dimensions move the end-effector tangent or orthogonal to the circle.

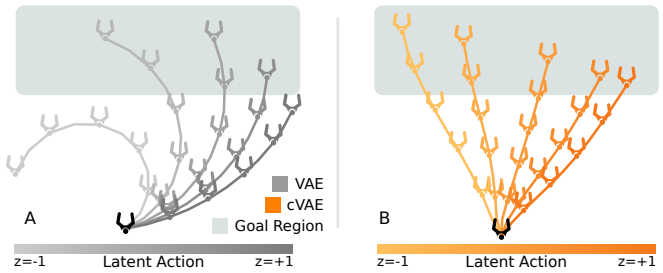

Fig. 5. Results for the *Reach* task. In both plots, we show the end-effector trajectory when applying constant inputs $z \in [-1, +1]$. The lightest color corresponds to $z = -1$ and the darkest color is $z = +1$. The goal region is highlighted, and the initial end-effector position is black. (A) trajectories with the VAE model. (B) trajectories with the cVAE model. The latent action z controls which part of the goal region the trajectory moves towards.

E. Reach Task

In the final task, a one-arm robot trains on trajectories that move towards a goal region (see Fig. 5). The robot learns a 1-DoF latent space, where z controls the direction that the trajectory moves (i.e., to the left or right of the goal region). We focus on *controllability*: can robots utilize latent actions to reach their desired goal? In order to test controllability, we sample 100 goals randomly from the goal region, and compare robots that attempt to reach these goals with either VAE or cVAE latent spaces. The cVAE robot more accurately reaches its goal: the L_2 distance between the goal and the robot's final end-effector position is 0.57 ± 0.38 under VAE and 0.48 ± 0.5 with cVAE. Importantly, using state conditioning improves not only the movement accuracy but also the movement *quality*. The average start-to-goal trajectory is 5.1 ± 2.8 units when using the VAE, and this length drops to 3.1 ± 0.5 with the cVAE model.

Summary. Viewed together, the results of our *Sine*, *Rotate*, *Circle*, and *Reach* tasks support hypotheses H1 and H2. The state conditioned models more *accurately* reconstruct high-DoF actions from low-DoF embeddings (H1), and also exhibit the user-friendly properties of *controllability*, *consistency*, and *scalability* (H2). We also observed that the latent dimensions naturally aligned themselves with the underlying DoF of the task, and that we could leverage the state conditioned models to embed robot trajectories.

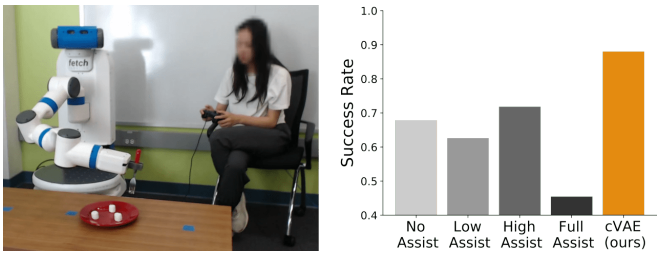

Fig. 6. Experimental setup for our first user study. (Left) in this feeding task, the participant uses a two-DoF joystick to guide the robot to pick up their desired morsel from a discrete set. (Right) we compare our latent action approach to shared autonomy strategies from the HARMONIC dataset.

VI. User Studies

To evaluate whether actual humans can use learned latent actions to teleoperate robots and perform everyday tasks, we conducted two user studies on a 7-DoF robotic arm (Fetch, Fetch Robotics). In the first study, we compared our proposed approach to state-of-the-art shared autonomy methods when the robot has a *discrete* set of possible goals. In the second study, participants completed a cooking task with *continuous* goals using either end-effector teleoperation or our learned latent actions. For both studies the users controlled the robot arm with a handheld joystick, and the tasks required careful coordination of each of the robot's joints.

A. Discrete Goals: Latent Actions vs. Shared Autonomy

In our first user study we implemented the assistive feeding task from the HARMONIC dataset [22] (see Fig. 6). Here the human is guiding the robot to pick up a bite of food. There are three morsels near the robot—i.e., three possible goals—and the human wants the robot to reach one of these goals. The HARMONIC dataset reports the performance of 24 people who completed this task while using the shared autonomy algorithm from Javdani *et al.* [7]. Under shared autonomy, the robot infers from the human's inputs which goal they are trying to reach, and then provides assistance towards that goal. We here conduct an additional experiment to compare our latent action method to this dataset.

Independent Variables. We manipulated the robot's teleoperation strategy with five levels: the four conditions from the HARMONIC dataset plus our proposed cVAE method. In the first four conditions, the robot provided no assistance (*No Assist*), or interpolated between the human's input and an assistive action (*Low Assist*, *High Assist*, and *Full Assist*). *High Assist* was the most effective strategy from this group: when interpolating, here the assistive action was given twice the weight of the human's input. In our *cVAE* approach the human's joystick inputs were treated as latent actions z (but the robot provided no other assistance). We trained our cVAE model on demonstrations from the HARMONIC dataset.

Dependent Measures. We measured the fraction of trials in which the robot picked up the correct morsel of food (*Success Rate*), the amount of time needed to complete the task (*Completion Time*), the total magnitude of the human's input (*Joystick Input*), and the distance traveled by the robot's end-effector (*Trajectory Length*).

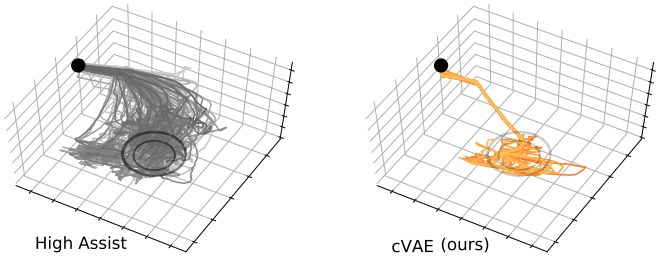

Fig. 7. End-effector trajectories from *High Assist* and *cVAE* conditions. The robot starts at the black dot, and moves to position itself over the plate.

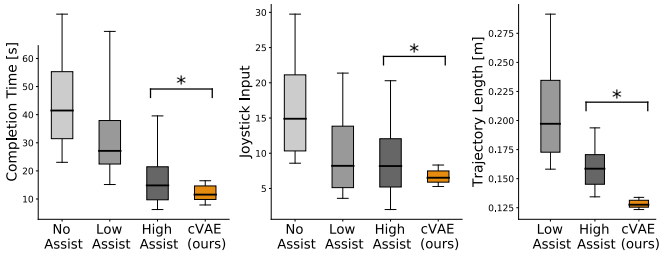

Fig. 8. Objective results from the feeding user study. We found that *cVAE* led to faster task completion with less user input and end-effector motion. The *Full Assist* condition performed worse than *High Assist* across the board (omitted for clarity). Error bars show the 10 and 90 percentiles, and $*$ denotes statistical significance ($p < .05$).

Hypothesis. We had the following hypothesis:

> **H3.** *Teleoperating with learned latent actions will improve task success while reducing the completion time, joystick inputs, and trajectory length.*

Experimental Setup. Participants interacted with a handheld joystick while watching the robotic arm. The robot held a fork; during the task, users teleoperated the robot to position this fork directly above their desired morsel. We selected the robot's start state, goal locations, and movement speed to be consistent with the HARMONIC dataset.

Participants and Procedure. Our participant pool consisted of ten Stanford University affiliates who provided informed consent (3 female, average participant age 23.9 ± 2.8 years). Following the same protocol as the HARMONIC dataset, each participant was given up to five minutes to familiarize themselves with the task and joystick, and then completed five recorded trials using our *cVAE* approach. At the start of each trial the participant indicated which morsel they wanted the robot to reach; the trial ended once the user pushed a button to indicate that the fork was above their intended morsel. We point out that participants only completed the task with the *cVAE* condition; other teleoperation strategies are benchmarked in Newman *et al.* [22].

Results. We display example robot trajectories in Fig. 7 and report our dependent measures in Figs. 6 and 8. Inspecting these example trajectories, we observe that the *cVAE* model learned latent actions that move the robot's end-effector into a region above the plate. Users controlling the robot with *cVAE* reached their desired morsel in 44 of the 50 total trials, yielding a higher *Success Rate* than the assistance baselines. To better compare *cVAE* to the *High Assist* condition, we performed independent t-tests. We found that participants that used the *cVAE* model took statistically significant lower *Completion Time* ($t(158) = 2.95$, $p < .05$), *Joystick Input* ($t(158) = 2.49$, $p < .05$), and *Trajectory Length* ($t(158) = 9.39$, $p < .001$), supporting our hypothesis H3.

Summary. Assistive robots that learn a mapping from low- to high-DoF actions can feed humans efficiently. Users teleoperating the *cVAE* robot reached their preferred goal more accurately than shared autonomy baselines, while requiring less time, effort, and movement.

B. Continuous Goals: Latent Actions vs. End-Effector

We have shown that learned latent actions can help robots reach discrete goals; next we want to assess our approach on tasks with *continuous* goals. In our second user study we therefore focus on a cooking scenario (see Fig. 9). The user wants their assistive robot to help them make a recipe: this requires picking up ingredients from the shelf, pouring into a bowl, recycling empty containers—or returning half-filled containers to the shelf—and then stirring the mixture. Shared autonomy is not suitable within this setting because the task involves completing specific robot motions, not reaching discrete goals. Hence, we compare our latent action method against direct teleoperation, where the joystick inputs control the position and orientation of the robot's end-effector. This end-effector strategy mimics current teleoperation interfaces available on wheelchair-mounted robotic arms [2].

Independent Variables. We tested two teleoperation strategies: *End-Effector* and *cVAE*. Under *End-Effector* the user inputs applied a 6-DoF twist to the robot's end-effector, controlling its linear and angular velocity. Participants interacted with two 2-DoF joysticks, and were given a button to toggle between linear and angular motion [2], [7], [22]. By contrast, in *cVAE* the participants could only interact with one 2-DoF joystick, i.e., the latent action was $z = [z_1, z_2] \in \mathbb{R}^2$. We trained the *cVAE* model using state-action pairs from kinesthetic demonstrations, where we guided the robot along related sub-tasks such as reaching for the shelf, pouring objects into the bowl, and stirring. The *cVAE* was trained with less than 7 minutes of demonstration data.

Dependent Measures – Objective. We measured the total amount of time it took for participants to complete the entire cooking task (*Completion Time*), as well as the magnitude of their inputs (*Joystick Input*).

Dependent Measures – Subjective. After participants completed the task with one condition, we administered a 7-point Likert scale survey. Questions on this survey were separated into six scales, including: adapting to the condition (*Adapt*), consistency between inputs and robot behavior (*Consistent*), ease of performing the task (*Ease*), and overall enjoyment (*Enjoyment*). Once users had completed both conditions, we asked comparative questions about which they preferred (*Prefer*), which was *Easier*, and which was more *Natural*.

Hypotheses. We had the following hypotheses:

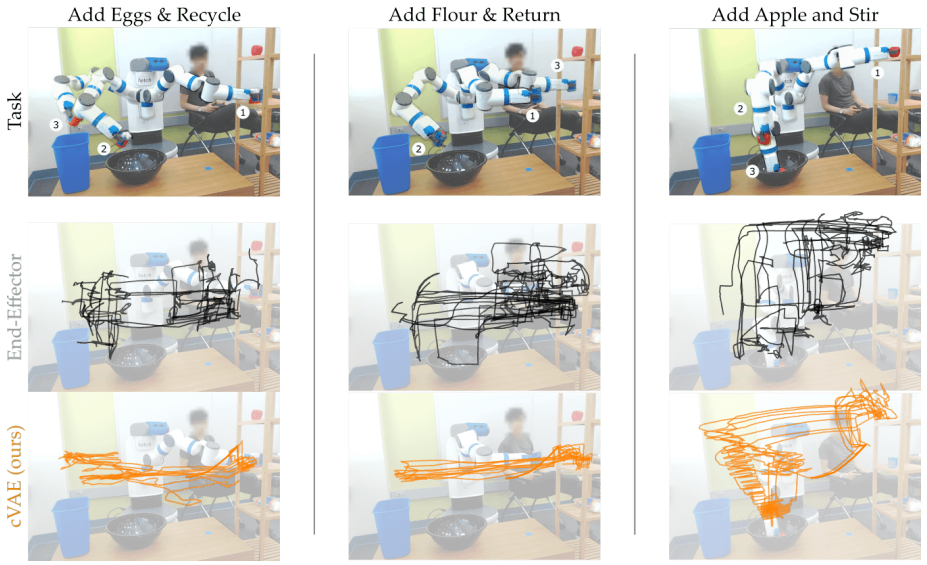

Fig. 9. Setup for our second user study. (Top row) the participant is teleoperating an assistive robot to make their recipe. This recipe is broken down into three sub-tasks. On left the robot picks up eggs, pours them into the bowl, then drops the container into the recycling. In middle the robot picks up flour, pours it into the bowl, then returns the container to the shelf. On right the robot grasps an apple, places it in the bowl, then stirs the mixture. (Middle row) example robot trajectories when the person directly controls the robot's *End-Effector*. (Bottom row) example trajectories when using *cVAE* to learn latent actions. Comparing the example trajectories, we observe that *cVAE* resulted in robot motions that more smoothly and directly accomplished the task.

H4. *Users controlling the robot arm with low-DoF latent actions will complete the cooking task more quickly and with less overall effort.*

H5. *Participants will perceive the robot as easier to work with in the cVAE condition, and will prefer the cVAE over End-Effector teleoperation.*

Experimental Setup. We designed a cooking task where the person is making a simplified "apple pie." As shown in Fig. 9, the assistive robot must sequentially pour eggs, flour, and an apple into the bowl, dispose of their containers, and stir the mixture. The user sat next to the robot and controlled its behavior with a handheld joystick.

Participants and Procedure. We used a within-subjects design and counterbalanced the order of our two conditions. Eleven members of the Stanford University community (4 female, age range 27.4 ± 11.8 years) provided informed consent to participate in this study. Four subjects had prior experience interacting with the robot used in our experiment.

Before starting the study, participants were shown a video of the cooking task. Participants then separately completed the three parts of the task as visualized in Fig. 9; we reset the robot to its home position between each of these sub-tasks. After the user completed these sub-tasks, we re-arranged the placement of the recycling and bowl, and users performed the entire cooking task without breaks. Participants were told about the joystick interface for each condition, and could refer to a sheet that labelled the joystick inputs.

Results – Objective. Our objective results are summarized in Fig. 10. When using *cVAE* to complete the entire recipe, participants finished the task in less time ($t(10) = -6.9$, $p < .001$), and used the joystick less frequently ($t(10) = -5.1$, $p < .001$) as compared to direct *End-Effector* teleoportation.

Results – Subjective. We display the results of our 7-point

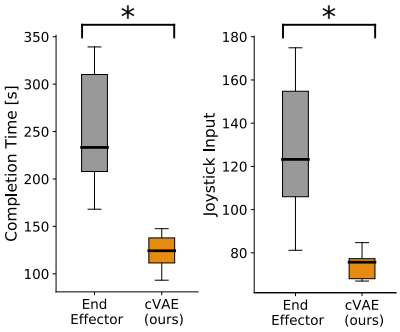

Fig. 10. Objective results from the cooking user study. These results were collected on the full task (combining each of the sub-tasks from Fig. 9).

Likert scale surveys in Fig. 11. Before reporting these results, we first confirmed the reliability of our six scales (such as *Adapt*, *Consistent*, and *Enjoyment*). We then leveraged paired t-tests to compare user ratings for *End-Effector* and *cVAE* conditions. We found that participants perceived *cVAE* as requiring less user effort ($t(10) = 2.7$, $p < .05$) than *End-Effector*. Participants also indicated that it was easier to complete the task with *cVAE* ($t(10) = 2.5$, $p < .05$), and that *cVAE* caused the robot to move more naturally ($t(10) = 3.8$, $p < .01$). The other scales were not significantly different.

Summary. Taken together, these results partially support our hypotheses. When controlling the robot with latent actions, users completed the cooking task more quickly and with less effort (H4). Participants believed that the *cVAE* approach led to more natural robot motion, and indicated that it was easier to perform the task with latent actions. However, participants did not indicate a clear preference for either strategy (H5).

VII. Discussion and Conclusion

Summary. We focused on assistive robotics settings where the robot has access to previous demonstrations from related

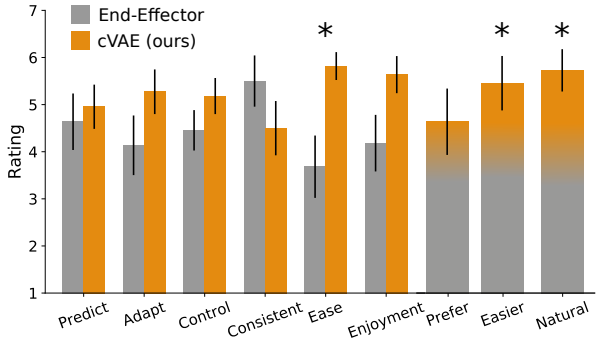

Fig. 11. Subjective results from the cooking user study. Higher ratings indicate participant agreement. For *Prefer*, *Easier*, and *Natural*, a 1 rating denotes that the user favored *End-Effector*, and a 7 denotes that the user favored *cVAE*. Participants thought our approach required less effort, made it easier to complete the task, and produced more natural robot motion.

tasks. In these settings, we showed that intelligent robots can embed their high-DoF, dextrous behavior into low-DoF latent actions for the human to control. We evaluated five different models for learning the latent actions, and determined that autoencoders conditioned on the system state accurately reconstructed the human's intended action, and also produced controllable, consistent, and scalable latent spaces.

One key advantage to latent actions is that—unlike current shared autonomy approaches—they can assist the human during tasks with either discrete or continuous goals. We validated this in our two user studies. In the first (discrete), latent actions resulted in higher feeding success than shared autonomy baselines. In the second (continuous), participants leveraged learned latent actions to complete a cooking task using 2-DoFs. Compared against a teleoperation strategy currently employed by assistive arms, latent actions led to improved objective and subjective performance.

How practical is this approach? In our cooking user study, the robot was trained with less than 7 minutes of kinesthetic demonstrations. We attribute this data efficiency in part to the simplicity of our model structure: we used standard cVAEs that we trained within the robot's on-board computer. We believe this makes our approach very efficient, accurate, and easy to use in practice as compared to alternatives.

Limitations and Future Work. Although the latent actions were intuitive when the robot's state was near the training distribution, once the robot reached configurations where we had not provided demonstrations the latent actions became erratic. For example, one participant unintentionally rotated the arm upside-down when trying to pour flour, and was unable to guide the robot back towards the cooking task. We were also surprised that users did not clearly prefer the latent action approach despite its improved performance. In their questionnaire responses, participants indicated that—while it was easier to work with latent actions—they enjoyed the increased freedom of control under end-effector teleoperation.

These limitations suggest that assistive robots should not always rely on latent actions. Our future work focuses on interweaving learned latent actions with standardized teleoperation strategies, so that robots can intelligently decide when latent actions are helpful. Our research aims to enable humans to seamlessly collaborate with assistive robots.

References

[1] D. M. Taylor, *Americans With Disabilities: 2014*. US Census Bureau, 2018.

[2] L. V. Herlant, R. M. Holladay, and S. S. Srinivasa, "Assistive teleoperation of robot arms via automatic time-optimal mode switching," in *ACM/IEEE Int. Conf. on Human Robot Interaction (HRI)*, 2016, pp. 35–42.

[3] B. D. Argall, "Autonomy in rehabilitation robotics: An intersection," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 441–463, 2018.

[4] D. P. Losey, C. G. McDonald, E. Battaglia, and M. K. O'Malley, "A review of intent detection, arbitration, and communication aspects of shared control for physical human–robot interaction," *Applied Mechanics Reviews*, vol. 70, no. 1, p. 010804, 2018.

[5] T. Carlson and J. d. R. Millan, "Brain-controlled wheelchairs: A robotic architecture," *IEEE Robotics & Automation Magazine*, vol. 20, no. 1, pp. 65–73, 2013.

[6] K. Muelling, A. Venkatraman, J.-S. Valois, J. Downey, J. Weiss, S. Javdani, M. Hebert, A. B. Schwartz, J. L. Collinger, and J. A. Bagnell, "Autonomy infused teleoperation with application to bci manipulation," *arXiv preprint arXiv:1503.05451*, 2015.

[7] S. Javdani, H. Admoni, S. Pellegrinelli, S. S. Srinivasa, and J. A. Bagnell, "Shared autonomy via hindsight optimization for teleoperation and teaming," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 717–742, 2018.

[8] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *The International Journal of Robotics Research*, vol. 32, no. 7, pp. 790–805, 2013.

[9] D. Gopinath, S. Jain, and B. D. Argall, "Human-in-the-loop optimization of shared autonomy in assistive robotics," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 247–254, 2016.

[10] R. M. Aronson, T. Santini, T. C. Kübler, E. Kasneci, S. Srinivasa, and H. Admoni, "Eye-hand behavior in human-robot shared manipulation," in *ACM/IEEE Int. Conf. on Human-Robot Interaction (HRI)*, 2018, pp. 4–13.

[11] A. Broad, T. Murphey, and B. Argall, "Learning models for shared control of human-machine systems with unknown dynamics," in *Robotics: Science and Systems (RSS)*, 2018.

[12] S. Reddy, A. D. Dragan, and S. Levine, "Shared autonomy via deep reinforcement learning," in *Robotics: Science and Systems (RSS)*, 2018.

[13] M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller, "Embed to control: A locally linear latent dynamics model for control from raw images," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015, pp. 2746–2754.

[14] J. Co-Reyes, Y. Liu, A. Gupta, B. Eysenbach, P. Abbeel, and S. Levine, "Self-consistent trajectory autoencoder: Hierarchical reinforcement learning with trajectory embeddings," in *Int. Conf. on Machine Learning (ICML)*, 2018, pp. 1009–1018.

[15] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play," *arXiv preprint arXiv:1903.01973*, 2019.

[16] A. Edwards, H. Sahni, Y. Schroecker, and C. Isbell, "Imitating latent policies from observation," in *Int. Conf. on Machine Learning (ICML)*, 2019, pp. 1755–1763.

[17] Y. Chandak, G. Theocharous, J. Kostas, S. Jordan, and P. Thomas, "Learning action representations for reinforcement learning," in *Int. Conf. on Machine Learning (ICML)*, 2019, pp. 941–950.

[18] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Int. Conf. on Learning Representations (ICLR)*, 2014.

[19] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[20] P. K. Artemiadis and K. J. Kyriakopoulos, "EMG-based control of a robot arm using low-dimensional embeddings," *IEEE Transactions on Robotics*, vol. 26, no. 2, pp. 393–398, 2010.

[21] M. T. Ciocarlie and P. K. Allen, "Hand posture subspaces for dexterous robotic grasping," *The International Journal of Robotics Research*, vol. 28, no. 7, pp. 851–867, 2009.

[22] B. A. Newman, R. M. Aronson, S. S. Srinivasa, K. Kitani, and H. Admoni, "HARMONIC: A Multimodal Dataset of Assistive Human-Robot Collaboration," *ArXiv e-prints*, July 2018.