The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Structured Unsupervised Kernel Regression for Closed-loop Motion Control

Jan Steffen[12], Erhan Oztop[23], and Helge Ritter[14]
[1] Neuroinformatics Group, Faculty of Technology, Bielefeld University, Bielefeld, Germany
[2] ATR Computational Neuroscience Laboratories, Dep. of Communication and Cognitive Cybernetics, Kyoto, Japan
[3] National Institute of Information and Communications Technology (NICT), Kyoto, Japan
[4] Cognitive Interaction Technology Center of Excellence (CITEC), Bielefeld University, Bielefeld, Germany
jsteffen@techfak.uni-bielefeld.de, erhan@atr.jp, helge@techfak.uni-bielefeld.de

***Abstract*— Transferring human skills to dextrous robots in an easy, fast and robust way is one of the key challenges that still have to be tackled in order to bring robots to our every-day life. However, many problems remain unsolved. In particular, researchers are seeking new paradigms along with efficient and robust task representations that facilitate adaptation to new contexts and provide a means to appropriately react to unforeseen situations.**

In this paper, we present a new method for robot behaviour synthesis, where intrinsic characteristics of 'Structured UKR manifolds' [13] are used to derive a closed-loop controller based on motion data obtained by the 'Robot Skill Synthesis via Human Learning' paradigm [10]. We apply the method to the task of swapping Chinese health balls with a real 16 DOF robotic hand. Our results indicate that the marriage of 'Structured UKR manifolds' with the 'Robot Skill Synthesis via Human Learning' paradigm yields an efficient way of realising a dexterous manipulation capability on real robots.

I. Introduction

The vision of the robot helper that assists the human in his every-day life has been the motor for research in many domains and since many years. One of the key challenges on the way to making this vision real is the possibility to transfer human skills to dextrous robots in an easy, fast and robust manner. In pursuit of this far reaching goal, a lot of work has been carried out in robotics in the fields of imitation learning and learning from observation/demonstration [1], [2], [3], [4], [5], [11], [12].

Perhaps the simplest form of transferring skills to a robot consists of directly copying the motor commands of the demonstrator to the robot. Whereas this approach can be very effective, its application generally turns out to be impossible in most cases as the motor commands are either not available or inappropriate for the robot.

While extensive research effort in imitation learning is indeed focussed on overcoming these two major problems, we follow the approach of Oztop et al. [10], who studied the skill transfer from human instructors to a robotic platform and proposed the idea to consider the robot as a tool for the human. By doing so, the human includes the robot in his body schema and automatically uses his visuo-motor learning capacity for generating appropriate training data directly on the robot. Following this paradigm, Oztop et al. recorded data from a robot hand - controlled by a human via a motion capture system - performing the swapping of Chinese health balls as described in Fig. 1. Afterwards, they were able to represent this manipulation using an open-loop controller which was able to achieve the same ball swapping task without human guidance. However, as the proposed controller framework did not allow for the incorporation of any sensory feedback, it was limited to actuating the represented motion in a feed-forward manner and thus was not able to react to unforeseen situations.

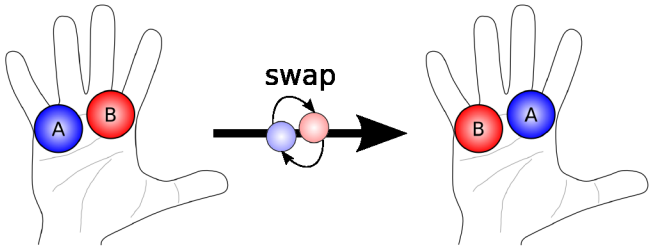

Fig. 1. Schematic visualisation of the Chinese health balls swapping. For our implementation on the real robot hand, we restricted the movements to the four fingers excluding the thumb.

In this paper, we propose a method to derive a closed-loop controller for the same ball swapping task taking the current positions of the balls into account and using them as control parameters for the underlying hand motion.

As a basis for the control framework, we introduce a new way of using Structured UKR manifolds for representing the motion data together with their corresponding manipulation states – in our case the 2D positions of the balls in the palm plane. In combination with the inherent characteristics of the Structured UKR manifold – a modified version of Unsupervised Kernel Regression which we have shown in previous work [13], [14] to be well suited to representing human motion capture data – the representation then lends itself to a simple and robust feedback control scheme that allows for closed-loop motion control.

This paper is primarily intended to present the Structured UKR closed-loop control as a new extension of the existing Structured UKR framework, which includes the robust representation of motions by construction [13], by learning [14] and its use for motion recognition and segmentation [16].

The robotic hand that we used for the implementation of the proposed controller framework is the five-fingered 16 degrees of freedom (DOF) Gifu Hand III ([8]; Dainichi Co. Ltd., Japan) consisting of a thumb and four fingers (Fig.

(a) (b) (c)

Fig. 2. **(a)** The Gifu Hand III. **(b)** The ball tracking setup. The camera for recording the pictures used by the colour blob tracker is positioned above the hand viewing in the direction of the palm plane. **(c)** Exemplary picture of the colour blob tracker camera shown in (b). Tracked blobs (ellipses) are marked by surrounding lines: the two balls with green and blue ellipses; the reference points with red and green.

2(a)). While the thumb provides four independent joints resulting in four DOF, the fingers only have three DOF as - in each case - the two distal joints are coupled. For the ball swapping task, the Gifu Hand was mounted on a PA-10 robot arm (Mitsubishi Heavy Industries) in order to adjust the orientation of the Gifu Hand in a similar way as in [10] (see Fig. 2(b)). In addition, a camera was placed above the scene and directed towards the palm of the hand (cf. Fig. 2(b)). Using the camera pictures, a colour blob tracker provides 2D positions in the palm plane of the two balls relative to the reference blobs near the wrist (see Fig. 2(c)).

The remainder of this paper is organised as follows: In Section II, we briefly review Structured UKR which we use as a basis for the training of the motion representation described in Section III. The feedback control for the manipulation task is detailed in Section IV, and Section V addresses the experimental results of the proposed framework using the Gifu Hand III. Section VI provides a short conclusion.

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 was 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 [6]. In its basic form, UKR uses the Nadaraya-Watson estimator [9], [18] as smooth mapping $\mathbf{f} : \mathbf{x} \in \mathbb{R}^q \rightarrow \mathbf{y} \in \mathbb{R}^d$ from latent to observed data space:

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

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

UKR treats Eq.1 as a mapping from 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 $\mathbf{\Theta}$ 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 objective function for the UKR training, the following reconstruction error is used:

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

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 $\mathbf{x}_i$ infinitely 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 [6]. Most notably, with UKR one can very efficiently perform leave-one-out cross-validation, that is, reconstruct each $\mathbf{y}_i$ without using the $\mathbf{y}_i$ term 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 density kernel, the highly non-linear reconstruction error (4) only depends on the set of latent parameters $\mathbf{X}$ and thus can be optimised with respect to $\mathbf{X}$ by gradient-based methods. As such methods often suffer from getting stuck in poor local minima, an appropriate initialisation is important. In the case of UKR, results from spectral embeddings, e.g. performed with Isomap [17], can easily be used for the initialisation. 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 $\hat{\mathbf{x}} = \mathbf{g}(\mathbf{y}; \mathbf{X}) = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{f}(\mathbf{x}; \mathbf{X})\|^2$ which approximates $\mathbf{f}^{-1}(\cdot)$.

In its original form, UKR is a purely unsupervised approach to continuous manifold learning. In order to enable to incorporate prior knowledge about the structure of the training data, we introduced a structured version of UKR training (e.g. [14]). With Structured UKR, it is possible to represent data with a temporal context, like trajectories of hand positions, in a very easy and robust way. In particular, due to the specific training of Structured UKR, the order of the represented time series of training observations $\mathbf{y}_i$ is reflected in their latent parameters $\mathbf{x}_i$ and is captured by one specific latent time dimension. In order to represent periodic

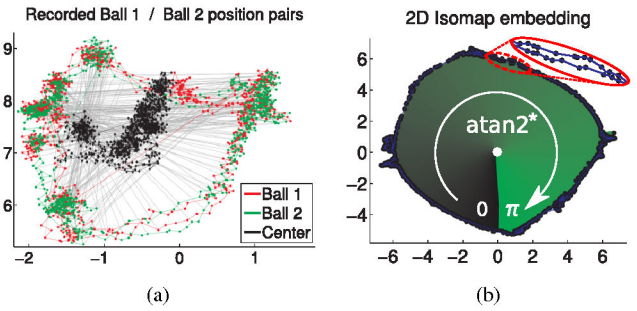

Fig. 3. **(a)** Recorded trajectories of both balls projected onto the x/y-palm plane. Corresponding ball 1 / ball 2 positions are connected by grey lines. The black points mark the trajectory of grey line centre points. Ball positions are normalised such that the wrist reference points (cf. Fig. 2(c)) are at $(-1,0)^T$ and $(1,0)^T$. **(b)** 2D Isomap embedding of all composed hand posture/ball positions observations. The periodic nature is clearly visible.

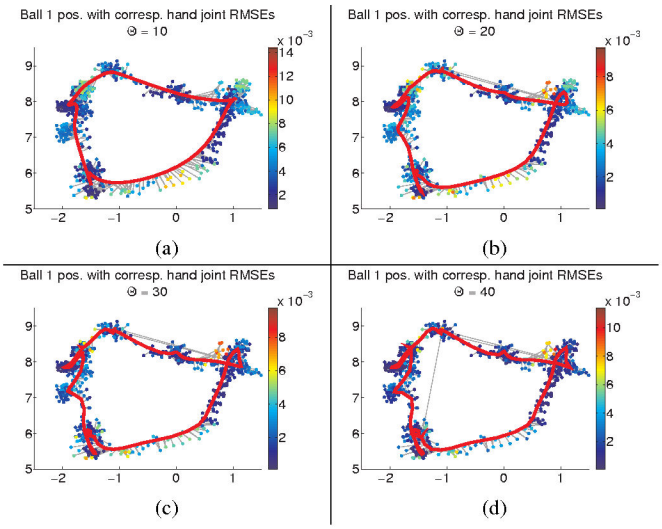

Fig. 4. Visualisations of the data and four trained models from the perspective of the ball positions. Depicted are the normalised positions of Ball 1 (coloured dots; normalised such that the wrist reference points – cf. Fig. 2(c) – are at $(-1,0)^T$ and $(1,0)^T$) in the recorded data and the Ball 1 trajectory as represented in the trained models (red line) for different bandwidth parameters Θ: **(a)** 10, **(b)** 20, **(c)** 30, **(d)** 40. The colours of the dots encode the RMSE between recorded and corresponding represented 16D hand posture as visualised in the colour legend. The gray lines depict the projection errors for the Ball 1 positions resulting from reproducing the recorded composed observations: $\hat{\mathbf{o}} = \mathbf{f}(\mathbf{g}(\mathbf{o}))$.

motions, we use the periodic kernel $K_{\circlearrowleft}(x_i - x_j; \Theta) = \exp\left[-\frac{1}{2}\Theta^2 \sin^2(x_i - x_j)\right]$.

The chronological order of periodic sequences $\mathcal{S}_\sigma = (\mathbf{y}^\sigma_1, \mathbf{y}^\sigma_2, \ldots, \mathbf{y}^\sigma_{N_\sigma})$, $\sigma = 1..N_\mathcal{S}$ with corresponding latent parameters $(\mathbf{x}^\sigma_1, \mathbf{x}^\sigma_2, \ldots, \mathbf{x}^\sigma_{N_\sigma})$ can be propagated by including a penalty term $P_{ord}(\mathbf{X}) = \sum_{\sigma=1}^{N_\mathcal{S}} \sum_{i=1}^{N_\sigma} \sin^2(x^\sigma_{i,d_t} - x^\sigma_{(i-1),d_t})$ in the UKR objective function. d_t denotes the latent time (or phase) dimension. For further details, see [7], [6], [14].

III. Training of the Structured UKR Model

As a basis for the training of the Structured UKR manifold, we recorded a sequence of hand postures together with the corresponding ball positions during four successful cycles of the ball swapping manipulation generated by the original open-loop controller from Oztop et al. [10] (see video [15]).

Figure 3(a) shows the recorded ball pair trajectories (each position normalised such that the reference points shown in Fig. 2(c) are $(-1,0)^T$ and $(1,0)^T$, respectively) and gives an impression of the symmetry in their movements. In order to represent the recorded motion, a 1D latent space ($q = 1$) has been chosen.

In the case of representing the ball swapping task in a way that inherently facilitates a closed-loop control, a fixed relationship between observed hand postures and the corresponding ball positions is required in order to maintain the association of one specific hand posture to the corresponding pair of ball positions that we observed during the data recording.

Correspondingly, we keep the ball positions as part of the observations by creating an 'observation vector' $\mathbf{o}$ consisting of the 16D hand posture $\mathbf{\Phi}$ and the 2D ball positions $\mathbf{p}_1$ and $\mathbf{p}_2$: $\mathbf{o} = (\mathbf{\Phi}^T, \mathbf{p}_1^T, \mathbf{p}_2^T)^T$. Including the ball positions in the observations however comes with an unfavourable side effect: the periodic cycle of the combined observations then spans over the complete cycle of the ball motion, that is, swapping the balls until each one has returned to its initial position. This however requires two swappings of the kind depicted in Fig. 1 and therefore two times the same hand motion. For the recorded data, this can be seen in Fig. 3(b) which depicts a 2D Isomap embedding of the training data of four single ball swappings (or bringing the balls back to their initial positions twice). A closer inspection of the embedding (see Fig. 3(b), zoomed area) reveals the expected effect: the data is embedded in only two cycles. In other words, the combination of ball and hand movements to observation vectors consequentially yields data whose periodicity spans over the whole period of the *ball* movement and thus over two periods of the *hand* movement. The net effect of these characteristics of the composed observations is that the basic hand movement is represented twice in the manifold, resulting in two (slightly different) movements for swapping ball 1 or ball 2, respectively.

Codifications in which the balls are indistinguishable would overcome this drawback, but at the same time break up the periodicity of the ball motion. Since this periodicity is a key characteristics of the manipulation, we decided to follow the straightforward approach described above which also produced good results for the closed-loop control. The motion represented with UKR, when synthesised in an open-loop fashion (see video [15]), however, performs worse than the original controller and only manages to swap the balls every second trial. Indeed, including the sensory feedback in the control loop afterwards yielded much better results, as will be detailed in the following section.

In order to train the manifolds, we applied the method for learning Structured UKR manifolds reviewed in Sec. II. The set of observations consisted of 1757 composed vectors comprising 16D hand postures (finger joint angle vectors; only 12 of the 16 dimensions have effectively varying values) and two 2D ball positions resulting in total in $d = 20$ dimensions. For the latent space, we specify the dimensionality of $q = 1$ which corresponds to the latent time dimension. To take

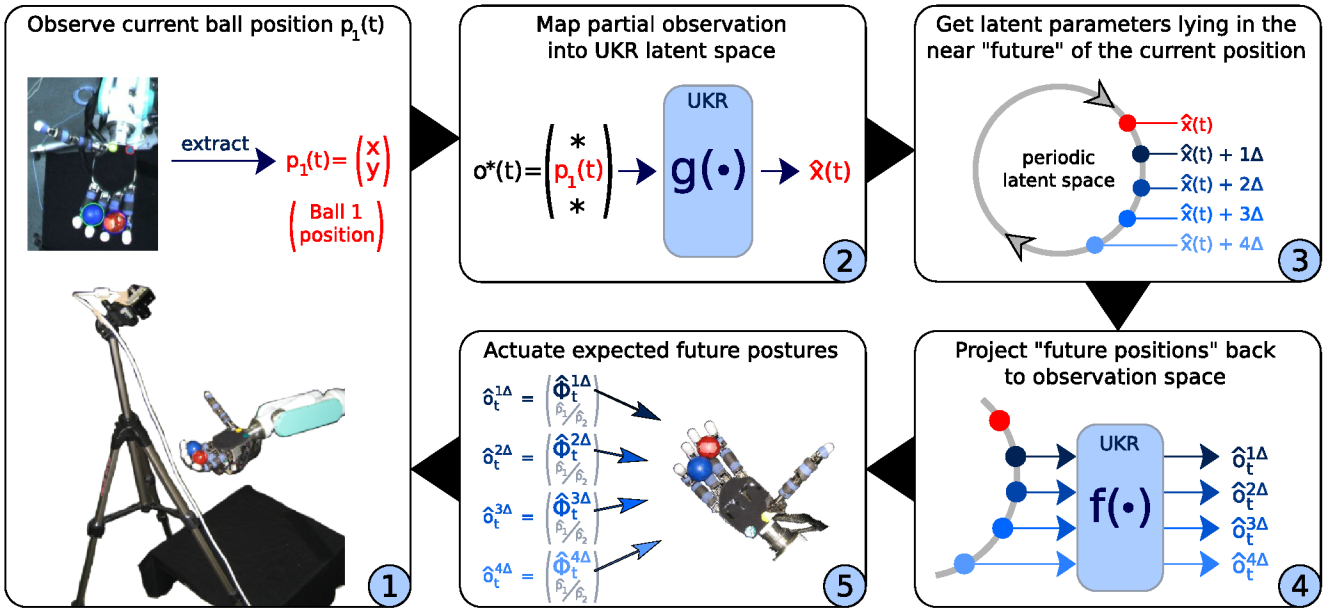

Fig. 5. Schematic overview over the different steps of the feedback control using Structured UKR manifolds (the corresp. step number is marked in the bottom right corner of each box). See the text for a detailed description.

the periodicity of the manipulation into account, we chose $K_1 = K_\circlearrowleft$ (cf. Sec. II). The latent initialisation is realised using the mapping $\mathrm{atan2}^\star(\cdot) = \frac{\pi}{2} + \frac{1}{2}\mathrm{atan2}(\cdot) \in [0; \pi]$ of the 2D Isomap embedding of the training data (see Fig. 3(b)).

With these settings, we trained the Structured UKR manifolds for different inverse bandwidth parameters Θ and evaluated them concerning their appropriateness for the specific task. To this end, we compute the manifold reproductions $\hat{\mathbf{o}}_i = \mathbf{f}(\mathbf{g}(\mathbf{o}_i))$ of the training data $\{\mathbf{o}_i\}$ as a basis, where $\mathbf{o} = (\mathbf{\Phi}^T, \mathbf{p}_1^T, \mathbf{p}_2^T)^T$ and $\hat{\mathbf{o}} = (\hat{\mathbf{\Phi}}^T, \hat{\mathbf{p}}_1^T, \hat{\mathbf{p}}_2^T)^T$.

Figure 4 shows the results for $\Theta = 10, 20, 30$, and 40 from the perspective of the ball-1-trajectories: the points depict the $\mathbf{p}_1$-parts of the training data; their colour encode the root mean square error (RMSE) between the corresponding hand postures $\mathbf{\Phi}$ and $\hat{\mathbf{\Phi}}$. The red lines visualise the $\hat{\mathbf{p}}_1$ trajectory resulting from mapping the whole range of latent parameters $\mathbf{X}$ back to the observation space, giving an impression of the smoothness of the trained manifold.

Whereas the model for the smallest inverse kernel bandwidth parameter $\Theta = 10$ (Fig. 4(a)) yields the smoothest representation, it also results in the highest RMSEs for the reproduced hand postures. Indeed, using this model for generating the ball swapping motion could not produce any ball swappings. By increasing Θ (i.e. decreasing the kernel bandwidth) to 20 (Fig. 4(b)), the RMSEs decrease, but at the price of a less smooth representation as shown by the ball-1-trajectory. However, only a further increase of Θ to 30 (Fig. 4(c)) sufficiently improved the hand posture synthesis ability, eventually leading to a model that is able to reproduce the ball swapping (see video [15]). A further increase of Θ as shown in Fig. 4(d) further decreases the RMSEs in most parts of the mapping, but also results in a more over-fitted solution. As this, however, did not improve the open-loop behaviour of the representation, we chose the model trained with $\Theta = 30$ (Fig. 4(c)) as basis for the feedback control presented in the next section.

IV. Feedback Control using Structured UKR

The training of the Structured UKR manifold described in the last section yields a representation of the underlying manipulation that provides us on the one hand with information about valid combinations of hand postures and ball positions, and on the other hand with knowledge about the chronological order of the data.

Exploiting these two main characteristics of the manifold, a closed-loop feedback control system can be implemented as shown in the schematic overview in Fig. 5:

In the first step (Fig. 5, box 1), the current ball positions $\mathbf{p}_1(t)$ and $\mathbf{p}_2(t)$ are extracted from the camera image as a description of the current manipulation state. These points are treated (separately) as two versions of partial observations of the manipulation: $\mathbf{o}^\star_{\mathbf{p}_1} = (\star, \mathbf{p}_1^T, \star)^T$ and $\mathbf{o}^\star_{\mathbf{p}_2} = (\star, \star, \mathbf{p}_2^T)^T$, respectively. Here, the $\star$ denotes unspecified values.

Considering both ball positions at the same time is also possible and also more logical at first sight. However, since usually not all possible combinations of ball positions are represented in manifold and the orthogonal projection of a non-represented combination onto the manifold may yield a latent position that is appropriate for neither of the actually observed ball positions, this approach yielded worse results in the experiments.

In the second step (Fig. 5, box 2), the partial observations are projected into the UKR latent space: $\hat{\mathbf{x}} = \mathbf{g}(\mathbf{o}^\star; \mathbf{X})$. Here, however, only the specified part of the data is considered ($\mathbf{p}_1$ or $\mathbf{p}_2$) by modifying the orthogonal UKR projection $\mathbf{g}(\cdot)$ accordingly: $\mathbf{g}(\mathbf{y}; \mathbf{X}) = \arg\min_{\mathbf{x}} \parallel \mathbf{S}(\mathbf{y} - \mathbf{f}(\mathbf{x}; \mathbf{X})) \parallel$ with $\mathbf{S}$ being a $d \times d$ diagonal matrix with elements $s_{ii} = 1$ if observation dimension i is specified and $s_{ii} = 0$ else.

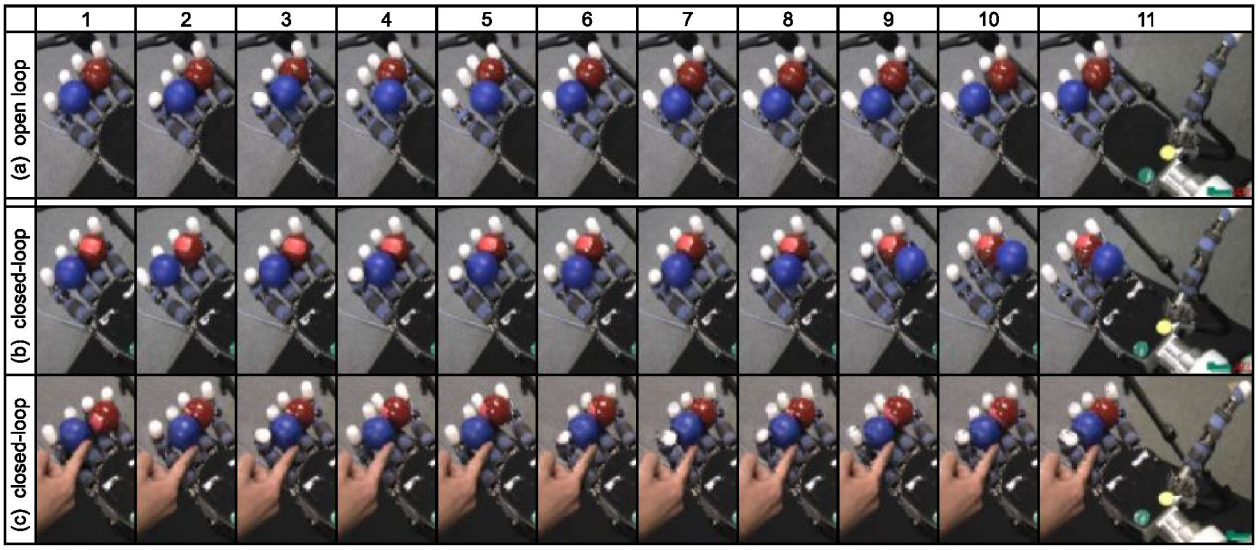

Fig. 6. Exemplary results of **(a)** the UKR open loop control and **(b-c)** the UKR closed-loop control. Whereas the open loop control does not react to unforeseen situations, the closed-loop control adapts to the current manipulation context. The results can be seen best in the corresponding video [15].

This 'partial observation projection' then yields the latent parameter – and thus the point in time or phase of the manipulation – which best matches to the observed ball position. In addition, it also provides means to determine the corresponding full reconstruction $\mathbf{o}$ from the manifold representation using the UKR mapping $\mathbf{f}(\hat{\mathbf{x}}; \mathbf{X})$. Since balls 1 and 2 are identical, we can use this reconstruction to find the ball whose position best matches with its reconstruction. The ball position yielding the smallest reconstruction error then defines the $\hat{\mathbf{x}}(t)$ used in the following process. In addition, if the reconstruction error of one of the balls exceeds a certain threshold, the manipulation can get halted.

For the manipulation control, however, we are more interested in the represented expected *future* development of the movement. Thus, in the third step (Fig. 5, box 3), starting from the identified current latent position within the manipulation (i.e. $\hat{\mathbf{x}}(t)$), we set up a list of k successive 'future' latent parameters using Δ as fixed sample distance: $\mathbf{X}^f_{\hat{\mathbf{x}}(t)} = (\hat{\mathbf{x}}(t)+1\Delta,\ \hat{\mathbf{x}}(t)+2\Delta, \ldots,\ \hat{\mathbf{x}}(t)+k\Delta)$. This list constitutes the latent representation of a plan for a short-term feed-forward motion whereas the combination of the parameters k and Δ specifies the coarseness/speed ("step size" Δ) and the length/duration ($k \cdot \Delta$) of the feed-forward part of the control.

In step four (Fig. 5, box 4), this plan is mapped back into the observation space by means of the UKR mapping $\mathbf{f}(\mathbf{x}; \mathbf{X})$. The resulting observations $\mathbf{O}^f_{\hat{\mathbf{x}}(t)} = (\hat{\mathbf{o}}^{1\Delta}_t,\ \hat{\mathbf{o}}^{2\Delta}_t,\ \ldots\ ,\ \hat{\mathbf{o}}^{k\Delta}_t)$ correspond to the expected actions (i.e. finger movements) and reactions (i.e. ball movements) from a proceeding manipulation.

Since the changes of the ball positions are directly induced by the observed finger movements, however, we also expect that the actuation of these actions result in the ball positions that are similar to the represented ones. Thus, in step five of the feedback control (Fig. 5, box 5), these expected future hand postures are actuated in their chronological order to perform the next steps of the desired manipulation.

After the finger movement, the resulting change of the manipulation state – i.e. the displacement of the balls – is observed by the camera and evaluated in order to adapt the hand motion and set up a new short-term feed-forward plan. In this way, the control loop is closed and restarts in box 1.

V. Experiments

The experiments focus on the qualitative evaluation of the behaviour of the closed-loop control using the presented new extension of UKR in comparison to the UKR control without this extension. They are intended to demonstrate the benefits and new possibilities of the extended control scheme.

The UKR *closed-loop* controller uses the control scheme presented in the last section with the manifold described in Sec. III. For the UKR *open loop* control, we use the same trained UKR manifold as a basis for the motion generation, whereas in this case, no feedback about the current ball positions is used. Instead, the latent space is regularly sampled with a fixed step size Δ. The resulting latent value $x(k) = x_0 + k \cdot \Delta, (k = 1, 2, ..)$ is mapped back to observation space by means of $\mathbf{o}(k) = (\mathbf{\Phi}(k)^T, \mathbf{p}_1(k)^T, \mathbf{p}_2(k)^T)^T = \mathbf{f}(x(k))$ and the corresponding hand joints angles $\mathbf{\Phi}(k)$ are actuated (while the ball positions $\mathbf{p}_1(k)$ and $\mathbf{p}_2(k)$ are ignored). Thus, the represented movement is produced in a purely feed-forward manner. The speed of the movement can be varied with the size of Δ.

As both controls are based on exactly the same UKR representation of the underlying motion, the capability of performing the ball swapping is equally present in both methods. However, by incorporating the ball position feedback with the new control scheme presented in the last section, the robot gains the ability to adequately react to unforeseen events or control failures. The exemplary sequences shown

in Fig. 6 illustrate the different behaviours of open loop and closed-loop control for the same initial ball configuration.

Figure 6(a) shows intermediate hand postures from the open loop control. Here, pics. 1-5 show an unsuccessful attempt to swap the balls. Indeed, as the open loop control cannot react to unachieved sub-goals of the control, the following movement (pics. 6-11) continues as if the blue (bottom left) ball had been correctly moved between red (top right) ball and the palm (cf. Fig. 6(b), pic. 11 for the targeted configuration) and thus tries in the following to bring the red ball to the position to which the blue ball returned.

Figure 6(b) depicts the same initial ball configuration as in Fig. 6(a), but using the closed-loop control (on the basis of the same UKR representation). Here, again, pics. 1-5 show an unsuccessful attempt to swap the balls. But, as the closed-loop control recognises that the blue ball returned to its initial position, the adequate part of the control is repeated (pics. 7-10) and the goal is eventually reached (pic. 11).

Figure 6(c) shows a control scenario in which the blue (left bottom) ball is manually pushed back to the initial position to prevent the ball from swapping. In this sequence, three attempts to push the ball to the correct position can be seen (pics. 1-4, 5-9, and 10-11).

Readers are encouraged to refer to the accompanying video [15] associated with Fig. 6 as it very intuitively demonstrates the effects of the closed-loop control and its superiority compared to the open loop version.

Whereas the closed-loop manipulations (Figs. 6(b-c)) are not perfect in the sense that no errors occur during the ball swapping, the closed-loop control scheme, however, better exploits the underlying UKR representation. It realises a hand motion which is adapted to the current ball configuration and better reacts to unforeseen disturbances during the manipulation. One interesting observation is that the repeated "trying" of the robot to accomplish the sub-goal of bringing one ball in a specific position gives the impression that the robot has a kind of awareness of the current situation yielding a very natural looking manipulation motion.

VI. Conclusion

In this paper, we presented a new closed-loop controller scheme which operates on the predefined clear structure of Structured UKR manifolds - a modified version of Unsupervised Kernel Regression which we have shown in previous work to be well suited to representing human motion data.

The new control scheme extends the existing framework of Structured UKR manifolds that provides means for representing, actuating, and recognising motions. The new mechanism now allows for considering partial observations as sensory input in order to perform a closed-loop control. Using this controller scheme, we have successfully implemented the ball swapping task on a real 16 DOF robot hand, namely the Gifu III Hand, using the positions of the balls as sensory feedback.

In addition to the proof of the presented concept on a real robot, we could also show that our new control is able to better exploit the underlying motion representation in order to realise a more sophisticated manipulation which adequately adapts to the current situation.

In future work, we plan to combine the closed-loop scheme presented in this paper with a more complex manifold providing more than one latent dimension and thus a broader range of different *versions* of the same manipulation for different motion parameters (e.g. ball radius).

Since the closed-loop control is based on the unified latent structure of Structured UKR Manifolds, we expect the control scheme to be applicable to all applications that can be represented in such manifolds. The control scheme then can be directly used in the presented way, only adjusting the control parameters k and Δ defining the length and durations of the feed forward steps of the control.

ACKNOWLEDGEMENT The authors would like to acknowledge support from the German Collaborative Research Centre "SFB 673 - Alignment in Communication" granted by the DFG, the Japanese National Institute of Information and Communications Technology (NICT), and the German Center of Excellence 277 "Cognitive Interaction Technology" (CITEC). Further on, the authors would like to thank Ales Ude for providing the colour blob tracking software.

References

[1] C. G. Atkeson and S. Schaal. Memory-based neural networks for robot learning. *Neurocomputing*, 9(3):243 – 269, 1995.

[2] D. C. Bentivegna, C. G. Atkeson, and G. Cheng. Learning tasks from observation and practice. *Robotics and Autonomous Systems*, 47(2-3):163 – 169, 2004. Robot Learning from Demonstration.

[3] C. Breazeal and B. Scassellati. Robots that imitate humans. *Trends in Cognitive Sciences*, 6(11):481 – 487, 2002.

[4] Y. Demiris and G. Hayes. Imitation as a dual-route process featuring predictive and learning components: a biologically-plausible computational model. In *Imitation in Animals and Artifacts*. MIT Press, 2002.

[5] A. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In *Advances in Neural Information Processing Systems*, pages 1523–1530. MIT Press, Cambridge, 2003.

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

[7] P. Meinicke, S. Klanke, R. Memisevic, and H. Ritter. Principal Surfaces from Unsupervised Kernel Regression. *Trans. on PAMI*, 27(9):1379 – 1391, 2005.

[8] T. Mouri, H. Kawasaki, K. Yoshikawa, J. Takai, and S. Ito. Anthropomorphic Robot Hand: Gifu Hand III. In *Proc. ICCAS*, pages 1288 – 1293, 2002.

[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] S. Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233 – 242, 1999.

[12] S. Schaal, A. Ijspeert, and A. Billard. Computational approaches to motor learning by imitation. *Philosophical Transaction of The Royal Society of London: Ser.B, Biological Sci.*, 358(1431):537–547, 2003.

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

[14] J. Steffen, S. Klanke, S. Vijayakumar, and H. Ritter. Realising Dextrous Manipulation with Structured Manifolds using Unsupervised Kernel Regression with Structural Hints. In *ICRA 2009 Workshop: Approaches to Sensorimotor Learning on Humanoid Robots*, 2009.

[15] J. Steffen, E. Oztop, and H. Ritter. IROS 2010: accompanying video for this paper. www.techfak.uni-bielefeld.de/∼jsteffen/iros2010/.

[16] J. Steffen, M. Pardowitz, and H. Ritter. Using Struct. UKR Manifolds for Motion Classification and Segmentation. In *Proc. IROS*, 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] G. S. Watson. Smooth Regression Analysis. *Sankhya, Ser.A*, 26, 1964.