In: Proc. of the 10th International Conference on Simulation of Adaptive Behaviour (SAB '08), Osaka, Japan (2008)

Synthesising Novel Movements through Latent Space Modulation of Scalable Control Policies

Sebastian Bitzer, Ioannis Havoutis, and Sethu Vijayakumar

Institute of Perception, Action and Behaviour, University of Edinburgh, Edinburgh EH9 3JZ, UK
s.bitzer@ed.ac.uk, I.Havoutis@sms.ed.ac.uk,
sethu.vijayakumar@ed.ac.uk

Abstract. We propose a novel methodology for learning and synthesising whole classes of high dimensional movements from a limited set of demonstrated examples that satisfy some underlying 'latent' low dimensional task constraints. We employ non-linear dimensionality reduction to extract a canonical latent space that captures some of the essential topology of the unobserved task space. In this latent space, we identify suitable parametrisation of movements with control policies such that they are easily modulated to generate novel movements from the same class and are robust to perturbations. We evaluate our method on controlled simulation experiments with simple robots (reaching and periodic movement tasks) as well as on a data set of very high-dimensional human (punching) movements. We verify that we can generate a continuum of new movements from the demonstrated class from only a few examples in both robotic and human data.

1 Introduction

As we design robots to become more anthropomorphic with an aim for them to co-exist in human friendly environments, the number of degrees of freedom and consequently the variety of movements that they can execute have grown significantly. This raises many issues concerning the control and planning in these robots: Who defines such a large set of movements for every new robot? How do you make those movements look natural? How do you cope with the large degree of redundancy?

A promising way out of this dilemma is for the robot (student) to learn the desired movements from a teacher (e.g., human demonstrator) through imitation [1]. There are several approaches to this problem depending on the information available to the student. For example, Grimes et al. [2] observe the movement of a teacher in joint angles and learn a probabilistic model which entails a common latent space between teacher and student to produce a stable movement of the student. Peters and Schaal [3], on the other hand, observe an imprecise, supervised movement in the student's own joint space and then, improve on it with reinforcement learning (which needs additional feedback). Such approaches might solve the problems of producing naturally looking movements and appropriate resolution of redundancy, but being only able to imitate one particular movement is rather limiting. An interesting possibility would be to use the demonstrated examples as a basis for generation of more generalised movements from the same class.

Here, we assume that a set of demonstrated examples belong to the same class of movements, i.e., follows a consistent optimisation or redundancy resolution principle in some lower dimensional (and common) *unobserved* task space. Additionally, we assume a rich repertoire of movements that achieve different task goals.

The problem of generating similar movements to a set of examples has been addressed in the computer graphics and animation communities. The aim there often is to generate natural looking human motion adapted to a certain situation given a database of recorded human motion. If the database is big enough and contains all the motions needed, it is often sufficient to use an efficient graph based search algorithm to generate desired movement sequences – however, we consider situations where extensive and exhaustive motion generation or capture is either expensive or infeasible. If two similar motions are available, linear interpolation between these works surprisingly well when they are represented as absolute positions and rotations of body parts in a global coordinate system [4]. Also linear combination of motion sequences has been shown to work reasonably well with the right representation [5]. These approaches, besides having to extrapolate movements in (usually) high dimensional movement space, have the problem of scalability and robustness under perturbation or goal modification because they generate an explicit, fixed movement plan indexed in time.

Ideally, we would want to represent and scale the movements in the corresponding task space, since such representations are very compact and interpretable. However, typically we only have access to the demonstrated movements in joint space. A potential solution to this problem is to find a low-dimensional space with similar properties as the task space by employing appropriate dimensionality reduction [6]. Tatani and Nakamura [7] apply autoassociative neural networks to find compact representations for motions from a humanoid robot, but they are missing a way to represent motion dynamics. While Wang et al. [8] incorporate dynamics in their dimensionality reduction to represent movements, this is not suitable for robotic applications, since it is not robust against perturbations and expensive to compute.

In this paper, we first investigate the qualitative relationship between latent spaces produced by the chosen dimensionality reduction technique and the task spaces of simple robotic setups. Then, we show that the resulting latent spaces can be used to encode and learn control policies which act as robust representations of the example movements and allow easy generalisation to new movements from the same class. Finally, we apply this methodology to human motion capture data to demonstrate its feasibility for complex, high-dimensional real world movement data.

2 Methodology

We adopt a 2-step approach, the schematic for which is laid out in Fig. 1. First, we explore a suitable latent space representation of the observed high dimensional movement data (e.g., in joint space) using appropriate dimensionality reduction techniques. Then, we formulate a representation of trajectories as control policies such that they are spatiotemporally scalable and robust against perturbations. In order to test the scalability of the methods, modulated control policies are then mapped back into the original movement space to generate novel target motion. While task space data (or constraints) are generally not accessible in real world demonstrated examples, we will exploit this formalism in artificial setups to test the viability of our methods against ground truth. Next, we explore the two essential components of our method: a dimensionality reduction algorithm which possesses an inverse mapping and a robust 'control policy' representation that can be easily modulated.

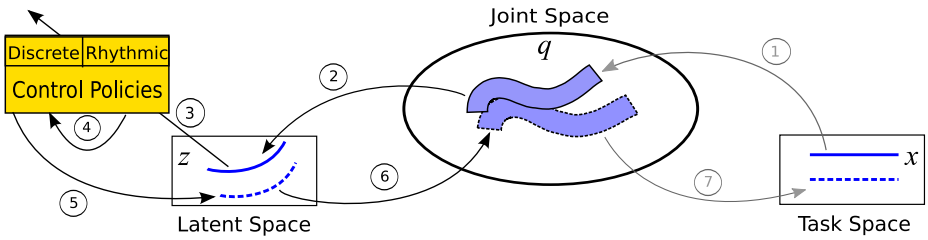


Fig. 1. Experimental methodology. The steps are: (1) inverse kinematics, (2) GPLVM learning, (3) control policy (CP) learning, (4) change of CP parameters, (5) generation of new trajectory with CP, (6) GPLVM mapping, (7) forward kinematics

2.1 Dimensionality Reduction

In general, joint and task spaces are nonlinearly related. Therefore, joint and latent spaces should be, too. Furthermore, we need a mapping from latent to joint spaces to generate new movements from modified trajectories in the latent space. Consequently, we have identified the Gaussian process latent variable model (GPLVM) as a promising candidate for our purposes, details of which are described below. An alternative method with similar properties is the Laplacian eigenmap latent variable model [9] – an extension of the spectral technique of Laplacian eigenmaps that adds continuous mappings between data and latent spaces. However, we do not follow up on this in this paper since our explorative experiments suggest that latent spaces recovered did not maintain a topology that was conducive to control policy modulation (see next subsection).

Gaussian Process Latent Variable Models. The Gaussian process latent variable model [10] is a nonlinear generalisation of probabilistic PCA. It is based on a generative model which uses Gaussian processes to map low-dimensional latent variables $\mathbf{z} \in \mathbb{R}^d$ to high-dimensional observed variables $\mathbf{q} \in \mathbb{R}^D$. The corresponding likelihood for a set of N latent variables $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_N]^\top$ can be written as

$$p(\mathbf{Q}|\mathbf{Z}, \boldsymbol{\beta}) \sim \prod_{i=1}^{D} \exp\left[-\frac{1}{2}\hat{\mathbf{q}}_i^\top \mathbf{K}^{-1}\hat{\mathbf{q}}_i\right]$$

(D independent Gaussian processes) with $\hat{\mathbf{q}}_i = [q_i^1, \ldots, q_i^N]^\top$ the collection of all observed points in dimension i and $\mathbf{K}$ a covariance matrix dependent on $\mathbf{Z}$. Here we use the standard squared exponential covariance matrix with independent, identically distributed noise on the observed variables $\mathbf{q}$:

$$K_{mn} = \beta_1 \exp\left(-\frac{1}{2}\|(\mathbf{z}_m - \mathbf{z}_n)/\beta_2\|^2\right) + \delta_{mn}\beta_3$$

where $\delta_{mn} = 1$ for $m = n$ and 0 otherwise.

Given the set of observed variables $\mathbf{Q}$, their latent representations and values for parameters are then computed by minimising the negative log-likelihood ($-\log p(\mathbf{Q}|\mathbf{Z}, \beta)$). This optimisation is highly susceptible to the initialisation of $\mathbf{Z}$. Usually, we use a PCA initialisation as suggested in [10], but where indicated, we also use initialisation with Laplacian eigenmaps or other results.

Table 1. Definitions of discrete and periodic control policies. For discrete CPs the dynamic variable governing the nonlinearity converges to 0 while it monotonically increases for periodic CPs.

discrete	**periodic**
$\frac{1}{\tau}\dot{v} = \alpha_v(\beta_v(g-z)-v) + \frac{g-z_0}{g^*-z_0^*}f(\xi)$	$\frac{1}{\tau}\dot{v} = \alpha_v(\beta_v(z_m-z)-v) + Af(\phi)$
$\frac{1}{\tau}\dot{z} = v \qquad \frac{1}{\tau}\dot{\xi} = -\alpha_\xi \xi$	$\frac{1}{\tau}\dot{z} = v \qquad \frac{1}{\tau}\dot{\phi} = \omega$
$f(\xi) = \xi\frac{\sum_{i=1}^n \Psi_i(\xi)w_i}{\sum_i \Psi_i(\xi)}$	$f(\phi) = \frac{\sum_{i=1}^n \Psi_i(\phi)w_i}{\sum_i \Psi_i(\phi)}$
$\Psi_i(\xi) = \exp\left(-h_i(\xi-c_i)^2\right)$	$\Psi_i(\phi) = \exp\left(-h_i(1-\cos(\phi-c_i))\right)$

A series of extensions to the GPLVM has been proposed in the literature. All of them lead to some kind of regularisation on the latent variables. This is mostly achieved by introducing a prior over the latents $p(\mathbf{Z})$. For example, the prior suggested in [8] defines dynamics on the latents.

2.2 Control policies

We use discrete and periodic control policies (CPs) to represent goal-directed and periodic movements as attractors of nonlinear dynamical systems [11]. The advantages of this approach are robust representation of movements and easy modifiability of movement parameters such as amplitude, goal point and baseline of oscillations while shape of the CPs is maintained. Alternative ways of representing dynamics, for example with HMMs or linear Gaussian models, do not provide the same level of robustness, suffer from being either restricted to a fixed set of discrete states, only allowing linear dynamics, or expensive computations. In the following, we present our adaptation of the formulation in [11] such that we can explicitly incorporate modifiable start and end positions. Note that only motion in one dimension (e.g. joint) is represented. Consequently, for motion in d dimensions d control policies must be learnt.

Discrete. Discrete movements (e.g., reaching) are characterised by a starting state, z_0, some state trajectory and a goal state, g. The formalisation of such a system is shown in Table 1(left). Ignoring the details of the modulating function f, this is a linear, two-dimensional dynamical system with a single, attracting stable point at $[g, 0]$. f is used to shape the trajectory of the dynamical system between z_0 and g. It can be represented as a weighted sum of RBF basis functions which depend on the state, ξ, of a canonical system that converges to 0. The number of basis functions, n and their width and centres, h_i, c_i, are chosen a priori. Given a complete movement $[\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}]$, the weights, w_i, of the nonlinear component are learnt. Once the movement is learnt (or encoded as a CP with start state z_0^* and goal g^*), we can change the start state and goal to produce a qualitatively equivalent dynamics of motion in different parts of the state space of z (which can either be a joint angle, or a dimension in our latent space).

Periodic. Periodic control policies work similarly, as shown in Table 1(right). Instead of a goal state, we have a baseline of oscillation, z_m. The nonlinearity, f, is now governed by a periodic, canonical system with phase velocity ω. Once the weights are learnt to fit a given periodic movement, we can adapt the amplitude, A, of that movement and move it around in state space by changing the baseline, z_m, without losing the shape of the CP. In our implementations, we choose the mean of a data set as an approximation for the initial baseline of the oscillation.

3 Experiments

We use the robotics toolbox for Matlab [1] to implement simulations of two different robots. Our first simulation features a 3 Degree of Freedom (DoF) **planar robot arm** that has a shoulder joint and 2 elbow joints, with the end effector constraint to move in a 2D plane. We resolve the redundancy in the inverse kinematics by choosing the joint space configuration, $\mathbf{q}$, closest to a default pose, $\mathbf{q}^*$, for which the task space constraints are fulfilled. In other words, we minimise $\|\mathbf{q} - \mathbf{q}^*\|^2$ subject to $\mathrm{k}(\mathbf{q}) - \mathbf{x} = 0$ with $\mathrm{k}(\mathbf{q})$ being the forward kinematics. The second platform that we use is the **PUMA-560 robot arm** with 6 DoFs joints (3 translational plus 3 rotational). However, we fix the rotation of the end-effector to a default value in our simulation. For the PUMA-560 robot, there are always 8 alternative joint angle configurations which all correspond to the same translation and rotation of the end-effector. Of these alternatives, we choose the solution which is right handed, has elbow up and non-flipped wrist.

3.1 Task Space vs. Latent Space

In our first experiment, we explore the relationship between the task space used to produce the example movements and the latent space resulting from nonlinear dimensionality reduction on such data. To begin with, we use a uniform grid data in task space to verify that the important properties of the task space are recovered. In particular, we sample 256 data points regularly spaced from a 2D task space. For the planar arm, the data points are spaced at 0.1m (see Fig. 2, left, blue +) while for the Puma arm[2] the points are separated by 0.027m (see Fig. 2, left, green +). For each of the 256 points in task space, we obtain a corresponding robot configuration in joint space using inverse kinematics and run the GPLVM on them to find a latent space configuration.

If for the same robot, a different inverse kinematic solution is chosen, i.e. existing redundancies are resolved in a different way, the data in joint space corresponding to the original task space points will change. Ideally, we would like the dimensionality reduction technique to show some sort of invariance to this source of variability. We investigate resulting latent spaces for 3 different simulations: we use the planar arm with the inverse kinematics as described above as well as one where the deviation from default for the first joint is weighted four times higher. In the third simulation, we use the Puma arm as described above.

Fig. 2 shows the resulting GPLVM latent spaces. Compared to the original grids, we see that the grids in latent space are nonlinearly distorted. However, the spatial topology of the original task space grids are maintained in the latent space. This suggests that interpolation between neighbouring points in latent space has a direct correspondence to modulation in the underlying task space. As expected, the GPLVM is sensitive to the exact choice of redundancy resolution (e.g., inverse kinematics) – the nonlinear distortions are subtly different in all three examples. However, the properties of all resulting latent spaces allow that a continuous trajectory in task space can be represented as a continuous trajectory in latent space, i.e., recovering a structure topologically similar to the unobserved task space is possible from joint data only.

[1] http://www.petercorke.com/Robotics%20Toolbox.html

[2] The Puma's workspace is 3D, here the data points lie in the X-Y plane with Z=0.

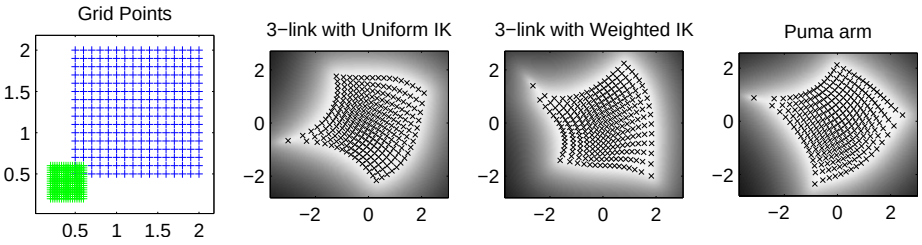


Fig. 2. GPLVM results on simulated robot poses with end-effector positions spanning a 2D grid. (a) The grid in task space (end-effector positions). GPLVM result on: 3DOF planar arm data with (b) standard inverse kinematics (c) weighted inverse kinematics; (d) 6 DOF PUMA arm

3.2 Reaching and Periodic Movements with Control Policies

Next, we investigate reaching movements which are constrained in specific ways in the task space. The aim of this investigation is three fold. Firstly, we would like to verify that topology is maintained in the extracted latent space. Secondly, we would like to investigate whether modulations of the CPs in the latent space recovers the same class of task (and joint) space movements that was used to train the GPLVM. Thirdly, we want to assess the level of generalisation to novel, unseen movements.

The following experiments are done with the simulated Puma robot which has more degrees of freedom than the planar arm. We start with a family of straight line, minimum jerk, reaching movement data in task space. To test whether we can reliably reconstruct a movement in latent (and task) space that was not used to produce the latent space, we leave one movement out when training the GPLVM (Step 2, Fig. 1).

After we obtain the latent space, we fit discrete control policies (Step 3, Fig. 1) to a single representative trajectory in latent space. We then generate new latent space movements through modulating the CPs (Step 4, Fig. 1) by reparametrising the start state and goal to match those of the remaining desired movements in latent space. Importantly, to test the generalisation ability, we generate a movement that was not used in the GPLVM training by interpolating the start and goal state of two neighbouring movements. We then evaluate movements generated by the CPs against the original one in latent, joint and task space.

Parallel Trajectories: We begin by considering parallel task space trajectories depicted by grey dots in Fig. 3(right) with the resulting trained latent space shown on the left. The shading visualises the probability that the GPLVM puts on a corresponding point in joint space. In both panels, the grey dots are the data points available in that space. The bold lines in latent space represent the fitted control policy while the thin lines are the result of CP modulation. The trajectories in task space result from mapping the latent space trajectories through joint space to task space (Steps 6-7, Fig. 1).

The deviation of the trajectories from the data points in task space has two possible sources: (i) discrepancy between the latent space data and CP modulated trajectories (thin lines); (ii) reconstruction errors of the GPLVM(i.e, Steps 2 & 6, Fig. 1). Statistics of the trajectory errors in various spaces are summarised in Table 2. We find that the GPLVM reconstruction error is negligible (see first column, Table 2). Consequently most of the deviation in task space is due to deviation of the CP modulation from latent space data exemplars. Overall, however, the generated movements fit the original task space and joint space movements exceptionally well.

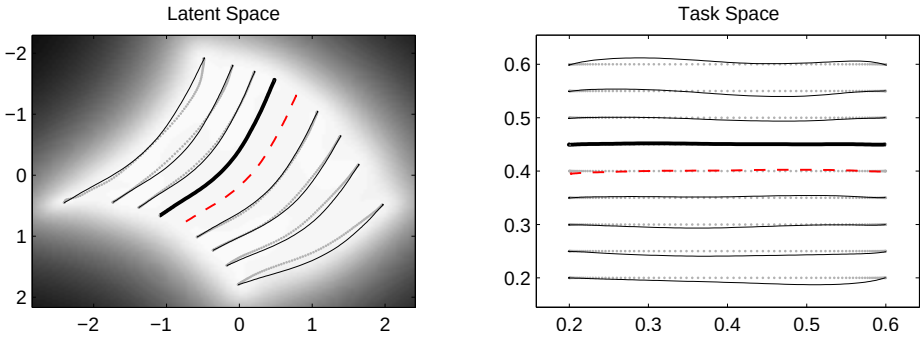


Fig. 3. Fitted (bold), modulated (thin) and interpolated (dashed) discrete CPs for the parallel trajectories of the PUMA arm in latent and task space.

Table 2. Reconstruction and Trajectory Errors (nMSE$\times 10^{-3}$ and standard deviation).

Parallel Trajectory

space	reconstruction	fitted	modulated	interpolated
latent	–	**0.14** $\pm$ 0.16	**4.12** $\pm$ 4.70	–
joint	**0.006** $\pm$ 0.009	**0.13** $\pm$ 0.17	**4.14** $\pm$ 5.19	**0.36** $\pm$ 0.41
task	**0.009** $\pm$ 0.013	**0.16** $\pm$ 0.21	**3.85** $\pm$ 4.95	**0.31** $\pm$ 0.42

Figure-8 Trajectory

space	reconstruction	fitted	modulated	interpolated
latent	–	**2.14** $\pm$ 5.44	**2.52** $\pm$ 2.58	–
joint	**0.000** $\pm$ 0.000	**1.81** $\pm$ 3.76	**2.83** $\pm$ 3.37	**7.96** $\pm$ 15.71
task	**0.000** $\pm$ 0.000	**0.61** $\pm$ 1.36	**3.17** $\pm$ 3.18	**4.98** $\pm$ 7.66

One can note that, as expected, the fitted control policy has smaller trajectory errors than the result of modulation of the CPs to other movements. This can be attributed to the slightly varying shapes that the representations of the movements have in latent space. Also, the topological relationship is preserved as can be seen by the fact that movements close by in latent space have similar 'shapes' - lending itself to better CP modulation. Indeed, that explains the very low error of the interpolated CP (being near the original fitted CP).

Star Trajectories: Next, we test whether these findings transfer to reaching movements where the task constraints are slightly more complex. Our data consists of 10 minimum jerk trajectories in task space where the start and end points are distributed along a quarter circle with radius 0.5 and 2, respectively (see Fig. 4(top right)).

We find comparable results to the earlier discussion – both qualitatively (see Fig. 4(top)) and quantitatively (the statistics of the error, which is very similar to Table 2, is left out in the interest of space). Consequently, we expect our method to be applicable to a wide range of reaching movements with diverse task constraints and orientation.

Periodic Movements: Having explored discrete, point-to-point movements, the natural question is whether the method extends to periodic movements? Again we utilise the Puma platform and simulate figure-8 movements in a 2D task space. We now fit, modulate and interpolate periodic control policies. The task space trajectories:

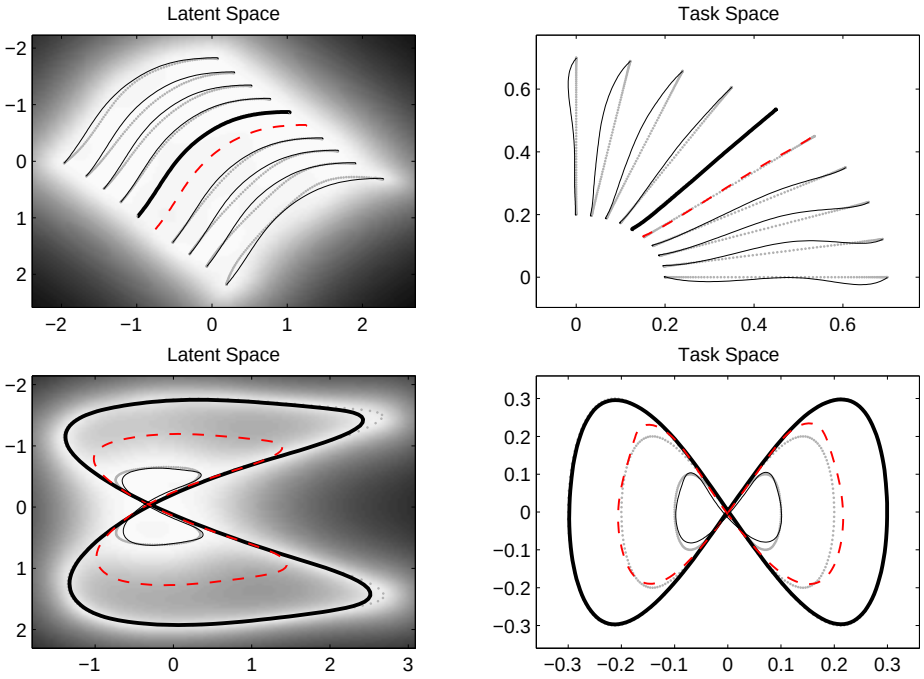


Fig. 4. Fitted (bold), modulated (thin) and interpolated (dashed) Control Policies for the (TOP row) star; (BOTTOM row) figure-8 trajectories of the PUMA arm in latent and task space.

$x(t) = A\sin(\pi t), y(t) = A\sin(2\pi t)$ are generated and then, translated and rotated to fit the Puma workspace. In the latent space, we fit CPs on the figure 8 with $A = 0.3$, modulate with $A = 0.1$ and interpolate for $A = 0.2$.

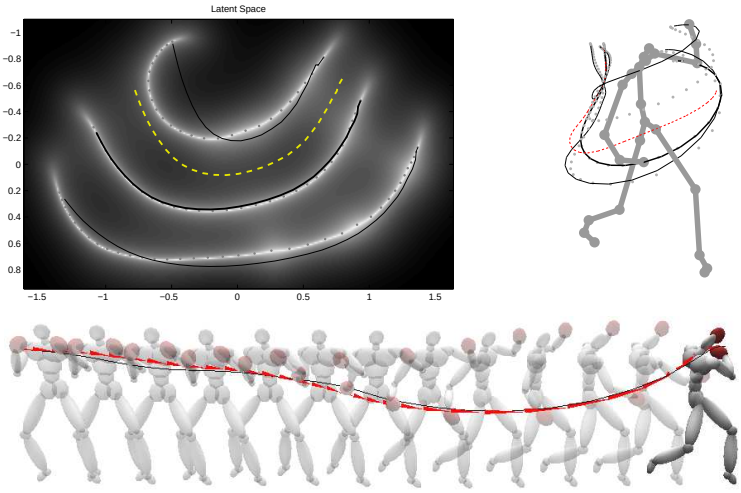
Results are shown in Fig. 4(bottom) and Table 2. Again, the generated movements follow the figure 8s in task space. However, movement shapes show larger variation in latent space, resulting in slightly higher error rates in task space. It is remarkable, though, that we can generate a continuum of complicated task space movements from just two examples.

3.3 Human Motion Capture

The simulation experiments are useful to compare our results to known ground truth, but compared to what we want to achieve the problem setting in these experiments is still easy with very regular movements in only a few degrees of freedom. A realistic setting is provided by real human data recorded with motion capture.

Here, we apply our method to 3 different punching motions from the same person. The 3 movements all have the same style of punch, but differ in the height that the punch hand (right) is travelling. In particular there is a high, a low and a very low punch (see Fig. 5, top right). The recorded data has 63 dimensions (60 angles plus the root offset).

First, we note that linear dimensionality reduction like PCA does not work for this data set. For a 2D PCA latent space the reconstruction error of the data in joint space is



space	reconstruction	fitted	modulated	interpolated
latent	–	**0.44** $\pm$ 0.81	**72.75** $\pm$ 65.56	–
joint	**0.001** $\pm$ 0.002	**2.78** $\pm$ 5.00	**351.55** $\pm$ 510.42	–
task	**0.000** $\pm$ 0.000	**0.67** $\pm$ 1.07	**399.42** $\pm$ 657.52	–

Fig. 5. Fitted (bold), modulated (thin) and interpolated (dashed) discrete Control Policies for the human punching motion in latent and end effector space. MIDDLE: Very low punch with path of right hand for original punch (red triangles) and result from CP modulation (solid line). BOTTOM: Errors for human movements (nMSE$\times 10^{-3}$ together with its standard deviation).

very high (nMSE: 4780.6e-3). Even for a 10D latent space this error is still significantly higher than for the GPLVM with a 2D latent space (86.8e-3 versus 0.001e-3).

Although a standard GPLVM with 2D latent space has virtually no problem reconstructing the data used to train it, the resulting latent space is not useful for learning control policies, because spatiotemporal topological relation is not well maintained, e.g. data points belonging to single punch sequences are broken up and spread discontinuously. Adding a dynamics prior on the data sequences as suggested in [8] improves results a little bit, but not sufficiently.

We find that a suitable initialisation of the latent space is of key importance. We carried out Laplacian Eigenmaps (LE) on a subset of the movement data that only contains motion of the punch arm and then trained a GPLVM on the same data, initialising with the LE result. This gives a good latent space in which topological invariance is maintained, but in this form, it did not provide a mapping to the full body; and furthermore, low and very low punches were switched in order in latent space. We overcome these problems by recomputing a GPLVM on the full data while using an initialisation based on the previous result which we bias towards correct order of the movements.

The resulting latent space is shown in Fig. 5(top left). We learn discrete CPs on the low punch and adapt their start state and goal to fit those of the high and very low punches. We also interpolate a new punch by taking the average of start state and goal between low and high punches. Using the position of the right hand (also compare Fig. 5, top right) to define the task space of these movements, we report nMSEs as presented in Fig. 5(bottom), which produce satisfying results. The learnt punch closely resembles

the original, although it was not possible to modulate the learnt control policies such that they match the other punches very precisely – especially , for the high punch scenario. However, playing a sequence of the generated movement creates natural looking punches that have slight offsets in the joint space. This is a consequence of the different intrinsic shape of the latent representation of the high punch compared to the low punch, which also influences the interpolation.

4 Conclusion

We have proposed a new method of generating a family of movements from examples which is suited for robotic applications with a large number of degrees of freedom. The method uses nonlinear dimensionality reduction to extract a low-dimensional space which captures the essence of the task space constraints and then, learns control policies on the resulting compact representations. New movements are generated by adapting parameters of the learnt control policies in the low-dimensional space and mapping the result back to the original joint space. We have demonstrated this approach in simulated experiments with simple robots and have shown its feasibility for more complicated movements with human motion capture data. In future work, we will investigate how to iteratively use feedback from the mapping to bias the dimensionality reduction such that representations of movements in the resulting latent spaces share stronger shape similarity and hence, allow better interpolation of new movements.

Acknowledgement. This work has been carried out within the SENSOPAC project which is supported by the European Commission through the Sixth Framework Programme for Research and Development.

References

1. Schaal, S., Ijspeert, A., Billard, A.: Computational approaches to motor learning by imitation. Phil. Trans. Royal Soc. London B Biol Sci **358**(1431) (2003) 537–547
2. Grimes, D.B., Chalodhorn, R., Rao, R.P.N.: Dynamic imitation in a humanoid robot through nonparametric probabilistic inference. In: Proc. Robotics: Science & Systems. (2006)
3. Peters, J., Schaal, S.: Reinforcement learning for parameterized motor primitives. In: 2006 International Joint Conference on Neural Networks, IJCNN. (2006) 73–80
4. Wiley, D.J., Hahn, J.K.: Interpolation synthesis of articulated figure motion. IEEE Computer Graphics and Applications **17**(6) (1997) 39–45
5. Giese, M.A., Poggio, T.: Morphable models for the analysis and synthesis of complex motion patterns. International Journal of Computer Vision **38**(1) (June 2000) 59–73
6. Grochow, K., Martin, S.L., Hertzmann, A., Popovic, Z.: Style-based inverse kinematics. In: ACM Transactions on Graphics (Proceedings of SIGGRAPH). (2004)
7. Tatani, K., Nakamura, Y.: Dimensionality reduction and reproduction with hierarchical NLPCA neural networks - extracting common space of multiple humanoid motion patterns. In: Proc. IEEE Intl. Conf. on Robotics and Automation, ICRA. (2003) 1927–1932
8. Wang, J.M., Fleet, D.J., Hertzmann, A.: Gaussian process dynamical models for human motion. IEEE Trans. on Pattern Analysis and Machine Intelligence **30**(2) (2008) 283–298
9. Carreira-Perpinan, M.A., Lu, Z.: The laplacian eigenmaps latent variable model. In: Proc. of the 11th Intl. Conference on Artificial Intelligence and Statistics, AISTATS. (2007)
10. Lawrence, N.: Probabilistic non-linear principal component analysis with gaussian process latent variable models. Journal of Machine Learning Research **6** (2005) 1783–1816
11. Ijspeert, A.J., Nakanishi, J., Schaal, S.: Learning attractor landscapes for learning motor primitives. In: Advances in Neural Information Processing Systems 15. (2003) 1523–1530