Separating Manners of Execution of Forceful Tasks when Learning from Multimodal Demonstrations

Yordan Hristov
School of Informatics
University of Edinburgh
Email: yordan.hristov@ed.ac.uk

Subramanian Ramamoorthy
School of Informatics
University of Edinburgh
Email: s.ramamoorthy@ed.ac.uk

***Abstract*—As physically embodied agents, robots can use a diversity of modalities to sense and affect their surrounding environment. Unfortunately, such high-dimensional streams of information are hard for human comprehension, making human-robot interaction and teaching nontrivial. Thus, we pose the problem of interpretable learning from demonstration as an optimisation one over a probabilistic generative model. To account for the high-dimensionality of the data, a high-parameter neural network is chosen to represent the model. Its latent variables are explicitly aligned with high-level notions and concepts, manifested in the demonstrations. We show that such alignment can be achieved through the usage of restricted user labels. The method is evaluated in the context of table-top dabbing (pressing against a surface with a sponge) with a PR2 robot which provides us with visual information, arm joint positions and arm joint efforts.**

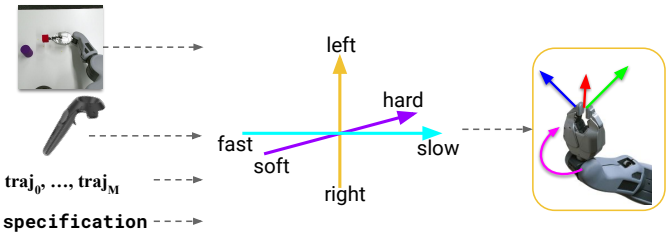

Fig. 1: User demos through teleoperation and a variety of modalities **(left)** are used to fit a common low-level disentangled manifold **(middle)** which contributes in an interpretable way to the generative process for robot behaviour **(right)**.

I. Introduction

Learning from demonstration (LfD) [3] is a commonly used paradigm where an inexperienced demonstrator desires to teach a robot how to perform a particular task in its environment. Most often this is performed through a combination of kinaesthetic teaching and supervised learning—imitation learning [24]. However, such approaches do not allow for elaborations and corrections from the demonstrator to be seamlessly incorporated. As a result new demonstrations are required when either the demonstrator changes the task specification or the agent changes its context—typical scenarios in the context of interactive task learning [19]. Such problems mainly arise because the demonstrator and the agent reason about the world by using *notions* and mechanisms at *different levels of abstraction*. A modifiable LfD setup requires establishing a mapping from the high-level notions humans use—e.g. spatial concepts, different ways of applying force—to the low-level perceptive and control signals robot agents utilise—e.g. joint angles, efforts and camera images. With this in place, any constraints or elaborations from the human operator, must be mapped to a behaviour on the agent's side that is consistent with the semantics of the operator's desires.

More concretely, we need to be able to ground [28, 10] the specifications and symbols used by the operator in the actions and observations of the agent. Often the actions and the observations of a robot agent can be very high-dimensional—high DoF kinematic chains, high image resolution, etc.—making the symbol grounding problem non-trivial. However, the concepts we need to be able to ground lie on a much-lower-dimensional manifold, contained in the high-dimensional data space [9]. For example, the concept of pressing softly against a surface technically lives in the 7 DoF real-valued space of joint efforts, spread across multiple time steps. However, the idea of what a soft press must look like can easily be summarised conceptually. The focus of this work is finding a nonlinear mapping (represented as a high-parameter neural model) between the low-dimensional manifold and the high-dimensional data space. Moreover, we argue apart from finding such a mapping, we can also shape and refine the low-dimensional manifold by imposing specific biases and structures on the neural model's architecture and training regime.

In this paper, we propose a framework which allows human operators to teach a PR2 robot about different spatial and force-related aspects of a tabletop dabbing task. Our main contributions are:

- A learning method which incorporates information from multiple high-dimensional modalities—vision, joint angles, joint efforts—in order to instill a disentangled low-dimensional manifold. We use weak expert labels during the optimisation process, the manifold eventually aligns with '*human common sense*' notions in a controller way without the need of post-hoc interpretation—see Figure 2 (left).
- A quantitative and qualitative evaluation of the eventual generative model and how closely do behaviours sampled from it, conditioned on an optional specification, align with the demonstrated concepts by the human—see Figure 2 (right).

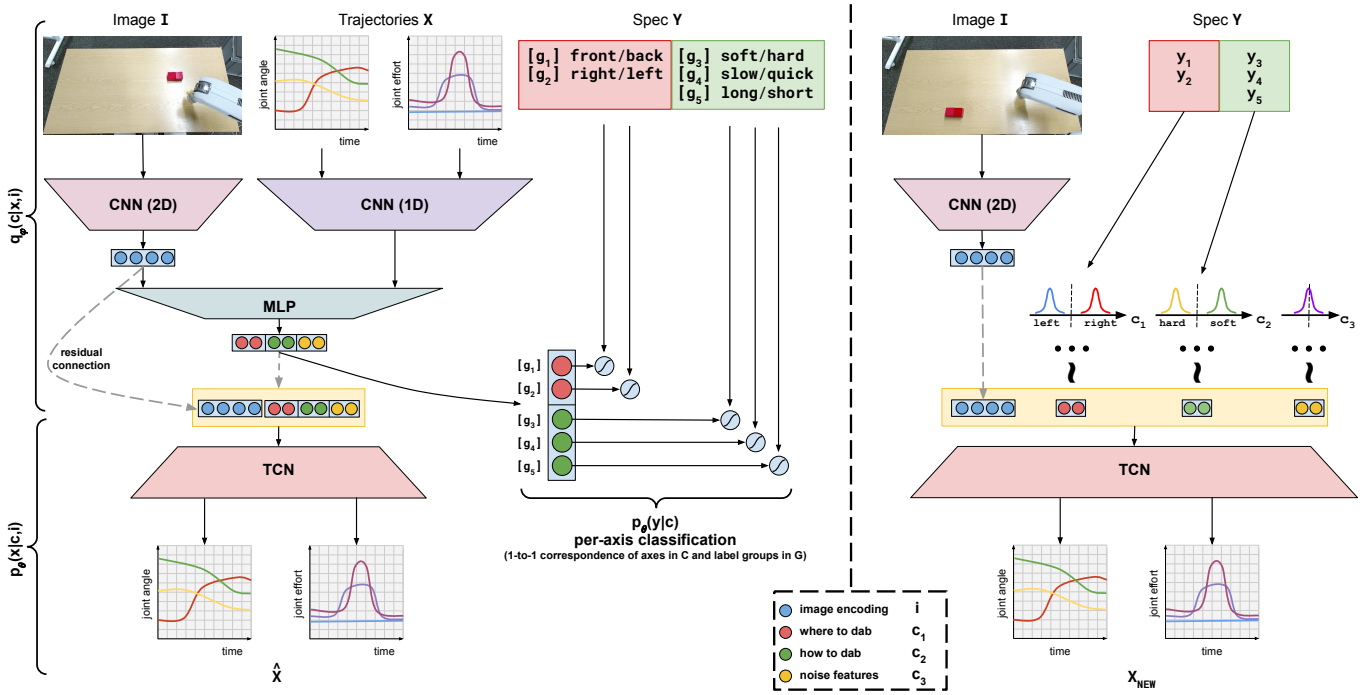

Fig. 2: Overall model sketch - multiple different modalities are encoded to a common latent space $\mathbf{c}$ which is used both for reconstructing trajectories $\mathbf{x}$ and predicting weak labels $\mathbf{y}$, coming from the demonstrator **(left)**; final generative model for new robot trajectories, given an environment context image and an optional specification. The optional specification plays the role of a hierarchical prior over some of the latent dimensions—we know what they mean as an artefact of the latent-axis-concept-group alignment—e.g. an axis in $\mathbf{c}_1$ determines if we'll dab to the `left` or to the `right` of the cube, an axis in $\mathbf{c}_2$ determines if we'll dab `softly` or `hardly`. Unaligned axes—$\mathbf{c}_3$—sample from their Gaussian prior **(right)**.

II. Relevant Work

Dynamic movement primitives [26] are commonly used to model dynamical systems as point-mass attractors with a non-linear forcing term. However, due to their high-dimensionality they don't easily scale outside of their training distribution. Probabilistic movement primitives [21] provide a way to sample a more diverse set of trajectories at test time by conditioning on the training data distribution, instead of using fixed control parameters that are fit over a single demonstration. However, the resultant control parameters can still be high-dimensional and unintuitive to manipulate. Performing dimensionality reduction on the fit control parameters [8] or imposing hierarchical priors over the parameter space [25] are both ideas present in the literature as a means of making the high-dimensional parameter space more meaningful to the human demonstrator, for the sake of having a clearer idea of how changes in the optimised parameters results in deterministic changes in the generated behaviours. However, most of these approaches limit themselves to end-effector trajectories, or at most making use of visual information [18].

Simultaneously, there has been a steady push in the representation learning community for methods which utilise high-parameter neural models to learn disentangled low-dimensional representation by either tuning the optimised loss function [11, 6], imposing crafted priors over the latent space [7, 2] or using weak supervision [13, 12, 22]. A disentangled representation in this context is any representation where independence between data-generative factors of variation is preserved as such in the learned latent codes—e.g. the size of an object in a given image is independent of its color and position, etc. While being able to produce representations which align with 'human common sense', most of these approaches focus on modelling visual data and respectively visually-manifested concepts, with minor exceptions—e.g. modelling motion capture data [2]. Disentangled representations from multiple modalities, which is much closer to the robotics community, is what we are focusing on in this paper.

The work of Noseworthy et al. [20] does explore disentangling task parameters from manner of execution parameters, in the context of pouring. They utilise an adversarial training regime, which helps for the better separation of the two types of parameters in the latent space. However, it is assumed that the task parameters are known a priori and only a single data modality is used—the end effector orientation along the Z axis. Moreover, interpretation of the learned codes is done post-training by perturbing latent axis values and qualitatively inspecting generated trajectories (standard evaluation technique for models which are trained unsupervised). We argue that through the use of weak discrete labels we have finer control and better guarantees about the meaning of the latent dimensions.

III. IMAGE-CONDITIONED TRAJECTORY & LABEL MODEL

We want to control **where** and **how** does a robot press against a table-top surface by using a set of coarse labels as a **specification**—e.g. *"press softly and slowly behind the cube in the image"*. Let, in that context, $\mathbf{x}$ denote a $K \times T$ dimensional trajectory for K robot joints and a fixed length T, $\mathbf{y}$ denote a set of discrete labels semantically grouped in N label groups $\mathcal{G} = \{g_1, \ldots, g_N\}$ (equivalent to multi-label classification problem) and $\mathbf{i}$ denote an RGB image[1]. The labels $\mathbf{y}$ describe qualitative properties of $\mathbf{x}$ and $\mathbf{x}$ with respect to $\mathbf{i}$—e.g. `left` dab, `right` dab, `hard` dab, `soft` dab, etc. We aim to model the distribution of demonstrated robot-arm trajectories $\mathbf{x}$ and corresponding user labels $\mathbf{y}$, conditioned on a visual environment context $\mathbf{i}$. This problem is equivalent to that of structured output representation [29, 27, 5]—finding a one-to-many mapping from $\mathbf{i}$ to $\{\mathbf{x}, \mathbf{y}\}$ (one image can be part of the generation of many robot trajectories and labels). For this we use a conditional generative model, whose latent variables $\mathbf{c}$ can accommodate the upper-mentioned mapping—see Figure 3. The meaning behind the different types of latent variables—$\mathbf{c}_s, \mathbf{c}_e$ and $\mathbf{c}_u$—is elaborated on in section IV. As a result we want the variability in the trajectory and label data to be concisely captured by $\mathbf{c}$. We design the dimensionality of the latent variables to be much lower than the dimensionality of the data. Therefore, $\mathbf{c}$ is forced to represent abstract properties of the robot behaviour which still carry enough information for the behaviour to be generated—absolute notions like speed, effort, length, and relative spatial notions which are grounded with respect to the visual context. More concretely, given a latent representation of a demonstration and a context, we should be able to tell whether the robot pressed `softly` or `hard` against the table, whether it pressed `in front of` or `behind` the visual landmark contained in $\mathbf{i}$.

Another way to think of $\mathbf{c}$ is as a continuous version of $\mathbf{y}$ which can therefore incorporate nuances of the same label. In the existing literature discrete labels are usually represented as discrete latent variables—e.g. digit classes [17]. However, the discrete labels humans use are a rather crude approximation to the underlying continuous concepts—e.g. we have the notion for a `soft` and a `softer` dab even though both would be labelled as `soft`. For this reason we use continuous latent variables, learned from discrete labels, to represent these subjective concepts.

The joint distribution over $\mathbf{x}$ and $\mathbf{y}$, conditioned on $\mathbf{i}$, is modelled according to Equations 1 and 2. We place an isotropic Gaussian prior over the latent variables $\mathbf{c}$ which are independent of the conditioned image.

$$p(\mathbf{x}, \mathbf{y}|\mathbf{i}) = \int p(\mathbf{x}, \mathbf{y}|\mathbf{i}, \mathbf{c}) p(\mathbf{c}) d\mathbf{c} \tag{1}$$

$$p(\mathbf{x}, \mathbf{y}|\mathbf{i}, \mathbf{c}) = p(\mathbf{x}|\mathbf{c}, \mathbf{i}) p(\mathbf{y}|\mathbf{c}) \tag{2}$$

[1]What $\mathbf{i}$ actually represents is a lower-dimensional version of the original RGB image. The parameters of the image encoder are jointly optimised with the parameters of the recognition and decoder networks

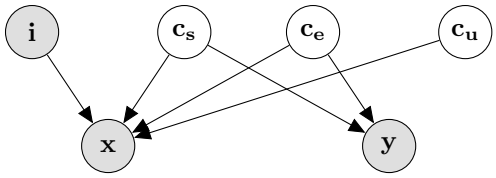

Fig. 3: The lower-dimensional encoding $\mathbf{i}$ of the environment context $\mathbf{I}$ is observed. Conditioned on $\mathbf{i}$ and the latent variables $\mathbf{c}$, sampled from the prior over $\mathbf{c}$, we get a distribution over possible robot trajectories $\mathbf{x}$ and user labels $\mathbf{y}$.

We choose to represent the distribution over $\mathbf{x}$ and $\mathbf{y}$ as Gaussian and Multinomial distributions respectively. The parameters of these distributions are defined as nonlinear functions (of the image context and latent variables) which are represented as the weights θ of a neural network p_θ—$\boldsymbol{\mu}_\theta(\cdot)$ is a vector of means, $\boldsymbol{\sigma}_\theta(\cdot)$[2] is a vector of standard deviations and $\boldsymbol{\pi}_\theta(\cdot)$ is a vector of probabilities.

$$p_\theta(\mathbf{x}|\mathbf{c}, \mathbf{i}) = \mathcal{N}(\boldsymbol{\mu}_\theta(\mathbf{c}, \mathbf{i}), \boldsymbol{\sigma}_\theta(\mathbf{c}, \mathbf{i})) \tag{3}$$

$$p_\theta(\mathbf{y}|\mathbf{c}) = \text{Cat}(\mathbf{y}|\boldsymbol{\pi}_\theta(\mathbf{c})) \tag{4}$$

The parameters of p_θ are optimised by maximising the variational lower bound (VLB) of the data distribution—see Equation 5. We additionally optimise a recognition network q_ϕ, parametrised by ϕ, which acts as an approximation to the posterior of $\mathbf{c}$, also modelled as a Gaussian.

$$\begin{aligned} \log p(\mathbf{x}, \mathbf{y}|\mathbf{i}) \geq \mathbb{E}_{q_\phi(\mathbf{c}|\mathbf{x},\mathbf{i})}(\log p_\theta(\mathbf{x}, \mathbf{y}|\mathbf{c}, \mathbf{i})) - \\ D_{KL}(q_\phi(\mathbf{c}|\mathbf{x}, \mathbf{i})||p(\mathbf{c})) \end{aligned} \tag{5}$$

$$q_\phi(\mathbf{c}|\mathbf{x}, \mathbf{i}) = \mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x}, \mathbf{i}), \boldsymbol{\sigma}_\phi(\mathbf{x}, \mathbf{i})) \tag{6}$$

The recognition network q_ϕ is used to find good candidates of $\mathbf{c}$, for given $\mathbf{x}$ and $\mathbf{i}$, thus making the optimisation of θ, tractable [16]. The posterior is conditioned on the environment context as well, since $\mathbf{i}$ and $\mathbf{c}$ are conditionally dependent, once $\mathbf{x}$ is observed [14]. Intuitively, we want for the some of the latent variables to represent relative spatial concepts—e.g. `left of` the cube in the image. For such concepts, under the same robot trajectory $\mathbf{x}$ we should get different latent codes, given different contexts $\mathbf{i}$. The fact that q_ϕ is not conditioned on $\mathbf{y}$ too means the recognition network has to utilise both the input image and the trajectory, in order to be able to "reconstruct" the trajectory **and the labels** (note that we don't reconstruct the image). The omission of $\mathbf{y}$ from q_ϕ also means that we might not be able to fully close the gap between the data distribution and its VLB (making the inequality into an equality), leading to a less efficient optimisation procedure. That is mainly owing to the fact that we use weak supervision—only for some demonstrations labels $\mathbf{y}$ are present. The full derivation of and comments on the optimised VLB is provided in the supplementary chapters.

[2]In practice we only set up μ as a function of x, parametrised by θ, and fix σ for numerical stability during optimisation.

IV. Weak Labelling and User Specification

A. Interpretability through Weak Labels

Even though the latent variables are constrained to be useful both for the tasks of trajectory and label generation, nothing constrains them to carry the intuitive absolute and relative notions described in section III in a factorised fashion—e.g. c_1 corresponds to encoding whether the robot pressed `softly` or `hard` against the table, c_2 corresponds to whether the robot pressed `quickly` or `slowly` against the table, etc. Those notions would still be contained in $\mathbf{c}$ but nothing guarantees that they will be contained in an **interpretable** way—namely $\mathbf{c}$ to be **disentangled**. We achieve disentanglement in the latent space by establishing a one-to-one correspondence between a subset of the latent axes—$\mathbf{c}_s, \mathbf{c}_e$—and the concept groups in $\mathcal{G}$ [12, 13]. The rest of the latent dimensions—$\mathbf{c}_u$—encode features in the data which don't align with the semantics of the labels and their concept groups but are still necessary for good trajectory reconstruction. Under these assumptions the label likelihood from Equation 4 is rendered as:

$$p_\theta(\mathbf{y}|\{\mathbf{c}_s, \mathbf{c}_e\}) = \prod_j^{|\mathcal{G}|} \mathbf{1}_{\{y_j \neq \emptyset\}} \text{Cat}(y_j|\boldsymbol{\pi}_j(c_j)) \tag{7}$$

Labelling is **weak/restricted** since each demonstration is given just a single label from a single concept group. This is meant to represent a more natural LfD scenario, where the main task of the expert is still to perform a number of demonstrations, rather than exhaustively annotate each demonstration with relevant labels from all groups. For example, a demonstrator might say *"this is how you press softly"* and do the demonstration. The shown behaviour might have also been `slow` and `short` but the human was not explicitly thinking about these notions while demonstrating. The missing label values for some of the concept groups is incorporated with an indicator function in Equation 7.

B. Condition on User Specification

Apart from being used in the optimisation objective, the weak labels are also used in an additional conditioning step at test time. Currently, the generative process we have consists of first sampling values for $\mathbf{c}$ from its Gaussian prior, passing those, together with $\mathbf{i}$ though p_θ in order to get distributions over $\mathbf{x}$ and $\mathbf{y}$, from which we can sample. However, what we are rather interested is being able to incorporate provided information about the labels $\mathbf{y}$ into the sampling of the semantically aligned parts of $\mathbf{c}$. Specifically, we are interested in being able to generate robot behaviours which are consistent with fixed values for $\mathbf{y}$ (what we call a **user specification**). Post-optimising the θ and ϕ parameters, we choose to approximate the posterior over c_j for each label i in each concept group j with a set of Gaussian distributions $\mathcal{N}(\mu_j^{(i)}, \sigma_j^{(i)})$. We use Maximum Likelihood Estimation for $\mu_j^{(i)}$ and $\sigma_j^{(i)}$ over the j-th dimension of a set of samples from $\mathbf{c}$. The samples are weighted with the corresponding label likelihood $p_\theta(y_i|\boldsymbol{\pi}_\theta^{(j)}(c_j))$. As a result, the process of generating a trajectory $\mathbf{x}$ is additionally conditioned on an optional user specification $\mathbf{y} = \{y_1, \ldots, y_{|\mathcal{G}|}\}$ and is governed by Equations 8 and 9.

$$c_j \sim p(c_j|y_j) = \begin{cases} \mathcal{N}(0,1), & \text{if } y_j = \emptyset \text{ or } j > |\mathcal{G}| \\ \mathcal{N}(\mu_j^{(i)}, \sigma_j^{(i)}), & \text{otherwise} \end{cases} \tag{8}$$

$$\mathbf{x} \sim p_\theta(\mathbf{x}|\mathbf{i}, \mathbf{c}, \mathbf{y}) = p_\theta(\mathbf{x}|\mathbf{i}, \mathbf{c}) \prod_j^{|\mathbf{c}|} p(c_j|y_j) \tag{9}$$

Conditioning on a user specification $\mathbf{y}$ is optional. If no label is given for group j, then the corresponding semantically-aligned latent dimension c_j also samples from the Gaussian prior. This revised generative process for $\mathbf{x}$ is graphically described by Figure 2 (right) and Figure 3 (after reversing the direction of the edges that lead from $\mathbf{c}_1$ and $\mathbf{c}_2$ to $\mathbf{y}$.

V. Methodology

In terms of a concrete model architecture, our model is a Conditional Variational Autoencoder (CVAE) [27] with an encoding network q_ϕ and a decoding network p_θ. The parameters of both networks are optimised jointly using methods for amortised variational inference and a stochastic gradient variational bayes estimation [16][3].

A. Encoding Networks

Due to the diversity of modalities that we want to use, q_ϕ is implemented as a combination of 2D convolutions (for the image input), 1D convolutions (for the trajectory input) and an MLP that brings the the output of the previous two modules to the common concept manifold $\mathbf{c}$. We additionally have a set of linear classifiers, each of which sources its input form a single dimension in $\mathbf{c}$. Even though we think of our model as a CVAE, the only part of the data that we are auto-encoding are the robot trajectories. During the parameter optimisation the images serve only as an input and the weak labels as outputs. In principle, assuming we had no labels $\mathbf{y}$, nothing prevents the model from ignoring the image modality and learning what a standard VAE would. This is sometimes accounted for by introducing adversarial terms in the optimised loss function which force different parts of latent space to pay attention to different input modalities [20]. However, in our case we can utilise the weak labels that the demonstrator has given us, some of which can only be inferred if information both from the trajectory and image inputs has been used—the labels of relative spatial nature. The only potential problem might be that since the labels are weak, the likelihood function that we are trying to maximise—Equation 7—won't be active for most of the time. We account for that by introducing a coefficient γ for the label likelihood term—see Equation 10.

[3]The models are implemented in PyTorch [1] and optimised using the Adam optimiser [15]

$$\min_{\theta,\phi,\mathbf{w}} \mathcal{L}(\mathbf{x},\mathbf{y},\mathbf{I}) = \beta D_{KL}(\underbrace{q_\phi(\mathbf{c}|\mathbf{x},\mathbf{I})}_{\substack{\text{amortised}\\\text{posterior}}} || \underbrace{p_\theta(\mathbf{c})}_{\text{prior}}) + \alpha \mathbb{E}_{q_\phi(\mathbf{c}|\mathbf{x},\mathbf{I})}(\log \underbrace{p_\theta(\mathbf{x}|\mathbf{c},\mathbf{I})}_{\text{MSE}}) + \gamma \sum_{i}^{|\mathcal{G}|} \underbrace{\mathbf{1}_{\{y_i \neq \emptyset\}} H(c_i \mathbf{w}_i^T, y_i)}_{\text{SCE}} \tag{10}$$

B. Decoding Network

For the decoding network we implement a temporal convolutional network [4] which is often used in seq2seq models. It takes a sequence of length T and transforms it to another sequence of length T by feeding it through a series of dilated convolutions. However, the concatenation $\mathbf{h}$ of a single concept embedding $\mathbf{c}$ and a single image encoding is equivalent to a sequence of length 1. Thus, we tile the concatenated vector T times and to each instance of that vector $\mathbf{h}_i, i \in \{1,\ldots,T\}$, we attach a time dimension $t_i = \frac{i}{T}$. This broadcasting technique has previously shown to achieve better disentanglement in the latent space both on visual [30] and time-dependent data [20]. As mentioned in section III, we only set up the mean μ_θ of the predicted distribution over $\mathbf{x}$ to be a non-linear function of $[\mathbf{h};\mathbf{t}]$ while σ is fixed, for better numerical stability during training. This allows us to use the $L2$ distance—Mean Squared Error (MSE)—between the inferred trajectories μ_θ (we use the mean as a reconstruction) and the ground truth ones.

C. Label Predictors

For each concept group g_j we take values from the corresponding latent axis c_j feed it through a single linear layer, with a softmax activation function, to predict a probability vector $\boldsymbol{\pi}^{(j)}$. Maximising the label likelihood is realised as a Softmax-Cross Entropy (SCE) term in the loss function. Optimising this term gives us better guarantee that the some of the latent axes will be semantically aligned with the notions in the label groups. The fact that some labels might be missing is accounted for with an indication function which calculates $\nabla\theta, \nabla\phi$ with respect to the SCE, only if y_j is present.

D. Weighted Loss

The full loss function that we optimise is presented in Equation 10. Instead of maximising the VLB (Equation 5), we choose to minimise its negation. Notably, we have three main terms—the trajectory MSE (equiv. to the negative trajectory log likelihood), the label SCE (equiv. to the negative weak label log likelihood) and the KL divergence between the fit amortised posterior over $\mathbf{c}$ and its standard Gaussian prior. The concrete values of the three coefficients—α, β, γ—are discussed in the Supplementary Materials.

VI. Experiments

Our experiments on a physical PR2 robot show that under the proposed model we can more reliably sample behaviours, consistent with a given spatial or force-related user specification, as compared to a baseline VAE model.

The setup used in the experiments consists of a PR2 robot, an HTC Vive controller and headset, and a tabletop with a single object on it—a red cube. Multiple dabbing motions on the surface of the table are demonstrated by teleoperating the right-arm end effector of the robot. The latter is achieved by mapping the pose of the end effector with respect to the robot's torso to be the same as the pose of the controller (held by the demonstrator) with respect of the headset (behind the demonstrator in Figure 4).

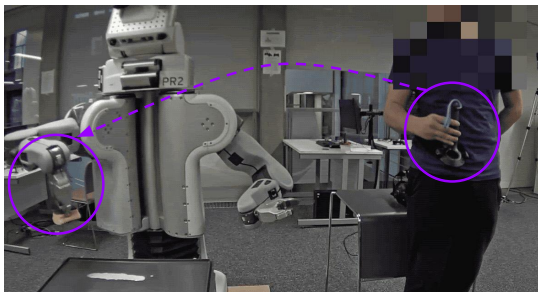

Fig. 4: Physical setup for teleoperating a PR2 end-effector through an HTV Vive controller.

A. Data

The data captured from the PR2, over all 200 total demonstrations, consists of 3 main modalities (+ weak labels):

- Scene image from a Kinect2 sensor—initial frame of each demonstration; 128x128 pixels, with RGB channels;
- Joint angle position information from the 7 joints of the right arm - fixed size trajectory of 240 time steps;
- Joint effort information from the 7 joints of the right arm - fixed size trajectory of 240 time steps;
- 5 groups of discrete weak labels, describing either the spatial or force-related aspects of a demonstration;

Each demonstration has a single label attached to it. In total we have 4 spatial symbols—where in the image, with respect to the red cube, do we dab—and 6 force-related symbols—how do we dab:

spatial (where) Initial Image + Joint Positions	effort (how) Joint Efforts
left & right **front & behind**	**soft & hard** **short & long** **quick & slow**

TABLE I: All labels given in the demonstrations together with the modalities necessary for the inference of the labels.

Figure 5 provides a graphical depiction demonstrating the differences between the different force-relate groupings qualitatively.

All trajectories are standardised to be of fixed length—the length of the longest trajectory—by padding them with the last value for each channel/joint. Additionally, both the joint positions and efforts are augmented with random noise or by randomly sliding them leftwards (simulating an earlier start)

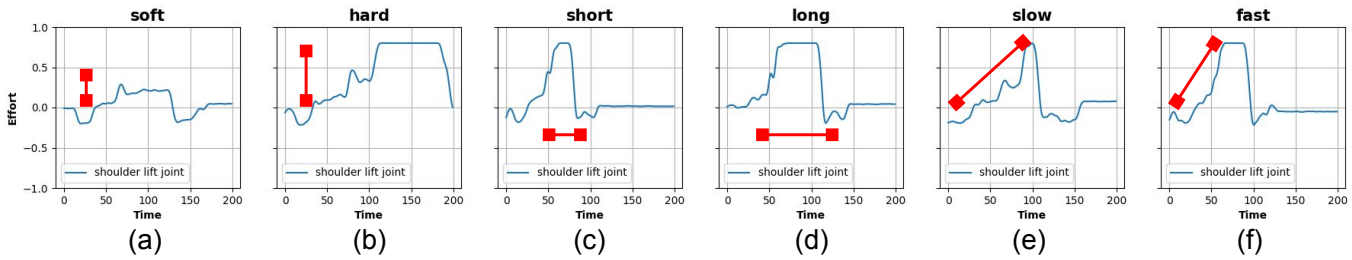

Fig. 5: Example effort trajectories from the training data for a single robot joint—`shoulder lift joint`—and the corresponding user-given labels across 6 different demonstrations—**(a)** and **(b)** designate the maximal exerted effort, **(c)** and **(d)** designate the length for which the maximal effort was maintained, **(e)** and **(f)** designate the time for which the maximal effort in the demonstration was reached (slope).

and padding accordingly. The size of the total dataset after augmentation is 2000 demonstrations which are split according to a 90-10 training-validation ratio.

The size of latent space is chosen to be $|\mathbf{c}| = 8$. In the context of the problem formulation in section III, $\mathbf{c}_s = \{c_0, c_1\}, \mathbf{c}_e = \{c_2, c_3, c_4\}$ and $\mathbf{c}_u = \{c_5, c_6, c_7\}$.

To fully close the loop, the trajectories which we sample from the model could further be executed on the physical robot through a hybrid position/force controller [23]. However, such evaluation is beyond the scope of the paper.

B. Evaluation

1) Quantitative: We use the Gaussian distributions we fit for each label i in each concept group j—$\mathcal{N}(\mu_j^{(i)}, \sigma_j^{(i)})$—to sample and generate trajectories that are consistent with the meaning of a particular label. For a set of 5 test images, which have not been seen during training, we sample 100 samples of $\mathbf{c}$ per image for each of the 10 labels. For each label we then judge whether the generated trajectories match the semantics of the corresponding label we have conditioned on through a series of heuristics. Average accuracy across all images and all samples is reported for each label. For the labels of spatial nature, for each 7-dimensional (7 DoFs) robot state $x_i, \mathbf{x} = \{x_1, \ldots, x_T\}$, we compute the corresponding end-effector pose p_i through the forward kinematics $\mathbf{K}$ of the robot. Afterwards, using the camera model of the Kinect2 sensor, each pose p_i is projected in the image we condition on. Both the kinematic and image projection transformations are deterministic ones. We report the Mean Absolute Error (MAE) in normalised pixel coordinates between the dab location and the landmark location (in pixel coordinates) for the misclassified trajectories. This gives us a sense of how far from the desired ones are the generated samples. The dab location is inferred as the point in the end-effector pose trajectory with lowest z coordinate. For example, if we have conditioned on the `left` label but some of the sampled trajectories result in a dab to the `right` of the cube in the image, we report *how far off* from being `left` did the robot touch the table.

2) Qualitative: We additionally report qualitative results from perturbing individual axes in the latent space which are meant to convey specific semantics—e.g. axes in $\mathbf{c}_1, \mathbf{c}_2$. We draw 5 consecutive samples for each axis in the range of $[-2, 2]$, keeping the other latent axes fixed at 0, and visualise the generated trajectories (joint positions and efforts). We also project the virtual end effector positions, following the generated trajectories, in the image plane, using the known robot model and its forward kinematics, allowing us to judge whether perturbing a force-aligned latent dimension results in spatial changes of the end-effector and vice-versa.

VII. Results & Discussion

Our evaluation shows that the model which uses the weak labels achieves better alignment between the latent axes and the high-level concepts, as compared to a model which ignores any user label information ($\gamma = 0$ in Equation 10). Tables II and III present the results from the qualitative evaluation of both models. We can see that the model which utilised the weak labels can sample consistently correct robot trajectories which satisfy the label we have conditioned on. We can also see that the VAE model, unlike the full one, experiences mode collapse, especially in the context of the effort labels. This can be partly explained by the fact that due to the lack of the weak labels, which utilise the image input, for the VAE there is no term in the loss function to incentivise that.

	`left`		`right`		`front`		`back`	
Model	Acc	MAE	Acc	MAE	Acc	MAE	Acc	MAE
VAE	0.80	0.026	0.82	0.031	0.86	0.050	0.96	0.030
Full	**0.95**	0.031	**0.87**	**0.026**	**0.92**	**0.027**	**0.99**	**0.018**

TABLE II: Accuracy (higher is better) and MAE (lower is better) for sampled trajectories under the two models, conditioned on a fixed spatial label.

Model	`soft`	`hard`	`short`	`long`	`slow`	`fast`
VAE	0.05	0.95	0.96	0.59	0.93	0.52
Full	**0.92**	**1.0**	**1.00**	**0.90**	**0.95**	**0.70**

TABLE III: Accuracy for sampled trajectories under the two models, conditioned on a fixed effort label.

Figures 6 to 10 present the qualitative results from perturbing in a controlled way the latent space optimised under the full model. For figure the first row of plots represents the generated joint angle positions for each of the 7 joints, for

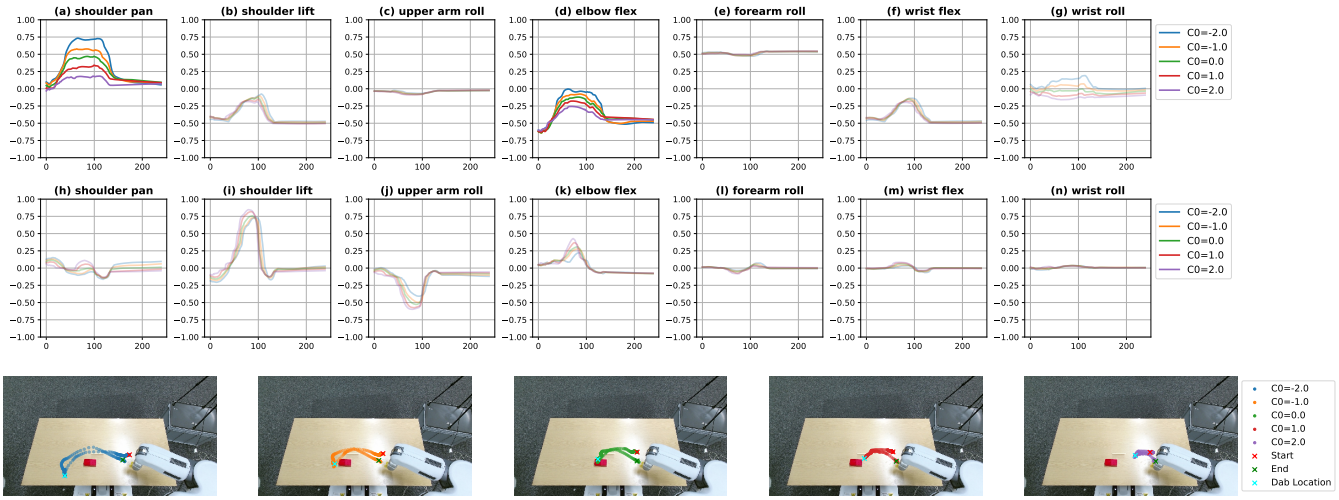

Fig. 6: Linearly interpolate c_0 ($\equiv$ from dabbing to the `left` to dabbing to the `right` of the visual landmark in the scene.)

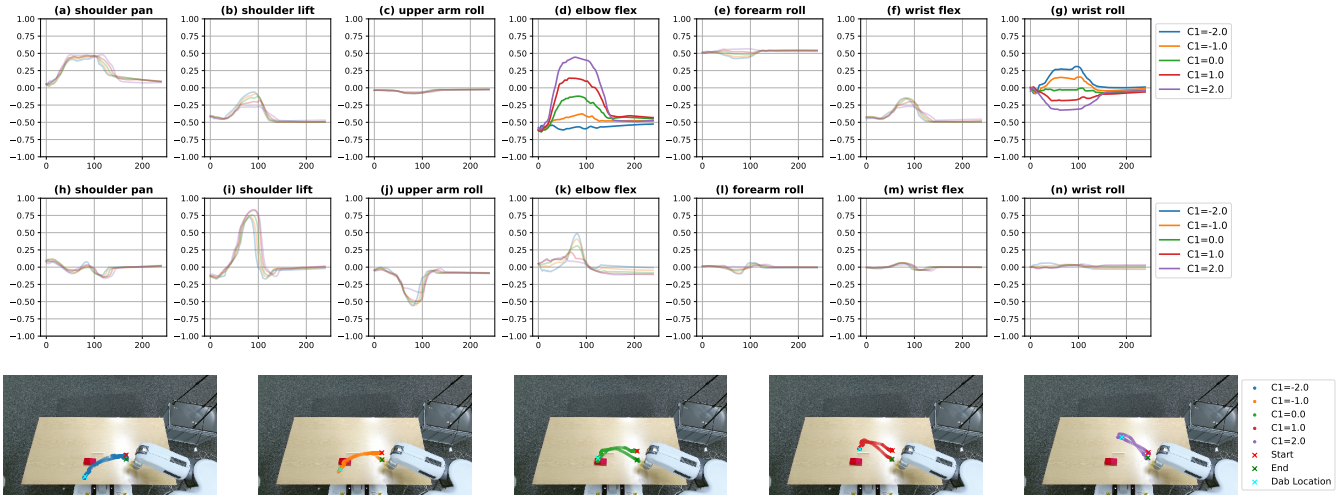

Fig. 7: Linearly interpolate c_1 ($\equiv$ from dabbing to the `front` to dabbing to the `back` of the visual landmark in the scene.)

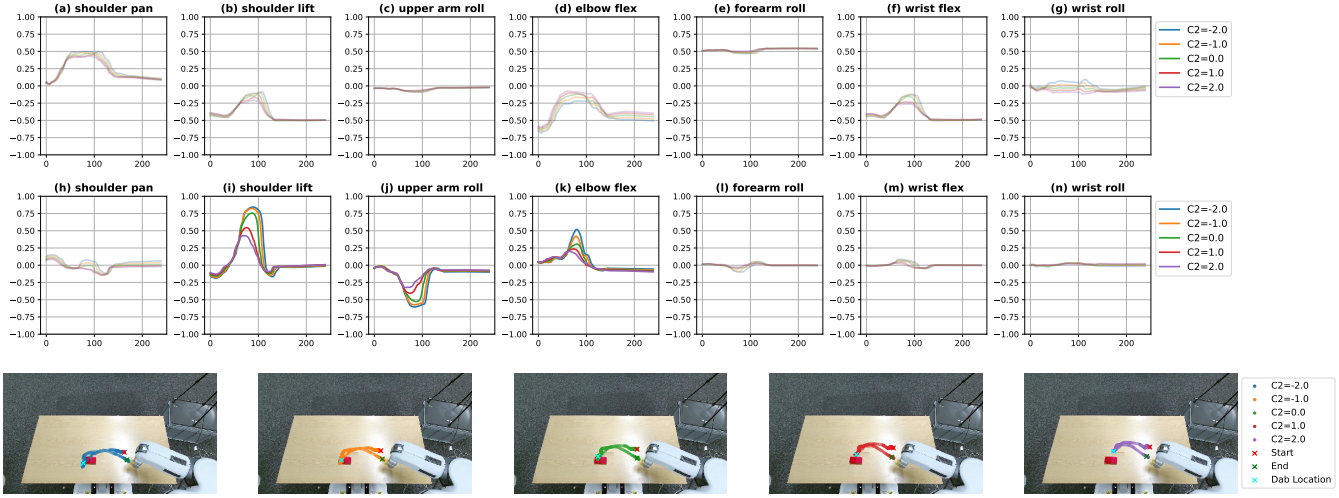

Fig. 8: Linearly interpolate c_2 ($\equiv$ from dabbing `hardly` to `softly`)

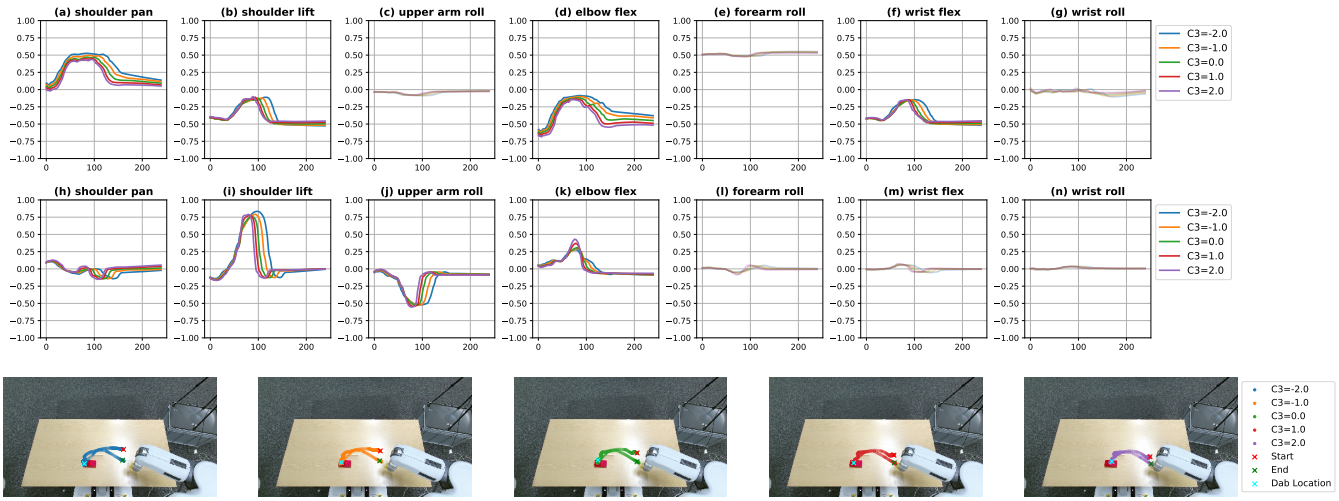

Fig. 9: Linearly interpolate c_3 ($\equiv$ from dabbing for a `long` period of time to dabbing for a `short` period of time)

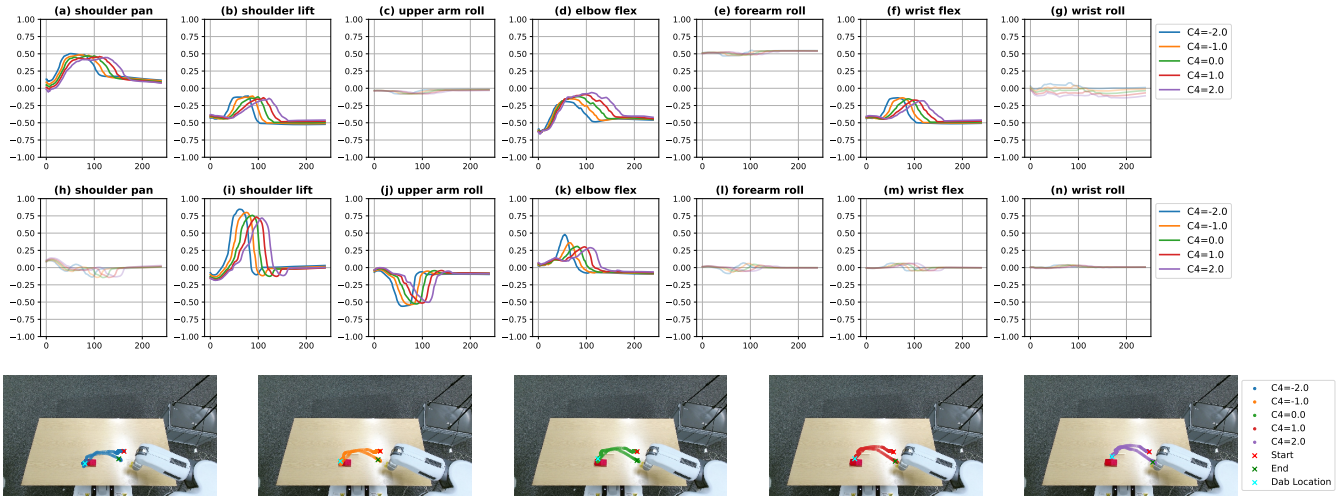

Fig. 10: Linearly interpolate c_4 ($\equiv$ from dabbing `quickly` to dabbing `slowly`)

each of the 5 drawn samples, the second row—the generated joint efforts—and the third row the corresponding end-effector positions, from the forward robot kinematics, projected in the image plane. Joints which exhibit more variance have been highlighted for clarity. As we can see, perturbing c_0 and c_1 corresponds to clear and consistent movement of the end effector (through the generated joint position profiles) while there is not much change in the generated effort profiles—Figures 6 and 7. Simultaneously, we observe the opposite effect in other 3 Figures. Perturbations in c_2 correspond solely changes in the output efforts in the shoulder lift and upper arm rolls joints—Figure 8 (i) and (j). Perturbations in c_3 correspond to changes in length for which the joint efforts are exerted but does not change their magnitude or where they are applied on the table—Figure 9. And finally, perturbations in c_4 correspond to changes on how quickly the maximum efforts are reached - most noticeable in the changing slope of the shoulder lift and upper arm rolls joints—Figure 10 (i) and (j). As a result, we can conclude that the latent dimensions, which we aligned with the semantically grouped weak labels, have indeed captured the notions and concepts which underlie the labels.

VIII. Conclusion

We have presented the problem of interpretable multi-modal LfD as equivalent to formulating a conditional probabilistic model. In the context of table-top dabbing, we utilise weak discrete labels from the demonstrator to represent abstract notions, manifested in a captured multi-modal robot dataset. Through the usage of high-parameter neural models and methods from deep variational inference we have demonstrated how to align some of the latent variables of the formulated probabilistic model with the demonstrated high-level notions.

IX. Acknowledgements

This work is partly supported by funding from the Turing Institute, as part of the Safe AI for surgical assistance project

References

[1] Paszke Adam, Gross Sam, Chintala Soumith, Chanan Gregory, Yang Edward, D Zachary, Lin Zeming, Desmaison Alban, Antiga Luca, and Lerer Adam. Automatic differentiation in pytorch. In *Proceedings of Neural Information Processing Systems*, 2017.

[2] Samuel K. Ainsworth, Nicholas J. Foti, Adrian K. C. Lee, and Emily B. Fox. oi-VAE: Output interpretable VAEs for nonlinear group factor analysis. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 119–128, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/ainsworth18a.html.

[3] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57 (5):469–483, 2009.

[4] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[5] Apratim Bhattacharyya, Bernt Schiele, and Mario Fritz. Accurate and diverse sampling of sequences based on a "best of many" sample objective. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8485–8493, 2018.

[6] Tian Qi Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, pages 2610–2620, 2018.

[7] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180, 2016.

[8] Adrià Colomé, Gerhard Neumann, Jan Peters, and Carme Torras. Dimensionality reduction for probabilistic movement primitives. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 794–800. IEEE, 2014.

[9] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2016.

[10] Stevan Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335–346, 1990.

[11] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2(5):6, 2017.

[12] Yordan Hristov, Alex Lascarides, and Subramanian Ramamoorthy. Interpretable latent spaces for learning from demonstration. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 957–968. PMLR, 29–31 Oct 2018. URL http://proceedings.mlr.press/v87/hristov18a.html.

[13] Yordan Hristov, Daniel Angelov, Michael Burke, Alex Lascarides, and Subramanian Ramamoorthy. Disentangled relational representations for explaining and learning from demonstration. In *Conference on Robot Learning (CoRL)*, 2019.

[14] Dirk Husmeier. Introduction to learning bayesian networks from data. In *Probabilistic modeling in bioinformatics and medical informatics*, pages 17–57. Springer, 2005.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[17] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[18] Jens Kober, Betty Mohler, and Jan Peters. Learning perceptual coupling for motor primitives. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 834–839. IEEE, 2008.

[19] John E Laird, Kevin Gluck, John Anderson, Kenneth D Forbus, Odest Chadwicke Jenkins, Christian Lebiere, Dario Salvucci, Matthias Scheutz, Andrea Thomaz, Greg Trafton, et al. Interactive task learning. *IEEE Intelligent Systems*, 32(4):6–21, 2017.

[20] Michael D. Noseworthy, Rohan Paul, Subhro Roy, Daehyung Park, and Nicholas Roy. Task-conditioned variational autoencoders for learning movement primitives. 2019.

[21] Alexandros Paraschos, Christian Daniel, Jan R Peters, and Gerhard Neumann. Probabilistic movement primitives. In *Advances in neural information processing systems*, pages 2616–2624, 2013.

[22] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, pages 3546–3554, 2015.

[23] MH Reibert. Hybrid position/force control of manipulators. *ASME, J. of Dynamic Systems, Measurement, and Control*, 103:2–12, 1981.

[24] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence*

and statistics, pages 627–635, 2011.

[25] Elmar Rueckert, Jan Mundo, Alexandros Paraschos, Jan Peters, and Gerhard Neumann. Extracting low-dimensional control variables for movement primitives. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1511–1518. IEEE, 2015.

[26] Stefan Schaal. Dynamic movement primitives-a framework for motor control in humans and humanoid robotics. In *Adaptive motion of animals and machines*, pages 261–280. Springer, 2006.

[27] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491, 2015.

[28] Paul Vogt. The physical symbol grounding problem. *Cognitive Systems Research*, 3(3):429–457, 2002.

[29] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European Conference on Computer Vision*, pages 835–851. Springer, 2016.

[30] Nick Watters, Loic Matthey, Chris P Burgess, and Alexander Lerchner. Spatial broadcast decoder: A simple architecture for disentangled representations in vaes. 2019.

APPENDIX A
VARIATIONAL LOWER BOUND DERIVATION

Below is a derivation of the Variational Lower Bound of the data likelihood, following Jensen's inequality. The inequality becomes and equality when the amortised posterior $q(\mathbf{c}|\mathbf{x},\mathbf{i})$ matches exactly the true posterior $p(\mathbf{c}|\mathbf{x},\mathbf{i},\mathbf{y})$. The posterior we optimise is not conditioned on $\mathbf{y}$, potentially meaning that we might never close the gap between the $\log$ of the data distribution and the VLB, measured by $D_{KL}(q(\mathbf{c}|\mathbf{x},\mathbf{i})||p(\mathbf{c}|\mathbf{x},\mathbf{i},\mathbf{y}))$. However, maximising the VLB still goes in the direction of maximising the data distribution.

$$
\begin{aligned}
\log p(\mathbf{x},\mathbf{y}|\mathbf{i}) &= \log \int p(\mathbf{x},\mathbf{y}|\mathbf{i},\mathbf{c})p(\mathbf{c})d\mathbf{c} \\
&= \log \int p(\mathbf{x}|\mathbf{i},\mathbf{c})p(\mathbf{y}|\mathbf{c})p(\mathbf{c})d\mathbf{c} \\
&= \log \int \frac{q(\mathbf{c}|\mathbf{x},\mathbf{i})}{q(\mathbf{c}|\mathbf{x},\mathbf{i})} p(\mathbf{x}|\mathbf{i},\mathbf{c})p(\mathbf{y}|\mathbf{c})p(\mathbf{c})d\mathbf{c} \\
&= \log \mathbb{E}_{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \frac{p(\mathbf{x}|\mathbf{i},\mathbf{c})p(\mathbf{y}|\mathbf{c})p(\mathbf{c})}{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \\
&\geq \mathbb{E}_{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \log \frac{p(\mathbf{x}|\mathbf{i},\mathbf{c})p(\mathbf{y}|\mathbf{c})p(\mathbf{c})}{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \qquad \text{[Jensen]} \\
&= \mathbb{E}_{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \log p(\mathbf{x}|\mathbf{i},\mathbf{c}) + \mathbb{E}_{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \log p(\mathbf{y}|\mathbf{c}) - \mathbb{E}_{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \log \frac{q(\mathbf{c}|\mathbf{x},\mathbf{i})}{p(\mathbf{c})} \\
&= \mathbb{E}_{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \log p(\mathbf{x}|\mathbf{i},\mathbf{c}) + \mathbb{E}_{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \log p(\mathbf{y}|\mathbf{c}) - D_{KL}(q(\mathbf{c}|\mathbf{x},\mathbf{i})||p(\mathbf{c})) \\
&= -(D_{KL}(q(\mathbf{c}|\mathbf{x},\mathbf{i})||p(\mathbf{c})) - \mathbb{E}_{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \log p(\mathbf{x}|\mathbf{i},\mathbf{c}) - \mathbb{E}_{q(\mathbf{c}|\mathbf{x},\mathbf{i})} \log p(\mathbf{y}|\mathbf{c})) \\
&= -(\mathcal{L})
\end{aligned}
\tag{11}
$$

APPENDIX B
ARCHITECTURE DETAILS

The model architectures are implemented in the PyTorch framework[4]. The image encoder network takes as input a single RGB 128x128 pixel image. The trajectory encoder takes a single 14-dimensional, 240 time-step-long trajectory. Their outputs both feed into an MLP network which, after a series of nonliner transformations, outputs parameters of a distribution over the latent space $\mathbf{c}$. Trough the reparametrisation trick [16] we sample values for $\mathbf{c}$. Through a residual connection, the output of the image encoder is concatenated to the sampled latent values. The resultant vector is tiled 240 time, extended with a *time* dimension, and fed into a TCN decoder network to produce reconstructions for the original 14-dimensional input trajectories.

Across all experiments, training is performed for a fixed number of 100 epochs using a batch size of 8. The dimensionality of the latent space $|\mathbf{c}|$ = 8 across all experiments. The Adam optimizer [15] is used through the learning process with the following values for its parameters—($learningrate = 0.001, \beta 1 = 0.9, \beta 2 = 0.999, eps = 1e-08, weightdecayrate = 0, amsgrad = False$).

For all experiments, the values (unless when set to 0) for the three coefficients from Equation 10 are:

- $\alpha = 1, \beta = 0.1, \gamma = 25$

The values are chosen empirically in a manner such that all the loss terms have similar magnitude and thus none of them overwhelms the gradient updates while training the full model.

[4]https://pytorch.org/docs/stable/index.html

Image Encoder
FC (4) $\mathbf{i}$
FC (256)
2D Conv (k=3, p=1, c=64)
2D Conv (k=3, p=1, c=64)
2D Conv (k=3, p=1, c=64)
2D Conv (k=3, p=1, c=32)
2D Conv (k=3, p=1, c=32)
Input Image $\mathbf{I}$ [128 x 128 x 3]

(a) Image Encoder

MLP
FC (2x8) $\mu, \log(\sigma)$
FC (32)
FC (32)
Concatenated $[\mathbf{i}; \boldsymbol{\tau}]$[1 x 36]

(c) MLP

Trajectory Encoder
FC (32) $\boldsymbol{\tau}$
FC (256)
1D Conv (k=7, p=3); 1D Conv (k=5, p=2); 1D Conv (k=3, p=1) [c=20]
Input Trajectory $\mathbf{x}$ [240 x 14]

(b) Trajectory Encoder

TCN Decoder
Temporal Block (dilation=4, k=5, c=14)
Temporal Block (dilation=2, k=5, c=20)
Temporal Block (dilation=1, k=5, c=20)
append time channel $\mathbf{t}$
tile (240, 10)
Concatenated $[\mathbf{i}; \mathbf{c}]$[1 x 10]

(d) TCN Decoder

TABLE IV: Network architectures used for the reported models. (a) is a 2D convolutional network, (b) is a 1D convolutional network, (c) is a fully-connected MLP network, (d) is a Temporal Convolution Network, made of stacked temporal blocks and dilated convolutions, described in [4]