In Workshop: Approaches to Sensorimotor Learning on Humanoid Robots, IEEE International Conference on Robotics and Automation (ICRA '09), Kobe, Japan (2009).

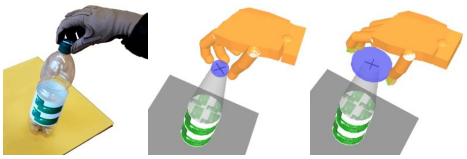
Realising Dextrous Manipulation with Structured Manifolds using Unsupervised Kernel Regression with Structural Hints

Jan Steffen(1), Stefan Klanke(2), Sethu Vijayakumar(2) and Helge Ritter(1)
(1) Neuroinformatics Group, Faculty of Technology, University of Bielefeld, Germany
{jsteffen, helge}@techfak.uni-bielefeld.de
(2) Institute of Perception, Action and Behaviour, School of Informatics, University of Edinburgh, United Kingdom
{s.klanke, sethu.vijayakumar}@ed.ac.uk

***Abstract*— Dextrous manipulation based on techniques for non-linear dimension reduction and manifold learning is an upcoming field of research and offers promising opportunities. Still, many problems remain unsolved and researchers are seeking for new representations that combine efficient learning of examples and robust generalisation to unseen situations. Here, we propose a manifold representation of hand postures, which due to its structural clarity lends itself to simple and robust manipulation control schemes. Focussing on cyclic movements, we describe extensions to the dimensionality reduction algorithm *Unsupervised Kernel Regression* (UKR) that allow to incorporate structural hints about the training data into the learning yielding task-related structures in the manifold's latent space. We present the resulting manifold representation and a simplified controller using this representation for manipulation in the example of turning a bottle cap in a physics-based simulation.**

I. Introduction

During the last decades, researchers have made huge advances in constructing and building anthropomorphic robot hands which have become more and more sophisticated. Together with these developments, researchers are facing the question of how to control such complex, dextrous robots with up to 20 degrees of freedom in up to five fingers plus wrist. It quickly became clear that implementing fixed manipulation programs does not lead to satisfying results as it is very time consuming and not robust against or generalisable to differences in the manipulation situation. Thus, various sophisticated – fundamentally different – approaches have been presented which address these problems. Michelman and Allen [8] implement simple object translations and rotations with the Utah/MIT Hand and combined them to more complex tasks. In this manner, they achieved to remove a child-proof bottle top with two fingers exploiting a decomposition into subtasks and explicit force and position control schemes. Zhang et al. [20] define a graph of vertices representing *canonical grasps* consisting of topological hand/object feature pairs having contact when the associated grasp is achieved. Directed edges between two grasps represent possible transitions which have to be designed as lower-level control laws. Manipulation planning then is implemented as path planning in the graph between defined start and end vertices. Platt et al. [12] address manipulation by sequencing concurrent combinations of hierarchically organised closed-loop controllers. Each of these is derived from potential functions and realises force-related objectives.



(a) (b) (c)

Fig. 1. **(a)** Hand-mounted CyberGlove II during cap turning movement. **(b)** Simulated manipulation scenario for the minimal cap size ($r = 1.5cm$). **(c)** Simulated scenario for the maximal cap size ($r = 3.5cm$).

By dint of operating subordinated controllers in the nullspace of superiors, higher-level conditions like wrench closure can be prioritised and sustained.

In recent years, less engineering motivated approaches to manipulation have been presented that aim at finding less complex representations of the targeted problems. To this end, i.e. *eigengrasps* have been taken into account [2] or lower-dimensional manifolds embedded in the hand posture space are used to facilitate hand control [18]. In previous work, we proposed similar manifolds as *experience* basis for a grasp strategy [14] which we used as motivation for representing manipulations in *Manipulation Manifolds* [15].

In this paper, we are concerned with manipulation movements that are highly structured in the sense that intermediate postures can be described by few parameters in the task space. Specifically, we focus on *cap turning* movements as carried out by a human or robotic hand for opening a bottle (cp. [15]): We aim at compactly representing a set of recorded movements, and from that representation generalise to similar movements (turning caps of unseen size).

Concerning the recording of robot-appropriate data, we follow the idea of Oztop et al. [10], [11] who propose to consider the robot as a tool for the human and to exploit the human learning capacity in learning to use this tool for generating appropriate training data directly on the robot. For the task-related representation of such data, we propose a manifold representation of hand postures, where distinct latent manifold dimensions reflect the natural parameters of the associated movement as directly as possible. For the cap turning example, these would be the (temporal) phase of the periodic manipulation and the bottle cap radius.

The paper is organised as follows: In Section II and III, we briefly review basic *Unsupervised Kernel Regression* and

the new extensions. Section IV will address the retrieval of our manipulation data. In Section V we describe the semi-supervised way of learning the manifolds followed by the evaluation and results in Section VI. In Section VII, we present a manipulation controller that directly exploits the simplicity of the previously trained manifold. We finish with a conclusion and an outlook on future work in Section VIII.

II. Unsupervised Kernel Regression

Unsupervised Kernel Regression (UKR) is a recent approach to learning non-linear continuous manifold representations, that is, to finding a lower dimensional (latent) representation $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) \in \mathbb{R}^{q \times N}$ of a set of observed data $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N) \in \mathbb{R}^{d \times N}$ and a corresponding functional relationship $\mathbf{y} = \mathbf{f}(\mathbf{x})$. It has been introduced as the unsupervised counterpart of the Nadaraya-Watson kernel regression estimator by Meinecke et al. in [7]. Further development has lead to the inclusion of general loss functions, a landmark variant, and the generalisation to local polynomial regression [5]. In its basic form, UKR uses the Nadaraya-Watson estimator [9], [19] as smooth mapping $\mathbf{f} \colon \mathbf{x} \in \mathbb{R}^q \to \mathbf{y} \in \mathbb{R}^d$ from latent to observed data space:

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{N} \mathbf{y}_i \frac{K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i)}{\sum_j K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_j)}. \tag{1}$$

The original estimator realises a smooth, continuous generalisation of the functional relationship between two random variables $\mathbf{x}$ and $\mathbf{y}$ described by given data samples $(\mathbf{x}_i; \mathbf{y}_i)$. Here, $K_{\mathbf{H}}(\cdot)$ is a density kernel (e.g., Gaussian) with associated bandwidth parameters $\mathbf{H}$.

UKR now treats (1) as a mapping from the latent space to the original data space in which the manifold is embedded and from which the observed data samples $\mathbf{Y} = \{\mathbf{y}_i\}, i = 1..N$ are taken. The associated set $\mathbf{X} = \{\mathbf{x}_i\}, i = 1..N$ now plays the role of the input data to the regression function (1). Here, they are treated as *latent parameters* corresponding to $\mathbf{Y}$. As the scaling and positioning of the $\mathbf{x}_i$'s are free, the formerly crucial bandwidth parameter $\mathbf{H}$ becomes irrelevant and we can use unit bandwidths. Thus, the regression function can be denoted as

$$b_i(\mathbf{x}; \mathbf{X}) = \frac{K(\mathbf{x} - \mathbf{x}_i)}{\sum_j K(\mathbf{x} - \mathbf{x}_j)} \tag{2}$$

$$\mathbf{f}(\mathbf{x}; \mathbf{X}) = \sum_{i=1}^{N} \mathbf{y}_i b_i(\mathbf{x}; \mathbf{X}) = \mathbf{Y}\mathbf{b}(\mathbf{x}; \mathbf{X}). \tag{3}$$

where $\mathbf{b}(\mathbf{x}; \mathbf{X}) = (b_1(\mathbf{x}; \mathbf{X}), b_2(\mathbf{x}; \mathbf{X}), \ldots, b_N(\mathbf{x}; \mathbf{X}))^T \in \mathbb{R}^N$ is a vector of basis functions representing the effects of the kernels parametrised by the latent parameters.

As loss function for the UKR training, the reconstruction error is considered and can be denoted as

$$R(\mathbf{X}) = \frac{1}{N} \sum_i \| \mathbf{y}_i - \mathbf{f}(\mathbf{x_i}; \mathbf{X}) \|^2 = \frac{1}{N} \| \mathbf{Y} - \mathbf{Y}\mathbf{B}(\mathbf{X}) \|_F^2 . \tag{4}$$

Here, $\mathbf{B}(\mathbf{X}) = (\mathbf{b}(\mathbf{x}_1; \mathbf{X}), \mathbf{b}(\mathbf{x}_2; \mathbf{X}), \ldots, \mathbf{b}(\mathbf{x}_N; \mathbf{X}))$ is an $N \times N$ *basis function matrix*. Note that moving the $\mathbf{x}_i$ infinitively apart from each other results in $\mathbf{B}(\mathbf{X})$ being the identity matrix which corresponds to a trivial minimisation solution $R(\mathbf{X}) = 0$. In order to prevent this undesired case, several regularisation methods are possible [5]. Most notably, UKR can very efficiently include leave-one-out cross-validation, that is, reconstruct each $\mathbf{y}_i$ without using itself. To this end, the only additional step is to zero the diagonal of $\mathbf{B}(\mathbf{X})$ before normalising its column sums to 1.

For a preselected kernel, the non-linear reconstruction error (4) only depends on the latent parameters $\mathbf{X}$ and thus can be optimised with respect to $\mathbf{X}$ by gradient-based methods. As such often suffer from getting stuck in poor local minima, an appropriate initialisation is important. Here, depending on the problem, PCA [4], Isomap [17] or LLE [13] are usually good choices. These eigenvector-based methods by themselves are quite powerful and efficient in uncovering low-dimensional structures in data sets. Contrary to UKR, however, PCA is restricted to linear structures and Isomap as well as LLE do not provide continuous mappings – a combination with UKR yields the best of both worlds.

An inverse mapping $\mathbf{x} = \mathbf{f}^{-1}(\mathbf{y}; \mathbf{X})$ from data space to latent space is not directly supported in UKR. Instead, one may use an orthogonal projection to define a mapping $\mathbf{x}^\star = \mathbf{g}(\mathbf{y}; \mathbf{X}) = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{f}(\mathbf{x}; \mathbf{X})\|^2$ which approximates $\mathbf{f}^{-1}(\cdot)$. For further details, please refer to [7], [5].

III. UKR with Structural Hints

To provide implicit mechanisms to incorporate given knowledge about training data structures, we presented several extensions to original UKR training [16]. These extensions enable the method to learn sequences of chronologically ordered data in a *semi-supervised* manner:

a) represent periodic movements: We assume latent parameters $\mathbf{x}$ consisting of one periodic *temporal* dimension and one or several (usually) non-periodic *data* dimensions. This inhomogeneous structure of the latent space can be captured by utilising different univariate kernels K_l (parametrised by Θ_l) for different latent dimensions l. The basis functions (2) then consist of the normalised products of these kernels:

$$b_i(\mathbf{x}_j; \mathbf{X}) = \frac{\prod_{l=1}^{q} K_l(x_{i,l} - x_{j,l}; \mathbf{\Theta}_l)}{\sum_k^N \prod_{l=1}^{q} K_l(x_{k,l} - x_{j,l}; \mathbf{\Theta}_l)}. \tag{5}$$

As kernel for the non-periodic dimensions, we use a standard Gaussian kernel with (inverse) bandwidth parameter Θ:

$$K_g(x_i - x_j; \Theta) = \exp\left[-\frac{1}{2}\Theta^2 (x_i - x_j)^2\right]. \tag{6}$$

As kernel for the periodic dimensions, we proposed a sin^2 kernel, periodic in $[0; \pi]$, again with parameter Θ:

$$K_{\circlearrowleft}(x_i - x_j; \Theta) = \exp\left[-\frac{1}{2}\Theta^2 \sin^2(x_i - x_j)\right]. \tag{7}$$

Up to normalisation and scaling, the kernel is equivalent to the von Mises distribution [6] which has been already used by Bishop et al. [1] to represent periodic data characteristics.

b) consider sequences of data: The affiliation of data to sequences enables us to influence the latent parameter adaptation such that sequence-specific mechanisms can be involved supplemental to the general holistic optimisation.

c) conserve sequence order in latent space: To propagate the temporal order of training sequences $\mathcal{S}_\sigma = (\mathbf{y}^\sigma_1, \mathbf{y}^\sigma_2, \ldots, \mathbf{y}^\sigma_{N_\sigma})$, $\sigma = 1..N_\mathcal{S}$, to the latent space structure, the corresponding latent parameters $(\mathbf{x}^\sigma_1, \mathbf{x}^\sigma_2, \ldots, \mathbf{x}^\sigma_{N_\sigma})$ need to reflect this order in their latent temporal dimension d_t. In the periodic case using *closed* sequences of training data ($\mathbf{x}^\sigma_0 = \mathbf{x}^\sigma_{N_\sigma}$), we can express this condition by adding a regularisation term to the reconstruction error (4) which penalises the sum of successor distances:

$$E_{cseq}(\mathbf{X}) = \sum_{\sigma=1}^{N_\mathcal{S}} \sum_{i=1}^{N_\sigma} \sin^2(x^\sigma_{i,d_t} - x^\sigma_{(i-1),d_t}). \qquad (8)$$

d) damp intra-sequence parameter variations: The basic idea is that the underlying movement parameters usually do not change during single sequences – that is, in terms of cap turning, the radius of the cap does not change during the motion. To involve such regularisation in UKR learning, we penalise high variances in the non-temporal dimensions $k \neq d_t$ – as before as additive penalty term to the reconstruction error (4):

$$E_{pvar}(\mathbf{X}) = \sum_{\sigma=1}^{N_\mathcal{S}} \sum_{k \neq d_t} \frac{1}{N_\sigma} \sum_{i=1}^{N_\sigma} \left(x^\sigma_{i,k} - \langle x^\sigma_{\cdot,k} \rangle\right)^2 \qquad (9)$$

The resulting overall loss function of UKR for periodic data sequences then can be denoted as

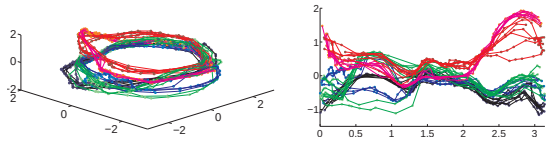
$$E(\mathbf{X}) = R(\mathbf{X}) + \lambda_{cseq} \cdot E_{cseq}(\mathbf{X}) + \lambda_{pvar} \cdot E_{pvar}(\mathbf{X}). \quad (10)$$

For further details on the UKR extensions for periodic data sequences and a profound analysis, please refer to [16].

IV. Manipulation Data

One usually critical issue of methods for manifold learning or non-linear dimensionality reduction, respectively, is that their (task-related) performance strongly depends on the underlying training data. Although UKR is able to handle noise in the training data to some extend, data collection should be conducted thoroughly to improve learning results.

Concerning the recording of robot-appropriate data, we follow the idea of Oztop et al. [10], [11] who propose to consider the robot as a tool for the human and to exploit the human learning capacity in learning to use this tool for generating appropriate training data directly on the robot. For our experiments, we generated hand posture data with our data glove – an Immersion CyberGlove II with 22 bend sensors for the different joints. We first map the sensor values onto a simulated hand model and perform the data generation in a simulated manipulation scenario (cp. Fig. 1). Here, we incorporate joint angle corrections provided by the collision detection of a physics-based simulation toolkit [3] and more general hand posture corrections performed by the user induced by visual feedback.



(a) (b)

Fig. 2. Structure of the cap turning hand posture data. Different colours encode different cap radii ($r = 1.5cm$ (black), $2cm$ (blue), $2.5cm$ (green), $3cm$ (magenta), and $3.5cm$ (red)). Connected points illustrate neighbouring hand postures within the sequences. **(a)** 3D Isomap embedding (K=10) of the hand posture data. The periodicity of the cap turning movement is clearly reflected in the cylinder-like embedding structure. **(b)** Top view of the cylinder-like embedding in (a). **(c)** 2D unfolding of the cyclic structure of the Isomap embedding in (a) using atan2. This form is used as initialisation of the UKR latent parameters.

By dint of this indirect method, we recorded sequences of hand postures during cap turning movements for five different cap radii ($r = 1.5cm,\ 2.0cm,\ 2.5cm,\ 3.0cm$ and $3.5cm$). For each of these radii, we produced five to nine sequences of about 30 to 45 hand postures each – in total 1204 for all sequences and all radii. Each hand posture consists of a vector of the 24 joint angles of the simulated hand model. For these initial experiments, we only generated data for a fixed position of the cap centre.

V. Training

The main target of this work was to generate manifolds similar to the *Manipulation Manifolds* presented in our earlier work [15], but in a more automatic manner: specific movement parameters – and especially the advance in time – are explicitly represented by specific and distinct manifold dimensions. Like this, the manifold structure itself represents knowledge about the manipulation and a manipulation controller which exploits this simplified structure (like the one presented in Section VII) can easily perform manipulation movements by just straightly navigating through the manifold's latent space. Incorporating the new UKR extensions for chronologically ordered data sequences and the results of the profound analysis [16] in the manifold training mechanism now enables a *semi-supervised* learning scheme for such *Manipulation Manifolds* that no longer requires manual constructions of the UKR latent parameters as before.

Since UKR requires a fix number of latent dimensions, we need to specify the adequate amount based on prior knowledge about the structure of the underlying training data. In our case of the manipulation data – corresponding to the constructed version of the *Manipulation Manifolds* – we aim at two-dimensional latent representations ($q = 2$) of the 24-dimensional hand posture data whereas one dimension corresponds to the cap radius and the other to the temporal advance within the cap turning movement. Taking the new features of UKR for periodic data sequences into account, we choose $K_1 = K_\circlearrowleft(x; \Theta_1)$ (7) as periodic kernel for the periodic temporal dimension and $K_2 = K_g(x; \Theta_2)$ (6) as non-periodic kernel for the non-periodic radius dimension. For initialising the model, we first compute a 3D embedding of the data using Isomap (here: neighbourhood radius K=10; results were very robust against different choices of

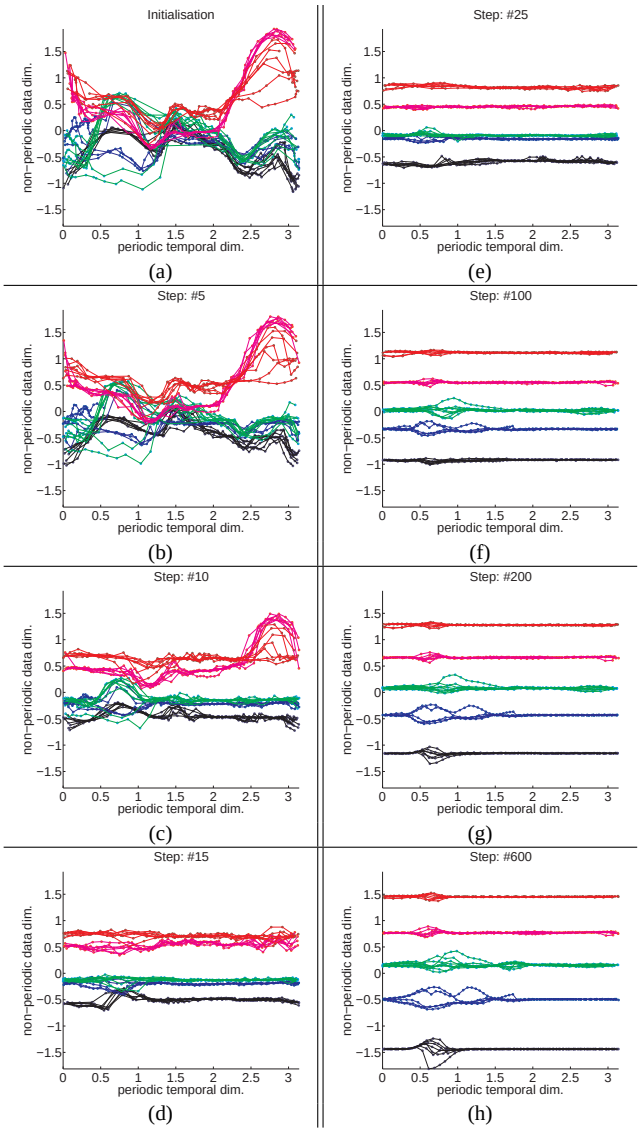


Fig. 3. Development of UKR latent variables during training (Colours as in Fig.2). **(a)** Initialisation with atan2-mapped Isomap-embedding. The horizontal direction corresponds to the periodic temporal latent UKR dimension, the vertical to the non-periodic latent data dimension. **(b-e)** after 5, 10, 15 and 25 optimisation steps: intra-sequence variances in latent data dimension are reduced, the separation of different sequences is not yet finished. **(f-g)** after 100 and 200 steps: sequence representations are now correctly ordered, flat in the (vertical) data dimension and clearly separated. Larger inter-sequence distances yield more intra-sequence distinctions (inducing larger intra-sequence data variances). **(h)** after 600 steps: training result.

K), which is visualised in Fig.2a. Here, because standard Isomap cannot directly embed into a periodic space, the extra dimension is necessary to preserve the local structure of the data. However, it is straightforward to detect the 2D subspace containing the periodicity, and to map these coordinates to an angular representation; in this case e.g. using $atan2$, leaving the non-periodic dimension unchanged (cf. Fig.2b). Following the parameter selection hints determined from the observations presented in [16], we set the parameters $(\Theta_1, \Theta_2, \lambda_{cseq}, \lambda_{pvar})$ of the UKR model to $(10, 4, 1, 1)$. We consider five different training sequences, each consisting of all available training data for one specific cap radius

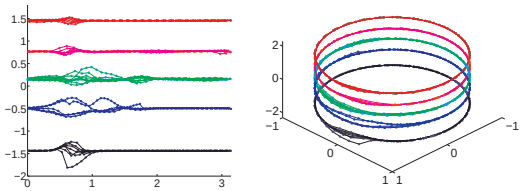


Fig. 4. Training result. (Colours as in Fig.2). **(a)** 2D plot of the latent parameters. The horizontal direction corresponds to the cap radius (*data*) dimension, the vertical to the periodic *temporal* latent dimension (period: $[0; \pi]$). **(b)** 3D mapping of (a). Here, sin/cos are used to generate the 2D-circular representation of the 1D-angular temporal latent dimension.

and run the gradient-based optimisation for 600 steps. Fig. 3 visualises the different stages of the latent parameters during the training. Fig. 3a depicts the utilised initialisation of the latent parameters. Fig. 3(b-e) show their development after 5, 10, 15, and 25 optimisation steps. Already in this early stage of the training, the representations of the single sequences are flattened in their radius dimension. This early effect results from the still tightly packed sequences pushing each other apart and the regularisation of the variance in the radius dimension of the single sequences by dint of the parameter variance penalty (9). The order of the sequence representations in radius direction corresponds to the correct order of the radii (from bottom to top): black ($r = 1.5cm$), blue ($2cm$), green ($2.5cm$), magenta ($3cm$), and red ($3.5cm$). However, the separation of the sequences is not yet finished – especially the sequences for $r = 2cm$ and $2.5cm$ (blue and green) are very close to each other and partly overlap. Indeed, the distinction then is intensified in the following training (cf. Fig. 3(f-g): after 100 and 200 steps, respectively) and finally very explicit after 600 steps, as depicted in Fig. 3h. However, the larger inter-sequence distances yield more space for intra-sequence distinction as well and thus, the variance in the radius dimension of the single sequence representations grows with progressing sequence separation and results in partly non-flat sequence representations.

VI. Evaluation and Results

The resulting latent parameters of the UKR training for periodic data sequences is visualised in Figure 4. The latent structure reflects the initially targeted characteristics: the representation of the single sequences of training data are well separated from each other and have low variance in their data (radius) dimension. However, from the application point of view, the task related abilities of the resulting manifold are of special interest and thus, the main focus of this evaluation will be the manifold's capability to represent and reproduce the underlying movements (or manipulations) and to synthesise new unseen motions.

Figure 5 visualises the training result in form of hand posture pictures. The postures correspond to the data space images $\mathbf{f}(\mathbf{x}; \mathbf{X})$ of regularly sampled positions $\mathbf{x}$ in latent space. Here, the bottom row corresponds to the minimal value of the latent radius dimension (which is also the smallest cap radius in this case) and the top row to the maximal

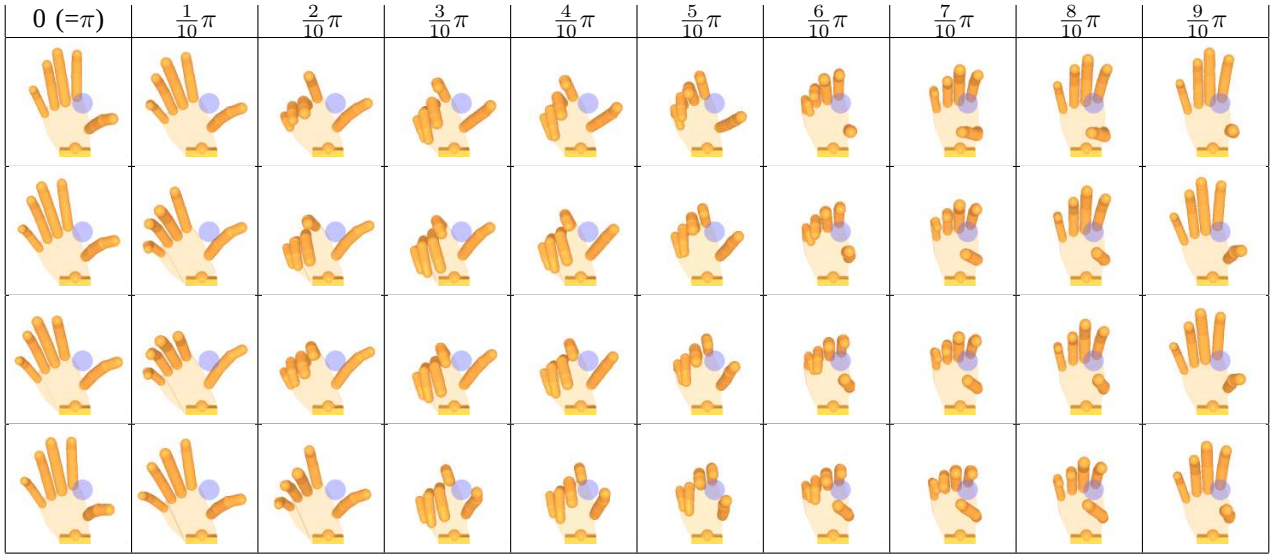


Fig. 5. Visualisation of the training result in the hand posture space. The depicted postures correspond to the reprojections $\mathbf{f}(\mathbf{x};\mathbf{X})$ of regularly sampled positions $\mathbf{x}$ in the trained latent space. In each picture, a bottle cap of the radius $r = 1.5cm$ is shown as comparison aid. The latent radius dimension is mapped onto the vertical direction, the latent temporal dimension onto the horizontal direction of the depicted grid. Remark that the temporal dimension is periodic in $[0;\pi]$ (and thus, that the first column is the successor of the last one). A more sophisticated impression of the manifold smoothness can be obtained by means of the movie available under `http://www.techfak.uni-bielefeld.de/~jsteffen/mov/icra2009/upkrturn/`.

value. In horizontal direction, the temporal dimension – periodic in $[0;\pi]$ – is sampled in steps of $\frac{1}{10}\pi$, whereas the last column is $\frac{9}{10}\pi$. Hence, the next step would be $\frac{10}{10}\pi$ which is equivalent to the first column ($= 0$) due to the periodicity.

The represented movement consists of different phases: 1) columns 0 to $\frac{2}{10}\pi$: the hand is opened and moves in the direction of the little finger (backward movement), 2) columns $\frac{2}{10}\pi$ to $\frac{4}{10}\pi$: the fingers close depending on the associated cap radius (closing movement), 3) columns $\frac{4}{10}\pi$ to $\frac{7}{10}\pi$: the closed hand moves in the direction of the thumb and rotates around the cap position (turning movement). Afterwards, the hand opens again (columns $\frac{7}{10}\pi$ to $\frac{9}{10}\pi$) and the motion smoothly transitions back to the beginning in column $0(=\pi)$. Whereas the open hand phase is very similar (or not characteristically differing from each other compared to the associated cap radius) for the different radii, the hand postures significantly differ in the turning phase (columns $\frac{4}{10}\pi$ to $\frac{7}{10}\pi$) where the fingers had contact with the cap in the training data generation. Here, different levels of hand closure (around the cap position) are clearly visible.

The results presented in Fig. 5 show that the cap turning movement described by the training data is – at least in principle – captured by the manifold learned in the new semi-supervised manner. The general characteristics are the same as for the previously *constructed* manifold presented in [15]. The video referenced in Fig. 5 shows manipulation movements for different latent radius values. In the next section, we present a simplified controller scheme and show that the manifold is able to perform manipulations as well.

VII. Simplified manipulation control

In [15], we already presented a simple control algorithm to perform manipulations with the *Manipulation Manifold* using a manifold approach to grasping to adapt the latent radius value to the presented manipulation context.

A more robust and at the same time less complex manipulation controller performs this radius adaptation in a simplified manner (cp. Fig. 6). The algorithm starts in an initial hand posture which is associated with a fixed latent position on the manifold lying on the 'maximum radius' border of the latent space in a temporal position where the fingers have contact with the cap (cf. Fig. 6, pos. 1). This initial position needs to be specified beforehand by visual inspection or – if present in the training data – by analysing the finger contacts for different latent time values. If thoroughly examined, this position serves as general starting posture for manipulations with all covered cap radii. The motion controller then is subdivided into two different phases of orthogonal, straight navigations through the latent space.

The first phase corresponds to grasping the cap and is realised by a straight navigation in direction of decreasing radii following the radius dimension. This is visualised in Fig. 6 as a vertical arrow and the trajectory part from point "1" to point "5". The corresponding closing movement of the fingers continues until thumb, fore finger and middle finger have contact (this is the case in point "5" in Fig. 6) which then yields an appropriate latent radius value for the subsequent manipulation movement.

As soon as this simple grasping method succeeds, the controller transitions to the manipulation phase, in which the adapted radius is fixed and the manipulation movement is performed by navigating through the latent space following the temporal dimension (Fig. 6, points "5" to "14"). As the latent space is periodic in this dimension, the cap turning movement controlled this way can be repeated several times by just further increasing the temporal latent value. Several manipulations for different radii produced with this simpli-

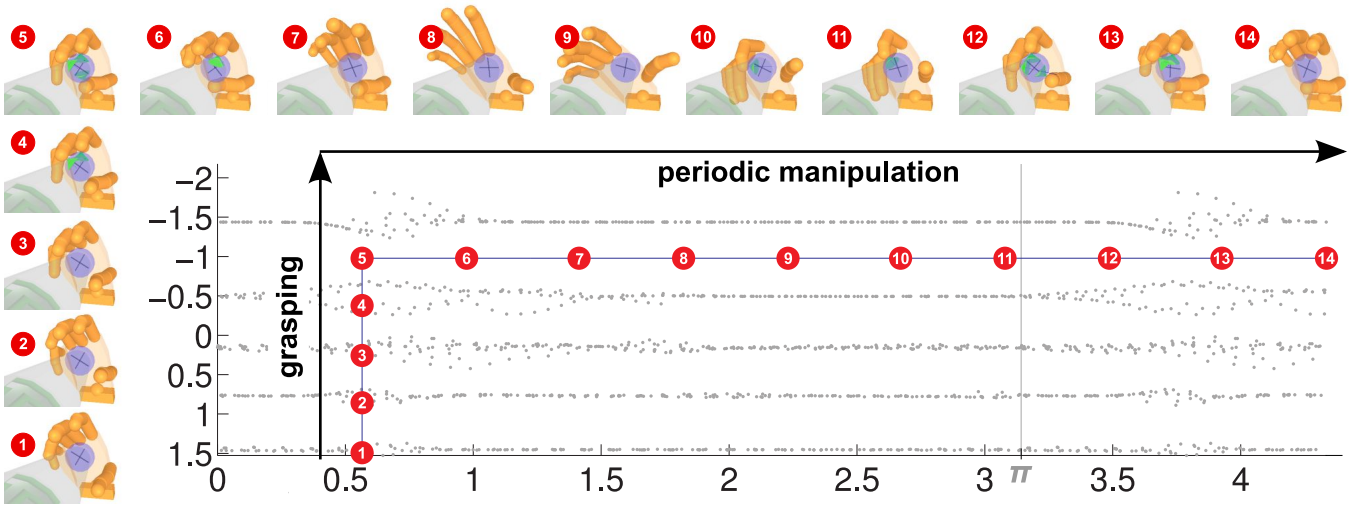


Fig. 6. Visualisation of an exemplary grasping/manipulation sequence using the simplified manipulation controller on the trained manifold. The graph depicts the distribution of the UKR latent parameters (remark the periodicity in $[0; \pi]$ in the horizontal time dimension and the reversed axis direction in the radius dimension). The hand pictures show 14 intermediate postures during the manipulation movement corresponding to the marks '1' to '14' in the graph. The arrows ("grasping", "periodic manipulation") represent the two very simple stages of the controller: a) grasp the cap by navigating through the manifold's latent space from a fix starting point (pic. 1) in the direction of decreasing radius (pic. 2-5) until thumb, fore finger and middle finger have contact (pic. 5) and then b) perform the turning movement by navigating in the orthogonal temporal latent dimension (pic. 5-14). As the temporal dimension is periodic, the turning movement can be reapplied (with smooth transition from one run to the next) by just further increasing the temporal value. There are two videos available at http://www.techfak.uni-bielefeld.de/∼jsteffen/mov/icra2009/ which visualise the manifold as well as the manipulation controller.

fied controller are shown in a video referenced in Fig. 6.

VIII. Conclusion

We presented a new approach to learning *Manipulation Manifolds* in which distinct dimensions represent distinct parameters of the associated manipulation movement. In [15], we have already shown that *Unsupervised Kernel Regression* is well suited to represent manipulation movements based on sequences of recorded human hand data. By incorporating the new extensions presented and analysed in [16], we now achieved to replace the former manifold *construction* by a semi-supervised *learning*. The evaluation of this learned manifold revealed similar characteristics as the manifolds resulting from the previous construction method. Further, we presented a new manipulation controller which uses the trained manifold as basis and represents a simplified but more robust version of the previous approach.

Future work will mainly follow two directions: Firstly, we will further investigate robustness and generalisation abilities of the new approach. Here, we are interested in representing data which additionally cover different positions of the bottle cap and in analysing the minimal amount of training data which still produces good training results. Secondly, we will address different kinds of data like motion capture data from whole body movements.

ACKNOWLEDGEMENT This work has been carried out with support from the German Collaborative Research Centre "SFB 673 - Alignment in Communication" granted by the DFG, from the EU FP6 SENSOPAC project to SK and SV, funded by the European Commission and from the german Cluster of Excellence 277 "Cognitive Interaction Technology (CITEC)".

References

[1] C. Bishop and C. Legleye. Estimating conditional probability densities for periodic variables. *Advances in Neural Information Processing Systems*, 7:641–648, 1995.

[2] M. Ciocarlie, C. Goldfeder, and P. Allen. Dimensionality reduction for hand-independent dexterous robotc grasping. In *Proc. IROS*, 2007.

[3] CM-Labs. Vortex 2.1. www.cm-labs.com/products/vortex, 2005.

[4] I.T. Jolliffe. *Principal Component Analysis*. Springer, New York, 2nd edition, 2002.

[5] S. Klanke. *Learning Manifolds with the Parametrized Self-Organizing Map and Unsupervised Kernel Regression*. PhD thesis, Bielefeld University, 2007.

[6] K. Mardia. Statistics of Directional Data. *Academic Press, London*, 1972.

[7] P. Meinicke, S. Klanke, R. Memisevic, and H. Ritter. Principal Surfaces from Unsupervised Kernel Regression. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(9), 2005.

[8] P. Michelman and P. Allen. Forming Complex Dextrous Manipulations from Task Primitives. In *Proc. ICRA*, 1994.

[9] E. A. Nadaraya. On Estimating Regression. *Theory of Probability and Its Application, Vol.9*, 1964.

[10] E. Oztop, L.-H. Lin, M. Kawato, and G. Cheng. Dexterous Skills Transfer by Extending Human Body Schema to a Robotic Hand. In *Proc. Intl. Conf. on Humanoid Robots (Humanoids)*, 2006.

[11] E. Oztop, L.-H. Lin, M. Kawato, and G. Cheng. Extensive Human Training for Robot Skill Synthesis: Validation on a Robotic Hand. In *Proc. Intl. Conf. on Robots and Automation (ICRA)*, 2007.

[12] R. Platt Jr., A. Fagg, and R. Grupen. Manipulation gaits: sequences of grasp control tasks. In *Proc. ICRA*, 2004.

[13] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, Dec. 2000.

[14] J. Steffen, R. Haschke, and H. Ritter. Experience-based and Tactile-driven Dynamic Grasp Control. In *Proc. IROS*, 2007.

[15] J. Steffen, R. Haschke, and H. Ritter. Towards Dextrous Manipulation Using Manifolds. In *Proc. IROS*, 2008.

[16] J. Steffen, S. Klanke, S. Vijayakumar, and H. Ritter. Towards Semi-supervised Manifold Learning: UKR with Structural Hints. In *Proc. WSOM*, 2009.

[17] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, December 2000.

[18] A. Tsoli and O. Jenkins. 2d subspaces for user-driven robot grasping. In *RSS Workshop on Robot Manipulation: Sensing and Adapting to the Real World*, Atlanta, GA, June 2007.

[19] G. S. Watson. Smooth Regression Analysis. *Sankhya, Ser.A*, 26, 1964.

[20] H. Zhang, K. Tanie, and H. Maekawa. Dextrous manipulation planning by grasp transformation. In *Proc. ICRA*, 1996.