Learning State Conditioned Linear Mappings for Low-Dimensional Control of Robotic Manipulators

Michael Przystupa*†, Kerrick Johnstonbaugh*†, Zichen Zhang†, Laura Petrich†,
Masood Dehghan†, Faezeh Haghverd†, Martin Jagersand†

***Abstract*— Identifying an appropriate task space can simplify solving robotic manipulation problems. One solution is deploying control algorithms in a learned low-dimensional action space. Linear and nonlinear action mapping methods have trade-offs between simplicity and the ability to express motor commands outside of a single low-dimensional subspace. We propose that learning local linear action representations can achieve both of these benefits. Our state-conditioned linear maps ensure that for any given state, the high-dimensional robotic actuation is linear in the low-dimensional actions. As the robot state evolves, so do the action mappings, so that necessary motions can be performed during a task. These local linear representations guarantee desirable theoretical properties by design. We validate these findings empirically through two user studies. Results suggest state-conditioned linear maps outperform conditional autoencoder and PCA baselines on a pick-and-place task and perform comparably to mode switching in a more complex pouring task.**

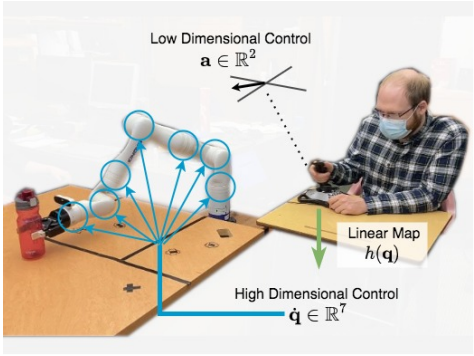

Fig. 1: A user controls a 7-DOF robotic manipulator through low dimensional actions of a 2-DOF joystick. The state-conditioned linear mapping $h(\mathbf{q}) \in \mathbf{R}^{7\times 2}$ transforms the joystick inputs $\mathbf{a}$ to high dimensional motor commands $\dot{\mathbf{q}}$.

I. Introduction

Robotic manipulation tasks involve controlling many actuators, or degrees-of-freedom (DOFs), simultaneously to perform complex motions like pouring liquids [1], unscrewing light bulbs [2], or even playing table-tennis [3]. These complex tasks are achievable by various methods from domains like control theory, visual servoing, reinforcement learning, or teleoperation. For control solutions that involve learning agents (artificial or human), the consequences of poor design decisions can have adverse effects in robotic manipulation problems, where safety [4] and sample efficiency [5] are desirable properties. In reinforcement learning, reducing sample complexity in environments with high-dimensional action spaces is a fundamental challenge [6]. For humans, previous research suggests that common control approaches such as mode-switching can be cognitively strenuous, with basic tasks like opening a door or picking an object requiring 30-60 mode switches [7], [8].

Action maps that transform low DOF inputs directly to relevant high-dimensional robotic commands have been explored both to reduce cognitive strain during teleoperation and to increase the learning efficiency of artificial agents [9], [10], [11], [12]. The assumption is that joints work together in the high-dimensional ambient space to produce concerted motions that lie on some lower-dimensional manifold. Previous research on hand poses – a high-DOF setting – supports this assumption, finding that principal component analysis (PCA) [13], [14] can capture 80% of configurations with only two principal components [15]. Authors have applied this result to develop linear action map teleoperation grasping control schemes [16], [17], [18] and planning algorithms [19]. Linear maps are advantageous because they are easy to analyze and provide intuitive mappings for teleoperation. However, globally linear approaches assume all task-relevant high-DOF commands exist in a single low-dimensional subspace, and therefore may not capture finer or subtler movements [18].

Other research has investigated nonlinear action mappings as an alternative. These mappings are typically learned with a conditional neural autoencoder (CAE) [12], [20]. Instead of a single linear map, low-dimensional actions are input to a state-dependent neural decoder to predict the corresponding high-dimensional motion. Assistive robotic research has demonstrated that CAE models enable higher success rates and faster completion times in assistive eating tasks compared to alternative systems [21], [22], [23], [20], [12]. However, CAEs are difficult to analyze because of their nonlinear structure. Authors have relied on engineering auxiliary loss terms to implicitly add desirable user-control properties [23], [24], but still require empirical evaluation to verify enforcement on deployment.

Given the potential benefits of both methods and their applications, we are interested in answering the question: *Can we combine the analytic advantages of linear mappings with the flexibility of nonlinear neural networks in a single model?*

† Department of Computing Science, University of Alberta, Edmonton AB., Canada, T6G 2E8. {przystup, kerrick, zichen2, llorrain, masood1, haghverd, mj7, }@ualberta.ca
* Equal contribution

We answer in the affirmative, summarizing our contributions as follows: 1) We propose *State Conditioned Linear Maps* (SCL maps), which use neural networks to predict a local linear basis for control. 2) We formulate the soft reversibility property for action mappings and prove that SCL maps guarantee this property by design. 3) The results of two teleoperation user studies reveal that participants found SCL maps more effective than PCA and CAE baselines, and competitive with mode switching.

II. Background

This section summarizes background information and defines relevant notations. Throughout this paper, we use small bold letters for vectors (e.g., $\mathbf{s}, \mathbf{a}$) and capital bold letters for matrices (e.g., $\mathbf{H}$). All vectors are assumed to be columns. Additional notations will be described as they become relevant.

A. Problem Formulation

We use $\mathbf{o}_r$ for robot observation information, and $\mathbf{o}_h$ for agent observations. In our experiments, $\mathbf{o}_r$ are joint angles $\mathbf{q}$ and gripper state $c \in [0, 1]$, but $\mathbf{o}_r$ could also be e.g. end effector location $\mathbf{x} \in \mathbb{R}^3$, or other state. We define $\mathbf{o}_h$ as the sensory inputs of an intelligent agent which can include visual, auditory, haptic, and other sensory inputs. The agent provides low-dimensional control inputs $\mathbf{a}$ which get transformed into high-dimensional robotic commands $\dot{\mathbf{q}}$ to complete certain manipulation tasks. We assume that the tasks have the Markov property, such that the action can be determined with only the current observation. The policy $\pi(\mathbf{o}_h) = \mathbf{a}$ denotes the mapping[1] from agent observation $\mathbf{o}_h$ to the low-dimensional action $\mathbf{a}$. We are interested in learning state specific mappings $g(\mathbf{o}_r, \mathbf{a}) = \dot{\mathbf{q}}$. We assume that the state transition is deterministic, and define the transition function as $\mathbf{q}' = T(\mathbf{q}, \mathbf{a}; g)$ which outputs the next state (joint angles) $\mathbf{q}'$ when taking a low-level action $\mathbf{a}$ from state $\mathbf{q}$. The transition function depends on the state-conditioned map g that is trained from demonstration data, and fixed during control. We write $T(\mathbf{q}, \mathbf{a})$ omitting g when it is clear from context that g is implied.

B. Previous Action Mapping Functions

Prior works have modeled $g(\mathbf{o}_r, \mathbf{a})$ with PCA and conditional autoencoders. Principal component analysis reduces the dimensionality of data by calculating the first d principal components and using them as an orthonormal basis with which to represent the data. We denote the matrix with the first d principal components as its columns as $\Sigma \in \mathbb{R}^{m \times d}$ [13]. The action map can then be defined as $g(\mathbf{q}, \mathbf{a}) = \Sigma\mathbf{a} = \hat{\dot{\mathbf{q}}}$, where the output is independent of $\mathbf{q}$. A conditional autoencoder is an unsupervised learning model that attempts to summarize high-dimensional data $\dot{\mathbf{q}} \in \mathbb{R}^m$ in some lower dimensional space, while conditioning the decoder on additional context information $\mathbf{o}_r$. The encoder $f(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{a}$ compresses the data, and a decoder reconstructs the data $g(\mathbf{o}_r, \mathbf{a}) = \hat{\dot{\mathbf{q}}}$. Both f and g are neural networks with parameters (ϕ, θ) respectively. They are trained with mean squared error loss: $\min_{\phi,\theta} \mathbb{E}[\|\dot{\mathbf{q}} - \hat{\dot{\mathbf{q}}}\|_2^2]$. At deployment, the decoder is the action map and $\mathbf{a}$ corresponds to user inputs.

[1]Throughout this work $\pi(\cdot)$ is always embodied by a human user.

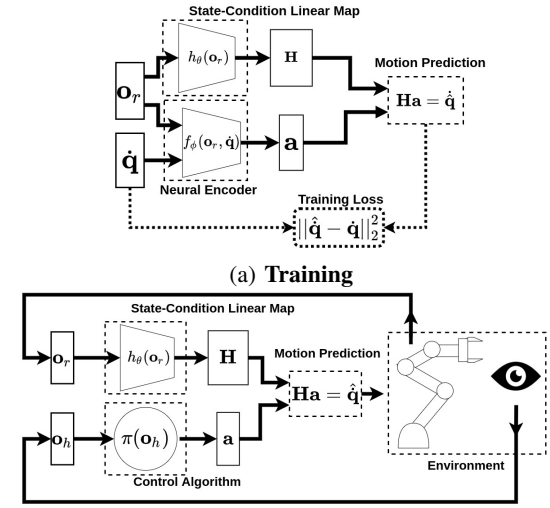

Fig. 2: State conditional linear maps are trained as autoencoders. At deployment, the encoder is replaced with the appropriate control policy. In our teleoperation user study, the control policy $\pi(\mathbf{o}_h)$ is embodied by a human.

III. Methods

This section describes our method for learning to control high DOFs with low-dimensional actions. We assume that there are available training tuples $\{(\mathbf{o}_r, \dot{\mathbf{q}})\}_i$ to learn state conditioned action mappings. We outline the model, training procedure, and a post-processing step.

A. State Conditioned Linear Mappings

A linear action map is desirable because it is easier to mathematically analyze. Knowing how to utilize a robot's context for a task can help determine the appropriate behavior, but this may be a nonlinear function. These two observations lead us to build neural networks ($h : \mathbb{R}^n \rightarrow \mathbb{R}^{m \times d}; \mathbf{o}_r \mapsto \mathbf{H}$) that output a linear map ($\mathbf{H} : \mathbb{R}^d \rightarrow \mathbb{R}^m$):

$$\hat{\dot{\mathbf{q}}} = \mathbf{Ha} = h_\theta(\mathbf{o}_r)\mathbf{a}. \tag{1}$$

The predicted high dimensional robotic velocity command is $\hat{\dot{\mathbf{q}}} \in \mathbb{R}^m$ and the user's low dimensional action is $\mathbf{a} \in \mathbb{R}^d$ (a column vector). The matrix $\mathbf{H} \in \mathbb{R}^{m \times d}$ is the state-conditioned linear map (SCL map). It represents a linear subspace of high-dimensional manipulation commands controllable by low-dimensional actions. State-conditioning allows the model to contextualize high-dimensional manipulation commands specific to the robotic state $\mathbf{o}_r$ [20]. The user's action form high-dimensional commands as linear combinations of the columns of $\mathbf{H}$. Linear map prediction has been previously considered for developing control algorithms [25], [26], [27], [28], [29].

B. Training SCL Maps

To train $h_\theta(\mathbf{o}_r)$, we optimize the system as a neural autoencoder with mean squared error loss:

$$\min_{\phi,\theta} ||\dot{\mathbf{q}} - h_\theta(\mathbf{o}_r) f_\phi(\mathbf{o}_r, \dot{\mathbf{q}})||_2^2, \tag{2}$$

where ϕ, θ are optimized jointly. The purpose of f is to produce appropriate actions that combine the SCL basis vectors to reconstruct the joint velocities. At deployment $f_\phi(\mathbf{o}_r, \dot{\mathbf{q}}) = \mathbf{a}$ is discarded and is replaced by an agent $\mathbf{a} \sim \pi(\mathbf{o}_h)$. Figure 2 shows the training and deployment stages of SCL maps.

C. Enabling Diverse Actions

A learned SCL map $\mathbf{H}$ may be sufficient to reconstruct high DOF control commands during training, but not facilitate exploration of the state space at deployment. Consider a two-joint robot. If $\mathbf{H}$ has columns $[1, 0]^\top$, and $[1, 0.1]^\top$, then the columns span all of $\mathbb{R}^2$. Yet, both $\mathbf{a} = [1, 0]^\top$ and $\mathbf{a} = [0, 1]^\top$ will give roughly the same motion, which could make it difficult to reach certain joint configurations. We address this issue by applying the Gram-Schmidt process on the columns of $\mathbf{H}$ to create an orthonormal representation $\mathbf{H}'$ [30]. The columns of $\mathbf{H}'$ form an orthonormal basis that spans the same sub-space as $\mathbf{H}$, but can lead to more varied actions. Thanks to the fact that the Gram-Schmidt process is differentiable, we leveraged auto-differentiation software to seamlessly include this post-processing step during the training of SCL.

IV. Properties of SCL

In this section, we discuss properties of SCL maps which include *proportionality* and *reversibility*. Both properties give an agent more control when interacting with a robot. Our analysis was motivated by the human-prior definitions from the work of Li et. al. [23]. Their interest was aligning action maps to user preference for teleoperation, whereas SCL is a novel action mapping method. SCL maps are also much simpler requiring no additional loss terms like previous work, but future work could combine both approaches. We use $\Psi(\mathbf{q}) = \mathbf{x}$ as the kinematics function of the end-effector's position and orientation [31]. The transition operator is defined as $T(\mathbf{q}, \mathbf{a}; g) = \mathbf{q} + g(\mathbf{q}, \mathbf{a}) = \mathbf{q} + h(\mathbf{q})\mathbf{a}$.[2] We dropped θ in $h(\mathbf{q})$ to ease notations.

A. Proportionality

With exceptions in shared autonomy [21], users expect small movements in an interface lead to small robotic motions and large inputs to larger motions. For scalar $\alpha \in \mathbb{R}$ the resulting action $\mathbf{a}' = \alpha\mathbf{a}$ will lead to a proportional change in the end effectors current position:

$$\alpha \|\Psi(T(\mathbf{q}, \mathbf{a}; g)) - \Psi(\mathbf{q})\| \approx \|\Psi(T(\mathbf{q}, \alpha\mathbf{a}; g)) - \Psi(\mathbf{q})\|.$$

Intuitively, the end-effector is expected to move in proportion to the magnitude of the input. As SCL maps predict a linear weight matrix, proportionality is locally achieved (where kinematics Ψ is approximately linear) by design. Suppose we have that $\mathbf{a}' = \alpha\mathbf{a}$ for $\alpha \in \mathbb{R}^+$. This will lead to a proportional increase of the response ($\mathbf{H}\mathbf{a}' = \alpha\mathbf{H}\mathbf{a} = \alpha\dot{\mathbf{q}}$).

[2]In practice, the inputs of g are not limited to joint angles and can still contain additional task-specific information.

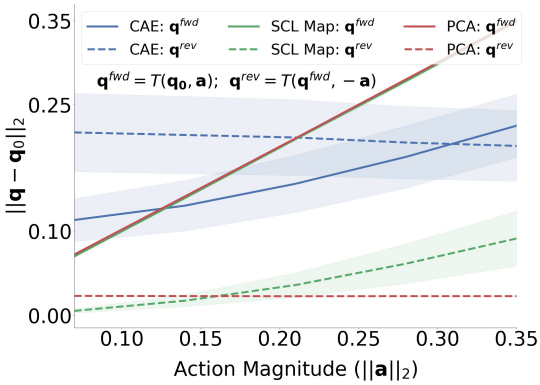

Fig. 3: Proportionality and soft reversibility experiments on a Kinova Gen-3 lite. We measure the Euclidean distance between the start and end joint positions. The line and shaded area show the mean and one-half standard deviation from 100 runs (10 trained models, 10 random states), respectively. Note $||\mathbf{q}^{fwd} - \mathbf{q}_0||_2$ for SCL and PCA overlap.

B. Soft Reversibility

The ability to revisit states is crucial for user experiences in case of mistakes or changes in the task. For two states $\mathbf{q}_i$ and $\mathbf{q}_j$ and an action $\mathbf{a} \neq 0$ such that $\mathbf{q}_j = T(\mathbf{q}_i, \mathbf{a}; g)$, *Reversibility* means that:

$$\mathbf{q}_j = T(\mathbf{q}_i, \mathbf{a}; g) \Rightarrow \mathbf{q}_i = T(\mathbf{q}_j, -\mathbf{a}; g).$$

For this definition to hold for SCL maps, $h(\mathbf{q}_j)$ must be equivalent to $h(\mathbf{q}_i)$, which is not guaranteed. Instead, we show that SCL satisfies a *soft reversibility* property: the state $\mathbf{q}_k$ reached by the inverse action from $\mathbf{q}_j$ will be closer than $\mathbf{q}_j$ to $\mathbf{q}_i$.

Theorem 1 (Soft Reversibility): Let $\|\mathbf{a}\|_2 < 1$ and $T(\mathbf{q}, \mathbf{a}; g) = \mathbf{q} + g(\mathbf{q}, \mathbf{a}) = \mathbf{q} + h(\mathbf{q})\mathbf{a}$. $h(\mathbf{q}) \in \mathbb{R}^{m \times d}$ is a matrix transformed from the vector form of hidden layer activations $v(\mathbf{q}) \in \mathbb{R}^{dm}$. Suppose v is Lipschitz continuous for some $L_v \leq 1$, that is, $\|v(\mathbf{q}_j) - v(\mathbf{q}_i)\|_2 \leq L_v \|\mathbf{q}_j - \mathbf{q}_i\|_2$. Then for some $\mathbf{q}_j = T(\mathbf{q}_i, \mathbf{a}; g)$ we have:

$$\|T(\mathbf{q}_j, -\mathbf{a}; g) - \mathbf{q}_i\|_2 < \|\mathbf{q}_j - \mathbf{q}_i\|_2$$

Proof: From our definition of $T(\mathbf{q}, \mathbf{a}; g)$:

$$\begin{aligned} \mathbf{q}_j &= \mathbf{q}_i + h(\mathbf{q}_i)\mathbf{a}_i \\ \mathbf{q}_k &= \mathbf{q}_j + h(\mathbf{q}_j)\mathbf{a}_j \end{aligned} \tag{3}$$

where $\mathbf{q}_i, \mathbf{q}_j, \mathbf{q}_k$ are three consecutive states, $\mathbf{a}_i$ and $\mathbf{a}_j$ are the actions at current time step and the next time step respectively. If the action $\mathbf{a}_j$ is the reverse action $\mathbf{a}_j = -\mathbf{a}_i$,

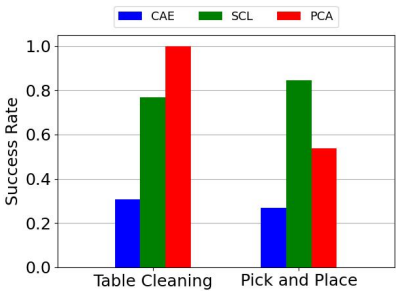

Fig. 4: Participant success rates for each control method in the Table Cleaning and Pick and Place tasks.

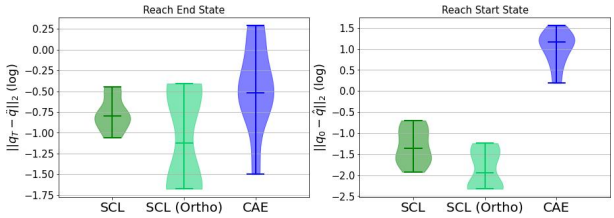

Fig. 5: Violin plots of euclidean distances across human prior loss combinations for training. Experiments were on a simulated 5-DOF planar reaching task. Reported in log scale for visualization.

we can rewrite Eq. (3) as

$$\begin{aligned}\mathbf{q}_k &= \mathbf{q}_i + h(\mathbf{q}_i)\mathbf{a}_i + h(\mathbf{q}_j)(-\mathbf{a}_i) \\ &= \mathbf{q}_i + (h(\mathbf{q}_i) - h(\mathbf{q}_j))\mathbf{a}_i\end{aligned}$$

From the Lipschitz continuity assumption, it follows that:

$$\begin{aligned}\|\mathbf{q}_k - \mathbf{q}_i\|_2 &= \|(h(\mathbf{q}_i) - h(\mathbf{q}_j))\mathbf{a}_i\|_2 \\ &\leq \|h(\mathbf{q}_i) - h(\mathbf{q}_j)\|_2 \, \|\mathbf{a}_i\|_2 \quad (4)\\ &\leq \|h(\mathbf{q}_i) - h(\mathbf{q}_j)\|_F \, \|\mathbf{a}_i\|_2 \quad (5)\\ &= \|v(\mathbf{q}_i) - v(\mathbf{q}_j)\|_2 \, \|\mathbf{a}_i\|_2 \quad (6)\\ &\leq L_v \|\mathbf{q}_i - \mathbf{q}_j\|_2 \, \|\mathbf{a}_i\|_2 \quad (7)\\ &\leq (L_v \|\mathbf{a}_i\|_2) \, \|\mathbf{q}_i - \mathbf{q}_j\|_2 \quad (8)\\ &< \|\mathbf{q}_i - \mathbf{q}_j\|_2 \quad (9)\end{aligned}$$

for any pairs of joint angles in a robot's set of joint configurations $\mathbf{q}_i, \mathbf{q}_j \in \mathcal{Q}$ such that $\mathbf{q}_j = T(\mathbf{q}_i, \mathbf{a}_i; g)$, where $\mathcal{Q}$ is the set of joint angles of the robot. We can tell from Eq. (9) that if $L_v \|\mathbf{a}_i\|_2 < 1$, taking the reverse action $\mathbf{a}_j = -\mathbf{a}_i$ brings the robot closer to the initial state $\mathbf{q}_i$. ■

C. Empirical Evaluation of Properties

We empirically validate our theoretical results for SCL map properties on a *Kinova Gen-3 lite* [32], and compare against CAE and PCA baselines. For the SCL and CAE algorithms, we trained 10 models each using the Adam optimizer for 200 epochs, with a learning rate of $1e$-3 and mini-batches of 256. Both SCL and CAE were trained with encoders and decoders that had two hidden layers each, with 256 neurons and tanh activation functions. The SCL decoder outputs a matrix as described in Section III. We apply the Lipschitz training procedure in Algorithm 1 of Gouk et. al. [33], $L_v = 1$, when training SCL maps to enforce Lipschitz continuity. We also apply the Gram-Schmidt process on $\mathbf{H} = h_\theta(\mathbf{q})$ in training and deployment.

For all algorithms, we generate an angle $\theta \sim \mathcal{U}(0, 2\pi)$ to create actions $\mathbf{a} = \alpha[\cos(\theta), \sin(\theta)]^\top$. We control the norm of the actions with the scalar α. For each value of α, we first apply the action $\mathbf{a}$ for one second, stop the motion, and then apply the inverse action $-\mathbf{a}$. We denote $\mathbf{q}^{fwd}$ and $\mathbf{q}^{rev}$ as the joint configurations of the robot after the forward and inverse actions, respectively. For each experiment run, we first measure the distance from the start state $\mathbf{q}^0$ to $\mathbf{q}^{fwd}$, and then the distance of $\mathbf{q}^{rev}$ to $\mathbf{q}^0$.

Our results are shown in Fig. 3. Ten random states from our demonstration data were chosen as initial states, and each algorithm (CAE, SCL, PCA) was used to compute joint velocities given a random action with increasing magnitude. The distance between the initial and final joint positions $||\mathbf{q}^{rev} - \mathbf{q}_0||_2$ increases with the magnitude of $\mathbf{a}$ for SCL because the maps begin to vary away from the initial position (the maps are state-conditioned). Our results suggest that SCL maps transformed by the Gram-Schmidt process are still soft reversibile despite Theorem 1 not accounting for this post processing. The error with PCA is constant because the map is fixed at all states.

V. User Study

We conduct two sets of user studies to validate the efficacy of SCL as a learnable action mapping approach. We chose to evaluate SCL in teleoperation because of previous research in assistive robotics [20], [12]. The first user study compares across the spectrum of action mapping approaches on two synthetic tasks that empirically could be solved in either a 2D or 3D action space. The second user study pushes the limits of SCL when compared to an assistive robotic's mode switching interface as well as a more advanced CAE model. The second user study task could not be solved with the first two principal components of PCA and was excluded. All experiments were conducted on a Kinova Gen3 lite robot with a control rate of 40 Hz [32]. Reported statistical significance used a 0.05 p-value. A Kruskal-Wallis test was performed before the reported Dunn test of significance.

A. Latent Action User Study

Our first user study compared SCL against other action mapping methods in the literature. We chose tasks that can be solved with 2D and 3D Cartesian control, which we refer to as *Table Cleaning* and *Pick and Place* in this section. Despite being simple, these tasks allow us to empirically evaluate the efficacy of SCL in general compared to existing action mapping approaches.

a) Experimental Set-up: The Table-Cleaning task is shown in Figure 9. Participants were asked to push five small objects on a table out of a designated boundary in 60 seconds.

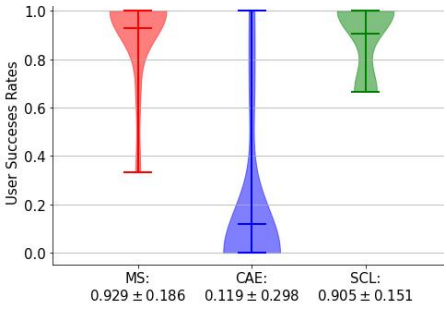

Fig. 6: User Success Rates.

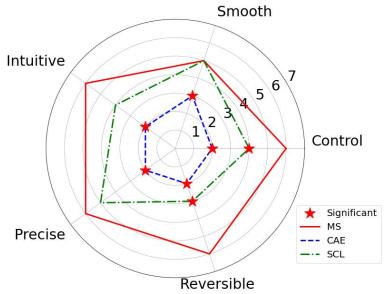

Fig. 7: Likert Scale Results (Median).

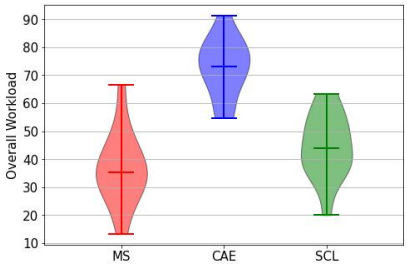

Fig. 8: NASA Workload Index.

Pouring User Study Results. Figure 6: Violin plots of completion with mean. Figure 7: Radar plot with stars indicating significance w.r.t. mode switching. Figure 8: Violin plots of NASA TLX scores with median.

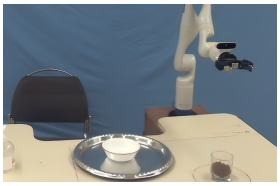

Fig. 9: Table Cleaning and Pouring Experiments.

We collected 5 demonstrations of circular scrubbing motions on a table for the experiment. Three trajectories were used to train the PCA, SCL and CAE models, and the remaining two were held out for validation. The total amount of training data was 509 ($\mathbf{q}$,$\dot{\mathbf{q}}$) tuples.

The Pick and Place task can be seen in Figure 1. The task was to reach and grasp the bottle and move it to another fixed, marked position on the table within 90 seconds. The bottle was initially placed on the black "x" marked on the table. Grasping was controlled by a trigger on the joystick controller. We collected eight demonstration trajectories, where six trajectories were for training and two for validation of each model. The six training trajectories accounted for 2001 ($\mathbf{q}$,$\dot{\mathbf{q}}$) tuples.

For each task, participants were given five minutes to practice, and two attempts at the task. There were 13 participants (4 females and 9 males, aged 23 - 33). We followed a single-blind experimental set-up, randomizing the order of models each participant used for each task.[3]

b) Models: We compared SCL against PCA [17], [18] and a conditional autoencoder (CAE) [12], [20]. Respectively, they represented a linear and non-linear approach for learning low-to-high dimensional mappings for control. Our CAE model is based on the open-source version of Karamcheti et. al. [22]. The conditional autoencoder and SCL maps were trained on demonstration data to fit the mean-squared error between the recorded and reconstructed joint velocities (i.e. with unsupervised learning). The SCL and CAE models were trained with the same data, learning parameters, and architectures described in Section IV.c. For each algorithm we chose the best model (out of ten) based on a validation loss for deployment. Participants used a 2-DOF joystick to provide low-dimensional control commands with inputs mapped as actions $\mathbf{a} = [a_0, a_1]^\top$, with $\|\mathbf{a}\| \leq 1$.

c) Results and Discussion: Figure 4 shows the success rates of the experiments. Our results suggest that PCA worked well for participants in the table-cleaning task, but it failed to perform consistently in the Pick and Place task. There were qualitative differences between the motions to pick up the bottle versus placing the bottle at the target location. PCA was forced to capture all variations in joint velocities for both pick and place motions with just two principal components. In comparison, SCL was free to adapt the subspace of joint velocities, given the current configuration of the manipulator. After reaching forward to grasp the bottle, the joint velocity commands available to participants (via output of action map $h(\mathbf{o}_r)$) changed to afford the placing motion, thanks to the state dependence. This adaptability resulted in SCL performing significantly better than the baselines in the Pick and Place task (Fig. 4).

The low success rate of the CAE model was attributed to two primary reasons during the experiments. First, the CAE internally uses affine mappings as opposed to linear mappings. When a user's input is $\mathbf{0} \in \mathbb{R}^2$ (no input), the joint velocity generated will be determined by the bias terms in the hidden layers of the model. SCL does not suffer from this problem, because it transforms actions with a linear map. Second, we observed that many CAE trials resulted in failure if users navigated to joint configurations outside of the training trajectory distribution. The robot's behavior was unpredictable in these states, and in many cases counter-productive (e.g. all actions further moved the end-effector away from the bottle). This issue of unpredictable behavior has been previously reported for CAEs [20]. In contrast, the soft reversibility property of SCL enabled participants to recover from configurations outside of the training distribution.

[3]The studies were approved by University of Alberta Research Ethics and Managements (Pro00054665).

B. Simulation Experiments for Human Prior Configuration

Prior to the second user study, we investigated ways to improve the CAE performance through additional loss functions. We re-implemented the *consistency*, *proportionality*, and *reversibility* loss functions proposed in Section 4.B of Li et al [23]. We performed a grid search over combinations of these loss functions as well as the sampling range for the *Proportionality* hyperparameter $\alpha \in [0.1, 1.0]$. We fixed the temperature parameter to $\gamma = 2.0$ in the *Consistency* loss.

We evaluated the different models in a simulation of a 5 DOF planar reaching task where targets varied along a single line. Trajectories were collected with inverse Jacobian control. For each configuration, we trained 10 models with the same hyperparameters as our first user study (see section IV.c) for 1000 epochs. Results are reported on the models with the best validation error. We compared against SCL with and without Gram-Schmidt orthonormalization.

Our experiment used a greedy user that at each step picked the best actions under the following loss:

$$\mathbf{a}^* = \min_{\mathbf{a} \sim \mathcal{N}(0,I)} ||\mathbf{q}^* - (\mathbf{q}_t + g_\theta(\mathbf{o}_r, \mathbf{a}))||_2, \qquad (10)$$

Where $\mathbf{q}^*$ is the goal joint configuration, and the decoder g_θ predicted joint velocities. Actions were selected from 4096 samples of an Isotropic Gaussian at each time step.

We report the distribution of average errors with respect to the end $\mathbf{q}_T$ and returning to the start states $\mathbf{q}_0$ on 100 test trajectories for each of the 10 evaluations per hyperparameter configuration in Figure 5. We found statistical significance between CAE results and both SCL versions. The performance between SCL and some configurations of CAE to reach the end state were comparable, but results suggest that even with these additional loss terms, SCL is better at reversing actions, as our theory suggested. We repeated this procedure with the collected pouring trajectories. Our user study uses the proportionality with $\alpha = 1.0$ and reversibility loss functions which worked best in simulation experiments.

C. Assistive Robotic User Study

Our second user study compares SCL to systems that have previously been used for assistive robotics including CAE and a mode switching system (MS) [34]. We set up a *Pouring* task (Figure 9), where participants had to pick up a cup full of beans, pour them into a bowl, and then replace the cup in a designated location on the table within two minutes. This task was highly nonlinear, consisting of several motions that involved both translation and rotation of the robot's end-effector. We collected 10 demonstration trajectories, where 8 were used for training (5200 training tuples) and 2 for validation (1191 validation tuples).

This study included 14 participants (ages 22 - 51, 2 female). Participants were given four two-minute practice trials with each of the control systems to familiarize themselves with the action mappings. We then collected results for each method over three test trials.

We report the average success rates of participants in Figure 6. Generally, we found users could pick up the glass with any interface, but struggled to pour with the CAE model thus failing the task. Both SCL and MS achieve statistically significant results compared to CAE. Between SCL and MS there were no statistical differences suggesting those methods performed comparably.

In addition, we also collected user subjective opinions which included a 5-question Likert scale survey (1 low and 7 high) featured in Figure 7. We found that only Control and Reversibility were statistically significant by Dunn's Test. We also collected the NASA Workload Index [35] featured in Figure 8. Again, by Dunn's test we found statistical significance between CAE and both SCL and MS.

Although our results show the promise of SCL in assistive robotic settings, further work is necessary to improve the user experience. With a single mode, SCL performed comparably to mode switching, which exposed three control modes (x-y, z-yaw, and roll-pitch, following [34]). On average, users switched modes 17.55 ± 3.91 times while using the mode switching interface in the pouring task. We conjecture that on more complex tasks (e.g. dynamic tasks, or tasks requiring simultaneous orientation and position control), the benefits of SCL would be more pronounced over typical assistive robotic systems. In previous works, tasks have consisted of several sub-tasks and were achievable with CAE models. Interestingly, our results would seem to disagree with existing work on CAE models for shared autonomy. One explanation could be that previous research included additional heuristics to account for CAE limitations. As discussed in several instances, we have found the CAE models move without user input. One possible solution would be sending 0 velocity if the action norm is small, but our focus was on achieving properties of the interface without such techniques.

VI. Conclusion

In this paper, we proposed SCL maps as a more controllable action mapping method which lies between a fixed linear map and an end-to-end nonlinear model. The local linearity of SCL maps guarantees proportional outputs and provides an intuitive control interface. We showed that SCL satisfies *soft reversibility*, where following an action with its inverse is guaranteed to bring the robot closer to its initial joint configuration. Soft reversibility enables users to more easily navigate the robot pose space compared to alternative mapping approaches. Our teleoperation results show the promise of SCL compared to alternative latent action models; however, future research is necessary to adapt SCL to related domains such as reinforcement learning. Given PCA's effectiveness in our first user study, we also believe that understanding action manifolds (i.e. when non-linearity is or is not necessary) is a promising direction.

References

[1] Y. Huang and Y. Sun, "Accurate robotic pouring for serving drinks," CoRR, vol. abs/1906.12264, 2019. [Online]. Available: http://arxiv.org/abs/1906.12264

[2] S. Manschitz, J. Kober, M. Gienger, and J. Peters, "Learning to unscrew a light bulb from demonstrations," in ISR/Robotik 2014; 41st International Symposium on Robotics, 2014, pp. 1–7.

[3] K. Mülling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis," The International Journal of Robotics Research, vol. 32, no. 3, pp. 263–279, 2013. [Online]. Available: https://doi.org/10.1177/0278364912472380

[4] S. Tosatto, G. Chalvatzaki, and J. Peters, "Contextual latent-movements off-policy optimization for robotic manipulation skills," in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 10 815–10 821.

[5] G. Li, L. Shi, Y. Chen, Y. Gu, and Y. Chi, "Breaking the sample complexity barrier to regret-optimal model-free reinforcement learning," CoRR, vol. abs/2110.04645, 2021. [Online]. Available: https://arxiv.org/abs/2110.04645

[6] Y. Chandak, G. Theocharous, J. Kostas, S. M. Jordan, and P. S. Thomas, "Learning action representations for reinforcement learning," CoRR, vol. abs/1902.00183, 2019. [Online]. Available: http://arxiv.org/abs/1902.00183

[7] L. V. Herlant, R. M. Holladay, and S. S. Srinivasa, "Assistive teleoperation of robot arms via automatic time-optimal mode switching," in 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 2016, pp. 35–42.

[8] F. Routhier and P. S. Archambault, "Usability of a joystick-controlled six degree-of-freedom robotic manipulator." in RESNA ANNUAL Conference 2010, 2010.

[9] A. Colomé, G. Neumann, J. Peters, and C. Torras, "Dimensionality reduction for probabilistic movement primitives," in 2014 IEEE-RAS International Conference on Humanoid Robots, Nov. 2014, pp. 794–800, iSSN: 2164-0580.

[10] Z. He and M. Ciocarlie, "Discovering synergies for robot manipulation with multi-task reinforcement learning," in 2022 International Conference on Robotics and Automation (ICRA). IEEE Press, 2022, p. 2714–2721. [Online]. Available: https://doi.org/10.1109/ICRA46639.2022.9812170

[11] N. Chen, M. Karl, and P. van der Smagt, "Dynamic movement primitives in latent space of time-dependent variational autoencoders," in 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), 2016, pp. 629–636.

[12] D. P. Losey, H. J. Jeon, M. Li, K. Srinivasan, A. Mandlekar, A. Garg, J. Bohg, and D. Sadigh, "Learning latent actions to control assistive robots," CoRR, vol. abs/2107.02907, 2021. [Online]. Available: https://arxiv.org/abs/2107.02907

[13] K. P. F.R.S., "Liii. on lines and planes of closest fit to systems of points in space," The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, vol. 2, no. 11, pp. 559–572, 1901. [Online]. Available: https://doi.org/10.1080/14786440109462720

[14] H. Hotelling, "Analysis of a complex of statistical variables into principal components." Journal of Educational Psychology, vol. 24, pp. 498–520, 1933.

[15] M. Santello, M. Flanders, and J. Soechting, "Postural hand synergies for tool use," The Journal of Neuroscience, vol. 18, pp. 10 105–15, 12 1998.

[16] G. Matrone, C. Cipriani, M. C. Carrozza, and G. Magenes, "Real-time myoelectric control of a multi-fingered hand prosthesis using principal components analysis," Journal of neuroengineering and rehabilitation, vol. 9, p. 40, 06 2012.

[17] A. Odest and O. Jenkins, "2d subspaces for user-driven robot grasping," Robotics, Science and Systems Conference: Workshop on Robot Manipulation, 2007.

[18] 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.

[19] M. Ciocarlie and P. Allen, "Hand posture subspaces for dexterous robotic grasping," I. J. Robotic Res., vol. 28, pp. 851–867, 06 2009.

[20] D. P. Losey, K. Srinivasan, A. Mandlekar, A. Garg, and D. Sadigh, "Controlling Assistive Robots with Learned Latent Actions," IEEE Int. Conf. Robotics and Automation (ICRA), 2020.

[21] H. Jun Jeon, D. Losey, and D. Sadigh, "Shared Autonomy with Learned Latent Actions," in Robotics: Science and Systems XVI. Robotics: Science and Systems Foundation, Jul. 2020. [Online]. Available: http://www.roboticsproceedings.org/rss16/p011.pdf

[22] S. Karamcheti, A. J. Zhai, D. P. Losey, and D. Sadigh, "Learning visually guided latent actions for assistive teleoperation," in Proc. Conf Learning for Dynamics and Control, 2021, pp. 1230–1241.

[23] M. Li, D. P. Losey, J. Bohg, and D. Sadigh, "Learning user-preferred mappings for intuitive robot control," IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS), pp. 10 960–10 967, 2020.

[24] A. Allshire, R. Martín-Martín, C. Lin, S. Manuel, S. Savarese, and A. Garg, "LASER: learning a latent action space for efficient reinforcement learning," CoRR, vol. abs/2103.15793, 2021. [Online]. Available: https://arxiv.org/abs/2103.15793

[25] M. Watter, J. T. Springenberg, J. Boedecker, and M. Riedmiller, "Embed to control: A locally linear latent dynamics model for control from raw images," in Proc. of the 28th Int. Conf. Neural Information Processing Systems, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 2746–2754.

[26] Y. Zhao and C. C. Cheah, "Neural network control of multifingered robot hands using visual feedback," IEEE Transactions on Neural Networks, vol. 20, no. 5, pp. 758–767, 2009.

[27] S. Lyu and C. C. Cheah, "Data-driven learning for robot control with unknown jacobian," Automatica, vol. 120, p. 109120, 2020.

[28] S. Lyu and C. C. Cheah, "Vision based neural network control of robot manipulators with unknown sensory jacobian matrix," in 2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), 2018, pp. 1222–1227.

[29] M. Przystupa, M. Dehghan, M. Jägersand, and A. Mahmood, "Analyzing neural jacobian methods in applications of visual servoing and kinematic control," ArXiv, vol. abs/2106.06083, 2021.

[30] E. Schmidt, "Zur theorie der linearen und nichtlinearen integralgleichungen," Mathematische Annalen, vol. 63, no. 4, pp. 433–476, Dec 1907. [Online]. Available: https://doi.org/10.1007/BF01449770

[31] J. J. Craig, Introduction to Robotics: Mechanics & Control. Addison-Wesley Publishing Company, 1986.

[32] Kinova Robotics, "Ros kortex." [Online]. Available: https://github.com/Kinovarobotics/ros_kortex

[33] H. Gouk, E. Frank, B. Pfahringer, and M. J. Cree, "Regularisation of neural networks by enforcing Lipschitz continuity," Mach Learn, vol. 110, no. 2, pp. 393–416, Feb. 2021. [Online]. Available: https://doi.org/10.1007/s10994-020-05929-w

[34] B. A. Newman, R. M. Aronson, S. S. Srinivasa, K. Kitani, and H. Admoni, "Harmonic: A multimodal dataset of assistive human–robot collaboration," The International Journal of Robotics Research, vol. 41, no. 1, pp. 3–11, 2022. [Online]. Available: https://doi.org/10.1177/02783649211050677

[35] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," in Advances in psychology. Elsevier, 1988, vol. 52, pp. 139–183.