Locally Linear Embedding for Markerless Human Motion Capture using Multiple Cameras

Therdsak Tangkuampien Tat-Jun Chin*
Institute of Vision Systems Engineering,
Monash University, Victoria, Australia.
{therdsak.tangkuampien tat.chin}@eng.monash.edu.au

Abstract

We investigate the possibility of applying non-linear manifold learning techniques to aid in markerless human motion capturing. We hypothesize that the set of segmented binary images (in a constrained environment) of a person in all possible poses lie on a low dimensional manifold in the image space. Since it is not feasible to densely sample the manifold by capturing real life images, we propose to learn the manifold by using synthetic images. An accurate 3D mesh of the actor can be used to generate the synthetic 3 dimensional virtual data. A set of poses (a collection of hierarchical joint angles defining the stance of a person at a point in time) ranging the space of possible human motion is used to animate the mesh and the synthetic images are then captured by virtual cameras. We hypothesize that these vectorized synthetic images lie on a low dimensional manifold shared by the pose vectors. We then align the synthetic image and pose pairs to form a common manifold by constraining them to be equivalent. Given a new set of real images of the actor, the system can then project the captured image onto the aligned common manifold and determine the closest synthetic poses to use to linearly generate the output pose. Our experiments exhibit promising results for our method.

1. Introduction

The ability to realistically capture human motion presents numerous opportunities for real world applications, ranging from human computer interaction interfaces to computer animation and control [9] in movies and computer games. Currently to accurately capture human motions, magnetic or optical markers are systematically attached to an actor and an expensive calibrated system is used to capture the positions of these markers as the actor performs the required motions. The main disadvantages of attaching markers are the restriction imposed on the actor's motion and the cost of a system specifically designed to track the markers.

Markerless motion capture from camera images usually involves generating voxel data [15, 11, 4] of the actor from multiple camera images. The joint orientations of the actor are then determined from the voxels at each time instance and a tracking filter, such as the Kalman filter [19] is applied to the captured joints to incorporate temporal constraints. We approach the problem of camera based motion capture differently and view it as a semi-supervised manifold learning problem. We plan to develop a flexible and inexpensive markerless multiple cameras capturing system that will be able to capture human poses without preprocessing to generate voxel data.

We investigate the possibility of applying non-linear manifold learning techniques like Locally Linear Embedding (LLE) [14] or Semi-definitive Programming (SDE) [17, 18, 16] to aid in motion capturing. We hypothesize that the set of segmented images (in a constrained environment) of a person in all possible poses lie on a low dimensional manifold, $\mathcal{M}_i$, in the image space, and that their corresponding poses (collection of joint angles) also lie on a low dimensional manifold $\mathcal{M}_p$, in the pose space. This intuition is based on the idea that in a constant and controlled environment, the images of an actor captured by cameras with static intrinsic and extrinsic parameters should mainly be dependent on the current pose of the person at that particular time. We then align image and pose pairs to form a common manifold, $\mathcal{M}_c$, by constraining them to be equivalent on that manifold. Given a new set of images of the actor, we project the image data onto $\mathcal{M}_c$ using the approach of [6]. On $\mathcal{M}_c$, the poses with corresponding images that are most similar to the projections can be determined and used to linearly generate the output poses of the actor.

Since it is not feasible to densely sample $\mathcal{M}_c$ nor $\mathcal{M}_p$

*Tat-Jun Chin is a recipient of the Australia-Asia Awards 2004 conferred by the Department of Education, Science and Training (DEST) of the Government of Australia.

by capturing real life images and determining the underlying poses (furthermore, we need a motion capture system to achieve this), we propose to learn them by using synthetic images and their corresponding poses. An accurate 3D mesh of an actor can be used to generate the synthetic 3 dimensional virtual data. A set of poses (a collection of hierarchical joint angles defining the stance of a person at a point in time) ranging the space of possible human motion is used to animate the mesh and the synthetic images are then captured by virtual cameras with the same intrinsic and extrinsic parameters as the real-world cameras.

Our technique is similar to [12] in that given a set of silhouettes of an actor in image space during post-processing, the system must determine the synthetic pose that are closest to it in pose space. The search in that case is performed on a motion capture database. The disadvantage of that method is that the output pose is constrained to a subspace spanned by the poses in the motion capture database. We present a manifold projection technique that will not only be able to determine neighbourhood criteria in pose space from image data, but the pose distances between them as well. We achieve this by learning a lower dimensional manifold ($\mathcal{M}_c$) where the pose distance ratios between neighbours are preserved. During post-processing, the input can be projected onto $\mathcal{M}_c$ where the distance ratios to the k-nearest images can be found. Since image and pose pairs are aligned on $\mathcal{M}_c$, the same distance ratios can then be applied to linearly combine the corresponding k-nearest poses to generate the output pose.

2. Motion Capture System Overview

The main steps performed within our markerless camera based motion capture system are summarized below:

1. **Accurate Mesh Generation of the Actor (Section 1)**
   In the initialization step, a laser scanner is used to obtained accurate point cloud data of the actor. Radial Basis functions [2, 3] are fitted to the point cloud and re-sampled to generate a static mesh of the actor in virtual space. This mesh is skinned [10] to create a deformable mesh in DirectX format.

2. **Generate Training Data with the Mesh (Section 4)**
   A training motion file containing poses sampled from all the possible stances reachable by the human skeleton is generated. Each pose in the training set is loaded into the deformable mesh to generate its representation in virtual space.

3. **Set up Virtual Cameras in Virtual Space**
   Virtual cameras with the same intrinsic and extrinsic parameters as the real cameras (for motion capture) are created in virtual space. For each pose, the set of synthetic images of the 3D mesh as it would appear in the real cameras are captured, segmented and linked to the original pose. All the other parameters like lighting, camera positions, etc are kept constant during the training process.

4. **Generating the Distance Preserving Aligned Manifold $\mathcal{M}_c$ (Section 5)**
   Determine the **distance preserving** aligned manifold $\mathcal{M}_c$ shared by the synthetic images and their corresponding poses via LLE [6]. This is possible because each set of synthetic images captured in the virtual cameras is mainly dependant on the pose that was loaded into the mesh (all the other variables are kept constant).

5. **Pose Estimation from Manifold Projection**
   Once the system has been trained with synthetic data, real images of the person can be captured by real cameras. The captured images are then pre-processed and projected onto $\mathcal{M}_c$, where the $\boldsymbol{k}$-nearest images and the relative distances between them on $\mathcal{M}_c$ can be determined. The same distance ratios can then be used to linearly combine the corresponding $\boldsymbol{k}$-nearest poses to generate the output pose.

3. Creating an Accurate Mesh of the Actor

Before the motion of the actor can be captured, the system needs to generate synthetic training data using an accurate representation of the actor in virtual reality. We create an accurate mesh representation of the actor using point cloud data obtain from the Riegl LMS-Z420i Terrestrial Laser Scanner. At system initialization, the front and back laser scans of the actor are recorded and combined. A radial basic function (RBF) is fitted to the point cloud data [2] and then re-sampled to create an accurate mesh of the actor. A generic skeletal structure is fitted inside the accurate mesh, which is then skinned [10] and transformed to a deformable mesh in DirectX format. See Figure 1 and 2 for example results. In our project we used an evaluation version of Farfield Technology's fastRBF Matlab toolbox to generated the accurate static mesh.

4. Generating Virtual Data for Motion Capture

Using the accurate deformable mesh, a set of joint orientations p^{tr}, densely sampled from the space of possible human motion, can be applied to create 3D virtual representation of the actor. Virtual cameras, with the same intrinsic and extrinsic parameters as the real cameras to be used in motion capture are then created. From the virtual

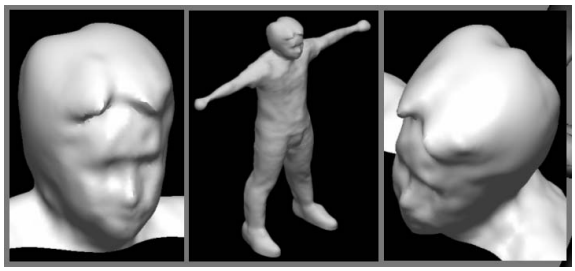

Figure 1. Resultant accurate mesh of an actor generated by sampling the Radial Basis Function of point cloud data obtained from a laser scanner.

Figure 2. Optical Marker Based Motion capture data (from the Carnegie Mellon University Graphics Lab Motion Capture Database) re-targeted to the deformable accurate mesh of the actor.

views, the system can then determine what the actor's silhouette, I_i, is expected to look like for each pose p_i^{tr} in each different cameras. Given a new set of images I of the actor captured form the real cameras, the system can segment the data and locate k-nearest silhouette images in the training set by using a silhouette similarity measure, $F(I_i, I_j)$, between two images I_i and I_j [12]. The motion capture system must then locate its k-nearest poses in the synthetic pose set as determined by a pose distance metric in the pose space, using **only** image data. We define our pose distance metric, $d(p_i, p_j)$, between two poses p_i and p_j, as the sum of euclidean distances between the joint angles in the human body. We approach this problem of mapping between images and poses by projecting the input image I onto a common manifold $\mathcal{M}_c$ which was trained to preserve the neighbourhood structure of both training images and their corresponding poses.

4.1 Synthetic Images Pre-processing

Before learning $\mathcal{M}_c$ using the training images and poses, we pre-process each set of synthetic images to convert it to a high dimensional vector X_i^{tr} in R^D. The pre-processing steps can be summarized as follows (see Figure 3):

1. **Image Segmentation and cropping** - For each pose p_i^{tr}, the set of synthetic images captured by the m virtual cameras are segmented and cropped to enclose the silhouettes as closely as possible.

2. **Image Resizing Concatenation** - All images are then resized to 60×60 pixels and horizontally concatenated to produce a single binary image I_i^{tr} for each training pose p_i^{tr}. Finally, the concatenated images are vectorized to create a high dimensional column vector X_i^{tr} for each pose p_i^{tr}. We expect that resulting column vectors generated by using different orderings of image concatenations should work as well, as long as the same procedure is consistently followed throughout the system.

During motion capture, the real images are pre-processed similarly to produce the high dimensional input vector X in R^D. The vector X is then projected onto the common manifold for the determination of distance ratios and neighbours.

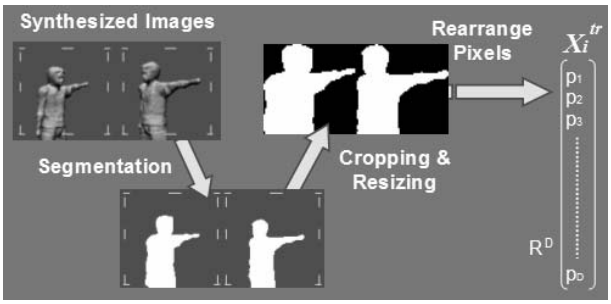

Figure 3. Pre-processing of the synthetic images to generate the training set X^{tr}.

5. Non-linear Manifold Learning and Alignment

In this section we show how Locally Linear Embedding (LLE) can be applied to map both the synthetic images, X_i^{tr} (in R^D) and their corresponding poses, p_i^{tr}, onto a common low-dimensional manifold $\mathcal{M}_c$ in R^d, where $d \ll D$. Each synthetic image and pose pair is then represented as Y_i^{tr} on $\mathcal{M}_c$. We then show how a new input vector X can then be projected onto $\mathcal{M}_c$. The projection of X should have its

corresponding Y on $\mathcal{M}_c$, though Y does not exist explicitly since it was not in the training database. We perform a euclidean distance search for the k-nearest neighbours of Y by searching for the k-nearest neighbours in Y^{tr}. We then calculate the distance ratios between Y and its k-nearest neighbours and use this distance ratio to linearly combine in pose space.

5.1 Locally Linear Embedding for motion capture

Locally Linear Embedding [14, 13] is a non-linear dimensionality reduction method whereby low dimensional data are generated from a **well-sampled** high dimensional set, whilst preserving euclidean distance ratio between local neighbours. Basically, given the training set A, LLE expresses each high dimensional vector A_i in the training set as a linear combination of its k-nearest neighbours. A cost function is then defined that measures the quality of this reconstruction,

$$\varepsilon^r(W) = \sum_i |\overrightarrow{A_i} - \sum_j W_{ij}\overrightarrow{A_j}\|^2 \ . \tag{1}$$

The weight W_{ij} summarizes the fraction of A_i in the training that can be synthesized from A_j in the same set. If the vector A_j is not one of the k-nearest neighbours of A_i, then W_{ij} will be set to zero. Another constraint imposed by LLE is that $\sum_j W_{ij} = 1$, which together with equation 1 ensures that the original neighbourhood structures are preserved on the learned manifold. Once the matrix W has been calculated, LLE then regenerate a set of lower dimensional vectors B_i in R^d by minimizing the embedding cost function,

$$\varepsilon^e(W) = \sum_i |\overrightarrow{B_i} - \sum_j W_{ij}\overrightarrow{B_j}|^2 \ , \tag{2}$$

subject to the constraint $\left(B_i B_i^T\right) = 1$. The main difference between this embedded cost function $\varepsilon^e(W)$ and the reconstruction cost function $\varepsilon^r(W)$ is that we are now keeping W_{ij} constant, whilst varying the low dimensional vectors in B. B_i and A_i are corresponding points on the learned manifold and the original manifold, respectively. The real advantage of LLE is the ability to solve this minimization as an eigenvalue problem by finding the eigenvectors of the matrix M, where

$$M = (I - W)(I - W)^T \ . \tag{3}$$

The lower dimensional representation B can then be obtained by taking d eigenvectors of M with the smallest d eigenvalues. Each eigenvector will then define a dimension for B.

5.2 LLE for Manifold Alignment and Projection

Performing LLE manifold learning separately on X^{tr} alone is not sufficient as it produces a neighbouring distance preserving manifold where the images and poses are not aligned. That is the neighbours found on the manifold are not neighbours in pose space. We solve this problem by aligning the manifold with its pose space representation via a technique proposed by [6, 5].

The problem of manifold alignment can be reformulated as follows: Given 2 sets of corresponding high dimensional vectors X^{tr} and p^{tr}, how do we project both data sets onto a common low-dimensional manifold $\mathcal{M}_c$ such that each corresponding pair X_i^{tr} and p_i^{tr} is constrained to be equally represented by a common point Y^{tr} on $\mathcal{M}_c$? Furthermore, how do we ensure that neighbourhood structures of both X^{tr} and p^{tr} are preserved on $\mathcal{M}_c$? Provided that this common manifold can be found, given a new point X not in the training set, we can then project it onto $\mathcal{M}_c$ and find the closest neighbours in the training set. The output pose can then be generated using linear distance ratios found on the manifold, but in pose space. A mathematical formulation of the combined matrix Z comprising of both X^{tr} and p^{tr} and the new input X is expressed below as

$$Z = \begin{bmatrix} X^{tr} & X \\ p^{tr} & p \end{bmatrix} \ . \tag{4}$$

The problem of determining a human pose can then be restated as a machine learning problem of calculating p in the matrix Z. An aligned manifold is created by initially performing LLE independently on the combined pre-processed image set X^{tr+} (n synthetic vectors and 1 input vectors) and another on the pose vectors p^{tr} (n synthetic poses),

$$Y^X = LLE([X^{tr}\ X]) = LLE([X^{tr+}]) \ , \tag{5}$$

$$Y^p = LLE([p^{tr}]) \ . \tag{6}$$

After separately performing LLE, the weight matrix, and therefore the M matrix (Equation 3) of both set (M^X of dimension of $n+1$ by $n+1$ and M^p of dimension of n by n) can be calculated. Letting $M^X_{(r_1..r_2,c_1..c_2)}$ represent the matrix given by the intersection of the r_1-th to r_2-th rows and the c_1-th to c_2-th columns, the projection of the two manifold Y^X and Y^p onto a common manifold Y^{tr} can be achieved by combining M^X and M^p as follows:

$$M^{tr+} = \begin{bmatrix} M^X_{(1..n,1..n)} + M^p_{(1..n,1..n)} & M^X_{(1..n,n+1)} \\ M^X_{(n+1,1..n)} & M^X_{(n+1,n+1)} \end{bmatrix} \tag{7}$$

The LLE aligned representation $[Y^{tr}\ Y]$ of the vectors $[X^{tr}\ X]$ mapped to a common manifold can be found by

taking the eigenvectors of M^{tr+} (The plus sign '+' emphasizes the fact that the matrix M^{tr+} includes the input point X whose correspondence in pose space is not known). This will produce a low dimensional manifold where the corresponding training points in X^{tr} and p^{tr} are constrained to be equivalent on the common manifold. Furthermore, the closest k neighbours and distance ratios (in pose space) to the input image I (pre-processed to become X) can be found by finding the nearest k neighbours of the mapped point Y on the manifold. The neighbours are then used to generate the output in pose space using the locally linear distance ratios found on the aligned manifold.

6. Results

In this section we present supporting results to confirm our intuition of a distance preserving (up to k neighbours) aligned manifold. A synthetic training set ranging the joint orientations attainable by the left arm is generated. The set is then applied to the accurate mesh of the actor to generate training data in virtual space. Two virtual cameras are set up at 45 degrees from the front axis of the actor to capture synthetic training images (Figure 6). The set of synthetic

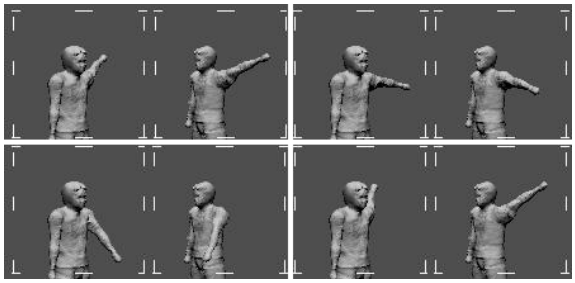

Figure 4. Example of 4 synthetically concatenated images captured from virtual cameras.

images are then segmented, cropped, resized and rearranged to create the training set X^{tr}. LLE is initially performed on X^{tr} to calculate an unaligned lower dimensional manifold in R^3. R^3 is chosen as the manifold's dimension because we are experimenting with a single ball and socket joint, which has 3 degrees of freedom. As shown in Figure 5, LLE can reduce the dimension of each set of synthetic images I_i^{tr} to a 3 dimensional point Y_i^{tr} in R^3. As mentioned previously, this manifold is not aligned and is not ideal for neighbour selection in pose space. For realistic motion capture, we need to find a lower dimensional manifold Y^{tr} in R^3 that is aligned with the pose set p^{tr}. In the case of a single ball and socket joint, the set of possible training poses in p^{tr} will span a unit sphere (Figure 6). The manifold learning process must therefore find a manifold Y^{tr} in R^3 such

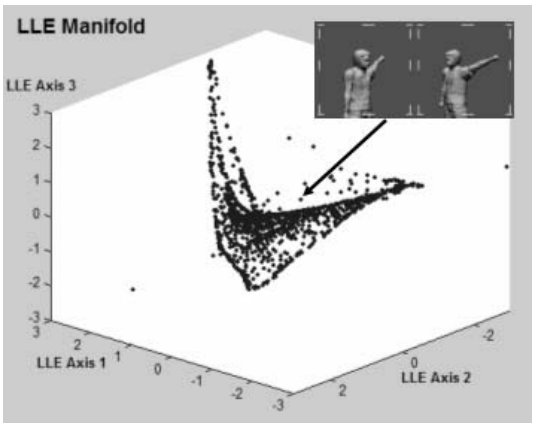

Figure 5. Unaligned manifold in R^3 calculated from LLE using 12 neighbours on the set of synthetic images with varying left arm orientation.

that it is aligned with this unit sphere. This spherical alignment is achieved using the alignment technique discussed in section 5.2. We initially perform LLE separately on X^{tr} and p^{tr} to obtain their corresponding M matrices, M^X and M^p respectively. We can find the aligned common matrix M^{tr} as follows:

$$M^{tr} = \begin{bmatrix} M^X_{(1..n,1..n)} + M^p_{(1..n,1..n)} \end{bmatrix} . \qquad (8)$$

The equation above is simply a special case of Equation 7 without the last rows and columns representing the input points with unknown representation in pose space. The 3 eigenvectors with the smallest eigenvalues of M^{tr} can be calculated and use as each dimension of the aligned lower dimensional manifold Y^{tr} in R^3. The resultant aligned manifold Y^{tr} is shown in figure 7. To confirm that this manifold is in fact aligned with the pose set p^{tr}, we select a random point Y_i^{tr} (represented as the end point at the end of the actor's left arm on the sphere - figure 8) and locate its nearest 150 neighbours using euclidian distances on the aligned manifold. The neighbours' corresponding representation in pose space are then highlighted by circles on the unit sphere as shown below. To further emphasize the benefit of finding neighbours using the aligned manifold Y^{tr}, we compare the same neighbour search to one without manifold learning and show the nearest 150 neighbours using euclidian distances in the pre-processed set X^{tr} (The euclidian distance calculation on X^{tr} is exactly the same as the hamming distance between the preprocessed segmented images in I^{tr} after cropping and resizing, but before image rearrangement to the vector X_i^{tr}). Note how almost half of the 150 synthetic points found using X_i^{tr} do not correspond to being neighbours in pose space (bottom highlighted part of the sphere - figure 9). Finally we test our manifold learn-

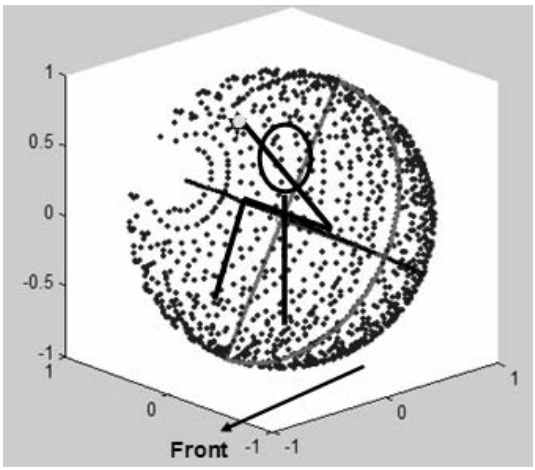

Figure 6. The set of synthetic poses p^{tr} used for training of the motion capture system.

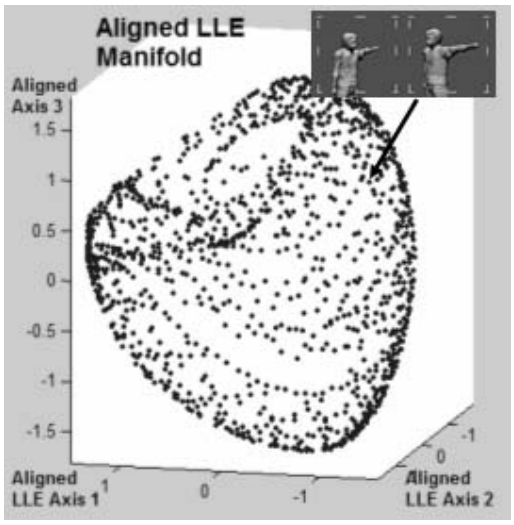

Figure 7. LLE manifold Y^{rt} aligned with the sphere spanned by the pose set p^{tr}.

ing technique with new synthetic images I, but remove its representation in pose space p from the manifold learning process and use that instead as the ground truth for error analysis. The image I is preprocessed to X and appended to X^{tr} to obtained the combined set X^{tr+}. The aligned M^{tr+} matrix is calculated using equation 5 and the the lower dimensional aligned manifold set Y^{tr+} generated. This process is equivalent to projecting the new input point X (with unknown pose space correspondence) onto a point Y on the aligned manifold in R^3. To generate the representation of Y in pose space, we find that linearly generating the output pose p^{out} from 8-12 neighbours gives the least errors. The Error function for this experiment $\varepsilon(p^{out}, p)$ is simply the shifted inverse cosine of the dot product between the output vector p^{out} and the ground truth input vector p, and has a range between 0 to 1.

$$\varepsilon(p^{out}, p) = \frac{cos^{-1}(p^{out} \cdot p)}{\pi/2} \tag{9}$$

An error value $\varepsilon(p^{out}, p)$ of 0 indicates zero error and that the pose vectors are completely aligned, where as an error value of 1 indicates complete perpendicular misalignment of 90 degrees. The resultant error plots on a test set I^{tst} of 200 concatenated images generated from the pose test set p^{tst} are shown in figure 10. In the plot, we highlight the maximum error of 0.1541, which converts to a maximum error 13.87 degrees between the system's output pose and the ground truth test set p^{tst}.

7. Discussions and Future Directions

In this paper we have presented ideas for a complete camera based system for human motion capture. The real advantages of our technique are its flexibility and lack of constraints in terms of movement of cameras and system initialization. There are also no need for any markers or preprocessing of image data to voxels in 3D space. Taking into account that it is possible to have constant preset intrinsic camera calibrations, all that is needed every time the cameras are moved around is to update the extrinsic parameters and generate training data with the mesh and re-capture the training images. The capturing accuracy can be easily increased by adding more cameras to the motion capture system, and concatenating the extra images in the training set. An interesting idea that will need further investigation is the ability for the system to capture human motion using training data from a generic mesh or mesh created from camera images [8, 7], instead of using an accurate mesh obtained from the laser scanner (Figure 1). This has the potential of an extremely portable and inexpensive motion capturing system, especially in areas of computer games and human computer interaction. Imagine being able to buy a computer game, which comes with 2-3 cameras, and only the extrinsic parameters of the cameras are required as input by the user at system initialization. Provided that good quality segmentation can be achieved, then the images can be mapped to the manifold and accurate poses found. The quality of the motion captured generated will obviously not be as accurate as the ones found by a marker based system, but the system makes up for this shortfall in terms of lower cost, portability and flexibility.

Future directions for out research includes extending the training and test motions to include all the joints on the upper part of the body, and testing with real camera images

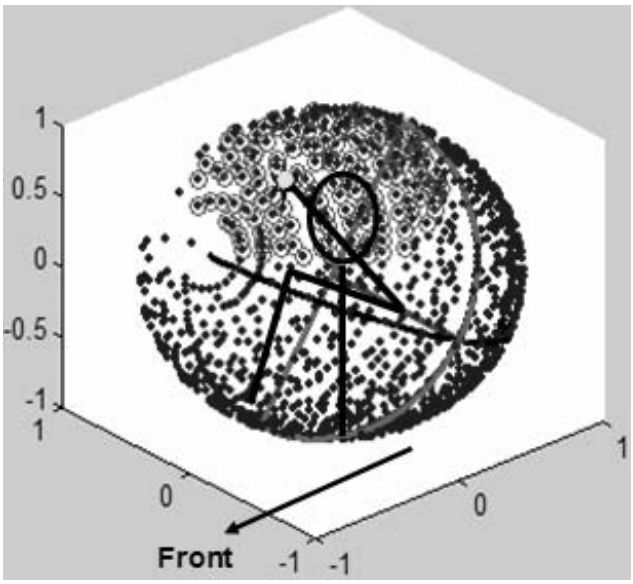

Figure 8. Plot of the pose set p^{tr} with the nearest 150 neighbours found on the aligned manifold Y^{tr} highlighted with circles.

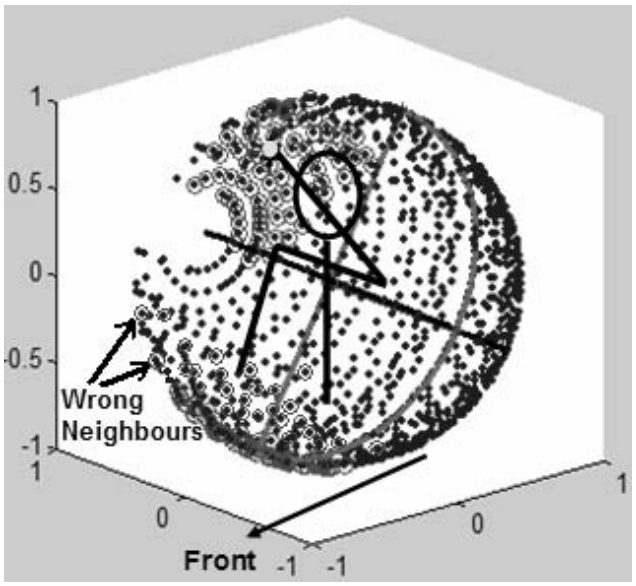

Figure 9. Plot of the pose set p^{tr} with the nearest 150 neighbours found using the distances on the pre-processed set X^{tr} before manifold learning. Note the incorrect neighbours found at the bottom of the sphere.

captured from firewire cameras. Currently the system uses segmented binary data for training and testing, another improvement we plan to investigate is to extend the system to work with colour image data. The system with then be trained with a **textured** skinned mesh rather than using a colourless mesh. The input images, in this case, will retain the foreground colours during pre-processing. This may further increase the accuracy of the system, without adding further to processing time, as the input dimensions for manifold learning would remain exactly the same as the case of binary segmented images.

Another possible improvement for the system is its speed. Currently when a new image is received, the system needs to solve for the eigenvectors of an $n+1$ by $n+1$ matrix at each frame for N training points. Instead of recalculating the eigenvalues every frame, an out of sample extension for LLE[1] can be implemented, where new points can be projected onto the manifold without fully recalculating for new eigenvectors. Another potential limitation of the system is that temporal information is not used at all in the capturing process. A possible way to fix this is to apply Kalman filters to the system's output joint angles in pose space. Another interesting area to investigate is the possibility of capturing motion from badly segmented data. In this case we propose that the noise as a result of bad segmentation will create a preprocessed input point X which will lie off the aligned manifold generated via LLE. The de-noising of the badly segmented data may then be performed by projecting the badly segmented image onto the manifold, and generating the output from the neighbours in pose space.

References

[1] 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. *Advances in Neural Information Processing Systems 16*, 2004.

[2] C. Carr, R. K. Beatson, J. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. *ACM SIGGRAPH 2001, Los Angeles, CA*, pages 67–76, 2001.

[3] J. C. Carr, R. K. Beatson, B. C. McCallum, W. R. Fright, T. J. McLennan, and T. J. Mitchell. Smooth surface reconstruction from noisy range data.

[4] K. M. Cheung, T. Kanade, J.-Y. Bouguet, and M. Holler. A real time system for robust 3d voxel reconstruction of human motions. In *Proceedings of the 2000 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '00)*, volume 2, pages 714 – 720, June 2000.

[5] J. H. Ham, D. D. Lee, and L. K. Saul. Learning high dimensional correspondences from low dimensional manifolds. *Proceedings of the ICML 2003 Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pages 34–41, 2003.

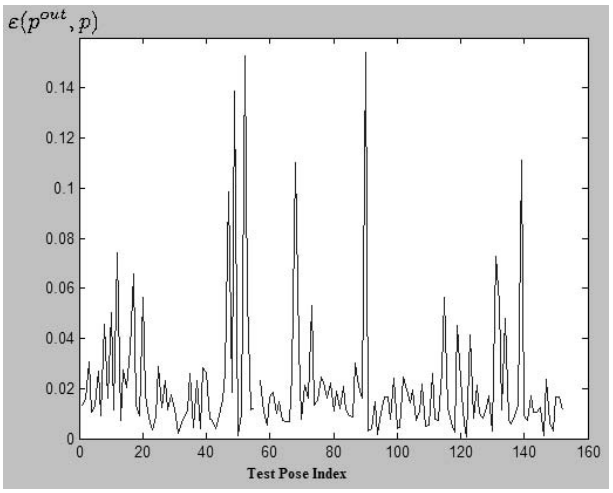

Figure 10. Error plots between the system's output pose and the ground truth pose in the test set p^{tst}.

[6] J. H. Ham, D. D. Lee, and L. K. Saul. Semisupervised alignment of manifolds. *Proceedings of the International Workshop on Artificial Intelligence and Statistics*, 2005.

[7] A. Hilton, D. Beresford, T. Gentils, R. Smith, and W. Sun. Virtual people: Capturing human models to populate virtual worlds. *In IEEE International Conference on Computer Animation,*, pages 174–185, 1999.

[8] A. Hilton, D. Beresford, T. Gentils, R. Smith, W. Sun, and J. Illingworth. Whole-body modelling of people from multiview images to populate virtual worlds. *Visual Computer: International Journal of Computer Graphics*, 16(7):411–436, 2000.

[9] J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard. Interactive control of avatars animated with human motion data. *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 491–500, 2002.

[10] F. Luna. Skinned mesh character animation with direct3d 9.0c. Technical report, moon-labs, www.moon-labs.com, September 2004.

[11] I. Mikić, M. M. Trivedi, E. Hunter, and P. C. Cosman. Human body model acquisition and motion capture using voxel data. In *AMDO '02: Proceedings of the Second International Workshop on Articulated Motion and Deformable Objects*, pages 104–118. Springer-Verlag, 2002.

[12] L. Ren, G. Shakhnarovich, J. Hodgins, H. Pfister, and P. Viola. Learning silhouettes features for control of human motion. *ACM Journal*, 4(3):1–33, 2004.

[13] L. K. Saul and S. T. Roweis. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2269, 2000.

[14] L. K. Saul and S. T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

[15] C. Theobalt, M. Magnor, P. Schler, and H.-P. Seidel. Combining 2d feature tracking and volume reconstruction for online video-based human motion capture. In *PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, page 96. IEEE Computer Society, 2002.

[16] K. Q. Weinberger, B. D. Packer, and L. K. Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, 2005.

[17] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition - CVPR*, 2:988–995, 2004.

[18] K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. *Proceedings of the Twenty First International Confernence on Machine Learning (ICML),Banff, Canada*, pages 839–846, 2004.

[19] G. Welsh and G. Bishop. An introduction to the kalman filter, siggraph 2001. *SIGGRAPH 2001*, 2001.