Eurographics/SIGGRAPH Symposium on Computer Animation (2003)
D. Breen, M. Lin (Editors)

An Example-Based Approach for Facial Expression Cloning

Hyewon Pyun[1], Yejin Kim[2], Wonseok Chae[1], Hyung Woo Kang[1], and Sung Yong Shin[1]

[1]Korea Advanced Institute of Science and Technology
[2]Electronics and Telecommunications Research Institute
e-mail: [1]{hyewon, wschae, henry, syshin}@jupiter.kaist.ac.kr, [2]yejink@etri.re.kr

Abstract

In this paper, we present a novel example-based approach for cloning facial expressions of a source model to a target model while reflecting the characteristic features of the target model in the resulting animation. Our approach comprises three major parts: key-model construction, parameterization, and expression blending. We first present an effective scheme for constructing key-models. Given a set of source example key-models and their corresponding target key-models created by animators, we parameterize the target key-models using the source key-models and predefine the weight functions for the parameterized target key-models based on radial basis functions. In runtime, given an input model with some facial expression, we compute the parameter vector of the corresponding output model, to evaluate the weight values for the target key-models and obtain the output model by blending the target key-models with those weights. The resulting animation preserves the facial expressions of the input model as well as the characteristic features of the target model specified by animators. Our method is not only simple and accurate but also fast enough for various real-time applications such as video games or internet broadcasting.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Three-Dimensional Graphics and Realism]: Animation; I.3.5 [Computational Geometry and Object Modeling]: Geometric Algorithms

Keywords: Facial Animation, Facial Expression Cloning, Example-based Synthesis, Scattered Data Interpolation, Motion Retargetting

1. Introduction

1.1. Motivation

Synthesis by reusing existing data has recently been popular in a wide spectrum of computer graphics, including shape modelling [29, 1], image or texture synthesis [16, 32], and motion generation [8, 19]. Inspired by motion retargetting [8, 14, 26], Noh and Neumann [18] posed the problem of cloning facial expressions of an existing 3D face model to a new model. Based on 3D geometry morphing, their solution to this problem is first to compute the motion vectors of the source model and then deform these to add to the target model. This approach works well for face models with similar shapes. In general, it is hard to simulate the imagination (or intention) of an animator by this rather mechanical manipulation of motion vectors.

There is a stream of research on parameter-driven facial animation such as facial action coding system or model-based persona transmission [20]. In particular, 3D facial animation is generated from 2D videos in performance-driven facial animation, which can be thought of as a process of transferring parameters from the source space (2D videos) to the target space (3D face models) [12, 24, 6]. In general, the target animation is obtained by deforming the target model or blending 3D face models with different expressions, to match the parameters transferred from the source space. The parameter-driven facial animation usually experiences time-consuming optimization in deformation or blending.

Based on the notion of parameter transfer, Bregler et al. [2] proposed an elegant scheme for cartoon motion capture and retargetting. They chose source example key-shapes from a given input cartoon animation and modelled their corresponding target key-shapes. Given the shape of an input cartoon character, they interpret it as a shape interpolated from

© The Eurographics Association 2003.

source key-shapes, which is deformed by an affine transformation. To capture a snapshot (posture) of cartoon motion, they extracted the affine transformation parameters together with the interpolation weight values of source example key-shapes at every time step by least squares approximation. They then applied the extracted parameters and weights to the target example key-shapes for motion retargetting. By allowing animators to model the target key-shapes explicitly, their imagination can be realized in the resulting cartoon animation.

Based on numerical optimization, their scheme is a little too time-consuming to be applied directly to the cloning problem, in particular, for real-time applications such as computer games and internet broadcasting. Furthermore, facial expressions result from local deformations of various parts of a face model, which may not be reflected properly by an affine transformation. To address these issues, we propose a novel scheme for cloning facial expressions of a source model while preserving the characteristic features of target expressions specified by animators. Based on scattered data interpolation, the proposed scheme is not only simple and efficient but also reflects animators' intention accurately.

1.2. Related Work

There have been extensive efforts on the development of 3D facial animation techniques since Parke's pioneering work [21]. An excellent survey of these efforts can be found in [20]. We begin with traditional approaches for generating facial animation from scratch and then move on to more recent work directly related to our scheme.

Facial Animation From Scratch: Physically-based approaches have been used to generate facial expressions by simulating the physical properties of facial skin and muscles [15, 25, 30, 31]. Parke proposed a parametric approach to represent the motion of a group of vertices with a parameter vector and used this approach to generate a wide range of facial expressions [22]. In performance-driven approaches, facial animations were synthesized based on facial motion data captured from live actors' performances [33, 9, 6]. Kalra et al. [11] used free-form deformations to manipulate facial expressions. Pighin et al. [23] presented an image-based approach to generate photorealistic 3D facial expressions from a set of 2D photographs. All of those approaches are common in that the same process needs to be repeated for animating a new face model, even when a similar expression sequence has already been available for a different model.

Retargetting and Cloning: Recently, there have been rich research results on reusing existing animation data. Gleicher [8] described a method for retargetting motions to new characters with different segment proportions. Lee and Shin [14] enhanced this idea by using a hierarchical displacement mapping technique based on multilevel B-spline approximation. Popovic and Witkin [26] presented a physically-based motion transformation technique that preserves the essential physical properties of a motion by using the spacetime constraints formulation. The approaches for motion retargetting focused on skeleton-based articulated body motions. Noh and Neumann [18] adopted the underlying idea of motion retargetting for reusing facial animation data. Based on 3D geometry morphing between the source and target face models, their approach transfers the facial motion vectors from a source model to a target model in order to generate cloned expressions on the target model. This method is suitable for mutually morphable face models with a strong shape resemblance to reproduce facial expressions as intended by animators.

Example-based Motion Synthesis: Example-based motion synthesis is another stream of research directly related to our approach. Rose et al. [27] and Sloan et al. [29] proposed example-based motion blending frameworks, employing scattered data interpolation with radial basis functions. Park et al. [19] proposed an on-line motion blending scheme for locomotion generation adopting this idea. Lewis et al. [17] introduced an example-based pose space deformation technique, and Allen et al. [1] applied a similar technique to range-scan data for creating a new pose model. While most of the previous techniques focused on the pure synthesis aspect, Bregler et al. [2] proposed an example-based approach for cartoon motion capture and retargetting. Based on affine transformations, their approach first extracts the transformation parameters and the interpolation weights of the source key-shapes at each frame of the input cartoon animation and then generates an output shape by applying both the parameters and weights to the corresponding target key-shapes. This approach is non-trivially adapted for cloning facial expressions in our work. We also note that there have been some similar example-based approaches in the domain of peformance-driven facial animation, for retargetting facial expressions from 2D videos to 2D drawings [3] or to 3D models [4].

1.3. Overview

Inspired by cartoon motion capture and retargetting [2], we adopt an example-based approach for retargetting facial expressions from one model to another. Figure 1 illustrates our example-based approach for expression cloning. Given an input 3D facial animation for a source face model, we generate a similar animation for a target model by blending the predefined target models corresponding to the example source face models with extreme expressions called the *key-models*. Our approach comprises three major parts: key-model construction, parameterization, and expression blending. The first two parts are done once at the beginning, and the last part is repeatedly executed for each input expression in runtime.

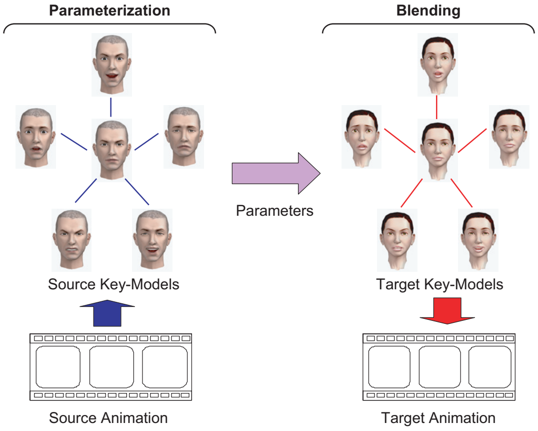


Figure 1: *Overview of our example-based cloning*

First, we identify a set of example extreme expressions for a given source model. These expressions should be generic enough to handle the facial animations of the source model effectively. Provided with the example extreme expressions, animators construct the corresponding key-models for both source and target face models. The source key-models should reflect the actual extreme expressions of the source model accurately. However, the animators can breathe their creativity and imagination into the target key-models while constructing them. To minimize the time and efforts of animators, we provide a novel scheme for compositing key-models. Next, the target key-models are parameterized to apply scattered data interpolation [27, 29]. We provide a simple, elegant parameterization scheme for effective motion blending. Finally, given an input model with some facial expression, the parameter vector for the corresponding output model is analytically computed to evaluate the predefined weight functions of target key-models. The output model with the cloned expression is obtained by blending the target key-models with respect to those weight values.

The remainder of this paper is organized as follows: In Sections 2, 3, and 4, we describe key-model construction, parameterization, and expression blending, respectively. We show experimental results and compare our approach with the previous one [18] in both quality and efficiency in Section 5. Finally, in Section 6 we conclude this paper and discuss future research issues.

2. Key-Model Construction

The facial expression cloning problem has a quite different nature than the cartoon motion capture and retargetting problem. In the latter problem, cartoon characters have the capability of changing their shapes dynamically while still preserving their identities. Bregler et al. tried to capture those dynamic shape changes with 2D affine transformations together with key-shape interpolation. In cartoon animations, it is hard to identify all generic extreme key-shapes due to their dynamic nature. Thus, the authors selected extreme key-shapes from a given source animation.

On the other hand, facial expressions are determined by a combination of subtle local deformations on a source face model rather than global shape change. Unlike cartoon motions, face motions (expressions) have been well characterized. In particular, we adopt two categories of key-expressions: emotional key-expressions and verbal key-expressions. The former category of key-expressions reflect emotional states, and the latter expressions mainly result from lip movements for verbal communications. We combine them to define generic key-expressions for a source model, and then create the corresponding key-models for both source and target face models, by deforming their respective base models with neutral expressions. Through crafting the target key-models, animators can realize their imaginations.

Referring to the emotion space diagram [28], we choose six purely emotional key-expressions including neutral, happy, sad, surprised, afraid, and angry expressions as shown in Figure 2. The neutral expression is chosen as the base expression. Based on the notion of *visemes*[†], that is, the distinct visual mouth expressions observed in speech [7], we take thirteen visemes as the purely verbal key-expressions as depicted in Figure 3. Combining these two categories of key-expressions, we have 78 (6 × 13) generic key-expressions together with six purely emotional expressions. Notice that the purely verbal expressions are regarded as the verbal expressions combined with the neutral emotional expression.

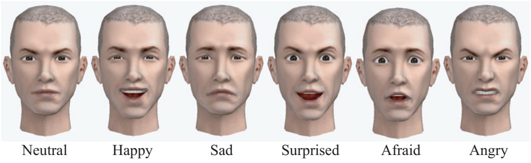


Figure 2: *Six emotional key-expressions.*

To facilitate example-based expression cloning, we need to have the corresponding 84 key-models for each of the source and target face models. It is time-consuming to craft all of them by hand. Instead, we take a semi-automatic approach: preparing the purely emotional and purely verbal key-models by hand and then combining them automatically. Now, our problem of automatic key-model creation is reduced to a geometry compositing problem: Given an emotional key-expression and a verbal key-expression, how can we obtain their combined key-expressions?

[†] The term ‘viseme’ is the abbreviation of ‘visual phoneme’.

© The Eurographics Association 2003.

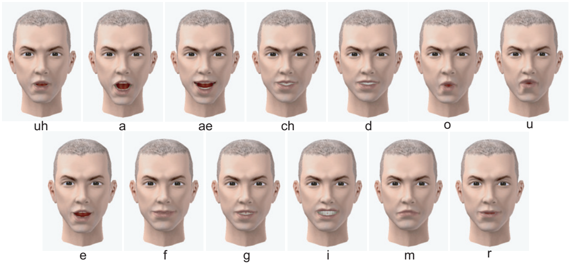


Figure 3: *Thirteen verbal key-expressions.*

Without loss of generality, suppose that the face models are represented as polygonal meshes (polyhedra). Then, expressions cause the movements of vertices on a face model. Analysing the purely emotional and verbal key-models with respect to the base model, we characterize the vertices in terms of their contributions to facial expressions. For example, vertices near eyes contribute mainly to making emotional expressions. Vertices near the mouth contribute to both emotional and verbal expressions. However, the movements are mainly constrained by verbal expressions to produce accurate pronunciations when there are any conflicts between two types of expressions.

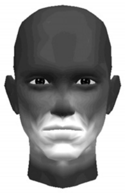


Figure 4: *A distribution of importance values.*

Based on this observation, we introduce the notion of importance, which measures the relative contribution of each vertex to the verbal expressions with respect to the emotional expressions. The importance value α_i of every vertex v_i is estimated empirically from the purely emotional and verbal key-models such that $0 \leq \alpha_i \leq 1$ for all i. If $\alpha_i \geq 0.5$, then the movement of v_i is constrained by the verbal expressions; otherwise, it is constrained by the emotional expressions. Figure 4 shows the distribution of importance values over a face model estimated by our scheme as given in the Appendix, to which we refer readers for details. Brighter regions contain vertices of higher importance values.

Now, we are ready to explain how to composite an emotional key-model E and a verbal key-model P derived from the base model B. Let

$$\begin{aligned} B &= \{v_1,\, v_2,\, \ldots,\, v_n\}, \\ E &= \{v_1^E,\, v_2^E,\, \ldots,\, v_n^E\}, \text{ and} \\ P &= \{v_1^P,\, v_2^P,\, \ldots,\, v_n^P\}. \end{aligned}$$

v_i^E and v_i^P, $1 \leq i \leq n$ are obtained by displacing v_i, if needed, and thus the natural correspondence is established for the vertices with the same subscript. For every vertex v_i, we define displacements Δv_i^E and Δv_i^P as follows:

$$\begin{aligned} \Delta v_i^E &= v_i^E - v_i, \text{ and} \\ \Delta v_i^P &= v_i^P - v_i. \end{aligned}$$

Let $C = \{v_1^C,\, v_2^C,\, \ldots,\, v_n^C\}$ and $\Delta v_i^C = v_i^C - v_i$ be the combined key-model and the displacement of a vertex $v_i^C \in C$, respectively. Since the problem of computing v_i^C can be reduced to the compositing of two vectors Δv_i^E and Δv_i^P, we assume that v_i^C lies on the plane spanned by Δv_i^E and Δv_i^P and containing v_i as shown in Figure 5.

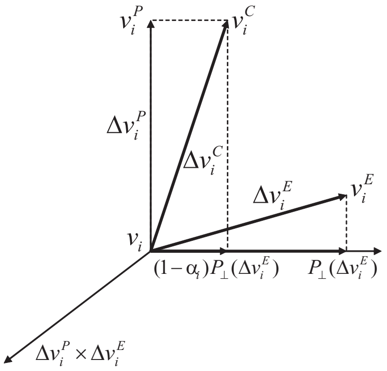


Figure 5: *Composition of the two displacements*

Consider a vertex $v_i^C, 1 \leq i \leq n$ of the combined key-model in C. If $\alpha_i \geq 0.5$, then the verbal component Δv_i^P should be preserved in Δv_i^C for accurate pronunciation. Therefore, letting $P_\perp(\Delta v_i^E)$ be the component of Δv_i^E perpendicular to Δv_i^P, only this component $P_\perp(\Delta v_i^E)$ of Δv_i^E can make a contribution to Δv_i^C on top of Δv_i^P. That is, we ignore the other component $P_{||}(\Delta v_i^E)$ of Δv_i^E which is parallel to Δv_i^P. Otherwise, the constraints on accurate pronunciation would not be satisfied when there are conflicts between the two types of expressions. (see Figure 5). If $\alpha_i < 0.5$, the roles of Δv_i^P and Δv_i^E are switched. Thus, we have

$$v_i^C = \begin{cases} v_i + (\Delta v_i^P + (1-\alpha_i)P_\perp(\Delta v_i^E)) & \text{if } \alpha_i \geq 0.5 \\ v_i + (\Delta v_i^E + \alpha_i E_\perp(\Delta v_i^P)) & \text{otherwise,} \end{cases}$$

where

$$\begin{aligned} P_\perp(\Delta v_i^E) &= \Delta v_i^E - \frac{\Delta v_i^E \cdot \Delta v_i^P}{|\Delta v_i^P|^2} \cdot \Delta v_i^P, \text{ and} \\ E_\perp(\Delta v_i^P) &= \Delta v_i^P - \frac{\Delta v_i^P \cdot \Delta v_i^E}{|\Delta v_i^E|^2} \cdot \Delta v_i^E. \end{aligned}$$

Figure 6 shows a combined key-model (Figure 6(c)) constructed from a key-model with a verbal expression (Figure 6(a)) and a key-model with an emotional expression

(Figure 6(b)). Note that the verbal expression is preserved around the mouth and the emotional expression is preserved in other parts.

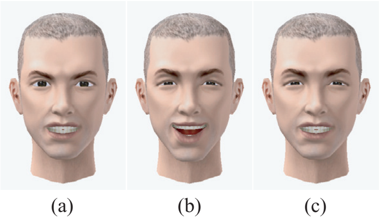


Figure 6: *Compositing key-models: (a) verbal key-model (vowel 'i') (b) emotional key-model ('happy') (c) combined key-model.*

3. Parameterization

We parameterize the target key-models based on the correspondences between the source base model and the source key-models. We interactively select a number of feature points on the source base model and then extract their displacements to the corresponding points on each of the source key-models. Concatenating these displacements, the displacement vector of each source key-model is formed to parameterize the corresponding target key-model. Most individual parameter components tend to be correlated to each other. Thus, based on PCA (principal component analysis) [10], the dimensionality of the parameter space can be reduced by removing less significant basis vectors of the resulting eigenspace.

As illustrated in Figure 7, we select about 20 feature points from the source base model. While the number of feature points depends on the shape and complexity of the base model, we believe empirically that two to four feature points around the facial parts such as the mouth, eyes, eyebrows, the forehead, the chin, and cheeks are sufficient to represent distinct facial expressions. Note that we only mark the feature points on the base model. Then, those on the other key-models are automatically determined from their vertex correspondences to the base model.

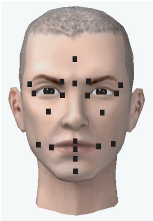


Figure 7: *The source base key-model with 20 manually selected feature points.*

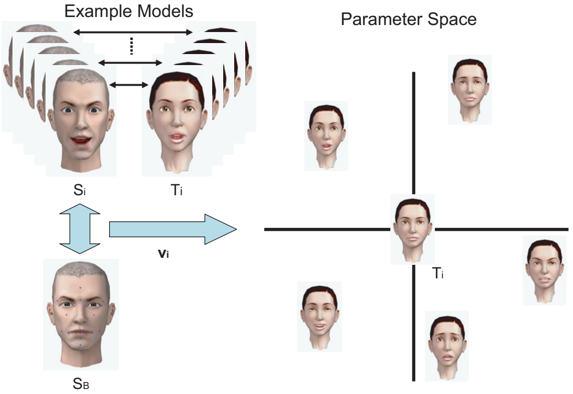


Figure 8: *The displacement vector of each source key-model* $\mathbf{S}_i$ *is used for parameterizing the corresponding target key-model* $\mathbf{T}_i$.

The displacement vector $\mathbf{v}_i$ of a source key-model $\mathbf{S}_i$ from the source base key-model $\mathbf{S}_B$ is defined as follows:

$$\mathbf{v}_i = \mathbf{s}_i - \mathbf{s}_B, \; 1 \leq i \leq M, \qquad (1)$$

where $\mathbf{s}_B$ and $\mathbf{s}_i$ are vectors obtained by concatenating, in a fixed order, the 3D coordinates of feature points on $\mathbf{S}_B$ and those on $\mathbf{S}_i$, respectively, and M is the number of source key-models. As shown in Figure 8, $\mathbf{v}_i$ places each target key-model $\mathbf{T}_i$ in the N-dimensional parameter space, where N is the number of components, that is, three times the number of feature points.

Since the dimensionality N of the parameter space is rather high compared to the number M of key-models, we employ PCA to reduce it. Given M displacement vectors of dimension N, we first generate their component covariance matrix, which is an $N \times N$ square matrix, to compute the eigenvectors of the matrix and the corresponding eigenvalues. These eigenvectors are called the *principal components* representing the principal axes that characterize the distribution of displacement vectors. The dimensionality of the parameter space can be reduced by removing less significant eigenvectors, which have small eigenvalues. In our experiments, we use an empirical threshold value of 0.00001 to remove those eigenvectors. The removal of such eigenvectors may cause some characteristics of the key-models not to be parameterized. With our choice of the threshold, we have observed that the effect is negligible. In experiments, the dimensionality of the parameter space can be reduced from 60 to 18 without any difficulty.

Let $\mathbf{e}_i, 1 \leq i \leq N$ be the eigenvector corresponding to the ith largest eigenvalue. Suppose that we choose $\bar{N}$ eigenvectors as the coordinate axes of the parameter space, where $\bar{N} < N$. To transform an original N-dimensional displacement vector into an $\bar{N}$-dimensional parameter vector, an

© The Eurographics Association 2003.

$\bar{N} \times N$ matrix $\mathbf{F}$ called the *feature matrix* is constructed:

$$\mathbf{F} = \left[\mathbf{e}_1\ \mathbf{e}_2\ \mathbf{e}_3 \ldots \mathbf{e}_{\bar{N}}\right]^\top, \tag{2}$$

Using the feature matrix $\mathbf{F}$, the parameter vector $\mathbf{p}_i$ corresponding to the displacement vector v_i of a target key-model $\mathbf{T}_i$ is derived as follows:

$$\mathbf{p}_i = \mathbf{F}\mathbf{v}_i,\ 1 \le i \le M, \tag{3}$$

which reduces the dimensionality of the parameter space from N to $\bar{N}$. This is equivalent to projecting each displacement vector $\mathbf{v}_i$ onto the eigenspace spanned by the $\bar{N}$ selected eigenvectors. We later use this feature matrix $\mathbf{F}$ to compute the parameter vector from a given displacement vector.

4. Expression Blending

With the target key-models thus parameterized, our cloning problem is transformed to a scattered data interpolation problem. To solve this problem, our expression blending scheme predefines the weight functions for each target key-model based on cardinal basis functions [29], which consist of linear and radial basis functions. The global shape of a weight function is first approximated by linear basis functions, and then adjusted locally by radial basis functions to exactly interpolate the corresponding key-model. Given the input face model with a facial expression, a novel output model with the cloned expression is obtained in runtime by blending the target key-models as illustrated in Figure 9. Our scheme first computes the displacement vector of the input face model and then derives the parameter vector of the output model from the displacement vector. Finally, the predefined weight functions are evaluated at this parameter vector to produce the weight values, and the output model with the cloned facial expression is generated by blending the target key-models with respect to those weight values.

The weight function $w_i(\cdot)$ of each target example model $\mathbf{T}_i, 1 \le i \le M$ at a parameter vector $\mathbf{p}$ is defined as follows:

$$w_i(\mathbf{p}) = \sum_{l=0}^{\bar{N}} a_{il} A_l(\mathbf{p}) + \sum_{j=1}^{M} r_{ji} R_j(\mathbf{p}). \tag{4}$$

where $A_l(\mathbf{p})$ and a_{il} are the linear basis functions and their linear coefficients, respectively. $R_j(\mathbf{p})$ and r_{ij} are the radial basis functions and their radial coefficients. Let $\mathbf{p}_i, 1 \le i \le M$ be the parameter vector of a target key-model $\mathbf{T}_i$. To interpolate the target key-models exactly, the weight of a target key-model $\mathbf{T}_i$ should be one at $\mathbf{p}_i$ and zero at $\mathbf{p}_j, i \ne j$, that is, $w_i(\mathbf{p}_i) = 1$ for $i = j$ and $w_i(\mathbf{p}_j) = 0$ for $i \ne j$.

Ignoring the second term of Equation (4), we solve for the linear coefficients a_{il} to fix the first term:

$$w_i(\mathbf{p}) = \sum_{l=0}^{\bar{N}} a_{il} A_l(\mathbf{p}). \tag{5}$$

The linear bases are simply $A_l(\mathbf{p}) = \mathbf{p}^l, 1 \le l \le \bar{N}$, where

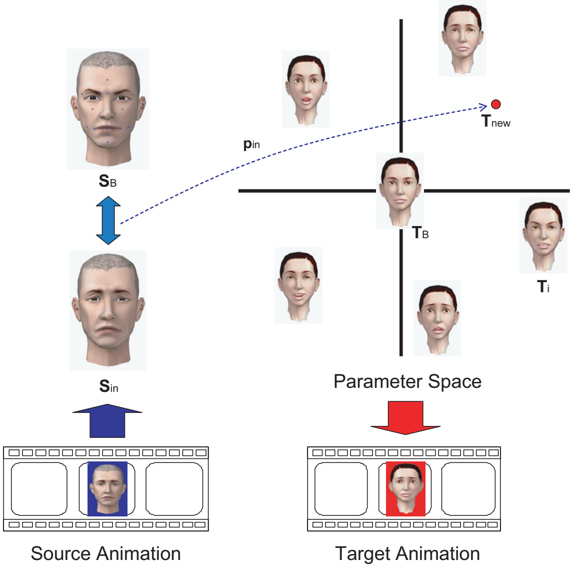


Figure 9: *Generating a new face model by blending target key-models*

$\mathbf{p}^l$ is the lth component of $\mathbf{p}$, and $A_0(\mathbf{p}) = 1$. Using the parameter vector $\mathbf{p}_i$ of each target key-model and its weight value $w_i(\mathbf{p}_i)$, we employ a least squares method to evaluate the unknown linear coefficients a_{il} of the linear bases.

To fix the second term, we compute the residuals for the target key-models:

$$w_i'(\mathbf{p}) = w_i(\mathbf{p}) - \sum_{l=0}^{\bar{N}} a_{il} A_l(\mathbf{p}) \text{ for all } i. \tag{6}$$

The radial basis function $R_j(\mathbf{p})$ is a function of the Euclidean distance between $\mathbf{p}$ and $\mathbf{p}_j$ in the parameter space:

$$R_j(\mathbf{p}) = B\left(\frac{\| \mathbf{p} - \mathbf{p}_j \|}{\alpha}\right) \text{ for } 1 \le j \le M, \tag{7}$$

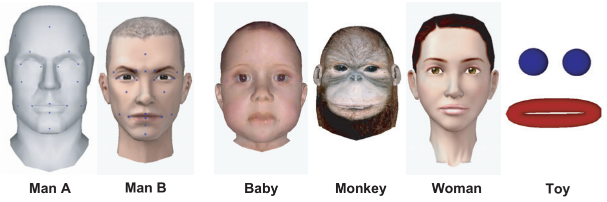
where $B(\cdot)$ is the cubic B-spline function, and α is the dilation factor, which is the separation to the nearest other example in the parameter space. The radial coefficients r_{ij} are obtained by solving the matrix equation,

$$\mathbf{r}\mathbf{R} = \mathbf{w}', \tag{8}$$

where $\mathbf{r}$ is an $M \times M$ matrix of the unknown radial coefficients r_{ij}, and $\mathbf{R}$ and $\mathbf{w}'$ are the matrices of the same size defined by the radial bases and the residuals, respectively, such that $\mathbf{R}_{ij} = R_i(\mathbf{p}_j)$ and $\mathbf{w}'_{ij} = w_i'(\mathbf{p}_j)$.

With the weight functions predefined, we are now ready to explain how to blend the target key-models in runtime. For the input face model $\mathbf{S}_{in}$ at each frame of an input animation, the displacement vector $\mathbf{d}_{in}$ is computed with respect to the source base model $\mathbf{S}_B$:

$$\mathbf{d}_{in} = \mathbf{s}_{in} - \mathbf{s}_B \tag{9}$$



(a) Source models (b) Target Models

Figure 10: *Models used for the experiments*

	Man A	Man B	Baby	Monkey	Woman	Toy
V	988	1192	1253	1227	1220	931
P	1954	2194	2300	2344	2246	957

Table 1: *Model specification*

(V:Vertices, P:Polygons)

where $\mathbf{s}_{in}$ and $\mathbf{s}_B$ are respectively vectors obtained by concatenating the 3D coordinates of feature points on $\mathbf{S}_{in}$ and $\mathbf{S}_B$ as explained previously. Given this N-dimensional displacement vector $\mathbf{d}_{in}$, we then obtain the corresponding $\bar{N}$-dimensional parameter vector $\mathbf{p}_{in}$ as follows:

$$\mathbf{p}_{in} = \mathbf{F}\mathbf{d}_{in}, \qquad (10)$$

where $\mathbf{F}$ is the feature matrix defined in Equation (2).

Using the predefined weight functions for the target key-models $\mathbf{T}_i$ as given in Equation (4), we estimate the weight values $w_i(\mathbf{p}_{in})$ of all target key-models $\mathbf{T}_i, 1 \leq i \leq M$ at the parameter $\mathbf{p}_{in}$ to generate the output face model $\mathbf{T}_{new}(\mathbf{p}_{in})$:

$$\mathbf{T}_{new}(\mathbf{p}_{in}) = \mathbf{T}_B + \sum_{i=1}^{M} w_i(\mathbf{p}_{in})(\mathbf{T}_i - \mathbf{T}_B), \qquad (11)$$

where $\mathbf{T}_B$ is the target base model corresponding to the source base key-model $\mathbf{S}_B$ with the neutral expression.

5. Experimental Results

As shown in Figures 10(a) and 10(b), we used two source models and four target models in our experiments. Table 1 gives the number of vertices and that of polygons in each model. We manually selected 20 feature points on each source model as described in Section 3. As input animations, we prepared two different facial animations: a facial animation of Man A with various exaggerated expressions, and a facial animation of Man B with verbal expressions combined with emotional expressions.

Our first two experiments were intended to qualitatively show the effectiveness of our approach. In the first experiment, we used Man A as the source face model and the baby and monkey models as the target face models. To clone the facial expression of Man A, we used six key-models for the source face model as well as the target face model. The first row of Figure 11 shows the input expressions of Man A sampled from the input animation. The baby and monkey models with cloned facial expressions are shown in the second and third rows of Figure 11, respectively. We can observe that the expressions of the source model were nicely cloned to the target models while reflecting the characteristic features of the target models.

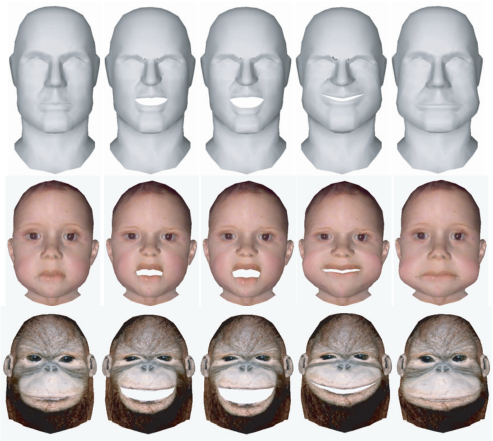


Figure 11: *Cloning expressions from Man A to the target models*

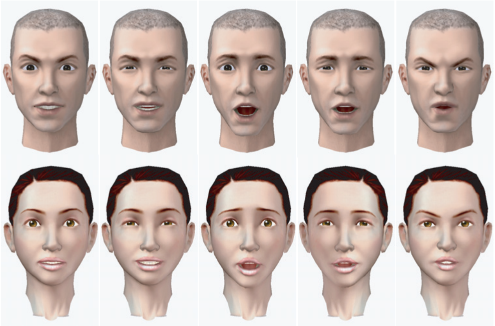


Figure 12: *Cloning expressions from Man B to the target model*

In the second experiment, we used Man B as the source model and the woman model as the target model to clone combined expressions. We prepared a total of 84 key-models: six purely emotional key-expressions (neutral, happy, angry, sad, surprised, and afraid expressions) and thirteen visemes (seven for vowels and six for consonants) for each of the six emotional key-expressions. After manually creating thirteen purely verbal expressions together with six purely emotional ones, the rest of them are obtained automatically by employing our expression compositing scheme.

© The Eurographics Association 2003.

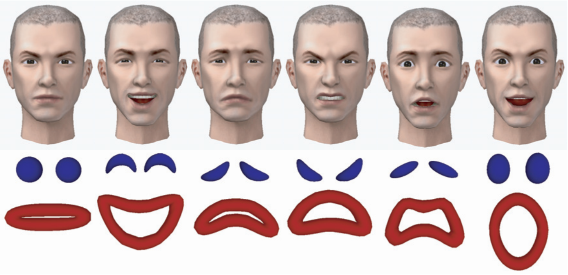


Figure 13: *Cloning expressions from Man B to the topologically different model*

As shown in Figure 12, the combined expressions of the source model were convincingly reproduced in the target model. We also cloned emotional expressions of Man B to the topologically different model as shown in the Figure 13.

	Man A ⇒ Man A		Man B ⇒ Man B	
	Ours	Noh et al.'s	Ours	Noh et al.'s
x	0.110%	0.234%	0.051%	0.176%
y	0.111%	0.196%	0.057%	0.133%
z	0.050%	0.077%	0.100%	0.213%

Table 2: *Average errors of cloned animations (source ⇒ source)*

	Man A ⇒ Baby Baby ⇒ Man A		Man B ⇒ Woman Woman ⇒ Man B	
	Ours	Noh et al.'s	Ours	Noh et al.'s
x	0.112%	2.120%	0.118%	3.076%
y	0.113%	1.936%	0.214%	3.893%
z	0.051%	1.004%	0.268%	4.183%

Table 3: *Average errors of cloned animations (source ⇒ target ⇒ source)*

The next two experiments were intended to quantitatively measure the effectiveness of our approach. In the third experiment, we used the same face model for both source and target models, that is, cloning expressions from Man A to itself and also from Man B to itself. We measured the difference of the resulting animation from the input animation for each of the models. The error at each individual frame is defined as follows:

$$\frac{\sum_{j=1}^{N} ||v_j - v'_j||}{\sum_{j=1}^{N} ||v_j||} \times 100, \qquad (12)$$

where v_j and v'_j are a vertex of the input model and the corresponding vertex of the cloned output model, respectively, and N is the number of vertices. The average error over all constituent frames is also computed for making comparisons with the previous work [18]. In the last experiment, we cloned expressions of Man A to the baby model and then the intermediate results back to Man A. For Man B, we repeated the same procedure with the woman model as the intermediate model. The average error is measured between the original and final animations.

Ideally, the vertex positions of cloned models in the resulting animations should be identical to those of the corresponding models in the input animation. Table 2 and Table 3 show the average errors of cloned animations for the x, y, and z coordinates in the last two experiments, respectively. In both experiments, our example-based approach made much smaller average errors than the previous approach [18]. This was mainly ascribed to the inherent accuracy of radial basis functions in scattered data interpolation, together with the fact that the previous method is based on 3D geometric morphing which may compromise the accuracy in trying to find a full surface correspondence between two models with just a small number of feature points.

The performance of our approach was summarized and compared with the same previous approach (see Table 4). Both schemes were implemented with C++ and OpenGL on an Intel Pentium® PC (P-4 2.4GHz processor, 512MB RAM, and GeForce 4®). As shown in the table, our scheme spent less than 1 millisecond in generating one frame for all experiments. Thus, the frame rate was over 1000Hz to guarantee a real-time performance. Compared to the previous approach, our approach required significantly less time in both preprocessing and retargetting steps. The efficiency of our approach was due to the supreme performance of the scattered data interpolation [29] that we have adopted for expression blending.

	Man A ⇒ Baby (1201 Frames)		Man B ⇒ Woman (880 Frames)	
	Ours	Noh et al.'s	Ours	Noh et al.'s
P	0.032 s	197.1 s	0.062 s	217.3 s
R	0.326 s	23.5 s	0.548 s	18.5 s
A	0.27 ms	19.6 ms	0.69 ms	21.0 ms

Table 4: *Computation time*

(P:Preprocessing, R:Retargetting, A:Average time / frame)

6. Conclusions

We have presented a novel example-based approach for cloning facial expressions from a source model to a target model while preserving the characteristic features of the target model. Our approach consists of three parts: key-model construction, parameterization, and expression blending. For key-model construction, we present a novel scheme for compositing a pair of verbal and emotional key-models. Based on a simple but effective parameterization scheme, we are able to place the target key-models in the parameter space. To predefine the weight functions for the parameterized target key-models, we adopt multi-dimensional scattered data interpolation with radial basis functions. In runtime, a cloned target model is generated by blending the target key-models using the predefined weight functions. As shown in the experimental results, our approach has accurately performed expression cloning with great efficiency.

One limitation of our method might be that it requires animators to prepare a set of key-models for source and target models as a preprocess, but at the same time it can be thought of as an advantage in that it allows for human control of the cloning results so that the characteristics of the target model are fully reflected. Another limitation is that our method cannot correctly clone an expression when it falls too far outside from the basis of the constructed source key-models. In this case, it would be better to select the source key-models from the source animation frames as in [2], rather than constructing them based on generic human facial key-expressions.

In future, we are planning to extend our approach to region-based expression cloning. According to results in psychology, a face can be split into several regions that behave as coherent units [5]. For example, the parts such as eyes, eyebrows, and forehead are used for emotional expressions, and those such as mouth, cheeks, and chin are used for verbal expressions. If we prepare example data for each of the parts separately, we could generate more diverse expressions with less example data. To achieve this, we need an effective way to combine separately-generated facial parts seamlessly.

References

1. B. Allen, B. Curless and Z. Popovic. "Articulated body deformation from range scan data", In *Proceedings of SIGGRAPH 02*, pp. 612-619, 2002.
2. C. Bregler, L. Loeb and E. Chuang and H. Deshpande. "Turning to the masters: Motion capturing cartoons", In *Proceedings of SIGGRAPH 02*, pp. 399-407, 2002.
3. I. Buck, A. Finkelstein, C. Jacobs, A. Klein, D. Salesin, J. Seims, R. Szeliski, and K. Toyama. "Performance-Driven Hand-Drawn Animation", In *Proceedings of Symposium on Non-Photorealistic Animation and Rendering*, 2000.
4. E. Chuang and C. Bregler. "Performance Driven Facial Animation using Blendshape Interpolation", *Standford University Computer Science Technical Report*, CS-TR-2002-02, April 2002.
5. P. Ekman and W. V. Friesen. "Unmasking the face: A guide to recognizing emotions from facial clues", Prentice-Hall Inc., 1975.
6. D. Fidaleo, J-Y. Noh, T. Kim, R. Enciso and U. Neumann. "Classification and Volume Morphing for Performance-Driven Facial Animation", In *Proceedings of Internatinoal Workshop on Digital and Computational Video*, 2000.
7. C. G. Fisher. "Confusions among visually perceived consonants.", *Jour. Speech and Hearing Research*, **11**, pp. 796-804, 1968.
8. M. Gleicher. "Retargetting motion to new Characters", *ACM SIGGRAPH 98 Conference Proceedings*, pp. 33-42, 1998.
9. B. Guenter, C. Grimm, D. Wood, H. Malvar, and F. Pighin. "Making Faces", *ACM SIGGRAPH 98 Conference Proceedings*, pp. 55-66, July 1998.
10. I. T. Jollife. "Principal components analysis", New York: Spinger, 1986.
11. P. Kalra and A. Mangili and N. M. Thalmann and D. Thalmann. "Simulation of facial muscle actions based on rational free from deformations", In *Proceedings of Eurographics 92*, pp. 59-69, 1992.
12. C. Kouadio and P. Poulin and P. Lachapelle. "Real-Time Facial Animation Based Upon a Bank of 3D Facial Expressions", *Proc. Computer Animation*, pp. 128-136, 1998.
13. G. A. Kalberer and L. V. Gool. "Face Animation Based on Observed 3D Speech Dynamics", *Computer Animation 2001*, pp. 20–27, November 2001.
14. J. Lee and S. Y. Shin. "A hierarchical approach to interactive motion editing for human-like figures", *Proceedings of SIGGRAPH 99*, pp. 39–48, 1999.
15. Y. C. Lee, D. Terzopoulos and K. Waters. "Realistic modeling for facial animation", In *Proceedings of SIGGRAPH 95*, pp. 55-62, 1995.
16. Marc Levoy and Pat Hanrahan. "Light Field Rendering", *Proceedings of SIGGRAPH 96*, pp. 31-42, 1996.
17. J. P. Lewis, M. Cordner, and N. Fong. "Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Drive Deformation", *ACM SIGGRAPH 2000 Conference Proceedings*, pp. 165-172, July 2000.
18. J. Y. Noh and U. Neumann. "Expression cloning", In *Proceedings of SIGGRAPH 01*, pp. 277-288, 2001.
19. S. I. Park, H. J. Shin and S. Y. Shin. "On-line locomotion generation based on motion blending", In *ACM SIGGRAPH Symposium on Computer Animation*, pp. 105-111, 2002.

20. F. I. Parke and K. Waters. *Computer Facial Animation*. A K Peters, 289 Linden Street, Wellesley, MA 02181, 1996.

21. F. I. Parke. "Computer generated animation of faces", Master's thesis, University of Utah, 1972.

22. F. I. Parke. "Parameterized models for facial animation", In *IEEE Computer Graphics and Applications*, Vol. 2, No. 9, pp. 61-68, 1982.

23. F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D.H. Salesin. "Synthesizing Realistic Facial Expressions from Photographs", In *ACM SIGGRAPH 98 Conference Proceedings*, pp. 75-84, July 1998.

24. F. Pighin, R. Szeliski and D. H. Salesin. "Resynthesizing facial animation through 3D model-based tracking", *Proceedings of International Conference on Computer Vision 99*, pp. 143-150, July 1999.

25. S. M. Platt and N. I. Badler. "Animating facial expressions", In *Computer Graphics*, Vol. 15(3), pp. 245-252, 1981.

26. Z. Popovic and A. Witkin. "Physically based motion transformation", In *Proceedings of SIGGRAPH 99*, pp. 11-20, 1999.

27. C. Rose, M. F. Cohen and B. Bodenheimer. "Verbs and adverbs: Multidimensional motion interpolation", In *IEEE Computer Graphics and Applications*, Vol. 18(5), pp. 32-40, 1998.

28. J. A. Russel. "A Circomplex Model of Affect", In *J. Personality and Social Psychology*, Vol. 39, pp. 1161-1178, 1980.

29. P. -P. Sloan and C. F. Rose and Michael F. Cohen. "Shape by example", In *Proceedings of 2001 Symposium on Interactive 3D Graphics*, pp. 135-144, 2001.

30. D. Terzopoulos and K. Waters. "Physically-based facial modeling, analysis, and animation", In *Journal of Visualization and Computer Animation*, Vol. 1, No. 4, pp. 73-80, 1990.

31. K. Waters. "A muscle model for animating three-dimensional facial expressions", In *Proceedings of SIGGRAPH 87*, pp. 17-24, 1987.

32. Li-Yi Wei and Marc Levoy. "Fast Texture Synthesis Using Tree-Structured Vector Quantization", In Proceedings of SIGGRAPH 00, pp. 479-488, 2000.

33. L. Williams. "Performance driven facial animation", In *Proceedings of SIGGRAPH 90*, pp. 235-242, 1990.

Appendix

To compute the importance value of every vertex, we have empirically derived the following three rules: First, the importance of a vertex is proportional to the norm of displacement vector. Second, even a vertex with a small displacement is considered to be important if it has a neighboring vertex with a large displacement. Finally, a vertex of high importance is constrained by verbal expressions, and a vertex of low importance drives emotional expressions.

According to those rules, we compute the importance of each vertex in three steps. In the first two steps, two independent importance values are computed from the verbal key-models and the emotional key-models, respectively. Then, they are combined to give the importance in the final step. Let $p_1(v_i)$ and $e_1(v_i)$, $1 \leq i \leq n$ be the verbal and emotional importances, respectively. In the first step, these importances are computed from the maximum norms of the displacement vectors of each vertex v_i over their respective key-models:

$$\begin{aligned} p_1(v_i) &= \max_j(|v_i^{P_j} - v_i|) / \max_{j,k}(|v_k^{P_j} - v_k|), \text{ and} \\ e_1(v_i) &= \max_j(|v_i^{E_j} - v_i|) / \max_{j,k}(|v_k^{E_j} - v_k|), \end{aligned}$$

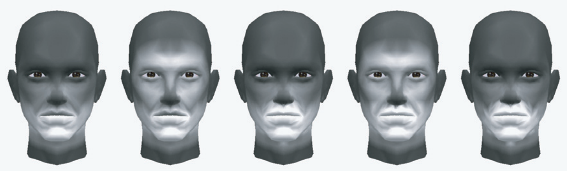
where $v_i^{P_j}$ and $v_i^{E_j}$ respectively denote the vertices in a verbal key-model P_j and an emotional key-model E_j corresponding to a vertex v_i in the base model. Note that we normalize the importances so that their values range from 0 to 1. We then propagate the importance value of each vertex to the neighboring vertices if it is big enough. Thus, in the second step, the importances $p_2(v_i)$ and $e_2(v_i)$ are obtained as follows:

$$\begin{aligned} p_2(v_i) &= \max(\{p_1(v_i)\} \cup \{p_1(v_j) \mid |v_i - v_j| < L_p,\ p_1(v_j) > S_1\}), \\ e_2(v_i) &= \max(\{e_1(v_i)\} \cup \{e_1(v_j) \mid |v_i - v_j| < L_e,\ e_1(v_j) > S_1\}), \end{aligned}$$

where S_1, L_p, and L_e are control parameters. In the final step, the importance α_i of the vertex v_i is obtained as follows:

$$\alpha_i = \begin{cases} p_2(v_i)(1 - e_2(v_i)), & \text{if } p_2(v_i) < S_2 \\ 1 - (1 - p_2(v_i))e_2(v_i), & \text{otherwise.} \end{cases} \quad (13)$$

Equation (13) adjusts importance values so that they are clustered near both extremes, that is, zero and one. Figure 14 shows the importance distributions for verbal and emotional expressions after the first, second, and third steps. Brighter regions indicate higher importance values.



(a) $p_1(\mathbf{v})$ (b) $e_1(\mathbf{v})$ (c) $p_2(\mathbf{v})$ (d) $e_2(\mathbf{v})$ (e) α

Figure 14: *Importance distributions*

© The Eurographics Association 2003.