Towards Dextrous Manipulation Using Manipulation Manifolds

Jan Steffen, Robert Haschke and Helge Ritter
Neuroinformatics Group, Faculty of Technology, University of Bielefeld, Germany
{jsteffen,rhaschke,helge}@techfak.uni-bielefeld.de

***Abstract*— In dextrous manipulation, the implementation of manipulation movements still is a complex and intricate undertaking. Often, a lot of object physics and modelling effort has to be incorporated into a controller working only for a very restricted task specification and performing quite artificially looking movements. In this paper, we present the first steps towards a representation of manipulation movements recorded from human demonstration which facilitates later application and promotes natural motion. We use manifolds of hand postures embedded in the finger joint angle space which are constructed such that manipulation parameters including the advance in time are represented by distinct manifold dimensions. This allows for purposive navigation within such manifolds. We present the manifold construction using the Unsupervised Kernel Regression (UKR) and the way of applying it for manipulation in the example of turning a bottle cap in a physics-based simulation.**

I. INTRODUCTION

During the last decades, researchers and engineers have made huge advances in constructing and building anthropomorphic robot hands which have become more and more sophisticated as one can see for example in the Salisbury Hand [9], the Utah/MIT Hand [6], the DLR II Hand [1] and the Shadow Dextrous Hand [18]. Together with these developments, researchers are facing the question of how to dexterously control such complex robots with up to 20 degrees of freedom in up to five fingers and a wrist.

It quickly became clear that implementing fixed grasp and manipulation programs does not lead to satisfying results as it is very time consuming on the one hand and not robust against or generalisable to differences in the grasping or manipulation situation. Thus, several sophisticated approaches have been presented to realise more robustness and generalisability. Michelman and Allen [11] implemented 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. Fuentes and Nelson [3] learn a mapping from *perceptual goals* – consisting of targeted object position/orientation and applied finger forces – onto robot commands realising these goals using an evolution strategy. Afterwards, manipulation can be performed by defining the task-specific perceptual goal and applying the learned mapping. Han et al. [4] propose a purely contact wrench analysis approach. They use a planner to generate a path in the space of *feasible configurations of the manipulation system* respecting hand/object constraints. A controller then incorporates sensor readings and system kinematics and statics to properly actuate the planned path. Platt et al. [13] address dextrous manipulation by sequencing concurrent combinations of hierarchical organised closed-loop controllers each derived from potential functions and realising force-related objectives. By dint of operating subordinated controllers in the nullspace of superiors, higher-level conditions like wrench closure can be prioritised and thus sustained.

To a certain extent, all these approaches require the manual design of (lower-level) controllers from scratch. In [15], Schaal argues that learning without incorporating prior knowledge is a mostly artificial approach rarely taken by humans and analyses the benefit of learning from demonstration. He applies reinforcement learning on balancing a pole with an anthropomorphic robot arm to find an optimal policy and solves the problem based on data from a 30 second demonstration. Nevertheless, he concludes from his experiments that not every learning problem can profit from prior knowledge in the same the way. Pollard and Hodgins [14] presented a different approach to incorporating human demonstration. They adapt quasistatic manipulation tasks to new friction conditions and untrained objects of known geometry by realising contacts and contact trajectories similar to former demonstration. The manipulation is planned such that the manipulator/object contacts combined with the extreme ground friction cones always produce force closure grasps arguing that intermediate configurations then are force closure too.

Although these approaches all realise robust dextrous manipulations to a certain degree, their implementations require considerable effort in problem modelling on the level of task definition and object characteristics.

In this paper, we present the first steps towards a new approach in dexterous manipulation with anthropomorphic dextrous robot hands using *manifolds of manipulation movements*. The main idea is to construct manifolds embedded in the finger joint angle space which represent the subspace of hand postures associated with a specific manipulation movement. Instead of learning these representations in a purely unsupervised manner yielding unpredictable manifolds, we

Final version in: Proc. Int'l. Conf. on Intelligent Robots and Systems (IROS), 2008

want to construct them such that specific movement parameters – and especially the advance in time – are explicitly represented by specific and distinct manifold dimensions. For our initial experiments, we focus on the manipulation movement of turning a bottle cap incorporating the advance in time and the cap radius as manipulation parameters. The training data consist of a set of vectors of finger joint angles generated in a physics-based simulation using a data glove as input device.

The paper is organised as follows: In Section II, we review the manifold representation that we chose for our experiments, namely the *Unsupervised Kernel Regression*. Section III will address the training data retrieval. In Section IV we describe the construction of the manifolds including results followed by an application example in Section VI. Finally, we end up with a conclusion and an outlook on future work in Section VII.

II. UNSUPERVISED KERNEL REGRESSION

Unsupervised Kernel Regression (UKR) is a recent approach for learning continuous manifold representations. It has been introduced as an unsupervised formulation of the Nadaraya-Watson kernel regression estimator by Meinecke, Klanke et al. in [10] and further developed by Klanke in [8], [7]. It uses the Nadaraya-Watson estimator [12], [19] to find a lower-dimensional (latent) representation of the original data and a smooth mapping from that latent space back to the original data space at the same time. The original Nadaraya-Watson estimator defines a mapping

$$\vec{y} = f(\vec{x}) = \sum_i \vec{y_i} \frac{K(\vec{x} - \vec{x_i})}{\sum_j K(\vec{x} - \vec{x_j})} \tag{1}$$

which realises a smooth, continuous generalisation of the functional relationship between $\vec{x}$ and $\vec{y}$ described by given data samples $(\vec{x}_i; \vec{y}_i)$. Here, $K(\cdot)$ is a density kernel. UKR now treats Eq. (1) as a mapping from a lower dimensional latent space $\mathcal{X}$ to the original data space $\mathcal{Y}$ which is described by a set of observed data $Y = \{\vec{y}_i\}$. Here, the corresponding $X = \{\vec{x}_i\}$ play the role of *latent parameters* of the regression function:

$$\vec{y} = f(\vec{x}; X) = \sum_i \vec{y_i} \frac{K(\vec{x} - \vec{x_i})}{\sum_j K(\vec{x} - \vec{x_j})}. \tag{2}$$

The training of the UKR manifold thus can be realised by gradient-based minimisation of the reconstruction error

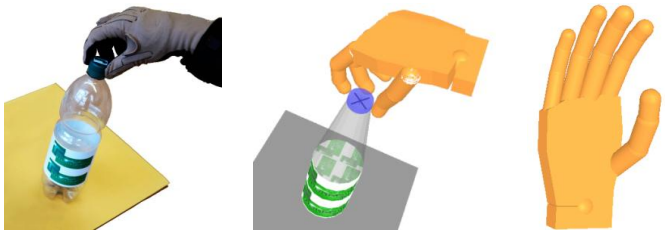
$$R(X) = \frac{1}{N} \sum_m \| \vec{y}_m - f(\vec{x}_m; X) \|^2 . \tag{3}$$

As special benefit, the UKR can very efficiently perform Leave-K-Out Cross-Validation by using a modified $f(\vec{x}_m; X)$ in Eq. (3):

$$f_m(\vec{x}; X) = \sum_{i \notin \mathcal{N}_m} \vec{y_i} \frac{K(\vec{x} - \vec{x_i})}{\sum_{j \notin \mathcal{N}_m} K(\vec{x} - \vec{x_j})} \tag{4}$$

where $\mathcal{N}_m$ is the set of neighbours excluded for the reconstruction of $\vec{y}_m$.

For further details, please refer to [10], [8], [7].



a) b) c)

Fig. 1. a) Hand-mounted CyberGlove II during cap turning movement. b) Simulated hand model with bottle. The hand joints are controlled by CyberGlove II. c) Close-up of the hand model.

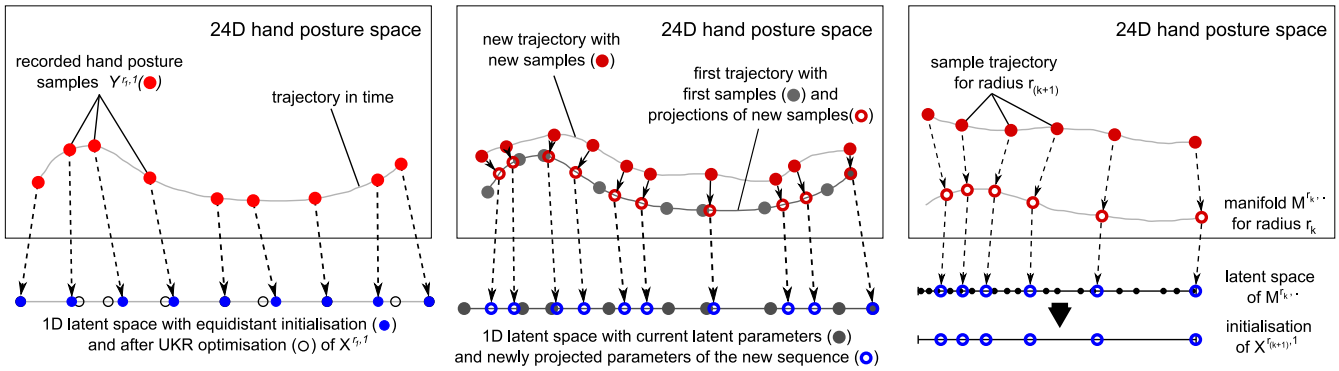
III. DATA RETRIEVAL

One bottleneck of manifold learning algorithms often is the need of a rather high amount of training data which is especially a problem when the generation of relevant data can not be performed in a direct manner.

In the case of dextrous grasping and manipulation with a robot hand, we are facing the problem that we need to generate hand postures or even trajectories of hand postures corresponding to a specific manipulation movement not yet implemented by any algorithm. As per-joint control or directly physically moving the fingers by hand does not yield natural trajectories, it is a quite obvious approach to use our own hands as perfect archetype to produce relevant data.

In general, there exist two ways to measure human hand postures in form of vectors of finger joint angles. One method is to use a vision system with integrated marker tracking. Applying it to hand posture retrieval during manipulation tasks, one has to cope with alternating marker occlusions due to the finger movements. Using numerous cameras watching the scene, occlusions can be minimised and precise joint information can be recorded. Another way of retrieving hand posture data is given by hand-mounted data gloves. In our group, we utilise an Immersion CyberGlove II [5] (cf. Fig. 1a) with 22 bend sensors for the different joints. As the sensors do not yield as precise data as possible with a sophisticated vision system, we map the sensor values onto a simulated hand model (Fig. 1b,c). The data generation itself then takes place in a simulated manipulation scene (Fig. 1b) incorporating the joint angle corrections provided by the collision detection module of the physics-based simulation toolkit [2] and the more general hand posture corrections performed by the user induced by visual feedback. The data retrieval conducted in this way exploiting interactive user control and dynamics and collision detection of the physics-based simulation yield rather realistic training data.

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 radius, we produced five sequences each of about 30 to 45 hand postures. Each hand posture consists of a vector of the 24 joint angles of the simulated hand model which can be mapped onto our anthropomorphic robotic *Shadow Dextrous Hand* [18].



(a) Equidistant initialisation of the latent parameters (blue) respecting the order of the hand posture samples in the movement trajectory (red). The black circles in latent space depict possible displacements of the latent parameters due to the UKR optimisation.

(b) Procedure of incrementally adding new sequences by projecting new samples into the latent space of the previously trained manifold. The new manifold consisting of the combination of old and new data samples and latent parameters gets retrained afterwards.

(c) Synchronising different radii r_k and $r_{(k+1)}$ by projecting new samples corresponding to $r_{(k+1)}$ into the latent space $X^{r_k,\cdot}$ corresponding to r_k. The projections then are used as initialisation of $X^{r_{(k+1)},1}$ for the subsequent training of $\mathcal{M}^{r_{(k+1)},1}$.

Fig. 2. Schematic description of different steps in the manifold construction process.

IV. MANIFOLD CONSTRUCTION

The problem using unsupervised manifold learning methods often is that there are usually only limited means of incorporating partial task knowledge or of controlling the way of how "the manifold is laid into the underlying data" respectively. On the other side, we can not provide completely specified training data that enables us to perform a purely supervised approach. Thus, our goal was to develop a mechanism that renders us possible to learn a manifold representation of the data in a partly unsupervised manner and additionally imprint specific meanings into the directions within the manifold. In terms of our scenario, the representation shall provide that every point on the manifold corresponds to one moment in time of a motion trajectory and additionally that the directions within the manifold, and especially its single dimensions, inhere the meaning of one specific motion parameter. Thus, in the example of turning a bottle cap, the goal is to realise a manifold in which one dimension controls the progress in time of the turning movement and another dimension specifies the radius of the cap. Then, performing the movement reduces to modifying the time component of the latent parameter.

A simple but - as presented in the following - effective approach to achieving this is to construct the final manifold out of several sub-manifolds each realising a manipulation movement of one motion parameter set.

In this first approach, we incorporate two parameters - the progress in time of the movement and the radius of the cap. The construction of the final manifold is performed iteratively starting with training sequences corresponding to the minimal cap radius successively increasing the radius of the subsequent sequences.

For the first sequence of hand postures $Y^{r_1,1} = \{\vec{y}_i^{\,r_1,1}\}$ corresponding to the minimal cap radius r_1 , we manually distribute the latent parameters of a 1D-UKR manifold equidistantly in a predefined interval of the latent space according to the intra-sequence order of the hand postures and perform the UKR training to optimise the latent parameters $X^{r_1,1} = \{\vec{x}_i^{\,r_1,1}\}$ afterwards (cf. Fig.2a). We denote the resulting UKR manifold as $\mathcal{M}^{r_1,1}$. The incorporation of the second sequence (representing another example of a movement for the same radius) is performed in an iterative manner: the hand posture vectors $Y^{r_1,2}$ of this sequence are projected pointwise into the latent space of the previously trained 1D-manifold $\mathcal{M}^{r_1,1}$ resulting in $X^{r_1,2}$ (cf. Fig.2b). By dint of this projection, we approximate a synchronisation of the temporal advance of the two movements. In the next step, we combine those data to a new UKR manifold $\mathcal{M}^{r_1,\{1,2\}}$ with observed data $Y^{r_1,\{1,2\}} = Y^{r_1,1} \cup Y^{r_1,2}$ and latent parameters $X^{r_1,\{1,2\}} = X^{r_1,1} \cup X^{r_1,2}$. A subsequent UKR training of $\mathcal{M}^{r_1,\{1,2\}}$ optimises the latent parameters subject to the whole combined data set.

By performing this procedure for all sequences of hand postures corresponding to similar cap turning movements for one specific cap radius, a 1D–UKR is trained representing a generalised radius-specific movement. Thus, by applying this method to all sets of radius-specific sequences, we generate one 1D–UKR per radius. To promote the synchronisation of the temporal advances also between the different radius-specific manifolds, we only initialise the first manifold $\mathcal{M}^{r_1,1}$ with equidistant latent parameters as describes above. When proceeding with sequences $Y^{r_{(k+1)},1}$ of a new radius $r_{(k+1)}$, we project the sequence onto the previously trained manifold $\mathcal{M}^{r_k,\{1,..,n\}}$ (as with sequences for the same radius) and utilise the resulting latent parameters as initialisation $X^{r_{(k+1)},1}$ of the new manifold $\mathcal{M}^{r_{(k+1)},1}$ instead of combining them to a manifold $\mathcal{M}^{r_k,\{1,..,n+1\}}$ (cf. Fig.2c). The training then continues as described above. Notice that the first sequence used to initially train the manifold for the first radius plays a special role and determines the relevant

subspace in the hand posture space. Therefore, it is important that this sequence represents a complete movement rather than only a specific phase.

The subsequent combination of all 1D-manifolds $\mathcal{M}^{r_i,\cdot}$ to one 2D-manifold $\mathcal{M}$ representing the complete cap turning movements for all radii r_i covered by the training data then is performed manually and without the usage of the UKR training. $\mathcal{M}$ then consists of all incorporated training data $\{Y^{r_i,j}\}_{i,j}$ together with the corresponding latent parameters $\{X^{r_i,j}\}_{i,j}$ and represents the whole manipulation movement described by the training data. Therefore, we denote it as *Manipulation Manifold*. The extension to two dimensions is realised by expanding each latent parameter $\vec{x}_i$ by a second dimension denoting the appropriate radius corresponding to the associated training sequence.

V. CONSTRUCTION RESULTS

We applied the method described in Section IV to all recorded training sequences (cf. Section III) starting with the set of sequences corresponding to the minimal radius $r_1 = 1.5cm$ and successively incorporating sequences of greater radii. After having trained one 1D-manifold for each of the training radii $r = 1.5cm,\ 2.0cm,\ 2.5cm,\ 3.0cm$ and $3.5cm$ in the described synchronised manner, we added the corresponding radius values as second dimension to the latent parameters (by manually extending each latent parameter by an extra dimension). The distribution of the latent parameters in the new latent space is depicted in Fig. 3. As constructed, the horizontal (first) dimension represents the temporal advance within the cap turning movement and the vertical (second) dimension denotes the associated cap radius. As no further UKR training is performed, the latent parameters only lie on the previously set discrete radius values. To get a more distinct impression of the movements represented by the manifold and its generalisation abilities, Fig. 4 depicts a matrix of hand postures corresponding to the positions in a regular grid covering the latent space of the manifold. Again, the temporal advance is depicted in the horizontal and the different radii in the vertical direction. To facilitate the comparison, a bottle cap with radius ($r = 1.5cm$) is depicted in each sub-figure. As shown in Fig. 3, only the radii $r = 1.5cm,\ 2.0cm,\ 2.5cm,\ 3.0cm$ and $3.5cm$ are directly supported by training data. Thus, the depicted intermediate radii $r = 1.75cm,\ 2.25cm,\ 2.75cm$ and $3.25cm$ in Fig. 4 visualise the generalisation ability of the constructed manifold to new cap radii. The corresponding movements for the intermediate radii are clearly similar to the training sequences. Secondly, it illustrates the effect of the temporal synchronisation between the different 1D-manifolds by projecting new sequences into the latent space of the previously trained manifolds before newly training as described in Section IV. The most distinct picture of this synchronisation can be seen in columns $3-5$ ($t = 20\% - 40\%$) where the fingers are shown in the moments (virtually) contacting the cap. Additionally, those columns give an impression of the smoothness in the $2nd$ manifold dimension. Remark the smooth finger opening with increasing cap radius in the column direction. In the row direction, all rows depict smooth transitions from left to right indicating a smooth manifold also in the row direction.

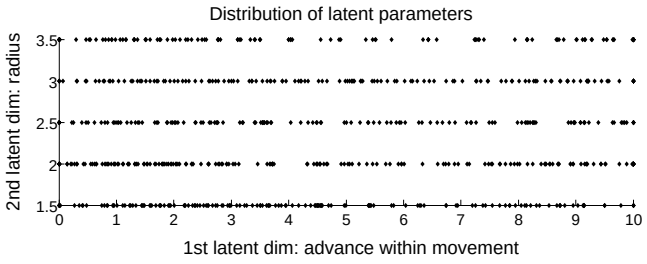


Fig. 3. Distribution of the latent parameters of $\mathcal{M}$. The $1st$ dimension represents the advance within the movement. As constructed, the $2nd$ dimension represents the defined cap radii.

One effect of the presented training method is not directly obvious in Fig. 4. When reaching the manifold border in the temporal dimension while performing the turning movement, the temporal position has to be reset to the beginning (go back from 100% to 0%) to restart the turning movement. As by now, there is no regulation for border synchronisation incorporated in the training, the 100%- and 0%-postures usually significantly differ from each other yielding an abrupt non-smooth hand movement when jumping back to the 0%-posture. As the beginning and the end of the movement are the phases where the fingers are the farthest away from the cap, the motion artifact resulting from the missing border synchronisation does not effect the cap turning and thus is not of particular relevance for the success of the cap turning manipulation. Nevertheless, we will address this issue in future work to optimise the natural impression of the manipulation.

VI. APPLICATION

The basic idea of the manifold construction in the way we presented it in the preceding sections was to create a robust representation of manipulation movements that facilitates to perform the associated manipulation in later applications. Thus, only simple steps need to be taken to reproduce the trained movement in absence of a cap and only few more steps to really perform the associated manipulation of a cap.

To reproduce the movement, we just have to manually select a desired cap radius (in the presented way of constructing the manifold, the values of the second manifold dimension correspond to cap radii in centimetres) and navigating through the manifold by increasing the temporal dimension value while keeping the radius value fixed. When reaching the manifold border, we reset the temporal dimension value to 0% and continue.

Performing a manipulation can be achieved in a similar manner. Instead of manually selecting a cap radius, we need to recognise the real radius of the presented bottle. In order to achieve this, we grasp the bottle cap by performing our previously developed *Experience-based Grasping* algorithm [17], [16] using our *Manipulation Manifold* from Section V which

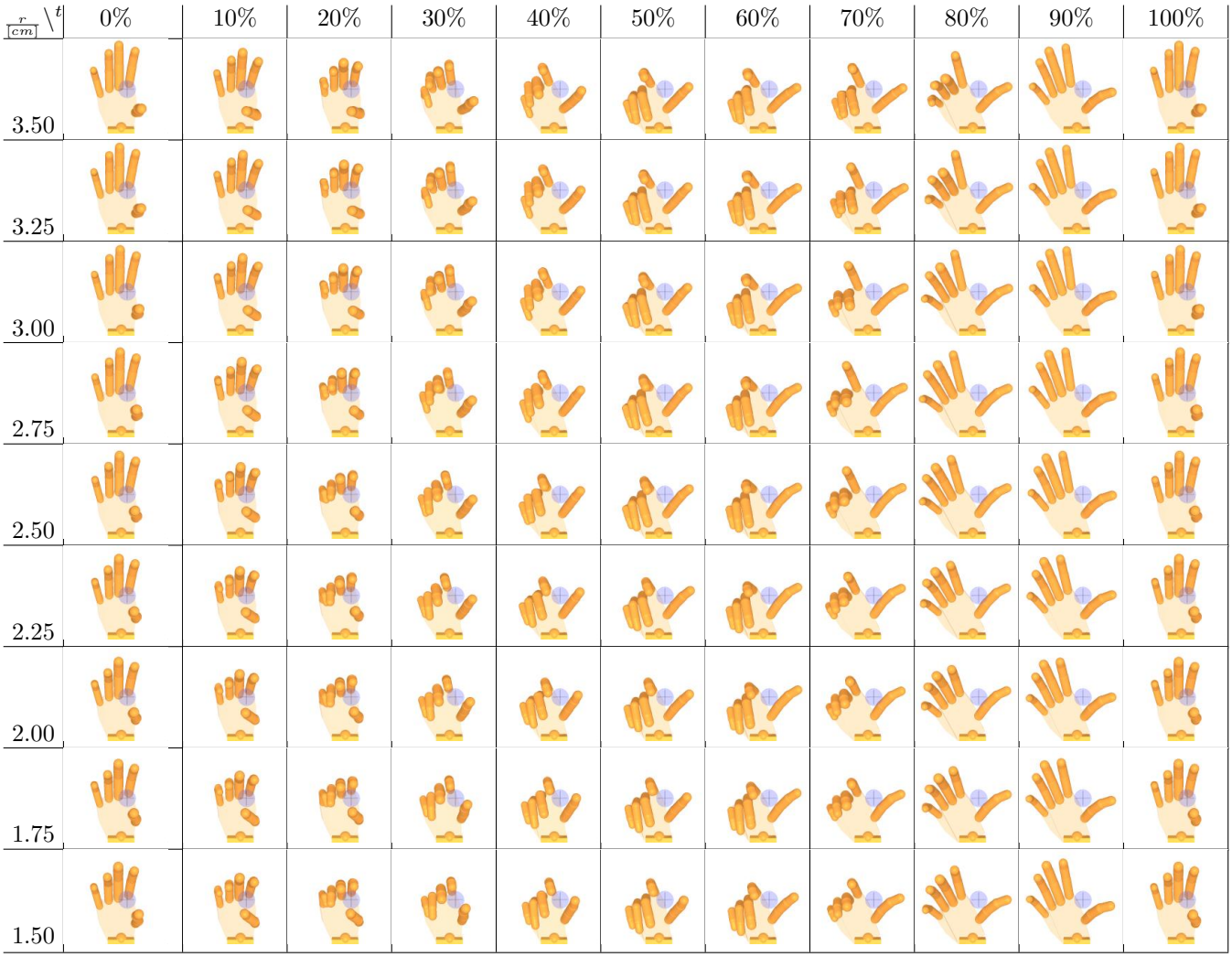


Fig. 4. Generalisation results of the UKR training/construction. The depicted hand postures correspond to the positions of a regular grid covering the latent space of the manifold. The horizontal direction is associated with the $1st$ (temporal) latent dimension and the vertical direction with the $2nd$ (radius) latent dimension. The size of the depicted bottle cap in all pictures is the same ($r = 1.5$) as a comparison aid. The radii $r = 1.5,\ 2.0,\ 2.5,\ 3.0$ and 3.5 correspond to radii covered by the training data, the radii $r = 1.75,\ 2.25,\ 2.75$ and 3.25 demonstrate the generalisation capability of the approach. The most distinct picture of the differences between the radii can be seen in the columns $t = 20\% \ldots 60\%$ where the fingertips are closest to the cap. A more sophisticated impression of the smoothness of the movement and the manifold respectively can be received by means of a movie available under `http://www.techfak.uni-bielefeld.de/~jsteffen/mov/SteffenHaschkeRitter_TurnCap_IROS2008.avi`.

– metaphorically speaking – pulls the current hand posture onto the manifold. As this approach originally has been designed to work with *Grasp Manifolds* consisting of hand postures all representing grasps, the originally presented algorithm does not fit optimally to *Manipulation Manifolds* but works sufficiently good for our first experiments. For future work, an adaptation to the new context will be realised. Already, the grasping algorithm results in an adequate grasp posture which can be projected into the latent space of the manipulation manifold. This yields a latent position which defines the radius to be used during the subsequent manipulation together with the current temporal position within the movement. Again, by increasing the temporal value while keeping the radius fixed, the manipulation movement is performed. By dint of the previous projection of an actuated grasp posture, the movement is adjusted to the presented cap radius and thus fits to the current manipulation context. By now, no further explicit contact conditions are incorporated. Figure 5 depicts two sequences of intermediate hand postures during a bottle cap manipulation for two different radii (a) $r = 2cm$ and (b) $r = 2.75cm$ using our algorithm. The first rows each depict one whole movement cycle ($0\% - 100\%$) whereas the second rows each show in detail the period of object manipulation in which the fingers have contacts to the bottle cap. Both manipulations are successful in terms of rotating the cap but as depicted in the second rows of Fig. 5a) and b), the period of the very manipulation where the cap is actively rotated is much longer for $r = 2cm$ (notice the different time scales in both rows!). In the first case, the manipulation takes place between 14% and 34% with at least two opposing contact fingers up to 42% where at least one finger contact remains. In the second case, this period only lasts from 36% to 46% with opposing contact fingers up to 50% with one finger contact.

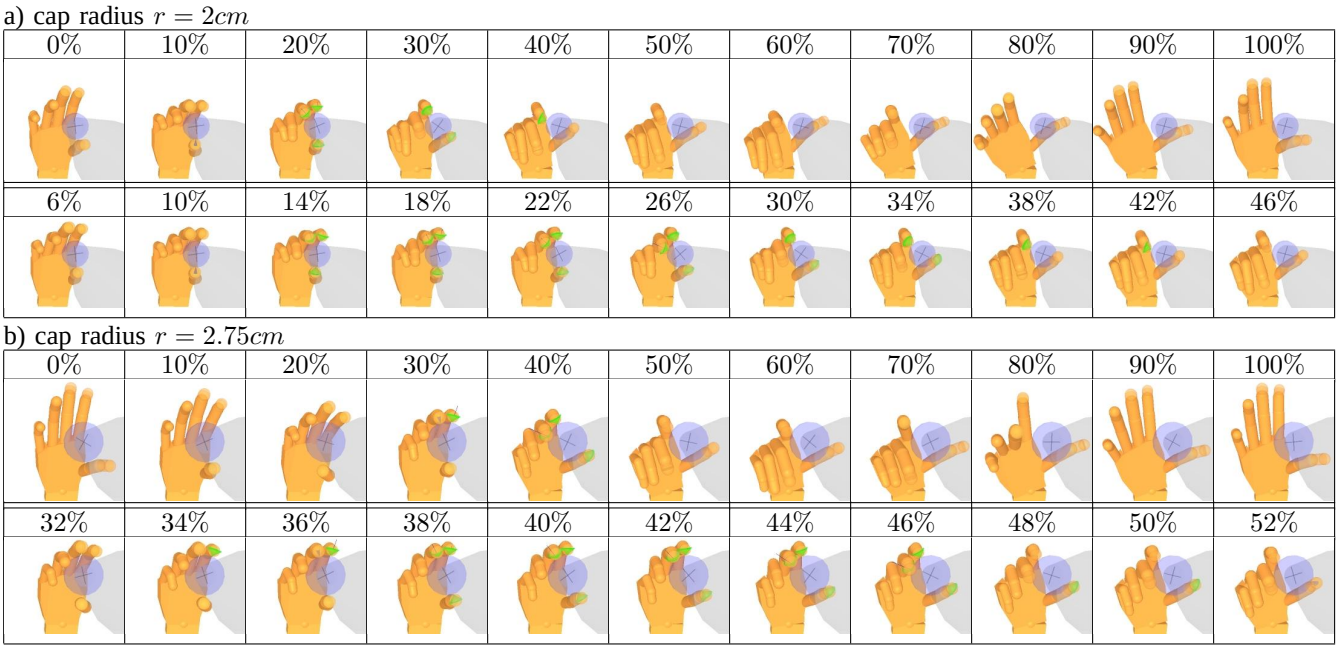


Fig. 5. Application results: hand posture sequences generated by applying the presented *Manipulation Manifolds*. Beforehand, the unmodified *Experience-based Grasp Control* algorithm [16] has been applied to find an initial grasp posture. The sequences depict intermediate hand postures during the turning movement for radii a) $r = 2cm$ and b) $r = 2.75cm$. In each case, the first rows visualise one complete movement cycle ($0\% - 100\%$) whereas the second rows each focus on the contact period. The green cones at the fingertips visualise the contact friction cones. Notice the different time scales in the second rows expressing different contact time proportions within the movements for the different manipulations.

VII. CONCLUSION AND FUTURE WORK

We presented the first steps towards a new approach in dextrous manipulation that uses sequences of hand postures recorded from human demonstration to learn and construct *Manipulation Manifolds* in which distinct dimensions represent distinct parameters of the associated manipulation. With the example of turning a bottle cap, we provided a proof of concept by incorporating two parameters – the cap radius and the temporal advance within the movement – in a manifold and applying it to perform the cap turning to two different bottle caps in an application. From our experiments, we conclude several subjects and aims for our future work. The most important for us will be to change the learning and construction mechanism such that it works in a more unsupervised fashion with the goal of completely replacing the manual construction part by an unsupervised learning. For this, we have several ideas in mind of how to modify the UKR learning to better fit to the problem of chronologically ordered data sequences. Other important objectives are to explicitly incorporate contact conditions and to remove artifacts due to the missing border synchronisation. While following our goals, we want to sustain our main principle of this work of constructing manifolds in which distinct dimensions have imprinted distinct and specific meanings like the radius of a bottle cap and the temporal advance within the manipulation movement.

REFERENCES

[1] J. Butterfass, M. Grebenstein, H. Liu, and G. Hirzinger. DLR-Hand II: next generation of a dextrous robot hand. In *Proc. ICRA*, 1986.
[2] CM-Labs. Vortex 2.1. www.cm-labs.com/products/vortex, 2005.
[3] O. Fuentes and R. Nelson. Learning Dextrous Manipulation Strategies for Multifingered Robot Hands Using the Evolution Strategy. *Machine Learning*, 31, 1998.
[4] L. Han, Z. Li, J. Trinkle, Z. Qin, and S. Jiang. The planning and control of robot dextrous manipulation. In *Proc. ICRA*, 2000.
[5] Immersion. CyberGlove II Wireless Data Glove User Guide. www.immersion.com/3d/docs/CyberGloveII_UserGuide_aug07v2.pdf.
[6] S. Jacobsen, E. Iversen, D. Knutti, R. Johnson, and K. Biggers. Design of the Utah/M.I.T. Dextrous Hand. In *Proc. ICRA*, 1986.
[7] S. Klanke. *Learning Manifolds with the Parametrized Self-Organizing Map and Unsupervised Kernel Regression*. PhD thesis, Bielefeld University, 2007.
[8] S. Klanke and H. Ritter. A Leave-K-Out Cross-Validation Scheme for Unsupervised Kernel Regression. In *ICANN 2006*, volume 4132 of *LCNS*. Springer, 2006.
[9] M. Mason and J. K. Salisbury. *Robot Hands and the Mechanics of Manipulation*. MIT Press, 1985.
[10] 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.
[11] P. Michelman and P. Allen. Forming Complex Dextrous Manipulations from Task Primitives. In *Proc. ICRA*, 1994.
[12] E. A. Nadaraya. On Estimating Regression. *Theory of Probability and Its Application, Vol.9*, 1964.
[13] R. Platt Jr., A. Fagg, and R. Grupen. Manipulation gaits: sequences of grasp control tasks. In *Proc. ICRA*, 2004.
[14] N. Pollard and J. Hodgins. Generalizing Demonstrated Manipulation Tasks. In *Proc. WAFR*, 2002.
[15] S. Schaal. Learning from Demonstration. In *Advances in Neural Information Processing Systems*, volume 9. MIT Press, 1997.
[16] J. Steffen, R. Haschke, and H. Ritter. Experience-based and Tactile-driven Dynamic Grasp Control. In *Proc. IROS*, 2007.
[17] J. Steffen, R. Haschke, and H. Ritter. SOM-based experience representation for Dextrous Grasping. In *Proc. WSOM*, 2007.
[18] The Shadow Robot Company. Shadow Dexterous Hand. www.shadowrobot.com/hand, 2002.
[19] G. S. Watson. Smooth Regression Analysis. *Sankhya, Series A*, 26, 1964.
[20] H. Zhang, K. Tanie, and H. Maekawa. Dextrous manipulation planning by grasp transformation. In *Proc. ICRA*, 1996.