NewtonianVAE: Proportional Control and Goal Identification from Pixels via Physical Latent Spaces

Miguel Jaques
School of Informatics
University of Edinburgh
Edinburgh, UK
m.a.m.jaques@sms.ed.ac.uk

Michael Burke
School of Informatics
University of Edinburgh
Edinburgh, UK
michael.burke@ed.ac.uk

Timothy Hospedales
School of Informatics
University of Edinburgh
Edinburgh, UK
t.hospedales@ed.ac.uk

Abstract

Learning low-dimensional latent state space dynamics models has been a powerful paradigm for enabling vision-based planning and learning for control. We introduce a latent dynamics learning framework that is uniquely designed to induce *proportional* controlability in the latent space, thus enabling the use of much simpler controllers than prior work. We show that our learned dynamics model enables proportional control from pixels, dramatically simplifies and accelerates behavioural cloning of vision-based controllers, and provides interpretable goal discovery when applied to imitation learning of switching controllers from demonstration.

1 Introduction

Vision-based control is highly desirable across numerous industrial applications, both in robotics and process control. At present, much practical vision-based control relies on supervised learning to build bespoke perception modules, prior to downstream dynamics modelling and controller design. This can be expensive and time consuming, and as a result there is growing interest in developing model-based approaches for direct vision-based control.

Model-based approaches for visual control tend to learn latent dynamics models that are subsequently used within suitable planning or model predictive control (MPC) frameworks, or to train policies for later use. In this paper, we argue that this decoupling of dynamics and control is computationally expensive and often unnecessary. Instead we learn a structured latent dynamical model that directly allows for simple proportional control to be applied. Proportional-Integral-Derivative (PID) feedback control produces commands that are proportional to an error or cost term, the cumulative error over some time horizon, and the rate of change in error term. Gain terms shape the controller response to errors. In goal-based control settings it is common to specify the error in terms of the difference between a (potentially dynamic) target state $\mathbf{x}^{target}$ and the current system state $\mathbf{x}$.

PID control is ubiquitous in industry, and broadly applicable across numerous domains, providing a simple and reliable off-the-shelf mechanism for stabilising systems. By structuring latent dynamics such that PID control can be applied to move between latent states, we remove the requirement for complex planning or reinforcement learning strategies. Moreover, we show that imitation learning from demonstrations becomes a simple goal inference problem under a proportional control model in this latent space, and can even be extended to sequential tasks comprising multiple sub-goals.

Imitation learning from high dimensional visual data can be particularly challenging [2]. Behaviour cloning, which seeks to reproduce demonstrations, is extremely vulnerable to generalisation failures, while inverse reinforcement learning (IRL) [38] strategies are challenging to train and extremely sample inefficient. By learning a structured dynamics model, we allow for more robust control in the presence of noise and simplify the inverse reward inference process. In summary, the primary contributions of this work are:

Preprint. Under review.

Embedding for proportional controllability We induce a latent space where taking an action in the direction between the current position and some target position, $\mathbf{u} \propto \mathbf{x}^{target} - \mathbf{x}$, moves the system towards the target position. Uniquely, this enables simple proportional control from pixels.

Imitation learning using latent switching proportional control laws We leverage the properties of this embedding, to frame imitation learning as a goal inference problem under a switching proportional control law model in the structured latent space. This enables one-shot interpretable imitation learning from high-dimensional observations.

Results show that embedding for proportional controllability produces more interpretable latent spaces, allows for the use of simple and efficient controllers that cannot be applied with less structured latent dynamical models, and enables one-shot learning of control and interpretable goal identification in sequential multi-task imitation learning settings.

2 Related Work

This paper takes a model-based approach to visual control, using variational autoencoding (VAE) [27]. Latent dynamical systems modelling using autoencoding is widely used, and has been proposed for Bayesian filtering [14, 32, 25], and as inverse graphics for improved video prediction and vision-based control [23]. Ha and Schmidhuber [20] train a latent dynamics model using a variational recurrent neural network (VRNN) in the latent space of a VAE, and then learn a controller that acts in this space using a known reward model. Hafner et al. [21] extend this approach to allow planning from pixels. Unfortunately, because these approaches decouple dynamics modelling and control, they place an unnecessary computational burden on control, either requiring sampling based planning or further RL policy optimisation. We argue that this burden can be alleviated by imposing additional structure on the latent space such that proportional control is feasible.

In doing so, we build on the control hypothesis advocated by Full and Koditschek [16], which seeks to model complex phenonoma and systems through simple template models and controllers, using anchor networks to abstract the complexity away from control. This also simplifies the challenges of imitation learning, allowing for the sequential composition of tasks in the spirit of Burridge et al. [7].

The addition of structural inductive biases into neural models has become increasingly important for generalisation. Injecting knowledge of known physical equations [19, 23] has been shown to improve dynamics modelling, while the inclusion of structured transition matrices was essential to learn Koopman operators [1] that model dynamical systems with compositional properties [35] (here, a block-wise structure with shared blocks was used to learn transition dynamics, which highlighted the importance of added structure in linear state space models, but was not applied to visual settings). Models like embed to control (E2C) [44] or deep variational Bayes filters (DVBF) [25] recover structured conditionally linear latent spaces which can be used for control, but, as will be demonstrated later, are still unsuitable for direct proportional control. PVEs [24] learn an explicit positional representation, but do so by minimizing a combination of several heuristic loss functions. Since these models do not use a decoder, it is not possible to visually inspect the learned representations in image space.

NewtonianVAE not only provides latent space interpretability, but also simplifies imitation learning. Inverse reinforcement learning (IRL) strategies for imitation learning typically struggle to learn from high dimensional observation traces as they tend to be based on the principle of feature counting and observation frequency matching [38], as in maximum entropy IRL [46]. Maximum entropy IRL has been extended to use a deep neural network feature extractor [46], but this is highly vulnerable to overfitting and has extensive data requirements. Recent adversarial IRL approaches [15, 17, 22] avoid the challenge of learning a global reward function by training policies directly, but these have yet to be successfully scaled to high dimensional problems. As a result, most imitation learning approaches tend to assume access to low dimensional states, avoiding the challenge of learning from pixels.

Standard imitation learning learning strategies can fail in multi-goal settings or on more complex tasks. In order to address this, many approaches frame the problem of imitation learning from these lower level states as one of skill or options [42, 29] learning using switching state space models. These switching models include linear dynamical attractor systems [10], conditionally linear Gaussian models [8, 33], Bayesian non-parametrics [39, 40], and neural variational models [28]. Kipf et al. [28] learn task segmentations to infer compositional policies, but the model uses environment states

directly instead of images. Burke et al. [5, 6] use a switching controller formulation for switching control law identification from image, proprioceptive state and control action observations. This work applies a similar strategy for goal inference, but, unlike the approaches above, makes use of a learned latent state representation and thus does not require proprioceptive or low level state information.

Despite this reliance on proprioceptive state information, there is a growing interest in direct visual imitation learning and control. Nair et al. [37] train a variational autoencoder (VAE) on image observations of an environment, and subsequently sample from this latent space in order to train goal-conditioned policies that can be used to move between different goal states. In contrast, we propose a latent dynamics model that allows for latent proportional controllability and eliminates the need to train a policy to move between goal states.

3 Preliminaries: Variational models for visual control

In order to learn a compact latent representation of videos that can be used for planning and control we use the variational autoencoder framework (VAE) [27, 41] and its recurrent formulation (VRNN), [9]. In this section we briefly present a general formulation of the VRNN, of which many recent models are particular cases or variations [44, 25, 14, 32, 21]. For derivation details please refer to [9].

Given a sequence of T images, $\mathbf{I}_{1:T}$, and actuations $\mathbf{u}_{1:T} \in \mathbb{R}^{d_u}$and the corresponding latent representations, $\mathbf{z}_{1:T} \in \mathbb{R}^{d_z}$, the marginal image likelihood is given by:

$$p(\mathbf{I}_{1:T}|\mathbf{u}_{1:T}) = \int p(\mathbf{I}_{1:T}|\mathbf{z}_{1:T}, \mathbf{u}_{1:T}) p(\mathbf{z}_{1:T}|\mathbf{u}_{1:T}) \mathrm{d}\mathbf{z}_{1:T} \tag{1}$$

where we factorize the terms above as $p(\mathbf{I}_{1:T}|\mathbf{z}_{1:T}, \mathbf{u}_{1:T}) = \prod p(\mathbf{I}_t|\mathbf{z}_t)$ and $p(\mathbf{z}_{1:T}|\mathbf{u}_{1:T}) = \prod p(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{u}_{t-1})$, with an approximate positerior given by $q(\mathbf{z}_{1:T}|\mathbf{I}_{1:T}) = \prod q(\mathbf{z}_t|\mathbf{I}_t, \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$. The model components are trained jointly by maximizing the lower bound on (1):

$$\mathcal{L} = \sum_t \mathbb{E}_{q(\mathbf{z}_t|\mathbf{I}_t, \mathbf{z}_{t-1}, \mathbf{u}_{t-1})} \left[p(\mathbf{I}_t|\mathbf{z}_t) + \mathrm{KL}\left(q(\mathbf{z}_{t+1}|\mathbf{I}_{t+1}, \mathbf{z}_t, \mathbf{u}_t) \| p(\mathbf{z}_{t+1}|\mathbf{z}_t, \mathbf{u}_t) \right) \right], \tag{2}$$

via the reparametrization trick, by drawing samples from the posterior distributions, $q(\mathbf{z}_t|\mathbf{I}_t, \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$. Under this framework, the various desired inductive biases are usually built into the structure of the transition prior $p(\mathbf{z}_{t+1}|\mathbf{z}_t, \mathbf{u}_t)$. In this work we will build on the formulation that uses a linear dynamical system as latent dynamics:

$$p(\mathbf{z}_{t+1}|\mathbf{z}_t, \mathbf{u}_t) = A(\mathbf{z}_t) \cdot \mathbf{z}_t + B(\mathbf{z}_t) \cdot \mathbf{u}_t + \mathbf{c}(\mathbf{z}_t) \tag{3}$$

which has been studied extensively in the context of deep probabilistic models [36, 14, 25, 32, 3].

4 Newtonian Variational Autoencoder

Motivation To motivate our model, we begin by examining the properties of an existing latent variable model used for control. We train an E2C model [44], since it applies a locally linear latent transition as in (3) and is highly representative of properties obtained in these types of model. We use a simple point mass system that can move in the $[x, y]$ plane and train the model on random transitions in image space (more details in the experiments section). Since the environment is 2D with 2D controls, we use a 4D latent space (2 dimensions for position and 2 for velocity). Our goal is to explore how the E2C model behaves when a basic proportional control law $\mathbf{u}_t \propto (\mathbf{x}^{goal} - \mathbf{x}_t)$ is applied, where $\mathbf{x}$ is the latent system configuration.

An immediate problem is that even though the latent coordinates corresponding to position are correctly learned (Fig. 1(right)), it is necessary to plot *every coordinate pair* and their correlation with ground truth positions in order to visually determine which 2 coordinates correspond to the position $\mathbf{x}$. Having determined such $\mathbf{x}$, we can use a random target position $\mathbf{x}^{goal}$ and see if successively applying an action $\mathbf{u}_t \propto (\mathbf{x}^{goal} - \mathbf{x}_t)$ will guide the system towards $\mathbf{x}^{goal}$ (which we term *proportional controllability*). Fig. 1(left) shows that this simple control law fails to guide the system towards the goal state, even though the latent space is seemingly well structured. These problems are present in many existing variational models for controllable systems, including E2C [44], DVBF [25] and the Kalman VAE [14].

To avoid the need for ground truth data and visual inspection, it is necessary to construct a model that explicitly treats position and velocity as separate latent variables $\mathbf{x}$ and $\mathbf{v}$. Furthermore, to ensure correct behaviour under a proportional control law[1] the change in position and velocity should be directly related to the force applied, i.e. given an external action $\mathbf{u}$ representing the force (=acceleration) acting on a system, $\mathbf{x}$ and $\mathbf{v}$ should follow Newton's second law, $d^2\mathbf{x}/dt^2 = \mathbf{F}/m$. Although this might seem like a trivial statement from a physical standpoint, this type of behaviour is not built into existing neural models, where the relationship between action and latent states can be arbitrary. This arbitrary relationship in turn complicates the control task, and it thus becomes necessary to learn downstream controllers or policies that both compensate for these dynamics and meet a control objective.

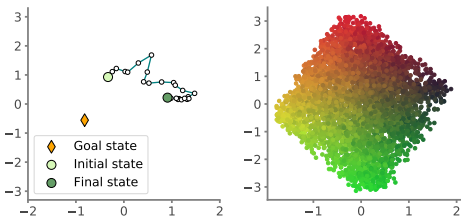

Figure 1: Trajectory of a point mass actuated using $\mathbf{u}_t \propto (\mathbf{x}^{goal} - \mathbf{x}_t)$ (left) in the latent space learned by an E2C model (right).

We make one additional observation: in many cases the external action $\mathbf{u}$ is typically applied along disentangled dimensions of the system. For example, for a 2-arm robot, actions correspond to torques on the angles of each arm relative to its origin[2]. These action dimensions correspond to the polar coordinates $[\theta_1, \theta_2]$, which are the ideal disentangled coordinates to describe such a robot. We will use this fact to formulate a model that not only provides an interpretable and P-controllable latent space, but also the correct disentanglement by construction.

Formulation We now formulate a model satisfying the above desiderata. For an actuated rigid body systems with D degrees of freedom, we model the system configuration (positions or angles) by a set of coordinates $\mathbf{x} \in \mathbb{R}^D$ with double integrator dynamics, inspired by Netwon's equations of motion:

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}\,, \quad \frac{d\mathbf{v}}{dt} = A(\mathbf{x},\mathbf{v}) \cdot \mathbf{x} + B(\mathbf{x},\mathbf{v}) \cdot \mathbf{v} + C(\mathbf{x},\mathbf{v}) \cdot \mathbf{u} \tag{4}$$

To build a discrete form of (4) into a VAE formulation, we use the instantaneous system configuration (or position) $\mathbf{x}$ as the stochastic variable that is inferred by the approximate posterior, $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{I}_t)$, with velocity a deterministic variable that is simply the finite difference of positions, $\mathbf{v}_t = (\mathbf{x}_t - \mathbf{x}_{t-1})/\Delta t$. The generative model is now given by

$$p(\mathbf{I}_{1:T}|\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) = \prod p(\mathbf{I}_t|\mathbf{x}_t) \tag{5}$$

$$p(\mathbf{x}_{1:T}|\mathbf{u}_{1:T}) = \prod p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}; \mathbf{v}_t) \tag{6}$$

where the transition prior is:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}; \mathbf{v}_t) = \mathcal{N}(\mathbf{x}_t|\mathbf{x}_{t-1} + \Delta t \cdot \mathbf{v}_t,\ \sigma^2\mathbf{Id}) \tag{7}$$

$$\mathbf{v}_t = \mathbf{v}_{t-1} + \Delta t \cdot (A \cdot \mathbf{x}_{t-1} + B \cdot \mathbf{v}_{t-1} + C \cdot \mathbf{u}_{t-1}) \tag{8}$$

with $[A, \log(-B), \log C] = \text{diag}(f(\mathbf{x}_t, \mathbf{v}_t, \mathbf{u}_t))$, where f is a neural network with linear output activation. Using diagonal transition matrices encourages correct coordinate relations between $\mathbf{u}$, $\mathbf{x}$ and $\mathbf{v}$, since linear combinations of dimensions are eliminated. In order to obtain the correct directional relation between $\mathbf{u}$ and $\mathbf{x}$, required for interpretable controllability, we set C to be strictly positive (in addition to diagonal). B is strictly negative to provide a correct interpretation of the term in $\mathbf{v}$ as friction, which aids trajectory stability. During inference, $\mathbf{v}_t$ is computed as:

$$\mathbf{v}_t = \frac{\mathbf{x}_t - \mathbf{x}_{t-1}}{\Delta t}, \text{with} \quad \mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{I}_t)\ ,\ \mathbf{x}_{t-1} \sim q(\mathbf{x}_{t-1}|\mathbf{I}_{t-1}) \tag{9}$$

This inference model provides a principled way to infer velocities from consecutive positions, similarly to [24]. We use Gaussian $p(\mathbf{I}_t|\mathbf{x}_t)$ and $q(\mathbf{x}_t|\mathbf{I}_t)$ parametrized by a neural network throughout.

We train all model components using the following ELBO (full derivation in Appendix B):

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{I}_t)q(\mathbf{x}_{t-1}|\mathbf{I}_{t-1})}\left[\mathbb{E}_{p(\mathbf{x}_{t+1}|\mathbf{x}_t,\mathbf{u}_t;\mathbf{v}_t)} p(\mathbf{I}_{t+1}|\mathbf{x}_{t+1}) + \text{KL}\left(q(\mathbf{x}_{t+1}|\mathbf{I}_{t+1}) \| p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t; \mathbf{v}_t)\right)\right] \tag{10}$$

[1] We use P-controllers here, though extension to PID-controllers is trivial and likely to improve performance further. For further analysis of the convergence and stability of second order systems see [12, 11].

[2] The example also applies more generally to any robot actuated with torques along its joints.

A crucial component of this ELBO is the fact that we perform not current step reconstruction, but future step reconstruction through the generative process (first term of the equation above). This idea has been used in several existing models [44, 25, 21] and it is known to encourage the use of the transition prior when learning the latent representation.

Further considerations Another key difference between a simple LDS and our Newtonian model is the fact that we consider velocity to be a deterministic latent variable that is uniquely determined by the stochastic positions. In contrast, independent inference through $\mathbf{z}$ means that position and velocity might not have the direct relation that is present in the physical world (velocity as the derivative of position). Both of these contribute to a lack of physical plausability, in the Newtonian sense, in existing models. Though technically our transition prior is a special case of the LDS (3), these added structural constraints are crucial in order to induce a Newtonian latent space that directly allows for proportional control to latent image states.

5 Learning Switching Proportional Controllers from Demonstrations

A key benefit of the Newtonian latent space is that it dramatically simplifies image-based imitation learning. Given a visual demonstration sequence $D_{\mathbf{I}} = \{(\mathbf{I}_1, \mathbf{u}_1), ..., (\mathbf{I}_T, \mathbf{u}_T)\}$, we encode the frames using the inference network $q(\mathbf{x}|\mathbf{I})$ described above in order to produce demonstrations in latent space, $D_{\mathbf{x}} = \{(\mathbf{x}_1, \mathbf{u}_1), ..., (\mathbf{x}_T, \mathbf{u}_T)\}$. We can fit a switching P-controller to a set of demonstration sequences in latent space using a Mixture Density Network (MDN), where the action likelihood given a state is given by a mixture of N Gaussian proportional controllers:

$$P(\mathbf{u}_t|\mathbf{x}_t) = \sum_{n=1}^{N} \pi_n(\mathbf{x}_t)\, \mathcal{N}\left(\mathbf{u}_t | K_n(\mathbf{x}_n^{goal} - \mathbf{x}_t),\, \sigma_n^2\right) \tag{11}$$

where K_n, $\mathbf{x}_n^{goal}$ and σ_n^2, $\forall n \in 1..N$, are learnable parameters, and $\boldsymbol{\pi}(\mathbf{z})$ is a parametric function like a neural network. Intuitively, fitting this MDN to the latent demonstrations splits the demonstrations into regions where a specific proportional controller would correctly fit that part of the trajectory. If the latent space is P-controllable (such as the one produced by the NetwonianVAE), the vectors $\mathbf{x}_n^{goal}$ will correspond to the intermediate goals or bottleneck states in the demonstration sequence. As an added benefit, we can pass the learned goals through NetwonianVAE's decoder in order to obtain their visual representation, providing an interpretable control policy.

Learning a finite-state machine Having identified the latent vectors corresponding to the goals, we determine the order in which they must be reached by analysing the goals visited during the demonstrations, directly extracting initiation sets and termination conditions. This produces a simple finite-state machine (FSM) that determines goal state transitions. The FSM and extracted P-controllers can then be used to reproduce demonstration behaviours by driving the environment to each goal in succession, but could also be used within an options framework [42] for reinforcement learning. Since the latent space induced by the NetwonianVAE empirically provides robust local (or even global) P-controllability, the switching P-controllers should correctly reach each sub-goal in succession.

6 Experiments

We validate our model on 2 simulated continuous control environments, to allow for better evaluation and ablations, and on a PR2 robot performing a reaching task, to show real-world applicability.

6.1 Simulated point mass and multi-goal reacher

To study and compare the latent space and P-controllability properties of the NewtonianVAE and baseline models we use two simulated environments adapted from the `dm_control` library. The first environment is a simple point mass system adapted from the `PointMass` environment. The mass is linearly actuated in the 2D plane and its movement bounded by the edges of the frame. This is a simple yet useful environment to verify whether basic physical properties hold in the latent space learned by each model. The second environment is a reacher robot adapted from the `Reacher` environment and inspired by [28]. We alter the environment so that the robot's middle joint can only bend in one direction, in order to prevent the existence of two possible arm configurations for every goal position. We also limit the origin joint angle range to $[-160, 160]$ so that the system

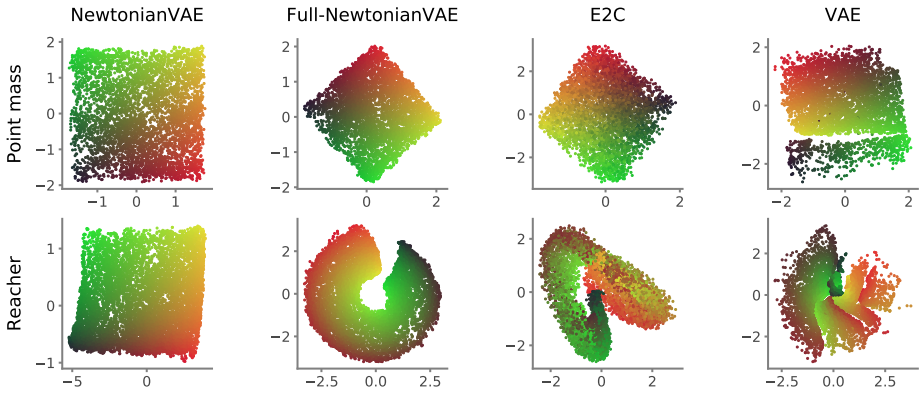

Figure 2: Latent spaces of various models in the point mass and reacher environments. Each dot corresponds to the latent representation of a frame in the test set, and the red-to-green color coding encodes the true 2D position/angle values. For E2C, we plot the two latent dimensions that best correlated with the true positions.

configuration can be described in polar coordinates by two variables corresponding to the angle of each arm, avoiding a discontinuity in case of full circular motion. For representation learning in both environments we generate 1000 sequences with 100 time-steps applying random actions. More implementation details can be found in Appendix C.

Baseline models We compare our model to E2C[3] and a static VAE (each frame encoded individually). Additionally, in order to better understand the effect of diagonality and positivity of the transition matrices in (8), we test Full-NewtonianVAE, where the matrices A,B,C are unbounded and full rank.

Comparing latent spaces We start by visualizing the latent spaces learned by the models in both environments. Fig. 2 shows that only the NewtonianVAE is able to learn a representation corresponding to the natural disentangled coordinates in both environments ($[x, y]$ in the point mass and $[\theta_1, \theta_2]$ in the reacher), and that these are correctly correlated with ground-truth values, coded in the red-green spectrum. This shows that the structure imposed on the transition matrices in (8) is key to learning correct latent spaces in both Cartesian and polar coordinates.

P-controllability Even though the models above produce different latent spaces, most are well structured and show a clear correlation with the ground truth state (color coded). Although their structure is visually appealing, we are primarily interested in verifying is whether they satisfy P-controllability. To do this, we sample random starting and goal states, and successively apply the control law $\mathbf{u}_t \propto (\mathbf{x}(\mathbf{I}^{goal}) - \mathbf{x}_t(\mathbf{I}_t))$. A space is deemed P-controllable if the system moves to $\mathbf{x}(\mathbf{I}^{goal})$ in the limit of many time-steps. For reference, we also apply model-predictive control to E2C.

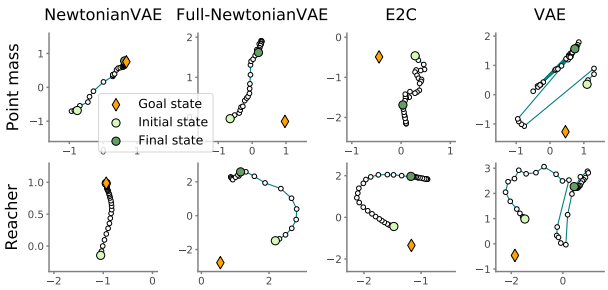

Figure 3: Examples of applying a P-controller to reach a goal state (orange diamond), $\mathbf{u}_t \propto \mathbf{x}^{goal} - \mathbf{x}_t$. Trajectory plots are in the latent space of Fig. 2. Only the NewtonianVAE produces a latent space allowing for P-control.

Convergence curves in the true state space are shown in Fig. 4, and example rollouts in the learned latent space in Fig. 3 (more examples in Appendix D). We can see that only NewtonianVAE produces P-controllable latent states, as all the remaining models diverge under a P-controller. This highlights the fact that even though the latent spaces learned by the Full-NetwtonianVAE and E2C are seemingly well structured for the point mass system, they fail to provide P-controllability. While these systems can still be stabilised using more complex control schemes such as MPC, this is entirely unnecessary with a P-controllable latent space, where trivial control laws can be applied directly.

[3]DBVF and E2C learn similar quality latent spaces, as both rely on an unstructured conditionally linear dynamical system.

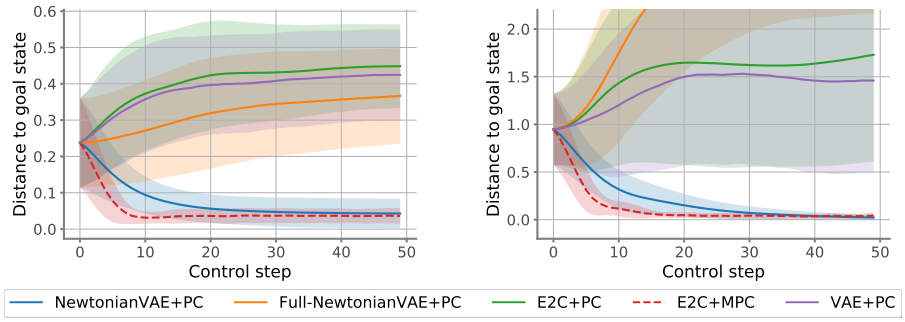

Figure 4: Convergence rates of proportional control (PC) using various latent embeddings for the point mass (left) and reacher (right) systems, over 50 episodes. For contrast, we also show Model Predictive Control (MPC, using CEM planning as per [21]).

Fitting MDN for goal and boundary visualization Having trained a NewtonianVAE on a dataset of random transitions we can use the learned representations to fit the mixture of P-controllers in (11) to the demonstration sequences. Here, we only consider the reacher environment. There are three colored balls in the scene and the task consists of reaching the three balls in succession, where the arm's starting location varies across demonstration sequences. We used the true reacher model with a custom controller to generate demonstration images. A full demonstration sequence is shown in Appendix C.3. For this experiment we use a linear $\boldsymbol{\pi}(\mathbf{x})$, though a MLP yields similar results.

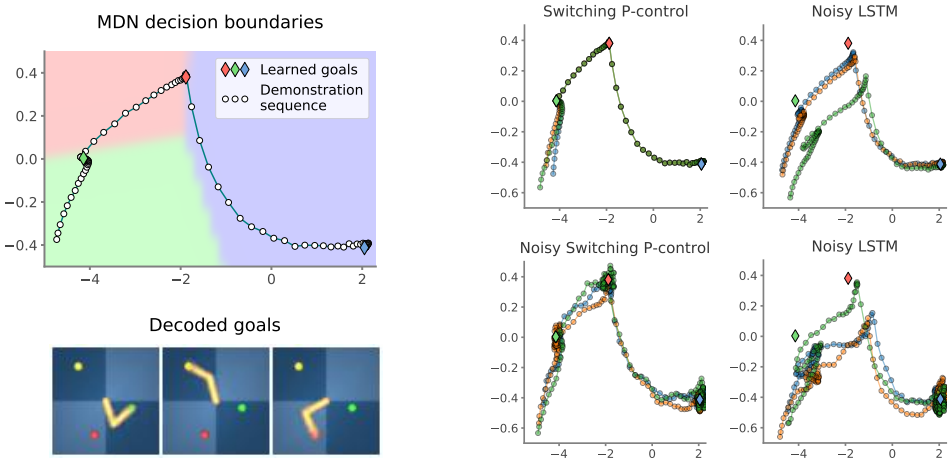

Figure 5: **Left:** Demonstration sequence and learned mixture of P-controllers (MDN). Each background color and corresponding diamond correspond to a component $\pi_n(\mathbf{x})$ and $\mathbf{x}_n^{goal}$, $\forall n \in \{1, 2, 3\}$, respectively. **Right:** Rollouts after imitation learning using switching P-controllers and LSTM policy, with a single demonstration sequence. In the noisy regime each action has an added noise $\mathcal{N}(0, 0.25^2)$. All plots are in the NewtonianVAE's latent space.

After fitting (11) on *a single* demonstration sequence, we visualize the goals $\mathbf{x}^{goal}$ and the decision boundaries of the switching network $\boldsymbol{\pi}(\mathbf{x})$ in Fig. 5(left). The figure shows that goal states are correctly identified (diamond markers), and that the three sub-task regimes are correctly segmented. Decoding $\mathbf{x}^{goal}$, we confirm that the goals are correctly represented in image space, adding an extra layer of interpretability to an upstream control policy.

Imitation learning performance We now compare various imitation learning methods in the simulated task described above. A reward of 1.0 is given when the system reaches a neighborhood of each target (as measured in the true system state), but the targets must be reached in sequence. A more detailed description of the task can be found in Appendix C.3. Our method (switching P-controller) uses a finite-state machine inferred from the MDN trained on latent demonstrations (Fig. 5(left)). We compare it to behaviour cloning with an LSTM with 50 recurrent units, in the NewtonianVAE's latent space, and GAIL [22], a state-of-the-art IRL method trained on ground truth proprioceptive states. Table 1 shows the imitation efficiency for increasing numbers of demonstration sequences, with example rollouts shown in Fig. 5(right). The results show that goal-driven P-control in a hybrid

control policy is significantly more data efficient and robust to noise than a standard behaviour cloning policy. Additionally, switching controllers dramatically outperform GAIL[4], even though this was trained on 5 times the number of environment interactions used by the NewtonianVAE.

Demonstration sequences	Switching P-controller		LSTM		GAIL from proprioception
	Clean	Noisy	Clean	Noisy	
1	3.0 ± 0.0	2.17 ± 0.32	0.81 ± 0.35	0.27 ± 0.20	—
10	3.0 ± 0.0	2.01 ± 0.34	3.00 ± 0.00	1.42 ± 0.34	—
100	3.0 ± 0.0	2.06 ± 0.30	3.00 ± 0.00	1.23 ± 0.30	0.62

Table 1: Efficiency of imitation learning methods for vision-based sequential multi-task control. Metric: Environment Reward (max = 3.0). The NewtonianVAE is used to encode the frames. 'Noisy': Added action noise $\mathcal{N}(0, 0.25^2)$ during the rollouts. Error ranges represent a 95% confidence interval across 100 rollouts. GAIL is trained for 5000 episodes.

6.2 Real multi-object reacher

We now apply our model to real robot data. Here, we record a 7-DoF PR2 robot arm that moves between 6 objects in succession in a hexagon pattern. A full sequence comprises approximately 100 frames. We use 636 frames to train the NewtonianVAE and an additional 100 held-out frames to train the MDN. Further model and dataset details can be found in Appendix C.4.

Fig. 6 shows the image representations of the learned goals (left) and the mode $\boldsymbol{\pi}(\mathbf{x})$ that is active for each frame in the demonstration sequence (right). We can see that the six goals are correctly identified by the MDN, and that the segmentations are correct in the sense that a frame is assigned to the learned goal to which the robot is moving at that time step. We note that the model is able to recover correct goals and segmentations even though not all of the joints are visible in every frame.

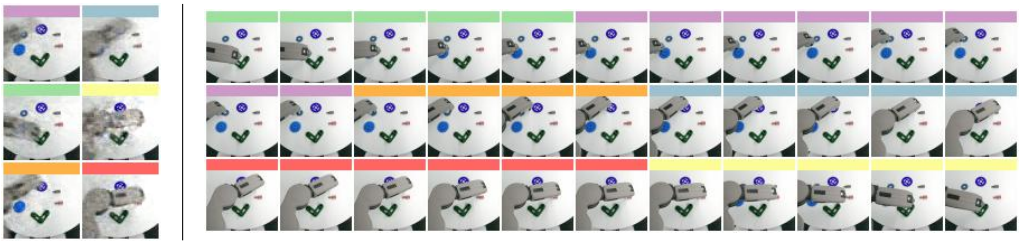

Figure 6: Decoded goals (left) and sequence segmentation (right) learned for a 6-goal visual trajectory of a PR2 robot. The sequence shows 33 equally spaced frames of a 100-frame demonstration.

6.3 Limitations and Future Work

This work assumes that underlying systems are proportional controllable, and follow Newtonian dynamics. However, in our opinion, the most notable limitation is the fact that the imitation learning model only learns a fixed set of goals. Ideally, the agent would learn a semantic goal, which would represent a command "fetch the yellow ball", for a variable position of the yellow ball and not a fixed state. However, this would require demonstration data with substantially more variety than considered here. We have also avoided multi-modal demonstrations for simplicity, though we believe it would be of interest to integrate our method with approaches like InfoGAIL [34].

7 Conclusion

This paper has introduced NewtonianVAE, a structured latent dynamics model designed to allow P-controllability from pixels. Results show that this structured latent space allows for trivial, robust control in the presence of noise and dramatically simplifies and improves imitation learning, which can be framed directly as a switching goal-inference problem in the latent space. Additionally, our model provides visually interpretable goal discovery and task segmentation under both simulated and real environments, without any labelled or proprioception data.

[4]Maximum Entropy IRL performed equally poorly, failing to reach a single goal. This is unsurprising, due to the connections between Maximum Entropy IRL and adversarial imitation learning [13].

Broader Impact

Vision-based control has numerous industrial applications, both in robotics and process control. The methods proposed in this work are directly applicable in many of these settings, and likely to be of considerable interest in industry, where PID control is ubiquitous and the ability to deploy controllers acting on high dimensional sensory data without the need to build custom state estimation tools is highly desirable. In industrial settings, PID controllers are typically sold as dedicated off-the-shelf hardware, equipped with auto-tuning functionality. By providing a mechanism for translating high-dimensional inputs to low dimensional states in which PID control can be applied, this work allows for direct integration with this hardware.

However, potential risks in the use of these strategies include vulnerabilities to adversarial image attacks, and the potential for overfitting and the inclusion of unplanned biases when learning dynamics models. Although we believe that the greater structure introduced in this work leads to more reliable latent spaces for control than previous approaches, the methods presented here still assume that sufficient data covering the operating regions of the system of interest is captured. Failure to do so could result in unexpected controller action. Moreover, the proposed approach implicitly assumes that the underlying systems of interest are proportional controllable, which may not always be true.

References

[1] Ian Abraham, Gerardo De, La Torre, and Todd D Murphey. Model-Based Control Using Koopman Operators. In *RSS*, 2017.

[2] J. Andrew (Drew) Bagnell. An Invitation to Imitation. Technical Report CMU-RI-TR-15-08, Carnegie Mellon University, Pittsburgh, PA, March 2015.

[3] Philip Becker-Ehmck, Jan Peters, and Patrick Van Der Smagt. Switching Linear Dynamics for Variational Bayes Filtering. In *ICML*, 2019.

[4] Filipe De A Belbute-Peres, Kevin A Smith, Kelsey R Allen, Joshua B Tenenbaum, and J Zico Kolter. End-to-End Differentiable Physics for Learning and Control. In *NIPS*, 2018.

[5] Michael Burke, Yordan Hristov, and Subramanian Ramamoorthy. Hybrid system identification using switching density networks. In *CoRL*, 2019.

[6] Michael Burke, Svetlin Penkov, and Subramanian Ramamoorthy. From explanation to synthesis: Compositional program induction for learning from demonstration. In *RSS*, 2019.

[7] Robert R Burridge, Alfred A Rizzi, and Daniel E Koditschek. Sequential composition of dynamically dexterous robot behaviors. *The International Journal of Robotics Research*, 18(6): 534–555, 1999.

[8] Silvia Chiappa and Jan R. Peters. Movement extraction by detecting dynamics switches and repetitions. In *NIPS*, 2010.

[9] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron Courville, and Yoshua Bengio. A Recurrent Latent Variable Model for Sequential Data. *Advances in Neural Information Processing Systems*, 2015.

[10] K. R. Dixon and P. K. Khosla. Trajectory representation using sequenced linear dynamical systems. In *ICRA*, 2004.

[11] Richard C Dorf and Robert H Bishop. *Modern control systems*. Pearson, 2011.

[12] Luu Duc, Achim Ilchmann, Stefan Siegmund, and Peter Taraba. On stability of linear time-varying second-order differential equations. *Quarterly of applied mathematics*, 64(1):137–151, 2006.

[13] Chelsea Finn, Paul Christiano, Pieter Abbeel, and Sergey Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852*, 2016.

[14] Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *NIPS*, 2017.

[15] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. 2018.

[16] Robert J Full and Daniel E Koditschek. Templates and anchors: neuromechanical hypotheses of legged locomotion on land. *Journal of experimental biology*, 202(23):3325–3332, 1999.

[17] Seyed Kamyar Seyed Ghasemipour, Richard Zemel, and Shixiang Gu. A divergence minimization perspective on imitation learning methods. *CoRL*, 2019.

[18] Sam Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian Neural Networks. In *NeurIPS*, 2019.

[19] Vincent Le Guen and Nicolas Thome. Disentangling Physical Dynamics from Unknown Factors for Unsupervised Video Prediction. In *CVPR*, 2020.

[20] David Ha and Jürgen Schmidhuber. World models. In *NeurIPS*, 2018.

[21] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *ICML*, 2019.

[22] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *NIPS*, 2016.

[23] Miguel Jaques, Michael Burke, and Timothy Hospedales. Physics-as-Inverse-Graphics: Unsupervised Physical Parameter Estimation from Video. In *ICLR*, 2020.

[24] Rico Jonschkowski, Roland Hafner, Jonathan Scholz, and Martin Riedmiller. PVEs: Position-Velocity Encoders for Unsupervised Learning of Structured State Representations. *CoRR*, abs/1705.09805, 2017.

[25] Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick Van Der Smagt. Deep variational Bayes filters: Unsupervised learning of state space models from raw data. In *ICLR*, 2018.

[26] Diederik Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2014.

[27] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[28] Thomas Kipf, Yujia Li, Hanjun Dai, Vinicius Zambaldi, Alvaro Sanchez-Gonzalez, Edward Grefenstette, Pushmeet Kohli, and Peter Battaglia. CompILE: Compositional Imitation Learning and Execution. In *ICML*, 2019.

[29] George Konidaris and Andrew G Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. In *NIPS*, 2009.

[30] Adam R Kosiorek, Hyunjik Kim, Ingmar Posner, and Yee Whye Teh. Sequential Attend, Infer, Repeat: Generative Modelling of Moving Objects. In *NIPS*, 2018.

[31] Jannik Kossen, Karl Stelzner, Marcel Hussing, Claas Voelcker, and Kristian Kersting. Structured Object-Aware Physics Prediction for Video Modelling and Planning. In *ICLR*, 2020.

[32] Rahul G. Krishnan, Uri Shalit, and David Sontag. Deep Kalman Filters. *arXiv preprint arXiv:1511.05121*, 2015.

[33] Sergey Levine and Pieter Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *NIPS*, 2014.

[34] Yunzhu Li, Jiaming Song, and Stefano Ermon. InfoGAIL: Interpretable Imitation Learning from Visual Demonstrations. In *NIPS*, 2017.

[35] Yunzhu Li, Hao He, Jiajun Wu, Dina Katabi, and Antonio Torralba. Learning compositional koopman operators for model-based control. In *ICLR*, 2020.

[36] Scott W Linderman, Matthew J Johnson, Andrew C Miller, Ryan P Adams David M Blei Liam Paninski Harvard, and Google Brain. Bayesian Learning and Inference in Recurrent Switching Linear Dynamical Systems. In *AISTATS*, 2017.

[37] Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. In *NIPS*, 2018.

[38] Andrew Y Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *ICML*, 2000.

[39] Scott Niekum and Andrew G. Barto. Clustering via dirichlet process mixture models for portable skill discovery. In *NIPS*. 2011.

[40] Pravesh Ranchod, Benjamin Rosman, and George Konidaris. Nonparametric bayesian reward segmentation for skill discovery using inverse reinforcement learning. In *IROS*, 2015.

[41] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.

[42] Richard S Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2): 181–211, 1999.

[43] Peter Toth, Danilo J Rezende, Andrew Jaegle, Sébastien Racanière, Aleksandar Botev, and Irina Higgins. Hamiltonian Generative Networks. In *ICLR*, 2020.

[44] Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *NIPS*, 2015.

[45] Nicholas Watters, Loic Matthey, Christopher P. Burgess, and Alexander Lerchner. Spatial Broadcast Decoder: A Simple Architecture for Learning Disentangled Representations in VAEs. In *ICLR Workshop*, 2019.

[46] Brian D Ziebart, Andrew Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, 2008.

A Additional related work

In addition to the works discussed in the main text, there are two research areas in the unsupervised learning literature worth discussing.

The first is object-centring representation learning. Several models that describe moving scenes using inverse-graphics [30, 23, 31] have access to explicit position and velocity representations, which is desirable for control. However, such models require full visibility of the objects in the environment and the ability to do visual segmentation and grouping of all components in the scene, which prevents their application to real control systems with arbitrary appearance and multiple, possibly occluded parts and joints.

The second is that of learning physically plausible representations (from video) by enforcing temporal evolution according to explicit or implicit physical dynamics [4, 18, 23, 43]. Though promising, these approaches have only been applied to very simple toy environments where dynamics are well known, and are still to be scaled up to real world scenes.

B ELBO derivation

We want to maximize the sequence marginal likelihood:

$$p(\mathbf{I}_{1:T}|\mathbf{u}_{1:T}) = \int p(\mathbf{I}_{1:T}|\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) p(\mathbf{x}_{1:T}|\mathbf{u}_{1:T}) \, \mathrm{d}\mathbf{x}_{1:T}. \tag{12}$$

We factorize the above terms as follows:

$$p(\mathbf{I}_{1:T}|\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) = \prod_t p(\mathbf{I}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \tag{13}$$

$$= \prod_t \int p(\mathbf{I}_t|\hat{\mathbf{x}}_t) p(\hat{\mathbf{x}}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}; \mathbf{v}_{t-1}) \, \mathrm{d}\hat{\mathbf{x}}_t \tag{14}$$

$$p(\mathbf{x}_{1:T}|\mathbf{u}_{1:T}) = \prod_t p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}; \mathbf{v}_{t-1}), \tag{15}$$

where $\mathbf{v}_t = (\mathbf{x}_t - \mathbf{x}_{t-1})/\Delta t$. Hence, $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}; \mathbf{v}_{t-1})$ depends on $\mathbf{x}_{t-2}$ through $\mathbf{v}_{t-1}$, but we will simply use $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}; \mathbf{v}_{t-1}) \equiv p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ for easy of readability. We use an approximate posterior factorized as:

$$q(\mathbf{x}_{1:T}|\mathbf{I}_{1:T}) = \prod_t q(\mathbf{x}_t|\mathbf{I}_t) \tag{16}$$

Using Jensen's inequality we can write (12) as:

$$\log p(\mathbf{I}_{1:T}|\mathbf{u}_{1:T}) = \tag{17}$$

$$= \log \left(\int \frac{\prod_t q(\mathbf{x}_t|\mathbf{I}_t)}{\prod_t q(\mathbf{x}_t|\mathbf{I}_t)} \prod_t \int p(\mathbf{I}_t|\hat{\mathbf{x}}_t) p(\hat{\mathbf{x}}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \, \mathrm{d}\hat{\mathbf{x}}_t \prod_t p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \mathrm{d}\mathbf{x}_{1:T} \right) \tag{18}$$

$$\geq \int \prod_t q(\mathbf{x}_t|\mathbf{I}_t) \left(\sum_t \log \left[\int p(\mathbf{I}_t|\hat{\mathbf{x}}_t) p(\hat{\mathbf{x}}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \, \mathrm{d}\hat{\mathbf{x}}_t \right] + \right.$$

$$\left. + \sum_t \log \frac{p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1})}{q(\mathbf{x}_t|\mathbf{I}_t)} \right) \mathrm{d}\mathbf{x}_{1:T} \tag{19}$$

$$= \sum_t \int q(\mathbf{x}_{t-1}|\mathbf{I}_{t-1}) q(\mathbf{x}_{t-2}|\mathbf{I}_{t-2}) \left(\log \left[\int p(\mathbf{I}_t|\hat{\mathbf{x}}_t) p(\hat{\mathbf{x}}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \, \mathrm{d}\hat{\mathbf{x}}_t \right] \mathrm{d}\mathbf{x}_{t-1} + \right.$$

$$\left. + \mathrm{KL}\left(q(\mathbf{x}_t|\mathbf{I}_t) \| p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \right) \right) \tag{20}$$

$$\geq \sum_t \mathbb{E}_{q(\mathbf{x}_{t-1}|\mathbf{I}_{t-1}) q(\mathbf{x}_{t-2}|\mathbf{I}_{t-2})} \Big(\mathbb{E}_{p(\hat{\mathbf{x}}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1})} \log p(\mathbf{I}_t|\hat{\mathbf{x}}_t) + \right.$$

$$\left. + \mathrm{KL}\left(q(\mathbf{x}_t|\mathbf{I}_t) \| p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \right) \Big) \tag{21}$$

C Full experimental details

C.1 Model architectures

All models used the encoder and decoder from Ha and Schmidhuber [20], except for the point mass dataset, where we use a spatial broadcast decoder [45]. All temporal models were trained using 2-step ahead prediction in the ELBO (instead of single step), which is a straightforward extension of (21), as done in latent overshooting [21]. All experiments use 64×64 RGB frames. To compute the transition matrices as a function of the state we use a fully connected network with 2 hidden layers with 16 units and ReLU activation, with the appropriate input and output dimensionality. In the NewtonianVAE variants, Δt was set to the known environment time step. All models were trained using Adam [26] with a learning rate of $3 \cdot 10^{-4}$ and batch size 1 (a single sequence per batch) for 300 epochs. In the point mass experiments we found it useful to anneal the KL term in the ELBO, starting with a value of 0.001 and increasing it linearly to 1.0 between epochs 30 and 60.

C.2 Simulated point mass environment

The point mass environment is adapted from the `PointMass` environment from the `dm_control` library. The mass is linearly actuated in the 2D plane and its movement bounded by the edges of the frame. The simulator uses a time-step $\Delta t = 0.5$ and the $[x, y]$ forces are in the range $[-1, 1]^2$.

C.3 Simulated reacher environment

The reacher environment is a adapted from the `Reacher` environment and inspired by the simulated reacher task in Kipf et al. [28]. We limit the rotation of the shoulder joint to the $[-160, 160]$ range, and the wrist joint to $[0, 160]$. The simulator uses a time-step of $\Delta t = 0.1$ and the torques are in the range $[-1, 1]$. When generating random rollouts we sample shoulder and wrist angles in the whole range, and when generating demonstrations these angles are sampling according to `0.5+(np.random.rand()-0.5)` and `-np.pi+0.3+np.random.rand()*0.5` (in radians), respectively. A full 100-step demonstrations sequence is shown in Fig. 7.

To evaluate the trained control policies in the simulator, we compute a sparse reward as follows. When the distance between the end affector and the initial target is lower than 0.015 and the joint velocity is lower than 0.2, the agent earns a reward of 1. The targets must be reached in sequence, i.e., if the agent goes straight for the second target without stopping at the first target, the reward is still 0. The optimal agent will thus have a maximum reward of 3. We use 120- and 220-step rollouts when evaluating the noiseless and noisy settings, respectively.

C.4 PR2 robot arm

Real robot experiments were conducted using the left arm of a PR2 robot, with images recorded using a downward facing Kinect 2 camera mounted on the PR2 head. Arm motion demonstrations were obtained by pre-programming the robot to move to various objects in the scene using the MoveIt! motion planning library in the robot operating system (ROS). The robot arm is actuated using 8 torque commands (7 for the joints in the robot arm and one for the robot torso height), which were recorded alongside images.

After preprocessing the images (rescaling to 64×64 and cropping to the region of interest), we are left with 836 frames, which we split into 636 training, 100 validation, and 100 testing frames. Training used a batch size of 20 frames.

Due to the very small amount of training data available, we had to impose further constraints on the model to allow for correct learning. Firstly, the transition matrices were set to $A = 0$, $B = 0$, $C = 1$. Secondly, we added an additional regularization term to the latent space, $KL(q(\mathbf{x}|\mathbf{I})\|\mathcal{N}(0, 1))$, to improve visualization of the goals (though this term was not necessary for obtaining correct sequence segmentations). Finally, we added a batch-wise entropy term in $\boldsymbol{\pi}(\mathbf{x})$ to encourage the use of all modes, as proposed by [5]:

$$\mathcal{L}^{\text{ENT}} = -\frac{1}{J}\sum_{j=1}^{J} \log\left(\frac{1}{T}\sum_{t=1}^{T} \pi_{j,t}\right). \tag{22}$$

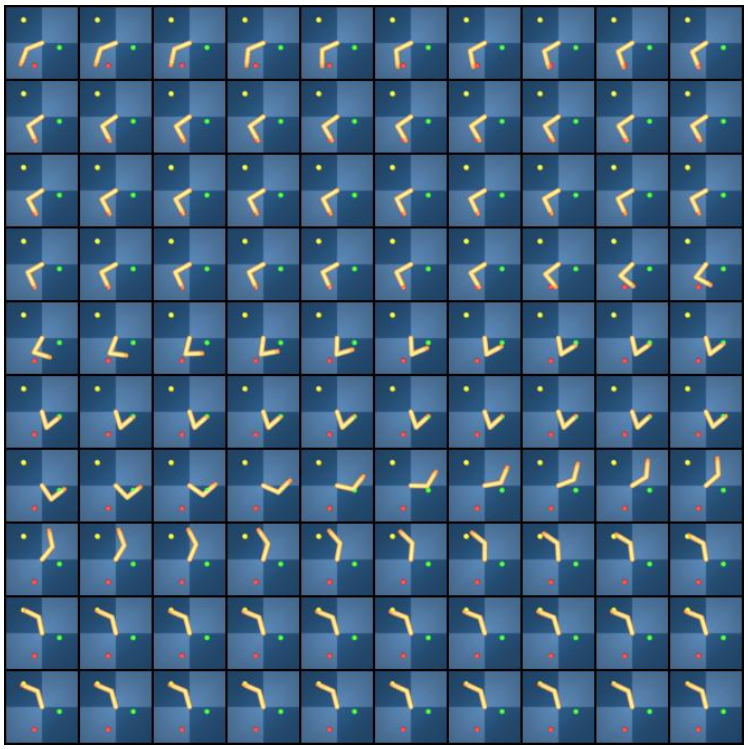

Figure 7: Full demonstration sequence for simulated reacher (progression left to right, top to bottom).

D P-control trajectories

Additional P-control trajectories for the point mass and reacher systems are shown in Figs. 8 and 9, respectively.

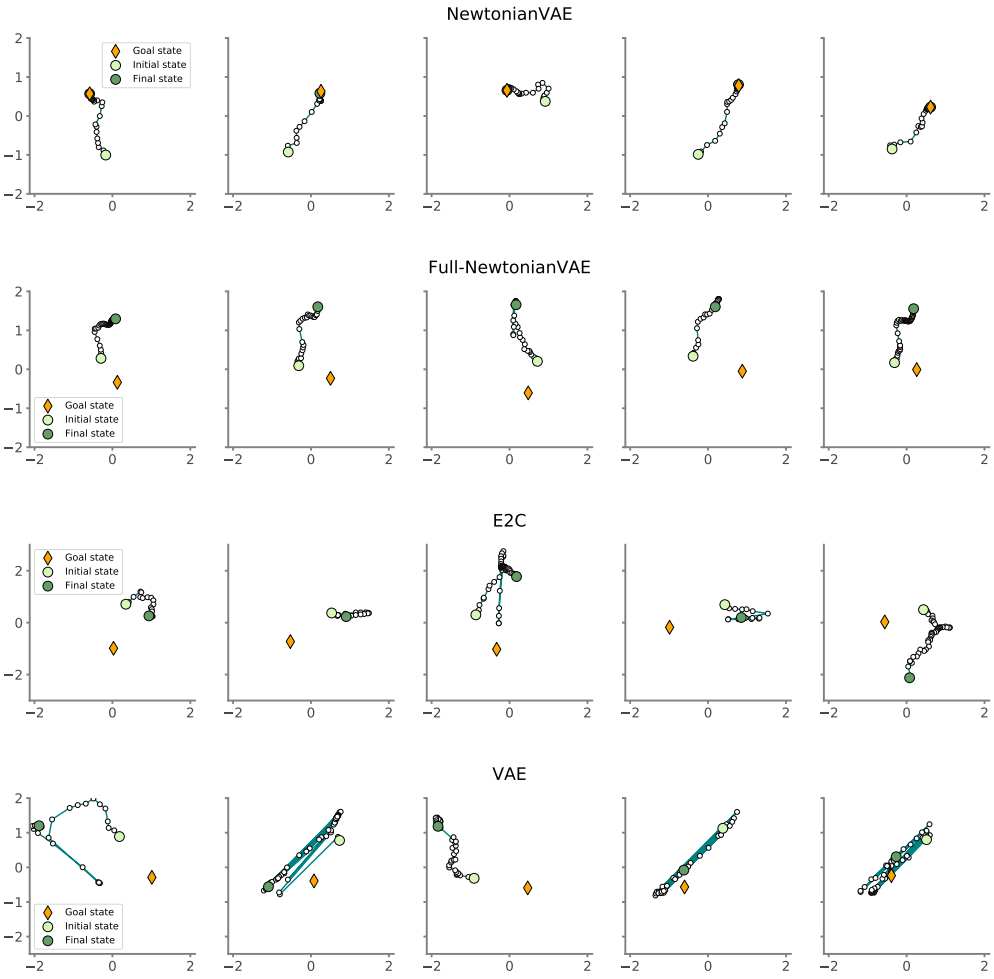

Figure 8: P-controllability in point mass system.

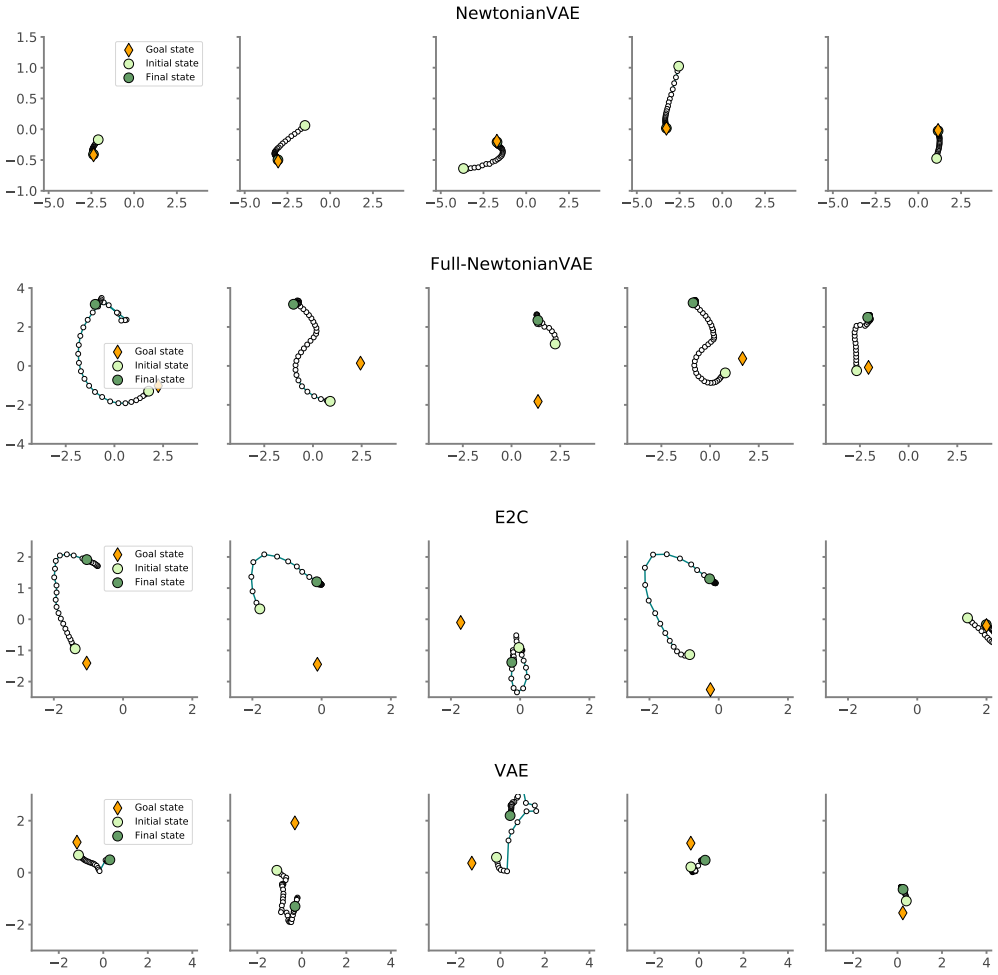

Figure 9: P-controllability in reacher system.