Detail Preserving 3D Motion Compression Based on Local Transformation

Chang Liu, Zhaoqiang Lai, Jiaxi Hu, Jing Hua
Department of Computer Science
Wayne State University
Detroit, MI, U.S.A.
changliu@wayne.edu

***Abstract*—With the advance of 3D acquisition techniques, more and more 3D motion data with fine details become available. In order to efficiently represent the data, we explored different motion compensation algorithms through conventional techniques and analyzed their performance both in theory and by experiments. A novel motion compression framework based on local transformation is proposed to solve the existing problems. Experiments show that our algorithm can achieve very high compression rate while preserving the fine details. The algorithm is easy to implement and is versatile to different applications including non-skeleton driven motion compression.**

***Keywords*-3D motion; compression; local transformation; details**

I. Introduction

With the advance of 3D acquisition techniques [1][2], more and more 3D models are available. On one hand, high resolution range scanners make 3D large terrain and cityscape data available; on the other hand, fast depth camera arrays enable us to capture 3D motion or facial expressions at a comparable speed with traditional video recorders [3]. Skeleton based animation and skinning animation are prevalent since they are easy to carry out and intuitive for manual editing in motion synthesis. However, compared to scanned mesh animation [4], they are less realistic since it is difficult to model the change of surface details during the motion and it is laborious to design sophisticated motions. Motion acquisition based on agents attached to humans provides an alternative, but it is still hard to capture the surface details especially when the motion is non-skeleton driven. The latest technology gives us a better stand. In [5], the subject in motion can be captured completely without using agents. The surface details are preserved at the same time. With this extra detail information acquired, 3D scanning motion data can be too large to store and process; thus, it prevents the data from being fast distributed to ordinary end users for entertainment or educational use. However, like 2D videos, the 3D scanned motion data usually contain large amounts of redundancies which can be reduced from the continuous sequence using proper algorithms. The core part of the algorithm is to use a limited number of shapes sampled at certain time stamps to predict the rest shapes in the series. In MPEG, this process is called motion compression. This term will be used in this paper to address the same problem but on 3D scanned data. In MPEG, motion is in the same structured domain as other properties such as color and shading; thus they can be compensated at the same time. However, it is not straightforward to use the same technique to compensate motion and surface details in general since the canonical domain of the surface is changing during the motion. Furthermore, most conventional methods working on a single 3D object are not suitable for motion data analysis or need to be verified. This paper mainly discusses how to implement efficient motion compression algorithms based on the new type of 3D motion data. The main contributions are:

1) Several algorithms derived from the state-of-the-art are implemented and compared.
2) A novel motion compression framework based on clustering and local transformation is presented.
3) The proposed algorithm is very easy to implement and high compression rates can be achieved without loss of surface details.
4) The algorithm is suitable for general motion including both skeleton driven and non-skeleton driven motion. The quality control is also easy to carry out.

The remaining paper is organized as follows: Sec. II surveys some work related to our proposed algorithms; Sec. III compared the results of some durable algorithms for motion compensation; Sec. IV discussed our motion compression framework in detail; Sec. V lists some experimental results based on our framework. Finally, Sec. VI presents a discussion and makes a conclusion.

II. Related work

To date, relatively less compression research work is carried out on real-world captured motion data than on image data. One reason is that using traditional techniques it is hard to retrieve the correlation between 3D data frames without proper post-registration; fortunately, more recent techniques have overcome the difficulty by incorporating tracking techniques and a deformable template into the scanning process. In [4], Vlasic et al. tried to match the deformable template to image silhouettes captured by a high definition camera array. Not a coincidence, Li et al. [5] used a template to match the partial scans from a depth camera.

Their methods are efficient and easy to apply. It is expected that more and more this kind of 3D motion data will be available in the near future. The registration-on-the-fly gives us a consistently meshed sequence to directly analyze the motion on the surface, which provides us with better details than skeleton based methods.

For the purpose of 3D motion data storage, some techniques can be considered. Mesh streaming [6] stressed the compression of connectivity information of triangle meshes were proposed. Compared to other methods, it is out-of-core and is aimed at processing a very large single mesh. Geometric compression can be considered in either the spatial or the frequency domain. Traditional methods use position prediction [7] and ad hoc data structure to remove visual redundancies. An MPEG style codec framework based on linear prediction was proposed in [8]. More recently, spectral analysis has introduced new ways to geometry representation [9]. The challenge of spectral analysis on the manifold is the lack of a common domain. Thus it is usually nontrivial to find a set of analytical basis for an arbitrary shape [10]. The latest research on Laplacian-Beltrami eigen analysis [11] has provided us with tools to decompose the geometry into details at different scales. However, all these methods has little concern on motion data and direct extension to motion compression needs validation. The most relevant work was from Yasmine et al. [12] who modeled the motion using local transformations. However, their method is preliminary with no surface detail preserving and smoothness control presented. Another similar work based on clustering was proposed in [13]; however, the motion is still modeled as the displacement of vertex coordinates which is lack of the power to describe local shapes. A thorough survey of dynamic mesh compression can be found in [14]. According to [14], our proposed work can be classified as multi-resolution geometry compression while existing topology compression methods can be overlaid with no extra effort. Compared to existing motion compression methods based on clustering, an analog in MPEG, our method does not rely on a predefined motion model but considers general local shape changes instead. The vertex prediction is replaced by transformation prediction as well. The low frequency information is well preserved by the decoding process.

Our work is motivated by the advancement of motion transfer and mesh editing. We can get one shape from another by applying the transformation between the two. Oscar et al. [15] have found that the face-based method has less distortion than vertex-based method since the neighborhood of each face form a simplex on which the local geometry can be easily defined. Fu et al. [16] further developed a method based on local vertex transformations, which preserves local shapes. However, this method needs certain perturbation on flat surfaces since the local transformation for a vertex cannot be estimated using its neighborhood in this case. Sumner et al. [17] solved this problem by adding a vertex in the normal direction of each face. Besides, Guo et al. [18] used a spectrum based method to deform the shape in a coarse scale and the details were added back afterwards. This makes the editing more efficient; However, to model the details as the hight function along the vertex normal will not be sufficient for complex patterns such as garment folds. The proposed work is also very related to [19] and [20] by chance where the techniques were used to build the skeletal structure of shapes. In contrast, our work focuses on motion data compression and is not restricted to articulated motions. Our framework is more rigorous and has more control over local geometry. In the following section we will compare some techniques which can be directly surveyed and used in 3D motion compression.

III. Comparison of spatial and spectral methods

A. Notations and terms

We'll use the following terms to refer to certain objects or operations:

1) Frame: a shape captured at one time stamp.
2) Key frame(s): the shapes captured at certain time stamps used to predict the rest shapes for compression purposes.
3) Intermediate frames: the shapes captured excluding key frames.
4) Predicted frame(s): the shapes that are predicted from the key frames to approximate intermediate frames.
5) Source mesh: the mesh before the current motion occurs. The terms, source mesh and target mesh, are used when we describe a general algorithm. When the algorithm is applied to motion compression, the source mesh is referred to as the key frame, and the target mesh becomes the intermediate frame.
6) Target mesh: the mesh as the result of the current motion.

In order to demonstrate our framework unique and essential, two types of techniques are implemented and compared in this section.

B. Motion compensation based on spectral decomposition

It is well known that the eigen functions of Laplacian-Beltrami operator form a complete basis on a manifold [11]. Furthermore, these bases are isometric invariant and can be used as a signature for similar shapes [11]. It is a widely used global representation of shapes. When the geometry is treated as a function defined on the manifold, the vertex positions can be mapped onto these basis; thus, compression can be achieved by quantizing the coefficients or discarding some of them in the same way as traditional discrete cosine transformation (DCT) encoding. Since articulated motions are near isometric, we can map vertex positions onto the basis of the key frames. For compression purposes, only the first 1000 coefficients are stored and used to recover the shapes. Two adjacent frames of a human body motion are

used to test this algorithm. In Figure 1, the key frame is on the upper left and the intermediate frame is on the upper right. Using the first 1000 coefficients we can recover the key frame as shown on the lower left. Based on the same basis and the first 1000 coefficients of the intermediate frame, the corresponding predicted frame is shown on the lower right. Although the two original frames are similar to each other, the recovered shapes are quite different. While the recovered key frame is a smoothed version of the original shape, the difference is visually significant between the intermediate frame and the predicted frame.

It is obvious that when the metric of the surface changes slightly, the original basis are not suitable for decomposing the geometry into different levels of detail anymore. As an alternative, we also tried to use the same basis to decompose the differences between the intermediate frames and the key frames. This approach works when the two shapes are very similar. However, it fails when they differ from each other. We keep the key frame unchanged as in Figure 1a and use another intermediate frame shown in Figure 2a. The predicted frame is shown in Figure 2b. As can be seen, the distortion is dramatic. The above experiments show that an arbitrary motion can not be predicted precisely when the shape is considered using only global functions.

C. Motion compensation based on spatial transformation

As an alternative to the global spectral representation, the motion of a shape can be modeled as a set of transformations, i.e., use a spatial decomposition instead of a spectral decomposition. For example, each vertex bears a transformation during the motion, when the vertices with a similar motion are clustered, a concise representation can be achieved. This idea was explored in [12]. A matrix in its homogenous representation is used to depict the motion of each cluster of vertices. However, representing the transformation in a global coordinate system will introduce unnecessary translation terms and a single transformation of such does not contain any information of the original shape; Thus, the algorithm in [12] is not ideal for shape preserving motion compression. This is extremely clear when only a few number of clusters are used. In Figure 3, the algorithm is implemented and tested on two adjacent frames. The model contains 10000 vertices, and 256 matrices are used to recover the motion of the same number of clusters. The discontinuity on the predicted frame can be easily perceived. For this kind of real world scanned data, the garment motion is very complex. Each vertex will undergo a very unpredictable different motion from others.

In order to solve this problem, local transformations can be used instead. In [16], a vertex and its neighboring vertices are used to estimate the transformation. This algorithm not only successfully preserves local details to a certain extend but also faithfully picks up the global motion. The underline assumption is that when the neighboring vertices bear the same one transformation as the current vertex, the estimation error will be minimized. However, the correlation between a vertex and its neighborhood is not consistent among all the vertices. When they are highly correlated, eg, co-planar, the transformation is not unique. In such cases, some local disturbance is introduce to regularize the local motion [16], which limits the application of the algorithm. One solution to this would be considering the transformation for each triangular face instead of each vertex as suggested in [15]. We will explore this idea in detail.

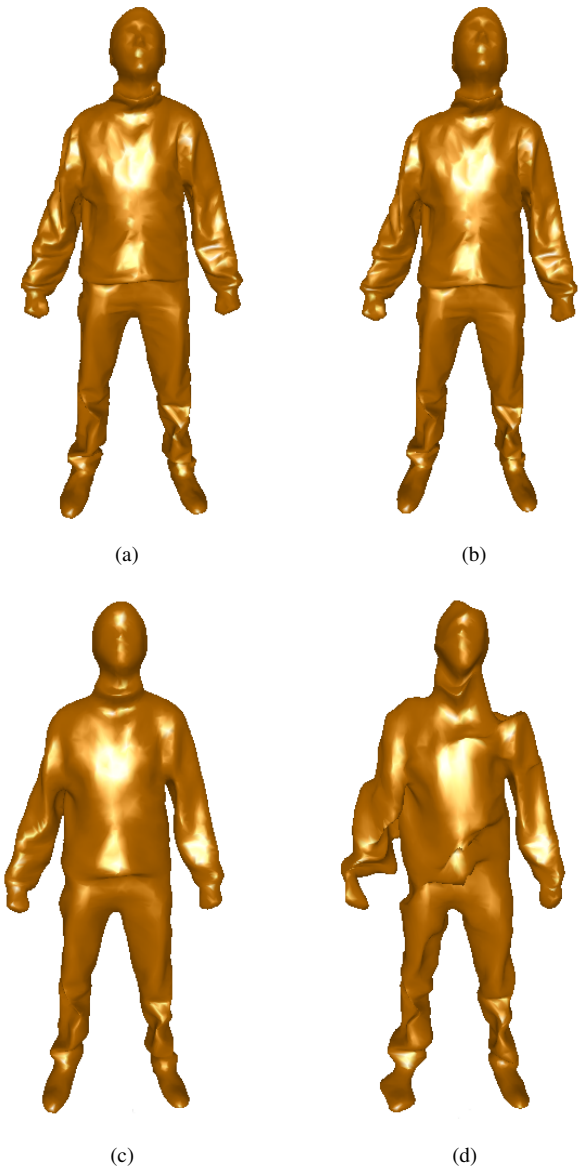
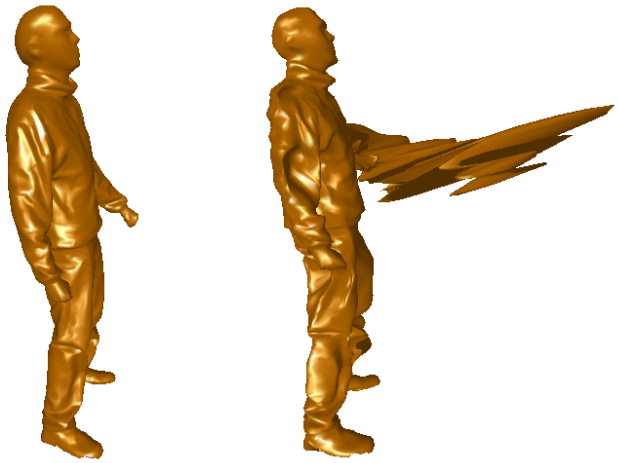


Figure 1: Recovered shape from spectrum decomposition: (a) - key frame; (b) - intermediate frame; (c) - recovered key frame; (d) - recovered intermediate frame (predicted frame).



(a) (b)

Figure 2: Another intermediate frame and the predicted frame based on the key frame in Figure 1a.

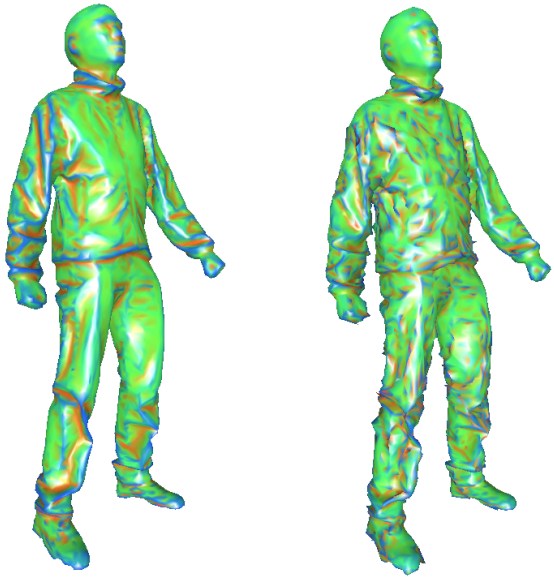


Figure 3: Right: an intermediate frame; left: the predicted frame based on [12] with 256 clusters, the details are distorted. The models are colored according to the mean curvature.

IV. A NOVEL MOTION COMPRESSION FRAMEWORK BASED ON LOCAL TRANSFORMATION

Earlier research shows that a set of local transformations can be used to recover the global motion. However, as shown in the previous section, the local transformations based on vertices are not informative enough to describe local shape changes. The fundamental cause can be found in geometry saying that the shape of a primitive is determined by its angles and edge lengths. Thus, when trying to preserve the local geometry of a triangular mesh, we are particularly looking at the transformation for each triangular facet. The algorithm is briefly introduced as below.

In order to estimate the local transformation of a triangular facet, an additional vertex is required. Let $\mathbf{v}_i$ and $\tilde{\mathbf{v}}_i, i \in 1,\ldots,3$, be the vertices of the triangle before the motion and after the motion, respectively. For a triangle in the mesh which has the configuration of $\mathbf{v}_i, i \in 1,\ldots,3$, we compute the fourth vertex as,

$$\mathbf{v}_4 = \mathbf{v}_1 + (\mathbf{v}_2 - \mathbf{v}1) \times (\mathbf{v}_3 - \mathbf{v}1)/\sqrt{|(\mathbf{v}_2 - \mathbf{v}1) \times (\mathbf{v}_3 - \mathbf{v}1)|} \tag{1}$$

and the same computation for $\tilde{\mathbf{v}}_4$. An affine transformation defined by the 3×3 matrix $\mathbf{Q}$ and displacement vector $\mathbf{d}$ transform these four vertices as follows:

$$\mathbf{Q}\mathbf{v}_i + \mathbf{d} = \tilde{\mathbf{v}}_i, \ \ i \in 1,\ldots,4. \tag{2}$$

If we subtract the first equation from the others to eliminate $\mathbf{d}$ and rewrite them in matrix by treating the vectors as columns, we obtain $\mathbf{QV} = \tilde{\mathbf{V}}$ where

$$\begin{aligned} \mathbf{V} &= [\mathbf{v}_2 - \mathbf{v}_1 \ \ \mathbf{v}_3 - \mathbf{v}_1 \ \ \mathbf{v}_4 - \mathbf{v}_1] \\ \tilde{\mathbf{V}} &= [\tilde{\mathbf{v}}_2 - \tilde{\mathbf{v}}_1 \ \ \tilde{\mathbf{v}}_3 - \tilde{\mathbf{v}}_1 \ \ \tilde{\mathbf{v}}_4 - \tilde{\mathbf{v}}_1]. \end{aligned} \tag{3}$$

So $\mathbf{V}$ can be calculated by

$$\mathbf{Q} = \tilde{\mathbf{V}}\mathbf{V}^{-1}. \tag{4}$$

Compared to other work, this local transformation Q contains solely the shape change of a triangular facet during the global motion. By clustering of Q for all the facet between intermediate and key frames, it is possible to group similar surface changes and remove redundancies accordingly.

The following algorithm carries out the motion compensation and recovers the shape based on all the known local transformations. We use the terms source mesh and target mesh as defined in Sec. III-A. Since the one-to-one correspondence is known between two arbitrary frames. Therefore, there are pairs of transformation $\{\mathbf{S}_i,\mathbf{T}_i\}$ for the source mesh and the target mesh. In order to maintain consistency, additional constraints are added to the transformation:

$$\mathbf{T}_j\mathbf{v}_i + \mathbf{d}_j = \mathbf{T}_k\mathbf{v}_i + \mathbf{d}_k, \forall i, \forall j, k \in p(v_i), \tag{5}$$

where $p(\mathbf{v}_i)$ is set of all triangles that share vertex $\mathbf{v}_i$. In order to solve the transformation, the difference between the source and target transformation under the consistency constraint has to be minimized:

$$\min_{\mathbf{T}_1+\mathbf{d}_1,\ldots,\mathbf{T}_{|T|}+\mathbf{d}_{|T|}} \sum_{j=1}^{|T|} \|\mathbf{S}_j - \mathbf{T}_j\|_F^2$$

subject to

$$\mathbf{T}_j\mathbf{v}_i + \mathbf{d}_j = \mathbf{T}_k\mathbf{v}_i + \mathbf{d}_k, \forall i, \forall j, k \in p(v_i), \tag{6}$$

where $\|\cdot\|_F$ is the Frobeniu norm.

For the target mesh, the $\mathbf{V}$ depends on the known, the elements of of $\tilde{\mathbf{V}}$ are the coordinates of the unknown deformed vertices. Thus, the elements of $\mathbf{T}$ are linear combinations of the coordinates of the unknown, deformed vertices $\mathbf{T} = \tilde{\mathbf{V}}\mathbf{V}^{-1}$. Based on this fact, the minimization problem can be rewritten as

$$\min_{\tilde{\mathbf{v}}_1,\ldots,\tilde{\mathbf{v}}_n} \sum_{j=1}^{|T|} \|\mathbf{S}_j - \mathbf{T}_j\|_F^2. \tag{7}$$

The solution to this problem is the solution to a system of linear equations. Rewriting the problem in matrix form yields

$$\min_{\tilde{\mathbf{v}}_1,\ldots,\tilde{\mathbf{v}}_n} \sum_{j=1}^{|T|} \|\mathbf{c} - \mathbf{A}\tilde{\mathbf{x}}\|_2^2, \tag{8}$$

where $\tilde{\mathbf{x}}$ is a vector of the unknown deformed vertex locations, $\mathbf{c}$ is a vector containing entries from the source transformations, and $\mathbf{A}$ is a matrix that relates $\tilde{\mathbf{x}}$ to $\mathbf{c}$. The final solution is the following form:

$$\tilde{\mathbf{x}} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^\mathbf{T}\mathbf{c}. \tag{9}$$

Before the above algorithm is applied to motion compression, we need to notice that a global translation may occur during the motion besides the shape change between frames. It is effortless to store the position of an arbitrary chosen vertex (called anchor vertex) for each intermediate frame and translate the predicted frame to meet this criteria. As an alternative, we can feed the information into Eq. 8 by moving the corresponding column from $\mathbf{A}\tilde{\mathbf{x}}$ to $\mathbf{c}$ to form another linear system as suggested by Sumner [17]. Furthermore, if the model has open boundaries such as the facial model we used, more than one anchor vertex is needed to regularize the motion along the boundary.

By utilizing the above techniques, our motion compression framework is described as follows:

Algorithm 1: Encoding:

1) Determine the key frames and intermediate frames from a motion series. All the intermediate frames prior to the next key frame will be predicted by the current key frame. The intermediate frames and predicted frames are paired with the current key frame.
2) For each key frame and intermediate frame pair, the local transformations between the couple are calculated according to Eq. 4.
3) Group the transformations for each pair using k-means clustering. Generate one transformation for each cluster and store them together with the cluster labels for each intermediate frame. Store an additional translation vector or a known vertex position for the intermediate frame as well.

Algorithm 2: Decoding:

1) Each predicted frame is obtained by assigning a transformation to each facet of the paired key frame accordingly to the stored information using Eq. 9.
2) Apply the global translation to each predicted frame if needed.

The transformation Q_k for each cluster $\mathbf{C}_k, k = 1, 2, \cdots$, can be generated by minimizing the prediction error.

$$\min_{\mathbf{Q}_k} \sum_{n=1}^{\|\mathbf{C}\|} \|\mathbf{Q}_k\mathbf{v}_n^k - \tilde{\mathbf{v}}_n^k\|^2, \mathbf{v}_n^k \in C_k. \tag{10}$$

The vectors $\mathbf{v}_n^k$ and $\tilde{\mathbf{v}}_n^k$ are defined as in Eq. 3.

In fact, since the consistent constrain will regularize the transformations, in practice, we can use the transformation of an arbitrary triangular facet in the cluster as $\mathbf{Q}_k$. Our experiments show that this algorithm works well for all the data we tested.

V. Experiments

We have examined our framework based on MIT's articulated mesh animation data sets [4]. It includes simple human motions such as marching (Figure 4) and jumping as well as complicated ones like handstand (Figure 6) and dancing (Figure 5). Each data set contains 148 to 250 frames. The characters for scanning all wear loose clothes which deform very randomly and create a lot of garment movements. The scanned models are triangulated with 10002 vertices and 20000 faces. We also tested our framework on a facial sequence with expression change (Figure 7), which is consist of triangular meshes with 1500 vertices and 3000 faces. For most of the body motion data, 256 clusters are created using k-means clustering on the 20000 faces, where the errors are bounded within 0.5% for all the data we have tested. For the facial data , 64 clusters are used. Thus, 5/8 byte is used to store the cluster label for each face and 1 transformation matrix of 3 by 3 is stored for each cluster. For each intermediate frame predicted, the bit rate is calculated according to the following formula:

$$\frac{B_M * N_C + \log N_C * N_F}{N_V}(bits/vertex/frame), \tag{11}$$

where B_M is the number of bits used to represent a matrix, N_C is the number of face clusters, N_F is the number of faces and N_V is the number of vertices. If 8 bytes are used for a floating point number, the bit rate will be $30.74bits/vertex/frame$ and $20.29bits/vertex/frame$ for the body motion data and facial data, respectively, without further compression. Compared to uncompressed data of $192bits/vertex/frame$, very high compression rates can be achieved. This result is comparable to that reported in [13] with the same error bounds. However, our work outperforms in terms of the visual errors. In most data sets we have tested, only one key frame (the first frame is used as the key frame) is needed to obtain visually acceptable predictions, i.e., all

the rest frames in a sequence are intermediate frames and are compressed by using **Algorithm 1**. Thus, the average bit rate of a sequence will be approximately the same as the numbers reported above.

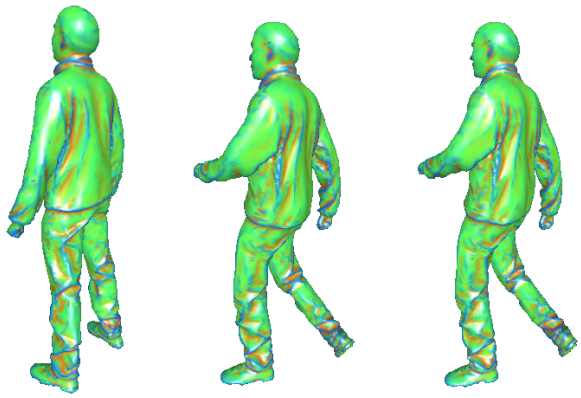


Figure 4: Two frames from a marching sequence. Left: key frame; middle: intermediate frame; right: predicted frame. The models are colored according to the mean curvature.

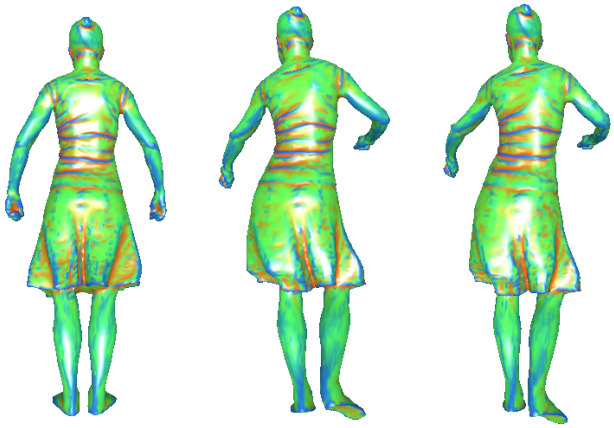


Figure 5: Two frames from a dancing sequence which contains fine garment motion. Left: key frame; middle: intermediate frame; right: the predicted frame based on 256 clusters. The models are colored according to the mean curvature.

Since the compression is lossy, the following formula is used to calculate the prediction error for a predicted frame:

$$Err_{predict} = (\sqrt{\sum_{i} \|\mathbf{v}_i - \tilde{\mathbf{v}}_i\|_2^2 / N_v})/\overline{L_e}, \qquad (12)$$

where $\mathbf{v}_i$ and $\tilde{\mathbf{v}}_i$ are vertices of the intermediate frame and the predicted frame, respectively, for $i = 1, 2, ..., N_v$, and $\overline{L_e}$ is the average edge length of the model over all the frames. This measurement mainly captures the variance of surface details. Table I shows the expectation of $Err_{predict}$ for all the predicted frames. The prediction error is proportional to

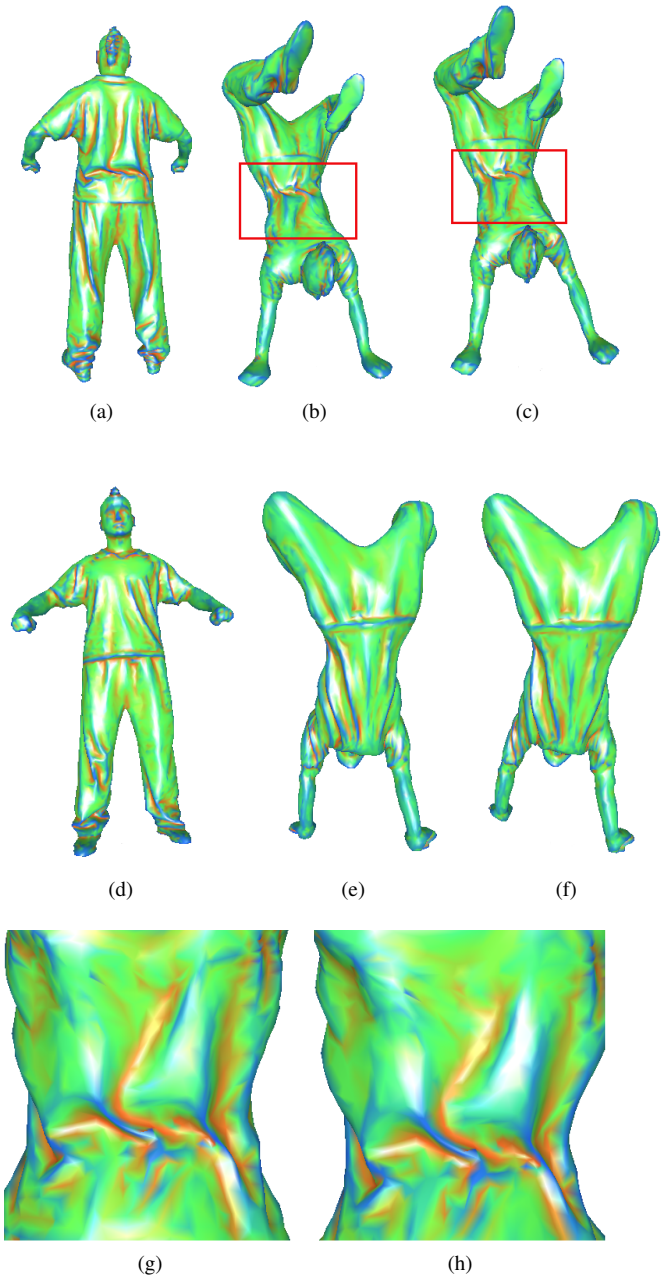


Figure 6: Two frames from a handstand motion where the surface changes more significantly than others. (a): key frame (b): intermediate frame; (c): the predicted frame based on 512 clusters, The models are colored according to the mean curvature; (d) another view of the key frame in (a); (e) another view of the key frame in (b); (f) another view of the key frame in (c); (g) zoom in view of the rectangular area in (b); (h) zoom in view of the rectangular area in (c).

the number of clusters used for encoding.

We have also calculated the prediction errors when different numbers of clusters are used. The $Err_{predict}$-N_C curve is show in Figures 8. Figure 9 shows the clusters calculated between the paired key frames and the intermediate frames in Figures 4,6,5, and 7, respectively.

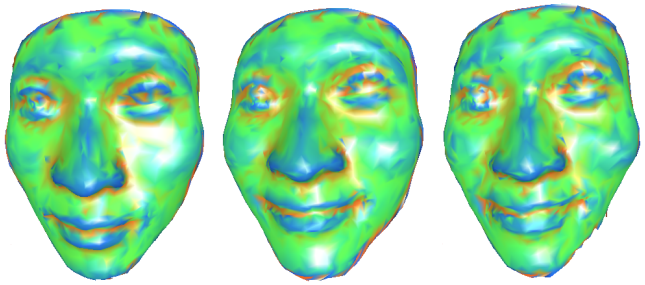


Figure 7: Two frames from a facial motion which is non-skeleton driven. Left: key frame; middle: intermediate frame; right: predicted frame.

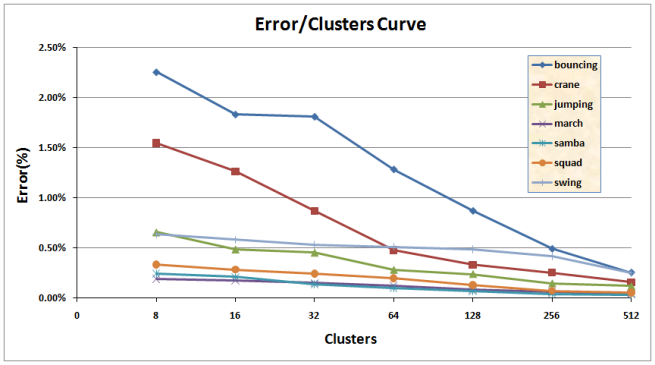


Figure 8: The prediction error $Err_{predict}$ change against the number of clusters used.

Table II shows performance of the algorithms in terms of speed. All the computations are conducted on a Core2 Quad CPU with 6G DDR2 memory.

VI. Conclusion and Future Work

In this paper, a novel 3D motion compression framework is presented. It works on registered triangular mesh sequences. The core algorithms are based on clustering of the local transformations of the triangular faces. Besides, we implemented and tested several state-of-the-art techniques on motion compression. Compared to other work, our approach can preserve the surface details during very complicated motions while achieving very high compression rates. The framework can also be used on non-skeleton driven motions directly. Our experiments show that our motion compensation algorithm successfully captures human motion with garment movements and facial motion with expression changes. In the future work, we will design

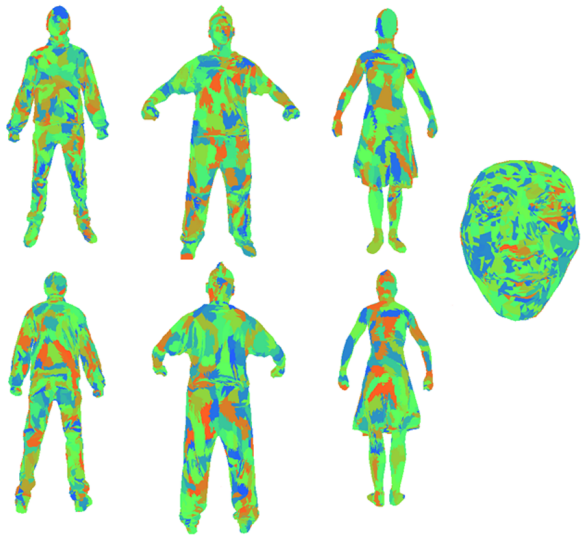


Figure 9: The clusters of local motion (calculated for each face) between the key frames and intermediate frames for the data shown in Figures 4,6,5,7. Different colors represent different clusters for each data set. The bottom row shows different views of the top row.

algorithms to compute the optimal number of clusters and key frames.

References

[1] Y. Furukawa and J. Ponce, "Dense 3D motion capture for human faces." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1674–1681.

[2] A. Golovinskiy, V. G. Kim, and T. Funkhouser, "Shape-based recognition of 3D point clouds in urban environments," in *IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 2154–2161.

[3] M. Liao, Q. Zhang, H. Wang, R. Yang, and M. Gong, "Modeling deformable objects from a single depth camera," in *IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 167–174.

[4] D. Vlasic, I. Baran, W. Matusik, and J. Popovic, "Articulated mesh animation from multi-view silhouettes," *ACM Transactions on Graphics*, vol. 27, no. 3, pp. 1–9, 2008.

[5] H. Li, B. Adams, L. J. Guibas, and M. Pauly, "Robust single-view geometry and motion reconstruction," *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2009)*, vol. 28, no. 5, December 2009.

[6] M. Isenburg, Y. Liu, J. R. Shewchuk, and J. Snoeyink, "Streaming computation of delaunay triangulations," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 1049–1056, 2006.

[7] J. Peng, C. Kim, and C. Kuo, "Technologies for 3D mesh compression: A survey," *Journal of Visual Communication and Image Representation (JVCIR)*, vol. 16, no. 6, pp. 688–733, December 2005.

Table I: Prediction error

Data set	bouncing	crane	jumping	March	samba	squad	swing	handstand	face
Error	5.1E-05	1.7E-05	0.000145	2.2E-05	1.1E-5	2.1E-05	1.4E-05	0.000215	0.01

Table II: Time taken for computing one frame (in seconds)

Data set	bouncing	crane	jumping	March	samba	squad	swing	handstand	face
Encoding	109	130	156	145	134	128	189	178	12
Decoding	22	23	21	22	25	21	22	20	0.3

[8] J. Ahn, C. Kim, and Y. Ho, "Predictive compression of geometry, color and normal data of 3D mesh models," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 2, pp. 291–299, February 2006.

[9] Z. Karni and C. Gotsman, "Spectral compression of mesh geometry," in *ACM SIGGRAPH*, New York, NY, USA, 2000, pp. 279–286.

[10] ——, "3D mesh compression using fixed spectral bases," *Graphics Interface*, pp. 1–8, 2001.

[11] M. Reuter, F.-E. Wolter, and N. Peinecke, "Laplace-beltrami spectra as "Shape-DNA" of surfaces and solids," *Computer-Aided Design*, vol. 38, no. 4, pp. 342–366, 2006.

[12] Y. Boulfani-Cuisinaud and M. Antonini, "Motion-based geometry compensation for dwt compression of 3D mesh sequence,," in *IEEE International Conference on Image Processing (ICIP)*, vol. 1, 2007, pp. 217–220.

[13] R. Amjoun and W. Straber, "Efficient compression of 2d dynamic mesh sequences," *Journal of WSCG*, pp. 99–106, 2007.

[14] A. Smolic, R. Sondershaus, N. Stefanoski, L. Vasa, K. Muller, J. Ostermann, and T. Wiegand, "A survey on coding of static and dynamic 3d meshes," *Three Dimensional Television-Capture, Transmission, Display*, pp. 239–312, 2008.

[15] O. K.-C. Au, C.-L. Tai, L. Liu, and H. Fu., "Dual laplacian editing for meshes," *IEEE Transaction on Visualization and Computer Graphics*, vol. 12, no. 3, pp. 386–395, May-June 2006.

[16] H. Fu and C.-L. Tai, "Mesh editing with affine-invariant laplacian coordinates," Hong Kong University of Science and Technology, Tech. Rep. HKUST-CS05-01, 2005.

[17] R. W. Sumner and J. Popovic, "Deformation transfer for triangle meshes," in *ACM SIGGRAPH*, 2004, pp. 399–405.

[18] G. Rong, Y. Cao, and X. Guo, "Spectral mesh deformation," *The Visual Computer*, vol. 24, no. 7-9, pp. 787–796, 2008.

[19] D. L. James and C. D. Twigg, "Skinning mesh animations," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 399–407, 2005.

[20] K. G. Der, R. W. Sumner, and J. Popovi, "Inverse kinematics for reduced deformable models," in *ACM SIGGRAPH*, 2006, pp. 1174–1179.