Exploring Nonlinear Relationship of Blendshape Facial Animation

Xuecheng Liu¹,², Shihong Xia¹, Yiwen Fan¹,² and Zhaoqi Wang¹

¹Institute of Computing Technology, Chinese Academy of Sciences
²Graduate School of the Chinese Academy of Sciences

Abstract

Human face is a complex biomechanical system and nonlinearity is a remarkable feature of facial expressions. However, in blendshape animation, facial expression space is linearized by regarding linear relationship between blending weights and deformed face geometry. This results in the loss of reality in facial animation. In order to synthesize more realistic facial animation, aforementioned relationship should be nonlinear to allow the greatest generality and fidelity of facial expressions. Unfortunately, few existing works pay attention to the topic about how to measure the nonlinear relationship. In this paper, we propose an optimization scheme that automatically explores the nonlinear relationship of blendshape facial animation from captured facial expressions. Experiments show that the explored nonlinear relationship is consistent with the nonlinearity of facial expressions soundly and is able to synthesize more realistic facial animation than the linear one.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1. Introduction

Creating realistic facial animation is one of the greatest challenges in computer graphics. Human face is a complex biomechanical system, so it is very hard to simulate facial motion accurately. Furthermore, human are so sensitive to facial expression that even a tiny flaw of facial animation could hardly escape from our attention. Despite realistic facial animation synthesis has gone a long way and many useful methods and tools have been developed, there is still space for improvement.

In both research and industry domains, blendshape method is widely used to synthesize facial animation due to its efficiency and intuition. In blendshape animation, the space of potential faces is a linear subspace defined by a set of key shapes, which are linearly blended to synthesize facial expressions. It is illustrated as follows.

$$E = \sum_{i=1}^{N_{bs}} w_i e_i, \quad 0 \leq w_i \leq 1$$

In above equation, facial expression $E$ is expressed as a vector of vertexes coordinates increments relative to the ones of neutral expression model. It is similar to key shape $e_i$. $N_{bs}$ represents the number of key shapes. $w_i$ is blending weight. In order to avoid meaningless facial expressions, blending weights are generally restricted in the closed interval $[0,1]$.

There is a drawback in synthesizing realistic facial animation through linear blending of key shapes. Human face is a complex biomechanical system composed by skeletons, muscles, flesh, skin and so on. Therefore, nonlinearity is a remarkable feature of facial expressions. However, this feature is ignored in blendshape method, and it inevitably results in the loss of reality in facial animation. Taking jaw motion as example, while adjusting the blending weight of key shape “Mouth Opening” gradually, the vertex on chin should move along an approximate arc centered at temporomandibular joint (Figure ??a) in reality. However in blendshape facial animation, the orbit is linearized and poorly simplified (Figure ??b). Another example is “Eyes Closing”: the trail of eyelid motion which should correspond with the contour of cornea (Figure ??a) is simplified as a section of line (Figure ??b) in blendshape facial animation.

As proposed by Pighin and Lewis, the relationship between blending weights and deformed face geometry should
be nonlinear to allow the greatest generality and fidelity of facial expressions [2]. However, the relationship is linearized as \( w \cdot e \) in blendshape facial animation. Through adopting nonlinear relationship functions \( f_i(w_i) \) that consist with the nonlinearity of facial expressions, it is able to synthesize more realistic facial animation (Equation 2). Taking jaw motion for example, the orbit of vertex on chin can be well reconstructed through setting \( f_i(w_i) \) as appropriate trigonometric functions (Figure 1c). Nonlinear relationship functions also apply to the case of "Eyes Closing" (Figure 1c).

\[
E = \sum_{i=1}^{N_b} f_i(w_i), \quad 0 \leq w_i \leq 1
\]  

Unfortunately, to our knowledge, few researchers have discussed the topic of exploring the nonlinear relationship of blendshape facial animation. Generally, given an arbitrary facial model, it is very difficult to construct nonlinear relationship functions \( f_i(w_i) \) due to the complex biomechanics of human face. A potential solution is physical simulation of anatomical face model such as techniques of [2] and [2]. However it is not easy to construct the accurate anatomy of the given face. Moreover, it is unexplored how to transform blending weights to facial muscles contract. Therefore, physics-based simulation is not suitable for exploring nonlinear relationship functions.

In this paper, we propose an optimization scheme that automatically explores the nonlinear relationship between blending weights and deformed face geometry of blendshape facial animation. The essence of proposed optimization is searching the nonlinear relationship functions that match captured facial expressions.

Through our scheme, the explored nonlinear relationship consists with the nonlinearity of facial expressions and is able to synthesize more realistic facial animation than the linear one. In the experiments, we firstly show the nonlinearity of facial expressions brought by the explored nonlinear relationship functions. Then, we contrast the facial animations which are respectively synthesized by linear relationship functions and the explored nonlinear ones. The results are assessed through both user study and quantitative evaluation. At last, we discuss the computational efficiency of the algorithms presented in this paper.

The remainder of this paper is organized as follows: Section 2 reviews related works of blendshape facial animation. Section 3 shows facial motion capture. Section 4 illustrates the optimization scheme in detail. Section 5 shows the re-
2. Related work

Since blendshape facial animation was firstly proposed by Parke’s pioneer work in [?] and [?], it has been widely used in both research and industry domains of facial animation due to its intuition and convenience. In this section, we will specifically review recent work related to blendshape facial animation.

Principle components analysis (PCA) based methods construct basic shapes of facial animation through dimension reduction of facial expression samples. Its primary advantage is the rigid orthogonality of the constructed space. The disadvantage is that the principle components don’t possess visual intuition and are not suitable for manual manipulation. Aiming at this problem, Chuang has designed three schemes that select key shapes from facial expression samples based on PCA results [?]; Li has used region-based PCA to automatically construct local orthogonal space from motion capture data [?].

Since facial action coding system (FACS) was firstly proposed by Ekman in [?], it has been popular in facial animation. According to FACS, facial expressions can be presented as combinations of distinct Action Units. Each Action Unit intuitively corresponds with a basic shape of facial animation. In specific applications, reduced or modified versions of FACS also animate face well with improved usability [?] [?] [?].

In light of the importance of facial animation, MPEG-4 has specified criterion of facial animation synthesize for network transmission [?]. In essence, MPEG-4 defines a piece linear blendshape facial animation. MPEG-4 has defined 68 parameters (FAP) to animate face. In these parameters, FAP1 and FAP2 depict fourteen static visemes and six basic facial expressions respectively, and the other FAPs defined 66 blending weights that enable synthesizing arbitrary facial expressions. In MPEG-4 facial animation, it is a vital issue that how to construct the facial animation table (FAT) that defines rules of facial motion. Kshirsagar, Fratarcangeli and Jiang have made their efforts to explore FAT [?] [?] [?].

Besides that, other works also have strived to construct linear space of facial expressions. For example, Joshi has proposed an automatic physically-motivated scheme that segments blendshapes into smaller regions [?]. Cao and Shin have used independent components analysis (ICA) to extract a set of meaningful parameters that were independent to each other as much as possible [?] [?].

3. Facial motion capture

A passive optical motion capture system is employed to capture facial motion. In this system, twelve infrared cameras are used to record coordinates of the optical markers attached on actor’s face, as shown in figure ??.

![Image of facial motion capture setup]

Figure 2: Illustration of facial motion capture. The left image shows the infrared cameras arrangement. The right one shows optical markers distribution on actor’s face, the smaller markers are used to reveal facial expressions and the four larger ones are used as reference of computing transformation matrix of head.

We process the captured facial expressions as follows. At first, we transfer the marker coordinates from world coordinate system to the facial local system. We take the larger optical markers which are stationary to actor’s head as reference and compute the transformation matrix of head using absolute orientation algorithm [?]. After that, we retarget the captured expressions to the facial model using [?] [?], due to that the facial model is different from actor’s face. Through above retargeting, the captured expressions from the actor can be used to explore the nonlinearity of different facial models. It makes the proposed method more applicable. At last, the neutral expression model is subtracted from each captured facial expression to obtain the marker coordinates increments. For ease of representation, in the latter section of paper, we refer the captured facial expressions as the data processed above.

In essence, the proposed scheme in this paper is independent of the specific data form of captured facial expressions. Coordinates of sparse markers, points cloud, retargeted or copied expressions, are all applicable.

Each captured facial expression is presented as following transversal vector:

\[ M_k = (M_{k,1,1}, M_{k,1,2}, \cdots, M_{k,Na,x}, M_{k,Na,y}, M_{k,Na,z}) \]

In above vector, \( M_k \) is composed of markers coordinates increments, \( k \) is captured expression index, \( Na \) is the number of markers. For ease of illustration, we rewrite above vector as follows:

\[ M_k = (M_{k,1}, M_{k,2}, \cdots, M_{k,j}, \cdots, M_{k,Na\times3}) \]

In this vector, \( j \) indicates the \( j \)th component of the vector, and \( Na \times 3 \) is the vector dimension.

We further present the captured facial expressions as the
We have captured two groups of facial expressions. The first group is used to analyze and parameterize the nonlinear relationship functions $f_i(w_i)$. To do that, the actor acts several basic facial actions such as “Open mouth”, “Close eyes”, “Raise lip corners” and so on. The motion capture data records the facial deformation during performing above basic actions. The second group is used to optimize the nonlinear relationship functions $f_i(w_i)$. We will optimize the nonlinear relationship functions from this group of captured facial expressions. For this purpose, the actor tries to contract all his facial muscles to perform enough facial expressions which are able to span the facial expression space.

4. Explore the nonlinear relationship functions

We explore the nonlinear relationship functions of blendshape facial animation from captured facial expressions through optimization technique. The scheme is illustrated in figure ??.

\[
M = \begin{bmatrix}
M_{1,1} & M_{1,2} & \cdots & M_{1,N_1} \\
M_{2,1} & M_{2,2} & \cdots & M_{2,N_2} \\
\vdots & \vdots & \ddots & \vdots \\
M_{N_1,1} & M_{N_1,2} & \cdots & M_{N_1,N_2}
\end{bmatrix}
\]  

(3)

In the matrix of equation ??, each row indicates a captured facial expression, and $N_f$ is the expressions number.

In above scheme, we firstly analyze and parameterize the nonlinear relationship functions in section 4.1. It is done through analysis of the first group expressions. Then, in section 4.2, we optimize the nonlinear relationship functions from the second group expressions. In the optimization, the parameters of nonlinear relationship functions are regarded as optimization variables. The captured facial expressions only record the motion of sparse markers, and above explored nonlinear relationship functions are only the ones of sparse markers. Therefore, at last, we expand the nonlinear relationship functions from sparse markers to facial model through Radial Basis Function interpolation in section 4.3.

4.1. Analyze and parameterize the nonlinear relationship functions

We analyze and parameterize the nonlinear relationship functions from the first group of captured facial expressions which record the facial geometry deformation during performing basic facial actions. For each facial action, we firstly selected out the expression $M_{\text{max norm}}$ which has maximum norm. It is regarded as the extreme expression of this facial action. Then, the others expressions are projected onto the extreme expression as illustrated in equation ??, and it forms samples of $(w_{ik}, M_k)$. In equation ??, $M_k$ is a transversal vector projected onto $M_{\text{max norm}}$, and $w_{ik}$ is the projection weights. Some representative samples of $(w_{ik}, M_{kj})$ are shown in figure ??a, where $M_{kj}$ is the $j$th component of expression $M_k$. They prove that the relationship between blending weights and deformed facial geometry is definitely nonlinear.

\[
w_{ik} = \arg \min \| M_k - w_{ik}M_{\text{max norm}} \|, 0 \leq w_{ik} \leq 1
\]

(4)

We choose cubic polynomials as the form of nonlinear relationship functions $f_i(w_i)$. It is based on the following considerations. Firstly, polynomials contribute to the computational efficiency of proposed optimization scheme in this paper, it will be illustrated in section 4.2. Base on that, cubic polynomials are further selected because they can represent the nonlinearity of facial expressions with fewer polynomial coefficients. They fit the $(w_{ij}, M_{kj})$ samples well, the fitting results are shown in figure ??b.

The cubic polynomial used in this paper is expressed as follows.

\[
f(w) = aw^3 + bw^2 + cw
\]

The constant item of above cubic polynomial is left out because zero blending weights correspond with neutral facial expression. The cubic polynomial is parameterized by the polynomial coefficients $a$, $b$ and $c$.

We set a cubic polynomial for each $f_{ij}(w_i)$ which is the $j$th component of relationship vector function $f_i(w_i)$, that is, polynomial coefficients $a_{ij}, b_{ij}, c_{ij}$ are used to parameterize $f_{ij}(w_i)$. We don’t get above parameters from the fitting
In this optimization, variables are composed of parameters of nonlinear relationship functions $a_{i,j}$, $b_{i,j}$, $c_{i,j}$ and blending weights of synthesized facial expressions $w_{i,k}$. Due to the large amount of optimization variables (more than 30,000 in our experiments), minimizing $E_{fitting}$ in global variables space results to overfitting. It means that the numerical optimal nonlinear relationship functions are able to synthesize captured facial expressions most accurately, but their visual meanings miss. The overfitted nonlinear relationship functions are not the desired ones, and the overfitted results will be illustrated in experiments.

In order to avoid overfitting, we restrict the optimization variables in an appropriate subspace rather than global space. In the application of blendshape facial animation, users usually design a set of key shapes $e_i$ to animate face. These shapes have specific visual meanings, such as facial action units, basic expressions or visemes. In the optimization, other than only being able to synthesize captured facial expressions accurately, the desired nonlinear relationship functions $f_i(w_i)$ should also preserve the visual meanings of their corresponding key shapes $e_i$. It is vital in blendshape facial animation.

In this paper, the subspace of optimization variables is established as follows. Firstly, construct a set of key shapes $e_i$ of blendshape facial animation according to the specific application. In our experiments, given the facial model, we have constructed 29 key shapes of basic facial action units [?] and visemes. Then, we rewrite the linear relationship functions $w_i e_i$ as nonlinear forms by setting $a_{i,j} = 0$, $b_{i,j} = 0$ and $c_{i,j} = e_{i,j}$. That is, the nonlinear form of $w_i e_i$ is $f_{i,j}(w_i) = e_{i,j} w_i$. In above formulation, $e_{i,j}$ and $f_{i,j}(w_i)$ are the $j$th components of key shape $e_i$ and nonlinear relationship function $f_{i,j}(w_i)$ respectively, and $a_{i,j}$, $b_{i,j}$ and $c_{i,j}$ are parameters of $f_{i,j}(w_i)$. At last, we restrict parameters $a_{i,j}$, $b_{i,j}$ and $c_{i,j}$ in the intervals $A_{i,j}$, $B_{i,j}$ and $C_{i,j}$, which are neighborhoods of 0, 0 and $e_{i,j}$ respectively. Also, we restrict blending weights $w_i$ in the closed interval $[0, 1]$ to avoid meaningless facial expressions.

From above, the complete optimization is expressed as follows:

$$\min_{w_{i,k}, a_{i,j}, b_{i,j}, c_{i,j}} \sum_{k=1}^{N_f} \sum_{i=1}^{N_f} \left| M_k - \sum_{i=1}^{N_f} f_i(w_{i,k}) \right|^2$$

subject to:

$$0 \leq w_{i,k} \leq 1,$$
$$a_{i,j} \in A_{i,j}, b_{i,j} \in B_{i,j}, c_{i,j} \in C_{i,j}$$

Above optimization is a bound-constraint nonlinear least
square one. It is difficult to directly solve due to large amount of variables. Aiming at efficient solving, we propose a two steps iteration optimization algorithm, as shown in figure 5.

1. **Initialization**
   - \( w_{i,j} = 0; \quad a_{i,j} = 0; \quad b_{i,j} = 0; \quad c_{i,j} = e_{i,j} \)

2. **Optimize \( E_{fitting} \) through \( w_{i,j} \)**
   - Minimize \( E_{fitting} \), subject to \( 0 \leq w_{i,j} \leq 1 \)

3. **Optimize \( E_{fitting} \) through \( a_{i,j}, b_{i,j}, c_{i,j} \)**
   - Minimize \( E_{fitting} \), subject to \( a_{i,j} \in A_{i,j}, b_{i,j} \in B_{i,j}, c_{i,j} \in C_{i,j} \)

4. If \( E_{fitting} \) converges, then stop; else jump back to step 2 for a further loop.

The flow of this algorithm can be illustrated as follows:

1. Initialize optimization variables by setting \( w_{i,k} = 0, a_{i,j} = 0, b_{i,j} = 0 \) and \( c_{i,j} = e_{i,j} \).
2. Minimize \( E_{fitting} \) through blending weights \( w_{i,k} \) of synthesized expressions. That is, to each captured facial expression \( M_k \), we solve following optimization:
   \[
   \min_{w_{i,k}} \left( M_k - \sum_{i=1}^{N_0} f(w_{i,k}) \right)^2 \\
   \text{subject to} : 0 \leq w_{i,k} \leq 1
   \] (6)

   This optimization is a nonlinear least square one but with a small amount \( (N_{0,29} \text{ in our experiments}) \) of variables. Therefore, it can be solved efficiently.
3. Minimize \( E_{fitting} \) through parameters of nonlinear relationship functions \( a_{i,j}, b_{i,j} \) and \( c_{i,j} \). It can be expressed as the following optimization.
   \[
   \min_{a_{i,j},b_{i,j},c_{i,j}} \left( M_k - \sum_{i=1}^{N_0} f(w_{i,k}) \right)^2 \\
   \text{subject to} : a_{i,j} \in A_{i,j}, b_{i,j} \in B_{i,j}, c_{i,j} \in C_{i,j}
   \] (7)

   Due to that polynomial function is linear to their polynomial coefficients, above optimization is a linear least square one. Although with a large amount of variables, it can be solved efficiently.

4. Judgment of convergence. If \( E_{fitting} \) converges, the algorithm terminates, else jump back to step 2 for a further loop.

In above algorithm, It is crucial to efficiently solving of the optimization in step 3 that nonlinear relationship functions are linear to their parameters. Otherwise, the optimization will become a nonlinear one with large amount variables which are difficult to optimize. Polynomials are linear to their coefficients, it is an important reason of choosing cubic polynomials as the nonlinear relationship functions, as illustrated in section 4.1.

### 4.3. Expand the nonlinear relationship to facial model

The explored nonlinear relationship functions are expanded from sparse markers to the whole facial model through Basis Function (BFB) interpolation. We train a RBF (Equation (8)) for each nonlinear relationship function \( f_i(w_i) \). In this RBF, the input is vertex coordinates \( v \) of a facial model; the output is the difference between parameters of nonlinear relationship functions and the linear ones corresponding to this vertex, it can be expressed as \( a_{i,j}, b_{i,j} \) and \( c_{i,j} - e_{i,j} \). Coordinates and nonlinear relationship functions of markers are regarded as the training samples. In this RBF, we choose inverse multiquadrics square root as the basis function \( h_m(v) \) (Equation (9)). The shortest path is set as the distance \( dist(v,v_m) \) between two vertexes \( v \) and \( v_m \) with regarding facial model as undigraph. The RBF training algorithm is elaborated in [8].

\[
F(v) = \sum_{m=1}^{N_m} p_{jm} h_m(v) \\
\]

\[
h_m(v) = (\text{dist}^2(v,v_m) + c_m)^{-1/2} \\
\]

### 5. Experiments

We show the experiment results of proposed scheme in this section. We firstly reveal the nonlinearity of facial expressions brought by the explored nonlinear relationship functions. Then, we contrast the facial animations respectively synthesized by linear relationship functions and the explored nonlinear ones. The results are assessed both through user study and quantitative evaluation. After that, we introduce the computational efficiency of the proposed algorithms in this paper. At last, we illustrated the overfitting results due to the optimization in global variable space. In the experiments, the facial model is composed of more than 6000 vertexes, and the main hardware of computer is 2.67GHz CPU and 2.0GB RAM. We have captured about 500 expressions to optimize the nonlinear relationship functions. To the intervals of optimization variables, \( A_{i,j} \) and \( B_{i,j} \) are set as \([-l_{e_{i,j}}, l_{e_{i,j}}] \), and \( C_{i,j} \) is set as \([0.8l_{e_{i,j}}, 1.2l_{e_{i,j}}] \).

By taking "Mouth Opening" as example, we show the

\( \copyright 2010 \text{ The Author(s)} \)

Journal compilation \( \copyright 2010 \text{ The Eurographics Association and Blackwell Publishing Ltd.} \)
nonlinearity of jaw motion brought by the explored nonlinear relationship functions. As shown in figure ??, the top face geometries are generated by linear relationship functions \( w_i e_i \); the bottom images are synthesized by the explored nonlinear ones \( f_i(w_i) \); from left to right, the blending weight \( w_i \) increase progressively from 0.0 to 1.0 with 0.2 intervals; the green spherules indicate the chin vertex positions of different blending weights; the purple ones reveal the orbit of chin vertex during blending weight changing gradually. Furthermore, the example of "Eyes Closing" is shown in figure ??; From figure ?? and ?? we can see that, through optimization, the explored nonlinear relationship between blending weights and deformed geometry is well consistent with the nonlinearity of facial expressions. More examples are shown in the accompanying video.

We contrast the facial animations which are respectively synthesized by linear relationship functions (Equation ??) and the explored nonlinear ones (Equation ??). The results are assessed through user study. To do that, we have captured another 13 clips of facial motion whose durations spread from 10 seconds to one minute. These clips of facial motion are independent from the captured expressions which are used to explore the nonlinear relationship functions. For each clip, we synthesize facial animations through linear relationship functions \( w_i e_i \) and nonlinear ones \( f_i(w_i) \) respectively. Given a frame of captured facial motion \( M_k \), to the linear one, blending weights are calculated through optimizations of equation ??; and facial expression is synthesized through equation ??; to the nonlinear one, blending weights are optimized through equation ??, and facial expression is generated according to equation ??.

From this table, we can see that, the explored nonlinear relationship functions are able to synthesize much more realistic facial animation than the linear ones (89.7% vs. 10.3%).

For intuition, we list several synthesized facial expressions of user study in figure ??; In this figure, the green spherules denote the marker positions of synthesized facial expressions, the red ones represent the marker coordinates of the captured facial motion. The second row and third row facial expressions are respectively synthesized by linear relationship functions and nonlinear ones from the clips of cap-
We also quantitatively evaluate the reality promotion brought by the explored nonlinear relationship functions relative to linear ones. We quantify the synthesis error as the markers average Euclidean distance between synthesized facial expressions and captured ones. From the thirteen clips of captured facial motion, the synthesis error brought by linear relationship functions is 3.52mm, and the one brought by the nonlinear ones is 0.71mm.

The algorithms proposed in this paper can be efficiently solved. At first, in the iterative algorithm of exploring nonlinear relationship functions, the optimization of equation ?? takes about 0.03 seconds; the one of equation ?? takes about 1.47 seconds; the iterative optimization algorithm totally spend about ten minutes to converge. Furtherly, it takes about another 3 minutes to expand the explored nonlinear relationship functions from sparse markers to facial model. At last, when synthesizing facial expressions through nonlinear relationship functions according to equation ??, it takes only about 0.0006 seconds per frame. Although the synthesis takes about three times as long as the linear blending method of equation ??, the high efficiency doesn’t cause any trouble of real-time application.
In the optimization, it results in overfitting to optimize nonlinear relationship functions in global variables space. The overfitted nonlinear relationship functions are able to synthesize the captured facial expressions which are used to explore them most accurately. However, they miss their visual meanings. It leads to unusability, as shown in figure ??a. Through setting the optimization intervals $A_{i,j}$, $B_{i,j}$ and $C_{i,j}$ as the neighborhoods of 0, 0 and $e_{i,j}$ respectively, the overfitting is avoided. As results, the explored nonlinear relationship functions possess their visual meanings and are available for synthesizing facial expressions, as shown in figure ??b.

6. Conclusion and discussion

In this paper, we work on the issue of exploring the nonlinear relationship between blending weights and deformed face geometry of blendshape facial animation. We explore aforementioned nonlinear relationship through optimization. The optimization searches the nonlinear relationship that can synthesize realistic facial expressions in the appropriate subspace of variables. Experiments show that the explored nonlinear relationship functions consist with the nonlinearity of facial expressions well. Also, more realistic facial animation can be synthesized by the explored nonlinear relationship functions than the linear ones.

The proposed scheme in this paper can be used in realistic facial animation synthesis as conveniently as the linear blending method. In blendshape animation, to animate a facial model, a set of key shapes is firstly constructed. It is an initialization procedure of blendshape facial animation. Then, blending weights are adjusted to synthesize desired facial animation. Slightly different from above flow, after construction of key shapes, a further step is needed to explore the nonlinear relationship functions $f_i(w_i)$. It is done efficiently with little manual operation. As another initialization step, it doesn’t cause trouble of usability. After exploration of $f_i(w_i)$, users can synthesize facial animation through nonlinear relationship functions as if linear ones. When adjusting blending weights, facial expressions are synthesized through nonlinear relationship functions as illustrated in equation ??.

We have applied the proposed scheme in the applications of both performance driven and key frame facial animation. In performance driven application, given one frame of captured facial motion, the blending weights are computed through optimization of equation ?? Then the facial expression is synthesized from blending weights through equation ?? We have applied it in experiments of user study, as shown in figure ??.

The scheme can also be used in key frame facial animation. Firstly, users may configure key frames expressions through manually adjusting blending weights, as shown in figure ?? Based on that, the other frames blending weights can be obtained by interpolating the ones between key frames. At last, the expression of each frame is synthesized through equation ?? We have demonstrated several expressions results of key frame facial animation in figure ??.

In our work, the geometric constraints are ignored, which may result in some artifacts. For example, when closing eyes, the upper and lower eyelids may penetrate into each other in extreme cases. In blendshape facial animation, user may achieve the geometric constraints through designing key shapes carefully. However the geometric constraints may be broken during optimization. So in future work, we will take the geometric constraints into consideration and try to add equality constraints into the optimization to improve the scheme.

7. Acknowledge

This paper was supported in part by the National Natural Science Foundation of China, No. U0935003 and No. 60970086.
Figure 12: Results of key frame facial animation. Given blending weights of key frames (red bars), the ones of other frames (slider block) are obtained through interpolation between key frames. The facial expressions are then synthesized from blending weights through nonlinear relationship functions.

References


[Sag06] SAGAR M.: Facial performance capture and expressive


Figure 13: The nonlinearity of upper eyelid motion. (a) The orbit (red curve) of vertex on upper eyelid during "closing eyes". (b) In blendshape facial animation, the orbit is linearly simplified (blue line). (c) Through setting appropriate nonlinear relationship functions (green curves), the orbit is well approximated.

Table 1: The vote percent of user study.

<table>
<thead>
<tr>
<th>Clip Index</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nonlinear</td>
<td>100</td>
<td>93.3</td>
<td>100</td>
<td>86.7</td>
<td>93.3</td>
<td>66.7</td>
<td>100</td>
<td>100</td>
<td>80</td>
<td>86.7</td>
<td>86.7</td>
<td>80</td>
<td>93.3</td>
<td>89.7</td>
</tr>
<tr>
<td>Linear</td>
<td>0</td>
<td>6.7</td>
<td>0</td>
<td>13.3</td>
<td>6.7</td>
<td>33.3</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>13.3</td>
<td>13.3</td>
<td>20</td>
<td>6.7</td>
<td>10.3</td>
</tr>
</tbody>
</table>