2D Subspaces for User-Driven Robot Grasping

Aggeliki Tsoli Odest Chadwicke Jenkins

Abstract— **Human control of high degree-of-freedom robotic systems is often difficult due to the overwhelming number of variables that need to be specified. Instead, we propose the use of sparse control subspaces embedded within the pose space of a robotic system. Using captured human motion for training, we address this sparse control problem by uncovering 2D subspaces that allow cursor control, or eventually decoding of neural activity, to drive a robotic hand. Considering the problems in previous work related to noise in pose graph construction and motion capture data, we introduce a method for denoising neighborhood graphs for embedding hand motion into 2D spaces. We present results demonstrating our approach to interactive sparse control for successful power grasping and precision grasping using a 13 DOF robot hand.**

I. INTRODUCTION

Developing human interfaces for controlling complex robotic systems, such as humanoid robots and mechanical prosthetic arms, presents an underdetermined problem. Specifically, the amount of information a human can reasonably specify within a sufficiently small update interval is often far less than a robot's degrees-of-freedom (DOFs). Consequently, basic control tasks for humans, such as reaching and grasping, are often onerous for human teleoperators of robot systems, requiring either a heavy cognitive burden or overly slow execution. Such teleoperation problems persist even for able-bodied human teleoperators given state-of-the-art sensing and actuation platforms.

The problem of teleoperation becomes magnified for applications to biorobotics, particularly in relation to prosthetic and assistive devices by users with lost physical functionality. In such applications, feasible sensing technologies, such as electroencephalogram (EEG) [1], electromyography (EMG) [2, 3, 4], and cortical neural implants [5, 6], provide a very limited channel for user input due to the sparsity and noise of the sensed signals. Specifically for **neural decoding**, efforts to decode this user neural activity into control signals have demonstrated success limited to 2-3 DOFs with bandwidth around 15 bits/sec. With such limited bandwidth, control applications have focused on low-DOF systems, such as 2D cursor control [7], planar mobile robots [1], and discrete control of 4 DOF robot arms [8, 9]. Additionally, Bitzer and van der Smagt [4] have performed high-DOF robot hand control by reducing the DOFs to a discrete set of poses that can be indexed through kernel-based classification.

Robotic systems geared for general functionality or human anthropomorphism will have significantly more than 2-3 DOFs, posing a **sparse control** problem. For instance, a prosthetic arm and hand could have around 30 DOF. While this mismatch in pose and control dimensionality is problematic, it is clear that the space of valid human arm/hand poses does not fully span the space of DOFs. It is likely that plausible hand configurations exist in a significantly lower dimensional subspace arising from biomechanical redundancy and statistical studies in human movement [10, 11, 12]. In general, uncovering intrinsic dimensionality of this subspace is crucial for bridging the divide between decoded user input and the production of robot control commands.

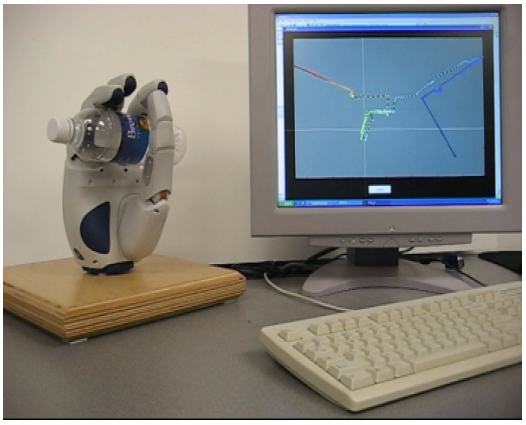


Fig. 1. **Snapshot of our sparse control system driving a DLR/HIT robot hand to grasp an object from a user's 2D cursor control.**

In addressing the sparse control problem, our objective is to discover 2D subspaces of hand poses suitable for interactive control of a high-DOF robot hand, with the longer-term goal of sparse control with 2D cursor-based neural decoding systems. We posit viable sparse control subspaces should be **scalable** (not specific to certain types of motion), **consistent** (two dissimilar poses are not proximal/close in the subspace), and **continuity-preserving** (poses near in sequence should remain proximal in the subspace). To uncover control subspaces, we follow a **data-driven** approach to this problem through the application of manifold learning (i.e., dimension reduction) techniques to hand motion data captured from real human subjects.

Our previous work [13] identified **noise in both the motion capture and pose graph construction procedures** as major limiting factors in uncovering subspaces for sparse control. In this paper, we address both of these limitations through 1) graph denoising using probabilistic belief propagation [14] and 2) more careful selection of motion capture data. We present results from embedding power grasps, precision grasps, and tapping motions into sparse control spaces and their use for interactive control of the DLR robot hand (13 DOFs).

II. THE SPARSE CONTROL PROBLEM

The essence of the sparse control problem is to estimate a control mapping $f : X \rightarrow Y$ that maps coordinates in a 2-dimensional control space $x \in \Re^2$ into the space of hand poses $y \in \Re^d$, where d is the number of DOFs expressing hand

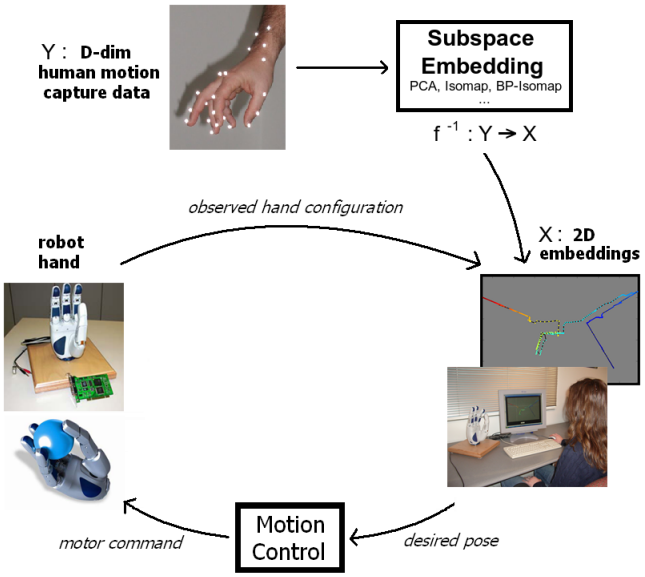


Fig. 2. **Diagram for hand control by the user using human hand motion capture data for training**

pose. The estimation of the mapping f is founded upon the assumption that the space of plausible hand poses for desired motion is intrinsically parameterized by a low-dimensional manifold subspace. We assume each hand pose achieved by a human is an example generated within this manifold subspace. It is given that the true manifold subspace of hand poses is likely to have dimensionality greater than two. With an appropriate dimension reduction technique, however, we can preserve as much of the intrinsic variance as possible. As improvements in user interfaces (namely for neural decoding) occur, the dimensionality of the input signal will increase but we will still leverage the same control mapping.

Tangenting briefly, the application of sparse control for interactive control of the DLR/HIT hand is illustrated in Figure 2. Human hand motion data in high-dimensional pose space is given as input. Using manifold learning, the hand pose data is embedded into a 2D space. The embedding space is presented to a human user through a Matlab graphical interface. Every time the user clicks on a point in this space, the 2D input coordinates are translated to a high-dimensional hand configuration, serving as the target joint angles for actuating the robot hand. The user can observe the results of his (or her) action and interactively guide the performance of the robot hand.

We create a control mapping by taking as input a set of training hand poses $y_i \in \Re^d$, embedding this data into control space coordinates $x_i \in \Re^2$, and generalizing to new data. The configuration of points in control space $x_i = f^{-1}(y_i)$ is latent and represents the inverse of the control mapping. Dimension reduction estimates the latent coordinates y such that distances between datapairs satisfy some criteria of similarity. Each dimension reduction method has a different notion of pairwise similarity and, thus, a unique view of the intrinsic structure of the data. Once embedded, the pose-subspace pairs (y_i, x_i) are generalized into a mapping through interpolation [15] to allow for new (out-of-sample) points to be mapped between pose and control spaces.

Discovery of the sparse control mapping is performed using Isomap [16]. We focus on Isomap, but have also explored the use of other dimension reduction techniques (PCA, Hessian LLE, Spatio-temporal Isomap) [13]. Isomap is basically a "geodesic" form of multidimensional scaling (MDS) [17], where shortest-path distances in pose space represent desired Euclidean distances in the control subspace. The geodesic distance between two points y, y' is approximated as:

$$D_{y,y'} = \min_p \sum_i D'(p_i, p_{i+1}) \tag{1}$$

where D' is a sparse pose graph of local distances between nearest neighbors and p is a sequence of points through D' indicating the shortest path between poses y and y'. MDS is performed on the matrix D containing all the geodesic distances to generate subspace coordinates x based on the distance preserving error E (which can be optimized efficiently through eigendecomposition):

$$E = \sqrt{\sum_x \sum_{x'} (\sqrt{(x - x')^2}) - D_{y,y'})^2} \tag{2}$$

A canonical Isomap example is the "Swiss roll" dataset (Figure 5), where input data generated by 2D manifold is contorted into a "roll" in 3D. Given a sufficient density of samples and proper selections of neighborhoods, Isomap is able to flatten this Swiss roll data into its original 2D parameterization, up to an affine transformation.

In practice, however, noise-free nearest neighborhood construction can be difficult and prohibit the application of Isomap to noisy datasets, such as motion capture data. In the Swiss Roll example, the inclusion of a noisy neighborhood edge between points at the start and end of the manifold creates a "short circuit" for shortest path computation. Consequently, the approximated geodesic distance is invalid and the resulting embedding lacks consistency with the manifold's true parameterization.

III. NEIGHBORHOOD DENOISING WITH BELIEF PROPAGATION

In order to produce control mappings when dealing with noisy data, we propose BP-Isomap — an extension to Isomap — using a denoised version of the neighborhood graph of hand poses. A description of BP-Isomap is provided in Algorithm 1. Step 2, neighborhood denoising, is the main differentation from Isomap and the primary focus of this section.

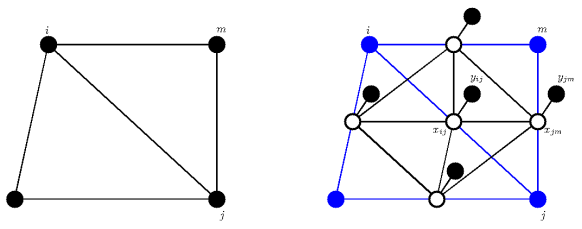
Step 2 associates each edge $\bar{ij}$ in the neighborhood graph with a latent distance $x_{\bar{ij}}$ and an observed distance $y_{\bar{ij}}$. Initially, relationships between neighborhood edges and the variables associated with them are determined through the formulation of a Markov Random Field (MRF) graphical model based on the neighborhood graph D' of step 1. Inference of the hidden variables $x_{\bar{ij}}$ is performed using the Belief Propagation (BP) algorithm as described by Yedidia et. al [14]. Once $x_{\bar{ij}}$

Algorithm 1 BP-Isomap

1. Construct a neighborhood graph D' between hand poses using the k-Nearest Neighbors algorithm.
2. Denoise the neighborhood graph edges.
   - Define a Markov Random Field (MRF) model over the neighborhood graph D'.
   - Run the Belief Propagation (BP) algorithm until convergence to determine the latent distance of the neighborhood edges
     (a) select two neighboring edges at random,
     (b) send a message from one to the other.
   - Remove edges from the neighborhood graph whose latent distance is above a threshold τ (the denoised neighborhood graph $\hat{D}'$ is produced).
3. Calculate the shortest-path (geodesic) distances between all pairs of input points in the denoised neighborhood graph $\hat{D}'$.
4. Embed the shortest-path distances using Multi-Dimensional Scaling (MDS).

has been estimated for all neighboring pairs $(i, j) \in D'$, edges with distances greater than an allowed threshold τ are removed creating the denoised neighborhood graph $\hat{D}'$.

The Markov Random Field (MRF) model over the neighborhood graph of step 1 is defined as shown in Figure 3. The MRF vertices associated with latent variables lie on the edges (shown in blue) of the neighborhood graph. The edges of the MRF connect latent variable vertices that correspond to adjacent neighborhood edges. They also connect latent variable vertices with vertices containing their corresponding observed variables. The observed variable $y_{\bar{ij}}$ of a neighborhood edge $\bar{ij}$ corresponds to its Euclidean distance and is calculated as $y_{\bar{ij}} = D'(y_i, y_j) = ||y_i - y_j||$. Variables y_i, y_j denote the d-dimensional coordinates of input hand poses i,j.



(a) neighborhood graph (b) MRF

Fig. 3. **(a) A neighborhood graph and (b) its corresponding MRF**

The local evidence function $\phi(x_{\bar{ij}}, y_{\bar{ij}})$ of the MRF model inclines the edge distance $x_{\bar{ij}}$ to preserve the observed Euclidean distance $y_{\bar{ij}}$. For that reason, this function weights possible values of $x_{\bar{ij}}$ with a Gaussian distribution centered at $y_{\bar{ij}}$ with variance σ^2:

$$\phi(x_{\bar{ij}}, y_{\bar{ij}}) \sim \mathcal{N}(y_{\bar{ij}}, \sigma^2) \tag{3}$$

The compatibility function $\psi(x_{\bar{ij}}, x_{j\bar{m}})$ of the model outputs a scalar value proportional to the compatibility of the latent distance $x_{\bar{ij}}$ of edge $\bar{ij}$ with the latent distance $x_{j\bar{m}}$ of its adjacent edge $j\bar{m}$. The compatibility of variables $x_{\bar{ij}}$, $x_{j\bar{m}}$ is determined in terms of the relationship between vertices i and m in neighborhood graph D' (note that vertices i,m are both adjacent to vertex j). This function considers two cases for the relationship between neighborhood graph vertices i and m: 1) neighborhood graph vertices i and m are adjacent or have at least one common neighbor $k \neq j$, or 2) neighborhood graph vertices i and m are neither adjacent nor have common neighbors. In both cases, we are concerned with weighting the compatibility of $x_{\bar{ij}}$ and $x_{j\bar{m}}$ by an imaginary triangle (Figure 4) that neighborhood graph vertices i, j, m form and more specifically via the distance d_{mi} of the triangle edge connecting neighborhood graph vertices i,m:

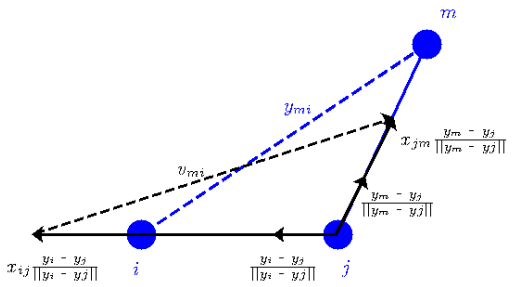


Fig. 4. **The triangle of vertices i, j,m used in the compatibility function $\psi(x_{\bar{ij}}, x_{j\bar{m}})$ of an MRF**

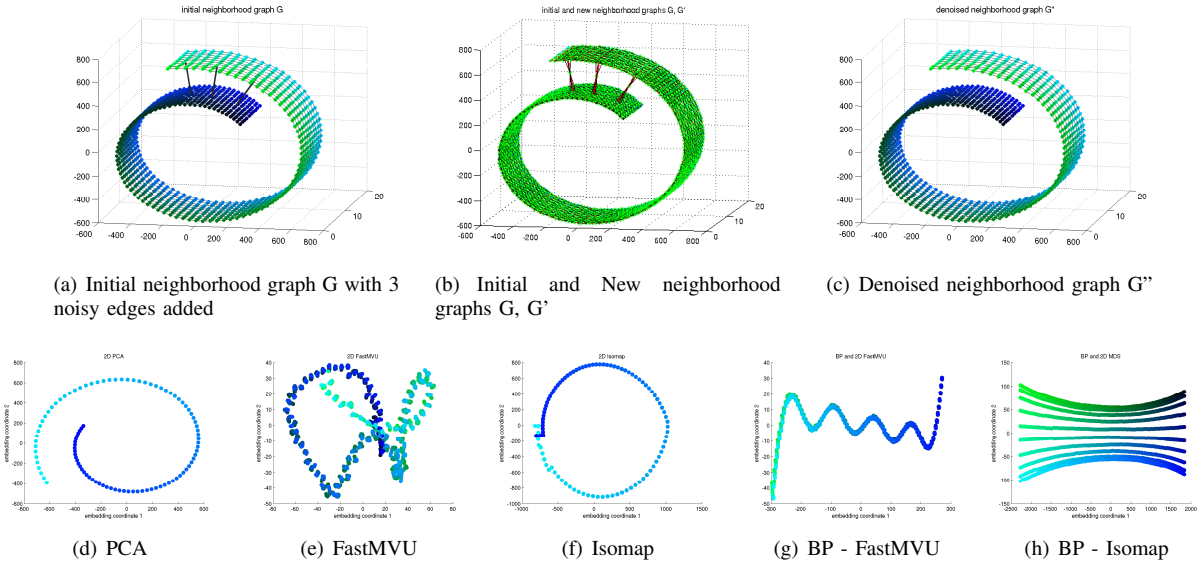
$$d_{mi} = ||v_{mi}|| \tag{4}$$

$$v_{mi} = -x_{\bar{ij}} \frac{y_i - y_j}{||y_i - y_j||} + x_{j\bar{m}} \frac{y_m - y_j}{||y_m - y_j||} \tag{5}$$

In the first case, adjacency or existence of common neighbors between i and m is an indication that all i,j,m are highly related. Consequently, the compatibility function prefers the triangle to be maintained and d_{mi} to be roughly equal to the observed Euclidean distance y_{mi}. In the second case, lack of common neighbors considers one of the neighboring edges $\bar{ij}$,$j\bar{m}$ to be noisy, thus preferring the distance d_{mi} to be as great as possible. We enforce these two cases in the compatibility function using a Gaussian distribution centered on y_{mi}, in the common neighbor case, and a logistic sigmoid function, in the distal case:

$$\psi(x_{\bar{ij}}, x_{j\bar{m}}) \sim \begin{cases} \mathcal{N}(y_{mi}, \sigma^2), & \text{if } m \in \text{adj}(i) \\ & \text{OR } \exists\ k \neq j \text{ s.t.} \\ & k \in \text{adj}(i), k \in \text{adj}(m) \\ \texttt{logsig}(0.2(d_{mi} - 1.8y_{mi}), & \text{otherwise} \end{cases} \tag{6}$$

In order to infer the marginals of the latent variables $x_{\bar{ij}}$, the Belief Propagation procedure ([14]) is used. In this procedure, for each latent variable $x_{\bar{ij}}$ (neighborhood edge distance), a probability distribution or belief is maintained. The belief $b(x_{\bar{ij}})$ which is an approximation of the marginal is formed as the product of the local evidence function $\phi(x_{\bar{ij}}, y_{\bar{ij}})$ and



(a) Initial neighborhood graph G with 3 noisy edges added

(b) Initial and New neighborhood graphs G, G'

(c) Denoised neighborhood graph G"

(d) PCA

(e) FastMVU

(f) Isomap

(g) BP - FastMVU

(h) BP - Isomap

Fig. 5. **Noisy "Swiss Roll" example (1000 points): the data initial noisy neighborhood graph (a), with denoised edges highlighted (b) and then removed (c). A comparison between 2D embeddings of the original neighborhood graph with (d) PCA, (e) Isomap, (f) FastMVU, (g) BP combined with Isomap, (h) BP combined with FastMVU. PCA and Isomap are unable to preserve consistency due to graph "short circuiting". Adding a denoising step allows for proper embedding into two dimensions.**

the incoming messages $m_{j\bar{m}\to\bar{ij}}$ from all the adjacent edges $j\bar{m}$:

$$b(x_{\bar{ij}}) = k\phi(x_{\bar{ij}}, y_{\bar{ij}}) \prod_{j\bar{m}\in \text{adj}(\bar{ij})} m_{j\bar{m}\to\bar{ij}}(x_{\bar{ij}}) \qquad (7)$$

where k is a normalization constant and adj$(\bar{ij})$ is the set of edges adjacent to edge $\bar{ij}$. For computational simplicity, we assume the belief $b(x_{\bar{ij}})$ is a discrete distribution representing probabilities over a given set of fixed distances.

Moreover, messages $m_{j\bar{m}\to\bar{ij}}$ to $\bar{ij}$ incoming from adjacent edges $j\bar{m}$ are formed using the following message update rule:

$$m_{j\bar{m}\to\bar{ij}}(x_{\bar{ij}}) \propto \sum_{x_{j\bar{m}}} \phi(x_{j\bar{m}}, y_{j\bar{m}})\psi(x_{\bar{ij}}, x_{j\bar{m}}) \prod_{x_{\bar{mk}}\in \text{adj}(j\bar{m})-\bar{ij}} m_{\bar{mk}\to j\bar{m}}(x_{j\bar{m}}) \qquad (8)$$

The denoising procedure begins by running the BP algorithm considering all belief distributions to be uniform — all latent distance values are equally probable. In each iteration of the BP algorithm, an MRF edge is selected at random and a message is sent from one of its vertices to the other. The procedure continues until convergence, with convergence properties described by Yedidia et al.[14]. Upon convergence, the mode of the distribution of the latent distance $x_{\bar{ij}}$ of each neighborhood edge $\bar{ij}$ is considered and it is compared to a threshold τ. Latent distances over τ result in the removal of the neighborhood edges they correspond to, thus, leading to a denoised neighborhood graph $\hat{D}'$.

The constants in all the above cases were found through informal experimentation and are considered user parameters. A typical value for the threshold τ is 1.2 times the maximum edge distance observed in the initial neighborhood graph.

Method	2D Embedding Error
PCA	1.2580×10^{12}
FastMVU	2.7928×10^{12}
Isomap	8.7271×10^{11}
BP - FastMVU	2.1027×10^{12}
BP - Isomap	$\mathbf{1.2099\times10^{8}}$

TABLE I

ERROR BETWEEN EUCLIDEAN DISTANCES IN THE NOISY SWISS ROLL EMBEDDINGS AND GROUND TRUTH DISTANCES.

IV. PRELIMINARY RESULTS

In this section, we present preliminary results from neighborhood denoising for manifold learning and interactive sparse 2D control of the DLR/HIT hand.

A. Swiss Roll Denoising

To evaluate our denoising procedure, we generated a 3D Swiss Roll dataset by transforming data parameterized by a planar 2D bordered manifold. The ground truth geodesic distances were known based on the 2D coordinates used to seed the Swiss Roll generation. The neighborhood graph of this data was corrupted by adding three non-adjacent noisy edges between disparate datapairs. Illustrated in Figure 5 and quantified in Table III, we compared the embeddings produced by PCA, FastMVU [18], Isomap [16] and our neighborhood denoising technique combined with MDS and FastMVU. Visually, it can be seen that PCA is simply an affine transformation of the data, due to embedding with all edges both valid and noisy. The noisy edges also present a problem for Isomap in that consistency is lost in a similar manner as PCA. In addition, Isomap brings into closer proximity points at the edges of the manifold giving the wrong impression that the edges of the manifold have continuity in the input space. Our denoising procedure was able to detect these three noisy edges and produce the proper embedding of the Swiss Roll.

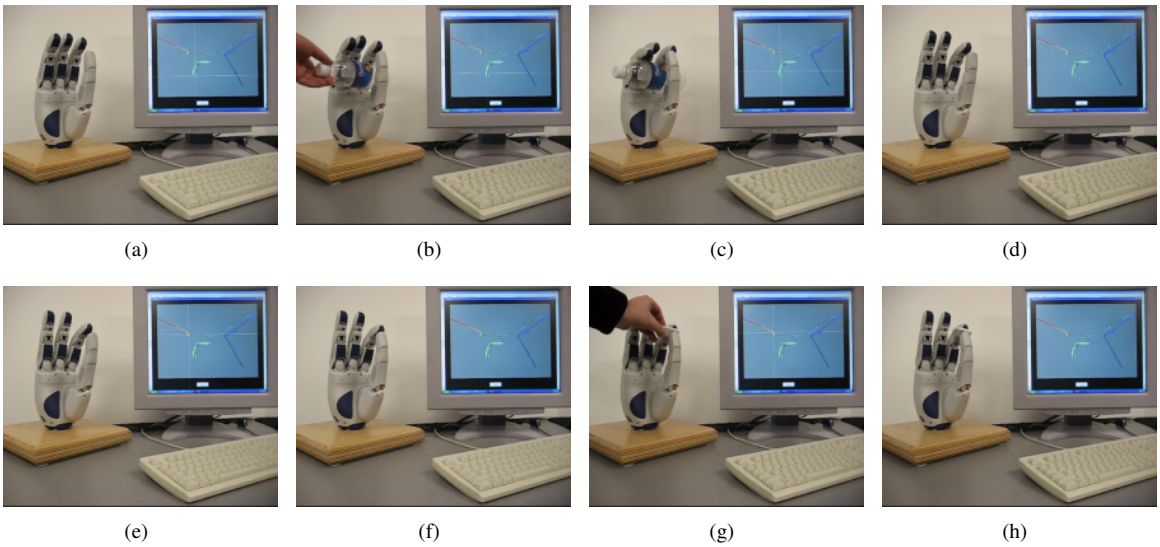


Fig. 6. **Performances of interactive sparse control for power grasping on a water bottle (top, a-d) and precision grasping of a small eraser (bottom, e-h).**

From our informal experience, FastMVU is the best of the non-denoising embedding techniques. Although in this case we were unable to produce quality results for both the original and denoised neighborhood graphs, we anticipate in the long-run that denoising with FastMVU will yield the best embedding results.

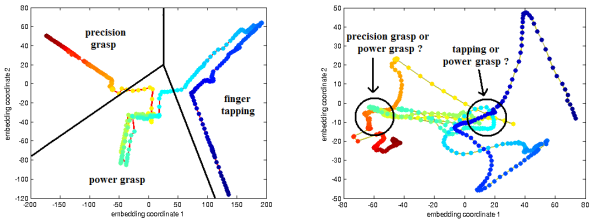
B. Interactive Control of a Robot Hand

Our sparse control and subspace embedding systems were implemented in Matlab. Mex executables formed the bridge between our Matlab implementation and the C++ interface provided by DLR for the control of the robot hand. The robot hand used in the experiments was the DLR/HIT anthropomorphic robot hand with 4 fingers and 13 DOFs. This hand has a form factor of roughly 1.5 times the size of a human hand. The human hand motion sequence that was used for training was a concatenation of finger tapping motions (one with each finger), 2 power grasps, and 3 precision grasps (one with each finger) captured by a Vicon optical motion capture system. The performer's hand was instrumented with 25 reflective markers, approximately 0.5cm in width, as shown in Figure 2. These markers were placed at critical locations on the top of the hand: 4 markers for each digit, 2 for the base of the hand, and 3 on the forearm. The resulting dataset consisted of approximately 500 frames and was intentionally selected to have at most one missing (occluded) marker at any instant of time. Each frame of hand motion was considered a point in the high-dimensional pose space. The pose space was defined as the 3D endpoints of the fingers in the hand's local coordinate system, resulting in a 12-dimensional vector. Because the DLR hand has only 3 fingers and a thumb, data for the fifth human finger (pinky) was omitted. Joint angles used for motion control of the hand were computed using an inverse kinematics procedure that minimized the distance between each finger's endpoint position with respect to the knuckle of the finger.

The 2D space of embeddings was produced by running our neighborhood denoising technique in combination with Isomap on a neighborhood defined from the high-dimensional input poses. The neighborhood graph was constructed by finding the eight nearest neighbors to each data point (k = 8). While the user was moving on the 2D space depicted on the screen, the high-dimensional robot configuration that each 2D point corresponded to was applied to the robot hand. The desired configuration of the hand was determined by the nearest neighboring point in the 2D embedding with respect to the current mouse position. Within these embeddings, continuity and consistency of the original poses were preserved (from visual inspection), placing the different types of grasps in different areas of space and facilitating user control (Figure 7a).

We performed two interactive tasks using our sparse control system: power grasp of a water bottle and precision grasp of an eraser between the thumb and index fingers. Figures 6(a-d) show the different phases of the power grasp before, during and after performing the actual grasp. The phases of the precision grasp are illustrated in figures 2(e-h). As these figures and the accompanying video (http://www.cs.brown.edu/∼aggeliki/rss2007_video.mov) show, control of the robot hand was performed in a reasonably consistent and "accessible" manner. The objects were successfully grasped and postures of the robot hand resembled the postures of a human hand during grasping.

In terms of the quality of the embedding space, we compared the embedding space of our method with the one from PCA and Isomap (Figure 7). We found that despite developing our neighborhood denoising procedure, neighborhood graphs for our grasping trials ended up not having noisy edges and, thus, denoising was not necessary. So, subspace embeddings produced by BP-Isomap and Isomap were identical. On the other hand, we saw that the embedding produced using Isomap was much better than the one produced using PCA. 2D trajectories in the Isomap embedding that corresponded to different motions as well as 2D points that corresponded to



(a) BP-Isomap/Isomap (b) PCA

Fig. 7. **(a) BP-Isomap/Isomap embeddings with annotations showing the areas of the trained hand motions, (b) PCA embeddings with annotations showing the violation of the consistency criterion.**

Method	Consistency Error	Continuity Error
PCA	0.2839	0.1037
BP - Isomap/Isomap	**0.2195**	**0.0861**

TABLE II

CONSISTENCY AND CONTINUITY ERRORS FOR THE HAND POSES EMBEDDINGS

frames of the same motion had only a small overlap. In this way, the performance of a single desired motion became much easier and more effective. The quality of all the embeddings was further assessed by quantifying the consistency and continuity criteria that we mentioned before (Table II). Consistency suggests that two poses that are dissimilar in the high-dimensional space should not be proximal in the low-dimensional space and our error metric for consistency evaluation was the following:

$$E = \sum_i \sum_{j:y_j \notin N(y_i)} g(i,j), \text{ where}$$

$$g(i,j) = \begin{cases} 1 & \text{, if } j \in N(x_i) \\ 0 & \text{, else} \end{cases}$$

where $N(y_i)$ is the set of the k nearest neighbors of point i in the high-dimensional space and $N(x_i)$ is the set of the k nearest neighbors of point i in the low-dimensional space.

Continuity over time suggests that poses that are near in time in the training motion sequence should remain proximal in the low-dimensional space. The error metric for continuity evaluation was

$$E = \sum_i \sum_{j=i+1} g(i,j), \text{ where } g(i,j) = \begin{cases} 1 & \text{, if } j \notin N(x_j) \\ 0 & \text{, else} \end{cases}$$

The largest facilitator of generating suitable control subspaces ended up being the careful selection of motion capture data. However, the noise-free character of this data came at the cost of diversity in the set of hand motion. The motion trials used by Jenkins [13] consisted of 3000 frames and contained much more variety in the style of grasps, although with up to eight missing markers at any point in time. In our future work, we plan to further experiment and extend graph denoising to handle such diverse, but unruly, datasets.

V. CONCLUSION

In this paper, we have attempted to address the problem of sparse control of a high-DOF robot hand. Considering the problems of noise in pose graph construction and motion capture, we introduced a method for denoising neighborhood graphs for embedding hand motion into 2D spaces. Such spaces allow for control of high-DOF systems using 2D interfaces such as cursor control via mouse or decoding of neural activity. Preliminary results were presented demonstrating our approach to interactive sparse control for successful power grasping and precision grasping using a 13 DOF robot hand.

REFERENCES

[1] J. del R. Millan, F. Renkens, J. Mourino, and W. Gerstner, "Brain-actuated interaction," *Artif. Intell.*, vol. 159, no. 1-2, pp. 241–259, 2004.

[2] M. Zecca, S. Micera, M. C. Carrozza, and P. Dario, "Control of multifunctional prosthetic hands by processing the electromyographic signal," *Critical Reviews in Biomedical Engineering*, vol. 30, no. 4-6, pp. 459–485, 2002.

[3] B. Crawford, K. Miller, P. Shenoy, and R. Rao, "Real-time classification of electromyographic signals for robotic control," in *Association for the Advancement of Artificial Intelligence (AAAI)*, 2005.

[4] S. Bitzer and P. van der Smagt, "Learning EMG control of a robotic hand: towards active prostheses," in *IEEE International Conference on Robotics and Automation*, 2006.

[5] J. Donoghue, A. Nurmikko, G. Friehs, and M. Black, "Development of neural motor prostheses for humans," *Advances in Clinical Neurophysiology (Supplements to Clinical Neurophysiology)*, vol. 57, 2004.

[6] D. Taylor, S. H. Tillery, and A. Schwartz, "Information conveyed through brain control: Cursor versus robot," *IEEE Trans. Neural Systems Rehab Eng.*, vol. 11, no. 2, pp. 195–199, 2003.

[7] M. Serruya, A. Caplan, M. Saleh, D. Morris, and J. Donoghue, "The braingate pilot trial: Building and testing a novel direct neural output for patients with severe motor impairment," in *Soc. for Neurosci. Abstr.*, 2004.

[8] L. Hochberg, M. Serruya, G. Friehs, J. Mukand, M. Saleh, A. Caplan, A. Branner, D. Chen, R. Penn, and J. Donoghue, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature*, vol. 442, pp. 164–171, 2006.

[9] E. Crawford and M. Veloso, "Learning to select negotiation strategies in multi-agent meeting scheduling," in *Working Notes of the AAAI Workshop on Multiagent Learning*, 2005.

[10] J. Lin, Y. Wu, and T. Huang, "Modeling the constraints of human hand motion," in *IEEE Workshop on Human Motion*, 2000.

[11] E. Todorov and Z. Ghahramani, "Analysis of the synergies underlying complex hand manipulation," in *Intl. Conference of the IEEE Engineering in Medicine and Biology Society*, 2004.

[12] C. R. Mason, J. E. Gomez, and T. J. Ebner, "Hand synergies during reach-to-grasp," *The Journal of Neurophysiology*, vol. 86, no. 6, pp. 2896–2910, Dec 2001.

[13] O. C. Jenkins, "2D subspaces for sparse control of high-dof robots," in *Intl. Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2006)*, New York, NY, USA, Aug-Sep 2006. [Online]. Available:

[14] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," pp. 239–269, 2003.

[15] Y. Bengio, J. Paiement, P. Vincent, O. Delalleau, N. L. Roux, and M. Ouimet, "Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Schölkopf, Eds. Cambridge, MA: MIT Press, 2004.

[16] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[17] T. Cox and M. Cox, *Multidimensional Scaling*. London: Chapman and Hall, 1994.

[18] K. Weinberger, F. Sha, Q. Zhu, and L. Saul, "Graph Laplacian methods for large-scale semidefinite programming, with an application to sensor localization," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hofmann, Eds. Cambridge, MA: MIT Press, 2007.