Facial Motion Capture Editing by Automated Orthogonal Blendshape Construction and Weight Propagation*

Qing Li and Zhigang Deng

Department of Computer Science
University of Houston
Houston, TX, 77204, USA
http://www.cs.uh.edu

Technical Report Number UH-CS-07-12

October 17, 2007

Keywords: Blendshape Animation, Facial Animation, Motion Capture, Data-Driven

Abstract

We present a novel data-driven 3D facial motion capture data editing system by automated construction of an orthogonal blendshape face model and constrained weight propagation, aiming to bridge the popularized facial motion capture technique and blendshape approach. In this work, 3D facial motion capture editing problem is transformed to a blendshape animation editing problem. Given a collected facial motion capture dataset, we construct a truncated PCA space spanned by retained largest eigen-vectors and a corresponding blendshape face model for each anatomical region of the human face. As such, modifying blendshape weights (PCA coefficients) is equivalent to editing their corresponding motion capture sequence. In addition, a constrained weight propagation technique allows animators to balance automation and flexible controls.

*This research has been funded by new faculty research start-up fund at University of Houston.

Facial Motion Capture Editing by Automated Orthogonal Blendshape Construction and Weight Propagation*

Qing Li and Zhigang Deng

Abstract

We present a novel data-driven 3D facial motion capture data editing system by automated construction of an orthogonal blendshape face model and constrained weight propagation, aiming to bridge the popularized facial motion capture technique and blendshape approach. In this work, 3D facial motion capture editing problem is transformed to a blendshape animation editing problem. Given a collected facial motion capture dataset, we construct a truncated PCA space spanned by retained largest eigen-vectors and a corresponding blendshape face model for each anatomical region of the human face. As such, modifying blendshape weights (PCA coefficients) is equivalent to editing their corresponding motion capture sequence. In addition, a constrained weight propagation technique allows animators to balance automation and flexible controls.

Index Terms

Blendshape Animation, Facial Animation, Motion Capture, Data-Driven

I. Introduction

Motion capture has became one of the popularized techniques for acquiring natural human motion with its intrinsic subtlety these days. Due to the costly expense of motion capture, how to efficiently and intuitively edit pre-recorded motion capture data for novel applications has been a hot research topic in computer animation community.

Striking research efforts have been pursued for efficiently editing recorded human body motion capture data; however, there has been relatively little research on facial motion capture data editing. Different from body motion capture data, the facial motion capture data do not have an obvious internal structure, and inverse kinematic models cannot be directly applied to the facial motion capture data editing without considerable efforts.

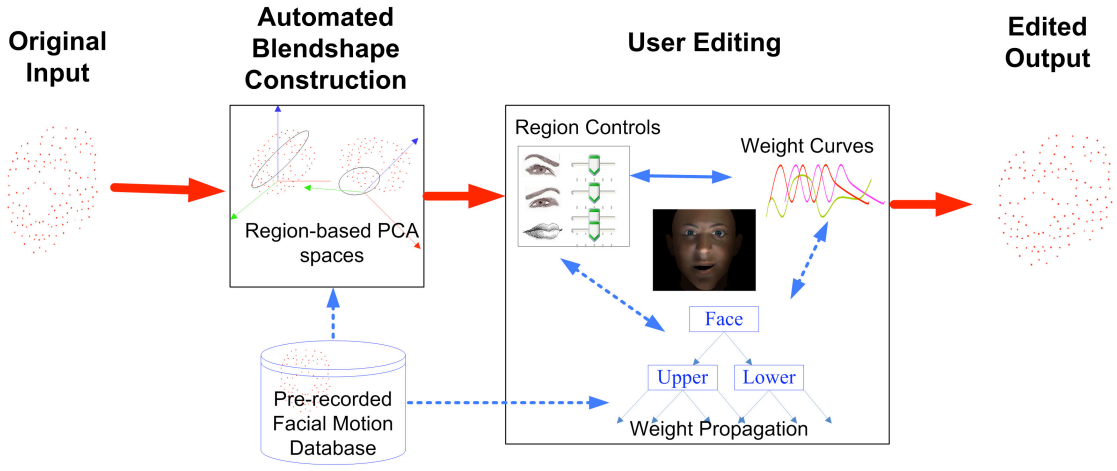

Fig. 1: Schematic overview of this facial motion capture editing system.

In this work, we present a data-driven 3D facial motion capture editing system by automated construction of an orthogonal blendshape face model and a constrained weight propagation. First, given a collected facial motion

*This research has been funded by new faculty research start-up fund at University of Houston.

capture dataset, we perform a region-based PCA decomposition and construct a truncated PCA space spanned by the largest eigen-vectors for each anatomical region of the human face (e.g. left eyebrow). An orthogonal blendshape face model is accordingly constructed as follows: each eigen-vector of an anatomical region corresponds to a blendshape basis, and its PCA coefficient is regarded as its blendshape weight. In addition, the headachy blendshape interference issue [1] that greatly affects the efficiency of the blendshape approach is minimized in our constructed orthogonal blendshape face model. Second, we transform each frame of a new facial motion capture sequence into blendshape weights by projecting it into the above PCA spaces on a per-region basis. As such, modifying the blendshape weights (PCA coefficients) is equivalent to editing the underlying motion capture sequence.

To automatically maintain the naturalness of the 3D face while a facial region is being edited, we extend the motion propagation algorithm [2] to a constrained weight propagation technique that allows animators to optionally specify which blendshape controls (and their weights) should not be affected by later user editing. Figure 1 shows the schematic overview of this 3D facial motion capture editing system.

The major contributions of this work include: (1) it bridges the popularized facial motion capture technique and the blendshape approach that is perhaps regarded as the most commonly employed technique in facial animation practice, and (2) it proposes a new, fully automated technique to construct an orthogonal blendshape face model from motion capture data and minimize the blendshape interference issue that often exists in hand-crafted blendshape models.

To make this work distinguished from previous facial motion editing techniques, here we briefly compare our approach with these methods. Data-driven statistical models [3]–[5] have proved their successes in learning high-level semantic attributes (e.g. expression), however, the applicability of these learned statistical models to regional facial editing has not been established and demonstrated.

Independent Component Analysis (ICA) based expressive facial motion editing technique [6] is essentially performed on a whole-face basis, where each ICA component is not purely local and often affects facial movements non-locally. As such, making local adjustments on specific facial regions is not well addressed in their work. Our approach not only offers flexible editing functions on local facial region and sequence-level basis, but also maximally maintains the global naturalness of the edited 3D face using a constrained weight propagation technique.

Comparing with the method of specifying explicit weights for hand-crafted blendshape models, our approach is more intuitive and efficient: (1) intensive manual efforts are generally required for building a hand-crafted blendshape model from a static 3D face model, even for skilled animator; however, while our approach constructs an orthogonal blendshape model automatically from a static 3D face model. (2) In our approach, input is a recorded facial motion capture sequence, and output is an edited facial motion capture sequence. Mappings between facial motion capture sequences and the weights of our automated blendshape models are seamless in our approach. However, if a hand-crafted blendshape model is used, the animators have to solve the mapping issue between the weights of hand-crafted blendshape models and facial motion capture sequences, which is a challenging issue. The work of [7] is not fully automatic in order to map facial motion capture data to existing, hand-crafted blendshape models.

The work of [8] learns controls through a physically-motivated face segmentation of a given blendshape face model and presents an algorithm to map a facial motion capture sequence to the blendshape face model. However, their approach requires a processed blendshape face model as an input, while our approach does not have this input requirement and adaptively constructs a blendshape face model from a static 3D face model. Furthermore, in order to output edited facial motion capture data (sequences) in [8], a mapping algorithm from the weights of the blendshape model to 3D marker motions is additionally required, which is not addressed in [8].

Our approach can lend itself well to facial animation practice. For example, synthesized facial animations (represented as 3D facial marker motions) by various data-driven facial animation synthesis algorithms [9] may not perfectly meet the needs of animators, and the animators want to further edit them (e.g. exaggerating marker motions on certain facial regions). To our knowledge, there are few existing tools or algorithms for this editing purpose. Our approach can be efficiently used for this application.

The remainder of this paper is organized as follows. A review of related work is given in Section II. Section III describes facial motion data acquisition and preprocessing. Section IV describes how to construct region-based PCA spaces and a corresponding orthogonal blendshape face model. Section V describes the constrained weight propagation algorithm that is used to automatically maintain the naturalness of the edited facial motion capture sequences. Section VI details how to perform sequence-level editing operations based on the constructed blendshape

model. Experimental results are reported in Section VII. Finally, in Section VIII, concluding remarks are given and future research directions are discussed.

II. Related Work

Geometrically deforming 3D face models is a natural way to generate facial animations. However, manually sculpting key faces every two to three frames is a painstaking process, and repeated manual work is needed to produce realistic facial animations with fine details. Due to its efficiency and simplicity, the blendshape approach has been widely used for keyframing facial animations, and significant research efforts were attempted. For example, some recent efforts attempt to reduce blendshape interference by selected motion attenuation [1] and map facial motion capture sequences to blendshape face models [7].

Essentially, the above approaches (geometric deformation or blendshape) are designed to simultaneously move and edit a group of relevant vertices of face geometry. However, human facial motion is the consequence of subtle skin deformation supported by the relaxation/contraction of tens of hidden facial muscles, and motions of different facial regions are intrinsically correlated each other. Local-region based or global deformation approaches often need users to switch editing operations on different facial regions in order to produce 3D facial animations with fine details. Furthermore, it is extremely difficult to judge which facial pose is closer to a real human face. A number of data-driven facial animation editing techniques [1], [2], [4]–[6], [8] were proposed to address this issue by exploiting rich correlations enclosed in collected dataset. Chuang et al. [4] apply the bilinear model to transform an input 2D facial animation sequence to a new one with a different expression. Most recently the multilinear model was also successfully applied for the purpose of facial animation transferring [5]. Cao et al. [6] apply the Independent Component Analysis (ICA) algorithm to recorded expressive facial motion capture data, and then interpret certain ICA components as expression or speech components, respectively. Based on a blendshape representation for 3D face models, Joshi et al. [8] present an interactive tool to edit 3D face geometry by learning controls through a physically-motivated face segmentation. A rendering algorithm for preserving visual realism in this editing was also presented in their approach. The geometry-driven face image editing technique [2] generates expression details on 2D face images by constructing a PCA-based hierarchical face representation from a selected number of training 2D face images. When users move one or several points on the 2D face image, the movements of other facial control points are automatically computed by a motion propagation algorithm. Most recently Chang and Jenkins [10] present a 2D sketch interface for posing 3D faces. In their work, users can intuitively draw 2D strokes that are used to search for the optimal pose of the face.

III. Facial Motion Capture

In the data collection stage, we captured high-quality 3D facial motions using a VICON motion capture system (the left panel of Fig. 2). An actress with 102 markers on her face was directed to speak a delicately designed corpus (composed of hundreds of sentences) four times, and each repetition was spoken with a different facial expression. In this data recording, total four basic facial expressions are considered: neutral, happiness, anger and sadness.

The facial motion data were recorded with a 120 Hz sampling rate. We collected approximately 135 minutes motion capture data. Because of tracking errors caused by rapid large head movement and the removal of unnecessary facial markers, we used only 90 of 102 markers for this work. The 90 markers were fully tracked. After data capture, we normalized the facial motion data by removing head motion from motion capture frames as follows. All the markers were first translated so that a specific marker was at the local coordinate center of each frame, and then the Singular Value Decomposition (SVD) based method [11] was used to calculate head motion. The middle panel of Figure 2 shows the 102 facial marker layout and the 90 kept markers.

IV. Region-based Principal Component Analysis

In this section, we describe how to apply principal component analysis on a per-region basis to construct an orthogonal blendshape face model.

As mentioned in Section III, each facial motion capture frame is composed of 90 facial markers. 3D positions of all these markers (of a facial motion capture frame) are concatenated in certain order to form a 270 dimensional facial motion vector. If a single truncated PCA space is constructed for these facial motion vectors, in order to keep

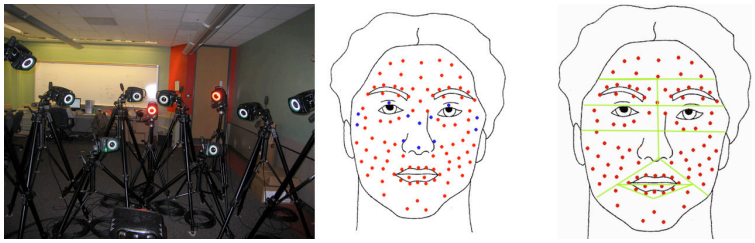

Fig. 2: The left shows a facial motion capture system, the middle panel shows the used facial marker layout, and the right panel shows the region-dividing scheme used in this work. In the middle panel, blue and red points together represent the total 102 captured markers, and the red points are the ninety markers used for this work.

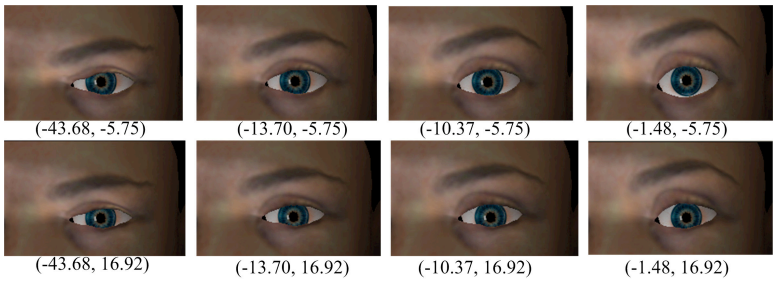

Fig. 3: Illustration of the first and second most dominant PCA coefficient controls for left eyebrow region. Duples in the figure show the PCA coefficients. As we can see from this figure, when the first PCA coefficient is increased, the left eyebrow is raised.

more than 90% of the variation, their retained dimensionality is still higher than twenty. Furthermore, since PCA is essentially a global transformation/reduction, there are no explicit and clear correspondences between PCA eigen-vectors and facial region movements. In other words, if users want to edit a specific facial region, it is extremely difficult for them to know how and which PCA coefficients should be adjusted. As such, facial motion editing in the single truncated PCA space is not intuitive for animators.

We adopt an anatomical segmentation scheme to divide the whole face (so its facial markers) into ten different regions (the right panel of Fig. 2). The ten facial regions are *forehead, left eyebrow, right eyebrow, left eye, right eye, left cheek, right cheek, upper lip, lower lip,* and *jaw*. For the markers in each of the ten regions, we construct a separate PCA space. We experimentally set the reduced dimensionality to three, because reducing dimensionality to three keeps more than 90% of the variation of the motions of every facial region.

In our experiments, we found that because PCA is applied to the motions of a specific facial region, increasing or decreasing the weights of its largest three eigen-vectors often corresponds to perceptual movements of the facial region. For example, when users increase the weight (PCA coefficient) of the largest eigen-vector for the left eyebrow region, the left brow is accordingly raised. Figure 3 and 4 show two examples of how these retained eigen-vectors approximately correspond to perceptual movements for certain facial regions.

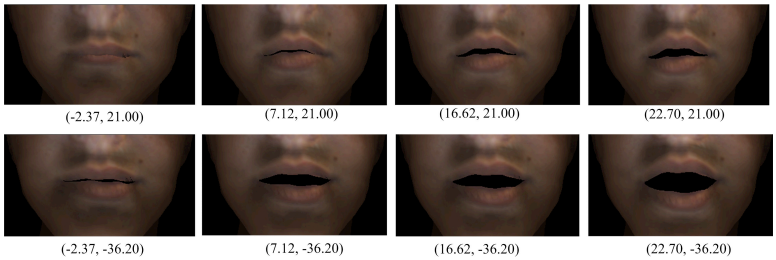

Fig. 4: Illustration of the first and second most dominant PCA coefficient controls for the lower lip region. Duples in the figure show the PCA coefficients. As we can see from this figure, when the first PCA coefficient is increased, the lower lip is more opener.

A. Automated Orthogonal Blendshape Construction

From the above region-based PCA decompositions, an orthogonal blendshape face model with 30 blendshape basis (each region has three blendshape basis and total ten regions) is automatically constructed as follows: a retained eigen-vector of any facial region corresponds to a blendshape basis, and its projected PCA coefficient is its weight. Eq. 1 describes the constructed blendshape model. Changing these retained PCA coefficients (equivalent to moving the sliders of blendshape weights) leads to the change of the underlying facial motion capture frame. In the remaining description of this paper, (blendshape) weights and PCA coefficients, blendshape basis and region-based PCA eigen-vectors, are used interchangeably.

$$MrkMotion = MeanMotion + \sum_{i=1}^{10}\sum_{j=1}^{3} C_{i,j} * EigV_{i,j} \quad (1)$$

The blendshape approach often suffers from the blendshape interference problem due to non-orthogonal blendshape basis [1]. In this work, the constructed blendshape basis are orthogonal, because motion markers in different facial regions do not intersect each other, and retained eigen-vectors in a facial region are orthogonal each other. As such, the blendshape interference is avoided or at least minimized in our system.

Now given a facial motion capture sequence as the input of an editing task, we project every frame of this sequence to the region-based truncated PCA spaces and obtain corresponding weights for all blendshape basis. In time domain, these continuous weight curves can be further edited by animators (Section VI).

V. Automated Weight Propagation

In this section, we describe how the weights of other blendshape basis are automatically adjusted when one or several blendshape weights are manipulated by users, in order to maximally maintain the global naturalness of the edited 3D face.

As illustrated in Figs 3 and 4, changing the weights of individual blendshape basis directly affects only the movement of certain facial regions. However, as discussed in Section II, the motions of different facial regions are intrinsically correlated each other. Based on the hierarchical PCA based motion propagation algorithm proposed in [2], we develop a constrained weight propagation technique to adjust weights automatically while offering flexible user controls.

We use a hierarchical principal component analysis [2] for blendshape weight propagation. Initially, we divide the face into ten leaf nodes (each corresponds to a divided region in the right panel of Fig. 2), and then construct two intermediate nodes (the upper face and the lower face). Finally, the whole face is considered as the root of this hierarchy. Fig. 5 shows the constructed hierarchical face structure.

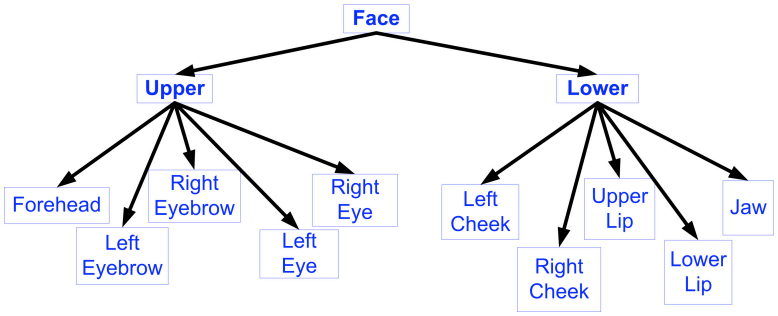

Fig. 5: Illustration of the face hierarchy for weight propagation.

For each node in Fig. 5, we construct a truncated PCA space. In this work, for the root node (the whole face), the most dominant twenty eigen-vectors (corresponding to the largest twenty eigen-values) are retained. For the two intermediate nodes (the upper face and the lower face), the most dominant ten eigen-vectors are retained. For the ten leaf nodes, the most dominant three eigen-vectors are retained.

The rules for this weight propagation procedure are 1) it chooses to move upward prior to downward in the hierarchy and 2) it visits each node only once. The propagation works as follows: first, when users change the weight of a blendshape basis, the propagation starts at the lowest hierarchy which is one of the ten left-nodes. Then,

it propagates upward to the middle of the hierarchy (the upper or the lower face node). Then, it moves upward to the root node (the entire face). After that, it moves downward again to the middle node that it has not visited yet and it keeps going upward and downward until all nodes are visited. For each node it visits, it projects the facial markers contained in the node to the PCA subspace spanned by the retained principal components of the node. In other words, the projection is the best approximation of the propagated motions in the PCA subspace of the node. For more details about this propagation algorithm, please refer to [2]. Figures 6 and 7 show two examples of how this weight propagation technique automatically adjusts weights to make the edited 3D face more natural.

To make this section more readable, we briefly summarize basic steps of the motion propagation algorithm [2] in Algorithm 1. In the following algorithm description, F represents any node in the hierarchy, δV represents the displacement vector of all markers, and $Proj(\delta V, F*)$ denotes the projection of the $F*$ part of δV to the truncated PCA space of the node $F*$.

Algorithm 1 WeightPropagation

Input: $F*$, the selected node in the hierarchy.

set h to the hierarchy level of $F*$.;
if *hasBeenProcessed*($F*$) **then**
    return;
end if
Compute *Proj*(δV,$F*$);
Update δV with *Proj*(δV,$F*$);
Set *hasBeenProcessed*($F*$) to be true;
for $\forall\ F \subseteq (level(F) = h - 1$ and $F \cap F* = NonEmpty)$ **do**
    WeightPropagation(F);
end for
for $\forall\ F \subseteq (level(F) = h + 1$ and $F \cap F* = NonEmpty)$ **do**
    WeightPropagation(F);
end for

Fig. 6: Illustration of the weight propagation results. The left panel shows the initial face (before editing), the middle panel shows the edited 3D face (the editing operation is the lower lip stretching, just before the weight propagation starts), and the right panel shows its propagated results (after weight propagation). Comparing the middle panel with the right panel, we can clearly see the weight propagation process properly adjusted the weights to make the whole face look more natural.

A. Constrained Weight Propagation

To automatically maintain the naturalness of the face, the above weight propagation algorithm affects all the facial regions when the weights (controls) of a facial region are being edited. However, in some scenarios, automated propagation results may not be exactly what users need, for example, they want to exaggerate the expressiveness of the edited 3D face or keep current configurations of certain facial regions as stationary as possible (not spoiled by later editing). Here is a specific example: animators put the mouth to the right shape (e.g. synchronized with a phoneme), but they want to edit the expressiveness of left/right cheeks to make the face look happier. In this case,

Fig. 7: Illustration of the weight propagation results. The left panel shows the initial face (before editing), the middle panel shows the edited 3D face (the editing operation is the right eyebrow raising, just before weight propagation starts), and the right panel shows the propagated results (after weight propagation). Note that after weight propagation, not only the right eyebrow is properly adjusted, the mouth is automatically adjusted accordingly - from fully close to slightly open.

if they move cheek controls, the above weight propagation algorithm will automatically adjust the mouth shape, which is not desired by the animators.

In this system, we introduce a constrained weight propagation technique to achieve this flexibility. It works as follows. First, animators specify which facial region(s) are the constrained regions (e.g., left/right eyebrows) that will not be affected by later weight propagations, and based on the pre-defined mappings between markers and regions, special flags are set to the markers of the constrained facial regions. Finally, we need to modify the above weight propagation procedure (Algorithm 1) accordingly: the leaf nodes of the constrained facial regions (in this example, left/right eyebrow nodes) are exempted from processing, and when their parent nodes (in this example, the upper face node and the whole face node) are processed ($Proj(\delta V, F*)$), markers with special flags will not be updated.

The constrained weight propagation provides an accessible tool for animators to balance the trade-off between user controls and automation. Figures 8 and 9 show two specific examples of the constrained weight propagation. As shown in Fig. 8, its left panel shows an original face (deformed based on a 3D facial motion capture frame), and its middle panel shows the edited results when users raise the left/right cheeks. Notice that in the middle panel, the mouth is automatically opened by the weight propagation procedure without constraints. Its right panel shows the result of the constrained weight propagation - three regions (the upper lip, the lower lip, and jaw) are set as the constrained regions. Notice that in the right panel, the mouth and jaw are perfectly kept while cheeks are moved to the right place.

Fig. 9 shows another application of constrained weight propagation. Its left panel shows an original face, and its middle panel shows the edited results when users close the mouth. Notice that in the middle panel, the eyebrows/eyes are affected (from anger-like expression to neutral-like expression) by the weight propagation procedure without constraints. Its right panel shows the result of the constrained weight propagation - five facial regions (the forehead, the left eyebrow, and the right eyebrow, the left eye, and the right eye) are set as the constrained regions. Notice that in the right panel, the eyebrows/eyes are perfectly kept while the mouth is moved to the expected close position.

Fig. 8: The first example of the constraint weight propagation application. The left panel shows an original face, the middle panel shows the edited result with weight propagation, and the right panel shows the edited result with constrained weight propagation.

VI. Motion Sequence Editing

In this section, we describe how to perform sequence-level facial motion capture editing. Besides modifying blendshape weights frame by frame, animators can also conveniently edit a sequence of frames by directly manipulating corresponding weight curves: a Catmull-Rom spline is first fitted based on the weight sequence of each blendshape control, and then the animators can edit control points of the Catmull-Rom spline. The Catmull-Rom spline is chosen in this work mainly due to its smooth interpolation and local control properties. This motion sequence editing tool can be effectively used for keyframing animations.

Fig. 9: The second example of the constraint weight propagation application. The left panel shows an original face, the middle panel shows the edited result with weight propagation, and the right panel shows the edited result with constrained weight propagation.

VII. Results and Experiments

We developed this 3D facial motion capture data editing system using VC++ on Microsoft Windows XP system. Figure 10 shows a snapshot of the running system. As shown in Fig. 10, its interface is composed of four panels: the top-left panel is a blendshape control window where users can select a facial region and then move its weight sliders, the bottom-left panel is the edited results without weight propagation, the top-right panel shows the original facial motion capture sequence, and the bottom-right panel shows the edited facial motion sequence (the final output) after constrained weight propagation is applied.

In three of the above panels (bottom-left, top-right, and bottom-right), facial motion capture data can be displayed in two different modes: point rendered (each marker is rendered as a 3D point) or a deformed 3D face. In the deformed 3D face display, a feature-point based deformation technique [12] is used to deform a static 3D face model based on 3D facial motion capture data.

We have conducted numerous facial motion capture data editing experiments using our system, including improving the animation quality of some initial synthesized or captured facial motion sequences, and exaggerating the expressiveness of some existing facial motion sequences.

VIII. Discussion and Conclusions

In this paper, we present a 3D facial motion capture editing system by automated orthogonal blendshape construction and constrained weight propagation. The success of this data-driven editing system is mainly due to the fact that it exploits rich correlations enclosed in a pre-recorded facial motion capture dataset. Based on a collected facial motion capture dataset, a region-based principal component analysis is applied to build an orthogonal blendshape face model. As such, we formalize the 3D facial motion capture editing problem to a blendshape animation editing problem. Our work bridges the popularized facial motion capture technique and the blendshape approach that is perhaps regarded as the most commonly employed technique in facial animation practice.

The traditional blendshape approach often suffers from the blendshape interference problem due to the non-orthogonal blendshape basis. Our system automatically avoids or at least minimizes this issue by constructing orthogonal blendshape basis. Furthermore, the employed constrained weight propagation provides powerful and flexible controls that minimize user interventions. When animators do local changes on a facial region, motions of other facial regions are self-configured based on the hierarchical principal component analysis based weight propagation. To offer flexible controls in the editing procedure, the constrained weight propagation empowers the animators further editing without spoiling previous editing efforts.

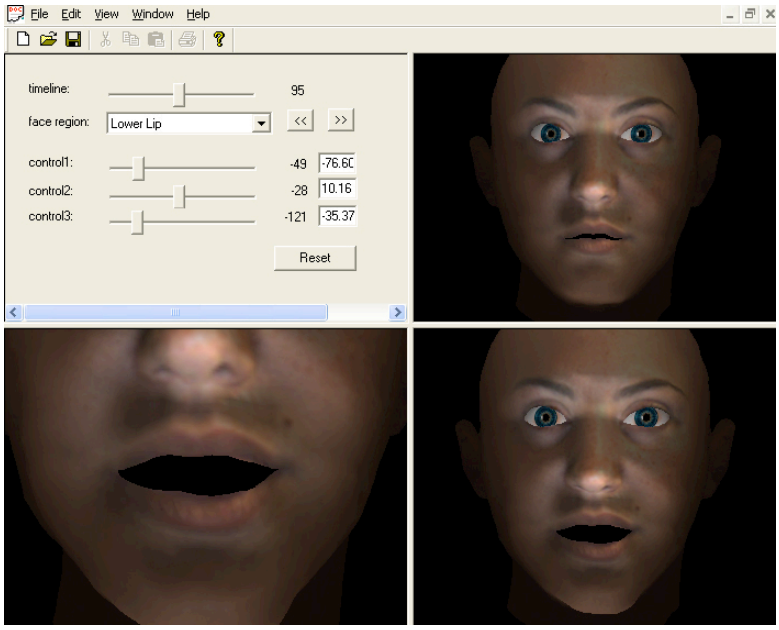

Fig. 10: A snapshot of the running facial motion capture editing system.

In the future, we plan to look into how to extend this data-driven system for editing and animating pre-designed blendshape face models, and its key issue would be the mappings between pre-designed blendshape face models and automatically constructed ones.

REFERENCES

[1] J. Lewis, J. Mooser, Z. Deng, and U. Neumann, "Reducing blendshape interference by selected motion attenuation," in *Proceedings of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3DG)*, 2005, pp. 25–29.

[2] Q. Zhang, Z. Liu, B. Guo, and H. Shum, "Geometry-driven photorealistic facial expression synthesis," in *Proc. of Symposium on Computer Animation*, 2003, pp. 177–186.

[3] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3d faces," in *Proc. of ACM SIGGRAPH'99*, 1999, pp. 187–194.

[4] E. S. Chuang, H. Deshpande, and C. Bregler, "Facial expression space learning," in *Proc. of Pacific Graphics'2002*, 2002, pp. 68–76.

[5] D. Vlasic, M. Brand, H. Pfister, and J. Popović, "Face transfer with multilinear models," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 426–433, 2005.

[6] Y. Cao, P. Faloutsos, and F. Pighin, "Unsupervised learning for speech motion editing," in *Proc. of Symposium on Computer Animation*, 2003.

[7] Z. Deng, P. Chiang, P. Fox, and U. Neumann, "Animating blendshape faces by cross mapping motion capture data," in *Proceeding of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, March 2006, pp. 43–48.

[8] P. Joshi, W. Tien, M. Desbrun, and F. Pighin, "Learning controls for blend shape based realistic facial animation," in *Symposium on Computer Animation*, 2003, pp. 35–42.

[9] Z. Deng and U. Neumann, "eFASE: Expressive facial animation synthesis and editing with phoneme-level controls," in *Proc. of Symposium on Computer Animation*. Vienna, Austria: Eurographics Association, 2006.

[10] E. Chang and O. Jenkins, "Sketching articulation and pose for facial animation," in *Proc. of Symposium on Computer Animation (SCA)*, 2006.

[11] C. Busso, Z. Deng, U. Neumann, and S. Narayanan, "Natural head motion synthesis driven by acoustic prosody features," *Computer Animation and Virtual Worlds*, vol. 16, no. 3-4, pp. 283–290, July 2005.

[12] S. Kshirsagar, S. Garchery, and N. M. Thalmann, "Feature point based mesh deformation applied to mpeg-4 facial animation," in *Proc. Deform'2000*, November 2000, pp. 23–34.