Expressive Speech-Driven Facial Animation

YONG CAO
University of California, Los Angeles
University of Southern California, ICT
WEN C. TIEN
University of Southern California, ICT
PETROS FALOUTSOS
University of California, Los Angeles
and
FRÉDÉRIC PIGHIN
University of Southern California, ICT

Speech-driven facial motion synthesis is a well explored research topic. However, little has been done to model expressive visual behavior during speech. We address this issue using a machine learning approach that relies on a database of speech-related high-fidelity facial motions. From this training set, we derive a generative model of expressive facial motion that incorporates emotion control, while maintaining accurate lip-synching. The emotional content of the input speech can be manually specified by the user or automatically extracted from the audio signal using a Support Vector Machine classifier.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Animation*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Graph and tree search strategies*; I.5.1 [**Pattern Recognition**]: Models—*Statistical*.

General Terms: Algorithms

Additional Key Words and Phrases: Facial animation, lip synching, expression synthesis, independent component analysis

1. INTRODUCTION

Facial animation is an essential component of many applications such as video games, online virtual reporters, and other interactive human-computer interfaces. However, realistic facial animation remains

This work was partially supported by the National Science Foundation Grant CCF-0429983 and by the Department of the Army under contract number DAAD 19-99-D-0046. Any opinions, findings and conclusions or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the Department of the Army. Intel Corp., Microsoft Corp., and ATI Corp. also supported this work with equipment and software grants.
Author's addresses: Y. Cao, UCLA Computer Science Department, 4732 Boelter Hall, Los Angeles, CA 90095-1596; email: abingcao@cs.ucla.edu; W. C. Tien, 13274 Fiji Way, Suite 600, Marina del Rey, CA 90292; email: tien@ict.usc.edu; P. Faloutsos, Boelter Hall 4531-F, University of California, Los Angeles, Department of Computer Science, Los Angeles, CA 90095-1596; email: pfal@cs.ucla.edu; F. Pighin, Institute for Creative Technologies, USC, 4640 Admiratly Way, Marina del Rey, CA 90292; email: pighin@ict.usc.edu.
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 1515 Broadway, New York, NY 10036 USA, fax: +1 (212) 869-0481, or permissions@acm.org.
© 2005 ACM 0730-0301/05/1000-1283 $5.00

Fig. 1. Synthesized visual speech snapshots with emotion variations (sad and happy).

one of the most challenging problems in computer graphics. Hundreds of individual muscles work in coordinated fashion to generate complex facial expressions and speech. Even though the dynamics of each of these muscles is well understood, their combined effect is very difficult to simulate precisely and efficiently. Motion capture provides us with the ability to record high-fidelity facial motions. But this technique is mostly useful for specific shots since the recorded motions are difficult to modify. Motion capture by itself cannot be used for efficient facial animation synthesis unless it is combined with some manual or automated editing tools.

A great deal of research addresses this issue using data-driven or machine learning approaches. In these approaches, face movement is no longer viewed as the result of a complex biomechanical system, but rather as the output of some abstract functions that can be estimated by the analysis of recorded motion capture data. The strength of data-driven approaches is to provide a yardstick against which to compare the synthesized motions: the quality of synthesized motions can be evaluated by how much they deviate from the recorded data. Machine learning approaches typically extract statistical or generative models from sets of high-fidelity recorded data. Novel animations are then synthesized by performing suitable operations within the interpolation space of the learned models.

Data-driven approaches have yielded some of the most high-fidelity facial animation systems to date. However, most of this work has focused on the issue of lip-synching or, in other words, synthesizing a face motion that matches the content of an input speech. In reality, facial motion is not only a function of speech. For example, the same sentence spoken in an angry way would look very different if spoken in a happy way. In computer graphics, it is traditional to refer to emotions in order to classify facial expressions. For instance, a smiling face is often associated with happiness or raised eyebrows with surprise (see Figure 1). This association between emotions and facial expression has long been studied by psychologists, anthropologists, and animal biologists. Emotions provide only a coarse parameterization of the space of facial expressions. However, because emotions have intuitive meanings that we can all relate to, they provide a simple way of restricting the space of facial expressions to a few intuitive categories.

In this article, we present a set of novel techniques for automatically synthesizing speech-driven expressive facial animation. Our approach relies on a database of high-fidelity recorded facial motions

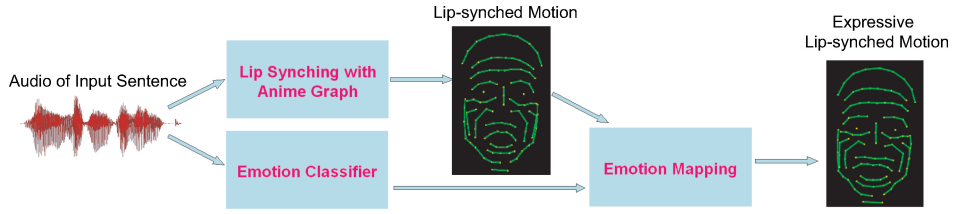

Fig. 2. High level overview of our approach.

which includes speech-related motions with variations across multiple emotions. The input of our system, shown in Figure 2, is a spoken utterance and a set of emotional tags. These emotional tags can be specified by a user or extracted from the speech signal using a classifier. Its output is a realistic facial animation that is synched to the input audio and conveys faithfully the specified emotions. Thus, our contributions are primarily twofold. First, we present a novel method for high-fidelity lip-synching based on motion capture data. Second, we develop a new approach to control the emotional content of the resulting facial motion.

The remainder of the article is organized as follows. Section 2 reviews the related literature. Section 3 presents the initial and laborious data collection and processing phase. In Section 4, we introduce our novel lip-synching approach. Section 5 describes how we map the emotional content of a synthesized facial motion to a desired emotion. Section 6 describes our emotion classifier. In Section 7, we present the details of our rendering method. Section 8 discusses our results. Lastly, Section 9 concludes this article and describes future work.

2. BACKGROUND

2.1 Speech Motion Synthesis

Speech motion synthesis, also known as lip-synching, refers to synthesizing facial motion that is synchronized with input speech. Most approaches first annotate the input speech signal based on standard speech unit such as phonemes. The phoneme annotation can be done manually, [Parke 1975] or automatically, [Lewis 1991; Bregler et al. 1997]. These speech units are then mapped to a set of lip poses, called *Visemes*. Visemes are the visual counterparts of phonemes which can be interpolated to produce smooth facial animation [Kalberer et al. 2002]. As an alternative representation of the speech units, Kshirsagar and Magnenat-Thalmann [2003] use syllables and their visual counterpart visyllables instead of phonemes and visemes.

The shape of the mouth during speech not only depends on the phoneme currently pronounced, but also on the phonemes coming before and after. This phenomenon, called *co-articulation*, is of particular importance for lip-synching. Co-articulation can affect up to 10 neighboring phonemes simultaneously. Considering that the English language has 46 phonemes, it is clear that a simple lookup table is not a practical solution.

There are many approaches that attempt to solve the lip-synching problem and model co-articulation. Most of the them fall in the following four categories.

Procedural approaches, the first category, model co-articulation using a set of rules [Pelachaud 1991; Cassell et al. 1994] or by establishing an order of visual importance over the phonemes [Kalberer et al. 2002].

The second category is the physics-based approach that use the laws of physics and muscle forces to drive the motion of the face. Although it is computationally expensive, it has been shown to be quite effective [Lee et al. 1995; Waters 1987; Albrecht et al. 2002].

Data-driven approaches that use a phoneme-segmented input speech signal to search within large databases of recorded motion and audio data for the closest matches are the third category. *Video Rewrite* [Bregler et al. 1997] is a representative example of such techniques. It constructs a large database of audiovisual basis units based on triphones. A new audiovisual sequence is constructed by concatenating the appropriate triphones from the database. In order to be useful, the method requires a large database of triphones which leads to a scaling problem.

A fourth class of techniques attempts to eliminate the need for large example databases by creating compact statistical models of face motion. Hidden-Markov and Gaussian mixture models are two machine learning techniques that are frequently used for this problem [Brook and Scott 1994; Masuko et al. 1998; Cohen and Massaro 1993]. For instance, *Voice Puppetry* [Brand 1999] uses a facial hidden markov model (HMM) to develop a mapping from voice to face. The model is learned from a face's observed dynamics using an entropy minimization method . It takes into account the position and the velocity of facial features and learns a probability distribution over the different facial configurations. Ezzat et al. [2002] develop a variant of the *Multidimensional Morphable Model* (MMM) that is represented as a set of optical flow vectors. It is used to describe images with local variations in shape and appearance. This model can be applied to statistically interpolate novel video frames corresponding to input speech segments. Although the technique could extend to 3D models, it has so far been tested only on the 2D cases. Saisan et al. [2004] learn a linear-dynamical system from recorded speech video clips. The system is driven by both a deterministic speech input and an unknown stochastic input. Because of the limitation of the model, only video clips for single word or very short sentences can be synthesized. Therefore, co-articulation can not be fully modeled with this approach.

2.2 Expression Synthesis

Facial expression reflects one's own motivations or emotional states. It can be used to supplement verbal communication and disclose personal feelings.

Facial expressions can be parameterized using a set of action units, such as the ones described in the *Facial Action Coding System* (FACS) [Ekman and Friesen 1978; Lien et al. 1998]. These action units define the facial poses and movements in small regions such as pulling lip corners. Video-based representations parameterize facial motion using images, while motion data-based approaches represent motion using the trajectories of mocap markers. Independently of the how the facial motion is parameterized, the main problem is how to control the expression of the face during speech.

To address this issue for video-based representations, researches have proposed factorization models to separate expressive facial motion from speech-related facial motion. Tenenbaum and Freeman [1999] present a general framework for separating content and style using a bilinear model. For example, they show how face images can be classified according to different personal identities or different facial poses and how to separate speech content from accent. Following a similar approach, Chuang et al. [2002] use a bilinear model to extract emotion and content from input video sequences of a talking head. However, the approach normalizes the signals, and as a result, important temporal information is lost. To distinguish more than two factors, Vasilescu and Terzopoulos [2003] extend *Principle Component Analysis* (PCA) and *Singular Value Decomposition* (SVD) to a novel multilinear reduction approach. They linearly factorize the facial image database according to expression, viewpoint, and illumination. Cao et al. [2003] decompose facial motion into style (emotion) and content (speech) using *Independent Component Analysis* (ICA). They also define a set of intuitive editing operations that can be performed on the data within the linear space defined by ICA.

In this article, we propose a novel data-driven approach for animating a high-quality 3D face model, inspired by the work of Kovar et al. [2002], Li et al. [2002], and Lee et al. [2002]. Most previous data-driven approaches are based on 2D motion data and they do not provide any control over the emotion.

Approaches based on statistical models [Brand 1999; Ezzat et al. 2002] produce great results, but they discard the original motion data after transforming it into compact generative statistical models. Although this compression reduces memory requirements, it also reduces, to some extent, the richness of the motion that such methods can produce. In contrast, our method preserves the entire database of motions. We organize the data into an efficient data structure that allows a novel search algorithm to locate appropriate motion segments. In contrast to previous techniques, we also develop an emotion mapping model that allows the user to specify the apparent emotional state of the speaking avatar. Lastly, we implement a *support vector machine* (SVM) classifier that allows us to automatically detect the emotional content of arbitrary input utterances.

3. DATA COLLECTION AND PROCESSING

3.1 Data Collection

To ensure high quality data, we hired a professional actor to perform during a series of motion capture sessions. We captured the motion data using a Vicon 8 optical system. Eight cameras tracked 109 markers on the face of the performer at a rate of 120 frames per second. The audio data was recorded at the same time with a sample rate of 44.1KHz. The performer uttered 442 sentences under 5 different emotional states as follows: frustrated 59, happy 85, neutral 86, sad 100, and angry 112. The actor performed each sentence with three different expression variations for each emotion. Ten of the sentences were performed with all five emotional states. In total, our motion database consists of 1,326 utterances. For modeling purposes, we took 5 pictures of the performer's face from different angles for each of 12 different expressions.

3.2 Data Processing

Geometric Face Model. Pighin et al. [1998] developed an imaged-based rendering technique that constructs high-quality 3D face models from photographs. We applied this technique to photographs of the performer and constructed a separate face model for each of the emotions we consider. In addition, Pighin et al's method [1998] produces a high-quality texture map of the face which we apply to our 3D geometric model to produce a realistic result.

Motion Data. Raw motion capture data typically contains noise, gaps, and errors. It, therefore, requires significant manual and automatic processing to fill in the gaps and compensate for the noise and errors. We semi-automatically labeled all motion sequences to ensure proper correspondence between the markers. Then, we used DIVA [House of Moves Inc.] to clean up the motions and fill in possible gaps. Lastly, the markers and their motion data were organized into three different motion groups: rigid head motion, upper face motion (eyebrows), and lip motion.

Audio Data. The audio data was down-sampled to 16KHz and segmented into sentence-long pieces. These sentences were then synchronized with the corresponding motion data.

The recorded data amounts to 53 minutes. The shortest sentence is a few seconds long, while the longest sentence has a duration of 15 seconds. The entire processing phase required two man-months of work.

4. LIP-SYNCHING

An important contribution of our facial animation system is its ability to produce accurate lip-synching while offering control of the emotional content of the facial motion. Although we can consider lip-synching and emotion modeling as separate parts, the data representation and algorithm that we propose are designed to satisfy the requirements of both problems. In this section, we describe in detail how we organize and use the data to perform accurate lip-synching. In a later section, we describe our

emotion mapping technique and the way it connects with the lip-synching module to produce accurate and expressive facial motion.

Lip-synching with correct co-articulation is a tedious task. We propose a novel graph structure called *Anime Graph* and a search-based technique that produce high quality speech-driven lip-synching.

4.1 Building the Anime Graph

Our training database consists of sentence-long utterances and their associated motion capture data. The motion of each of these utterances is first compressed using principal components analysis (PCA). We retain enough components to capture 95% of the variance of the motion. It is then processed with independent component analysis (ICA) [Hyvarinen et al. 2001] to provide a semantical decomposition of the data. This decomposition becomes critical later, at the emotion mapping stage. We then automatically segment each utterance into a sequence of phonemes using *Festival* [Speech Group, Carnegie Mellon University]. Since the audio is synchronized with the motion, we can easily extract the trajectories of the markers that correspond to each phoneme in the sequence. Each recorded utterance is converted to a sequence of nodes with one node per phoneme which we call *animes*. An anime captures a phoneme instance and contains a phoneme label, the associated motion segment, and other audio information. Like a viseme, an anime is a visual representation of a phoneme. Unlike a viseme that captures a single frame of a facial pose, an anime holds a number of motion frames. The number of frames depends on the sampling frequency (120fps or 60fps) and the duration of the phoneme. To organize these sequences of phoneme/anime pairs into a useful structure, we need to associate each pair with an appropriate feature vector that we can later use during matching and searching.

For each phoneme P in the sequence, we extract a prosody feature vector $\mathbf{C}$ using RASTA-PLP [International Computer Science Institute, Berkeley, CA]. This vector consists of the trajectories of the first nine parameters returned by RASTA-PLP. In addition, each example utterance has an emotion label E that indicates one of five emotions: Happy, Angry, Neutral, Sad, Frustrated. All phonemes in a sequence share the same label E.

We can now transform a phoneme into an *anime node*, a, by grouping the previous pieces together as follows:

$$a =< P, \mathbf{C}, \mathbf{M}, E >, \tag{1}$$

where P is a phoneme label, $\mathbf{C}$ the trajectories of the prosody features, $\mathbf{M}$ the compressed anime motion curves, and E the emotion label. Thus, each utterance in our training data-base becomes an anime node sequence. Within an anime node sequence, we connect two adjacent animes with a directed edge. Each node sequence is then converted into a directed link list. The set of link lists that correspond to our recorded set of utterances forms the *Anime Graph*.

4.2 Lip Motion Synthesis

Our lip motion synthesis approach is summarized in Figure 3. Given an input utterance, we search in our example database for a motion that satisfies a set of matching criteria. We first compute the emotion label of the input utterance, E, automatically using a support vector machine classifier described in Section 6. We then segment the input speech into a phoneme sequence using *Festival*. For each phoneme in the sequence, we assemble a *search node*, sn, that consists of a phoneme label P, the prosody features $\mathbf{C}$ (from RASTA-PLP) and an emotion label E.

$$sn =< P, \mathbf{C}, E > .$$

Thus, we transform the input utterance into a sequence of search nodes,

$$S_i = sn_1, \cdots, sn_n$$

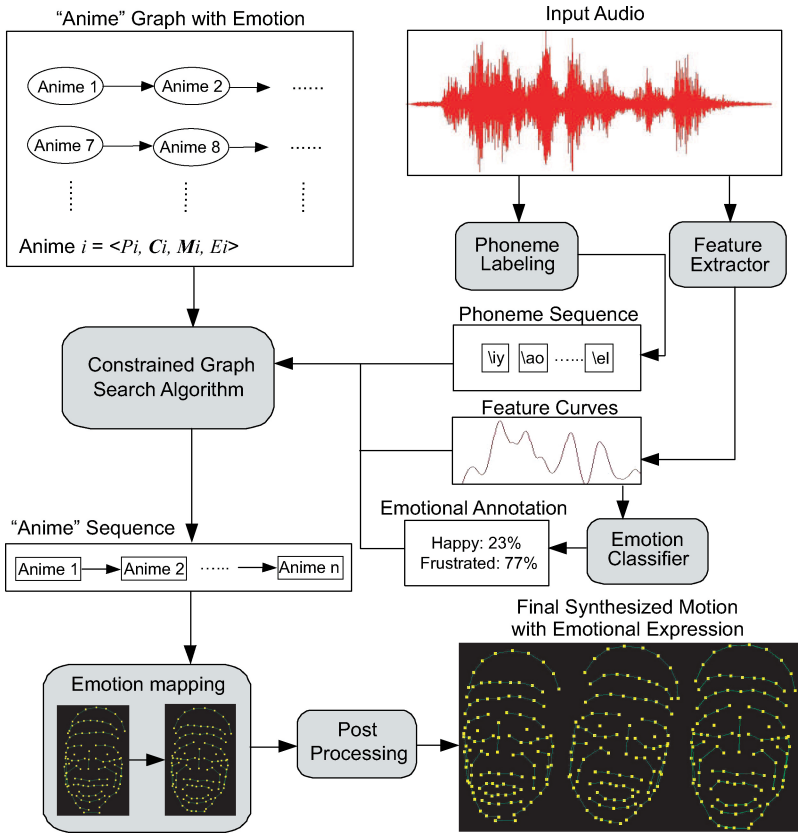

Fig. 3. Motion synthesis process.

which we can use to search in the Anime Graph for a matching sequence of anime nodes. A matching sequence, $S_m = a_1, .., a_n$, must have the same phoneme labels as the search sequence S_i, that is, $(a_i.P = sn_i.P, 1 \leq i \leq n)$. There are normally multiple matching sequences. To find the best matching sequence, we develop a set of evaluation criteria. Before we can describe them, we need to define the concept of a jump match.

Unless the input utterance has the same speech content as one of the recorded utterances, any matching sequence includes nodes from different connected branches in the graph. When there are two adjacent nodes in the sequence, (a_i, a_{i+1}) from different branches of the Anime Graph, we have what we call a *jump match*. A jump match indicates that the associated motions, $a_i.\mathbf{M}$ and $a_{i+1}.\mathbf{M}$, may not join smoothly. In this case, the resulting speech motion may have the wrong co-articulation. Figure 4 shows an example of a jump match. Phonemes P_1 and P_3 of the input sequence are matched on connected nodes on the top branch of the graph, while P_2 needs a jump match to the second branch.

We can now define our search criteria. Our search algorithm minimizes the following costs, presented in order of highest priority: a) number of jump matches (b) jump match cost and (c) audio feature curve difference. Intuitively speaking, the algorithm first minimizes the number of jump matches to ensure the best possible co-articulation effect. Among sequences that have the same number of jump matches, it chooses the one with minimum jump cost. If there are still multiple choices, it chooses the one that best matches the audio curves.

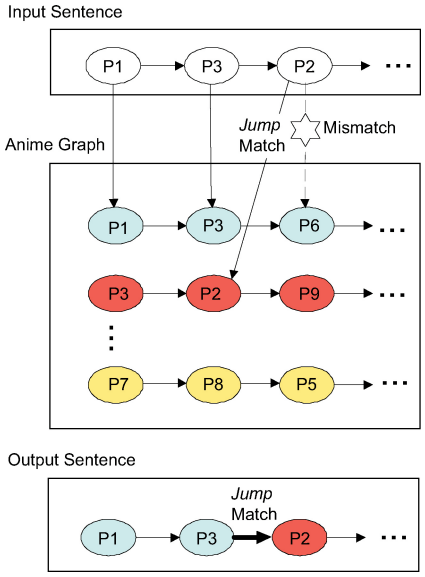

Fig. 4. Jump match: Phoneme P_2 matches on a different branch of the graph than P_1 and P_3.

Given a matching sequence of n anime nodes, our cost function C is defined as follows,

$$C = w_1 \sum_{i=1}^{n-1} C_i^{Jump} + w_2 \sum_{i=1}^{n-1} C_i^{Jumpcost} + w_3 \sum_{i=1}^{n} C_i^{Audio}, \tag{2}$$

where the weights w_i reflect the priorities of the associated costs. We experimentally choose w_1, w_2, and w_3 to be 100, 1, and 0.01, respectively. In Equation (2) costs C_i^{Jump}, $C_i^{Jumpcost}$, and C_i^{Audio} are defined as follows:

$$C_i^{Jump} = \begin{cases} 0, & \text{if perfect match,} \\ 1, & \text{if jump match,} \end{cases} \tag{3}$$

$$C_i^{Jumpcost} = Dist(\mathbf{M}_i(t_{end}), \mathbf{M}_{i+1}(t_{start})), \tag{4}$$

$$C_i^{Audio} = \frac{1}{m} \sum_{k=1}^{m} \left\| \mathbf{C}_i^{input}(k) - L(\mathbf{C}_i^{anime})(k) \right\|. \tag{5}$$

In Equation (4), function $Dist(\mathbf{x}, \mathbf{y})$ is the *Root Mean Square* (RMS) distance between two motion frames $\mathbf{x}$ and $\mathbf{y}$, and t_{end} and t_{start} are, respectively, the time stamps of the first and last frames of a motion. In Equation (5), L is a *Dynamic Time-Warping* (DTW) [Sankoff and Kruskal 1983] operator that warps an audio feature curve $\mathbf{C}_i^{anime}$ in order to match the input audio curve $\mathbf{C}_i^{input}$.

There exist many constraint-based or cost-based graph search algorithms. We have chosen to implement a *Priority-Queue* algorithm which is a variation of the Branch and Bound algorithm for tree-like structures [Sahni 1999]. Our Priority-Queue algorithm takes an input search sequence $S_i = (sn_1 sn_2 \cdots sn_n)$ and generates a matching anime sequence $S_a = (a_1 a_2 \cdots a_n)$ by taking the following steps.

(1) *Initialization*. Set a variable *mincost* to infinity and create an empty queue PQ. Find all anime nodes, a_i', with phoneme label $sn_1.P$. For each of these nodes, a_i', create a queue node $q_i =< seq, c, index >$,

where *seq* is an anime sequence that contains only anime node (a'_i), c is the matching cost of sequence *seq* defined by Equation (2) and *index* is set to 1. Put all nodes q_i in the queue PQ.

(2) *Main search step.* In PQ, find and remove the node $q_{\min}$ with the minimum cost $q_{\min}.c$. In the Anime Graph, find all anime nodes, $a'_1, a'_2, \cdots$, with phoneme label $sn_{q_{\min}.index+1}.P$. Create a queue node q_i for each of these anime nodes a'_i. Set $q_i.seq = (q_{\min}.seq, a'_i)$, set $q_i.index = q_{\min}.index + 1$, and compute the matching cost $q_i.c$ of sequence $q_i.seq$. Add these nodes to the queue PQ.

(3) *Queue pruning.* If at Step (2) $q_{\min}.index$ equals to n, then we have found a matching anime sequence for the input search sequence. Delete any node q_i from PQ for which $q_i.c$ is larger than $q_{\min}.c$. If $q_{\min}.c$ is less than *mincost*, set the resulting anime sequence S_a to $q_{\min}.seq$ and set *minscost* to $q_{\min}.c$.

(4) If queue PQ is empty, return S_a as the output and terminate the algorithm. Otherwise, go to Step (2).

The above algorithm returns the optimal sequence of anime nodes with respect to the cost function. However, the size of our graph (approximately 10,000 nodes) and the average length of the input search sequences (30) prevents it from running in real time. To improve the efficiency of the search, we consider only 3–6 phonemes ahead of the current phoneme, effectively bounding the length of the input sequence. Thus, we trade off quality for efficiency and find a local optimum that consists of shorter optimal sequences.

The resulting anime sequence, $S_a = a_1, .., a_n$, provides a sequence of motion segments: $a_1.\mathbf{M}, \ldots, a_n.\mathbf{M}$. The next section describes how to join these motion segments into a continuous facial motion.

4.3 Postprocessing

Producing continuous facial motion from an input search sequence, S_i, and the corresponding anime sequence, S_a, requires three steps: time-warping, blending and smoothing.

Time-Warping. The duration and timing of the phonemes in the anime sequence, a_i, is, in general, different from those of the corresponding phonemes in the input sequence, sn_i. To align them, we apply the *Dynamic Time-Warping* algorithm on the associated audio feature curves, $(sn_i.\mathbf{C}, a_i.C)$. We then apply the resulting time-warping function to the associated motion curves $a_i.\mathbf{M}$.

Blending. The quality of the continuous facial motion depends significantly on how we string together the sequence of motion segments. Connecting the motion segments of two anime nodes a_i and a_{i+1} is trivial if these nodes are connected within the Anime Graph. If not, then they correspond to a jump match and may not join smoothly. For such nodes, we search the Anime Graph for any pair of connected nodes that capture a transition from phoneme $a_i.P$ to phoneme $a_{i+1}.P$. If such nodes, a_m and a_n, exist, then the associated motion curves, $a_m.\mathbf{M}$ and $a_n.\mathbf{M}$, join properly. They essentially serve as an example of how the motion of phoneme $a_i.P$ transitions to the motion of $a_{i+1}.P$. We use this example to improve the way we join $a_i.\mathbf{M}$ and $a_{i+1}.\mathbf{M}$. First, we time-warp $a_m.\mathbf{M}$ and $a_n.\mathbf{M}$ based on the audio curves so that each aligns properly to the motions $a_i.\mathbf{M}$ and $a_{i+1}.\mathbf{M}$, respectively. We then create two motion curves $\mathbf{M}_{anime}$ and $\mathbf{M}_{help}$ by concatenating the time-warped motions as follows: $\mathbf{M}_{anime} =< a_i.\mathbf{M}, a_{i+1}.\mathbf{M} >$ and $\mathbf{M}_{help} =< a_m.\mathbf{M}, a_n.\mathbf{M} >$, where operator "$< *, * >$" indicates concatenation (sequencing) of motion frames. Note that since $a_m.\mathbf{M}$ and $a_n.\mathbf{M}$ join smoothly, $\mathbf{M}_{help}$ is a smooth curve. The final curve set is a piecewise linear blend of the two curves sets:

$$\mathbf{M}(t) = (1-w)\mathbf{M}_{anime}(t) + w\mathbf{M}_{help}(t),$$
$$w = \begin{cases} (t - t_{start})/(t_{joint} - t_{start}), & t \le t_{joint} \\ (t_{end} - t)/(t_{end} - t_{joint}), & t > t_{joint}, \end{cases}$$

where t_{start} and t_{end} are, respectively, the time stamps of the first and last frames of motions $\mathbf{M}_{anime}$ and $\mathbf{M}_{help}$, and t_{joint} is the time stamp of the frame where $a_i.\mathbf{M}$ and $a_{i+1}.\mathbf{M}$ should join.

When we cannot find a pair of connected anime nodes a_m and a_n, we proceed with the following steps. We collect the next q frames of motion following the animes that proceed a_i in the Anime Graph where q is the number of frames of motion $a_{i+1}.\mathbf{M}$. We denote these frames as $\mathbf{M}_i$. Similarly, we collect the p frames of motion that precede anime a_{i+1} in the Anime Graph and denote them as $\mathbf{M}_{i+1}$, where p is the number of frames of motion $a_i.\mathbf{M}$. If such frames do not exist because a_i doesn't have a child anime or a_{i+1} doesn't have a parent anime, we create them based on the velocity of the motion curves. We then create the motion curves $\mathbf{M}'_i =< a_i.\mathbf{M},\ \mathbf{M}_i >$ and $\mathbf{M}'_{i+1} =< \mathbf{M}_{i+1},\ a_{i+1}.\mathbf{M} >$. Motions $\mathbf{M}'_i$ and $\mathbf{M}'_{i+1}$ have the same number of frames, $p+q$, and are linearly blended together to produce the final transition from $a_i.\mathbf{M}$ to $a_{i+1}.\mathbf{M}$:

$$\begin{aligned}\mathbf{M}(t) &= (1-w)\mathbf{M}'_i(t) + w\mathbf{M}'_{i+1}(t),\\ w &= (t - t_{start})/(t_{end} - t_{start}),\end{aligned}$$

where t_{start} and t_{end} are, respectively, the time stamps of the first and last frames of motions $\mathbf{M}'_i$ and $\mathbf{M}'_{i+1}$.

Smoothing. The blending stage creates continuous motion curves for the entire utterance. However, jump matches often introduce high frequencies that create visible artifacts in the resulting motion. To eliminate them, we apply a low-pass filter. The cutoff frequency of the filter is crucial since it can significantly affect the motion. To ensure that only the undesirable frequencies are eliminated, we learn a suitable cutoff frequency from the data. We scan the entire motion database, and for each of the independent components of the motion curves, we identify the range of frequencies that contain 99% of the total energy of that component. The highest frequency of that range is the cutoff frequency of our filter.

5. EMOTION MAPPING

Perhaps the most interesting contribution of our work is the ability of our system to model and control the emotional content of the generated facial motion. Our emotion model is learned from a rich set of recorded motions that exhibit variation across multiple emotions. This section describes how we build this model and how use it to manipulate facial expression within a synthesized speech motion.

Our model considers disconnected emotion spaces. In particular, each motion in our database belongs to a specific emotion: neutral, sad, angry, happy, and frustrated. Consider two motions $\mathbf{M}^i$,$\mathbf{M}^j$ that have the same speech content but different emotions, i, j. The goal of our emotion mapping approach is to learn a transformation that maps $\mathbf{M}^i$ to $\mathbf{M}^j$:

$$\mathbf{M}^i = T_{ij}(\mathbf{M}^i). \tag{6}$$

To learn this mapping from data, we record a training set of speech-related motions so that each of m sentences has an incarnation in all five emotion spaces. By comparing how the motion $\mathbf{M}$ varies across these emotion spaces for every pair of motions, $(\mathbf{M}^i_k, \mathbf{M}^j_k),\ 1 \le k \le m$, we can learn a transformation that maps motions from emotion space i to emotion space j. This mapping is built in two stages. First, we take the pair of motions and decompose them into two types of components, one that captures the speech content, and the other one that captures emotion. This decomposition is performed using a recently-developed technique by Cao et al. [2003]. We then create a set of correspondences between the two emotion components. From this set of sparse correspondences, we learn a smooth mapping through a *Radial Basis Function* (RBF) approximation [Buhmann 2003]. An overview of our emotion-mapping learning approach is shown in Figure 5.

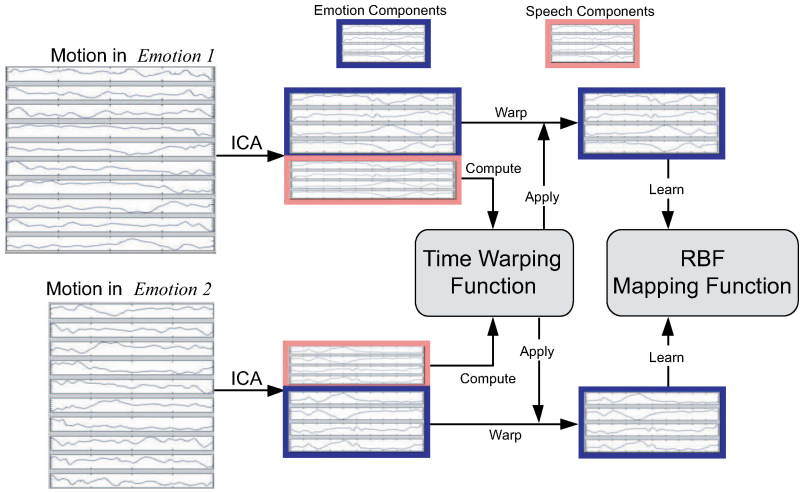

Fig. 5. Learning emotion mapping.

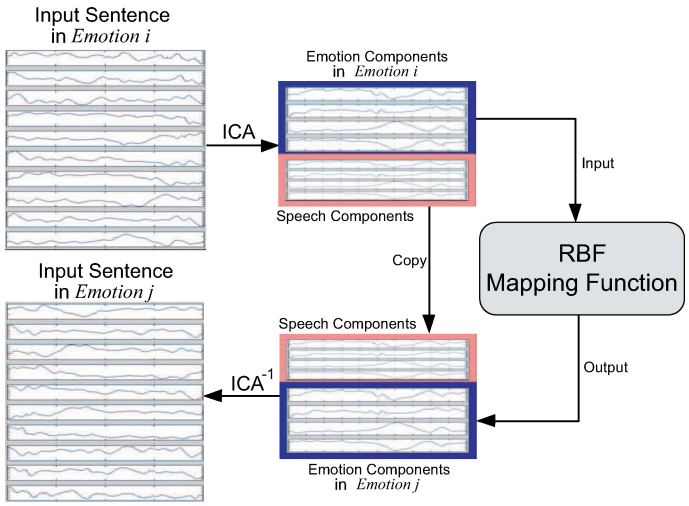

Fig. 6. Performing emotion mapping.

Figure 6 shows how we use the resulting mapping models to change the emotion of a facial motion. We essentially perform two projections. The first projection splits the facial motion into speech components and emotion components. The second projection takes as input emotion components from one emotion space and maps them onto another emotion space using the mapping function we learned from the data. We then combine the original speech components with the new emotion components to produce the final motion. ICA is central to our algorithm since it allows us to separate style (emotion) from content (speech).

The rest of this section describes our approach in detail.

5.1 Motion Decomposition with ICA

In order to decompose a facial motion into speech components and emotion components, we use Independent Component Analysis (ICA) [Hyvarinen et al. 2001], following the approach described in Cao et al. [2003]. This process decomposes speech motions into a set of sources that have intuitive meaning. In particular, we separate the data into style and content components. In our case, we equate style with expressiveness or emotion and content with speech motion. While Cao et al. [2003] edit the emotion components interactively by scaling or translating the corresponding time series, we use this decomposition to learn a mapping between emotion components belonging to different emotion spaces.

5.1.1 *Independent Component Analysis.* To understand the motivation behind ICA, let us first start with the following problem quoted from Hyvarinen and Oja [2000].

> Imagine that you are in a room where two people are speaking simultaneously. You have two microphones, which you hold in different locations. The microphones give you two recorded time signals, which we could denote by $x_1(t)$ and $x_2(t)$, with x_1 and x_2 the amplitudes, and t the time index. Each of these recorded signals is a weighted sum of the speech signals emitted by the two speakers, which we denote by $u_1(t)$ and $u_2(t)$. We could express this as a linear equation:
>
> $$x_1(t) = a_{11}u_1(t) + a_{12}u_2(t)$$
> $$x_2(t) = a_{21}u_1(t) + a_{22}u_2(t),$$
>
> where a_{11}, a_{12}, a_{21} and a_{22} are some parameters that depend on the distances of the microphones from the speakers. It would be very useful if we could now estimate the two original speech signals $u_1(t)$ and $u_2(t)$, using only the recorded signals $x_1(t)$ and $x_2(t)$. This is called the *cocktail-party problem*.

Such problems are often solved with a family of techniques that perform what is called *Blind Source Separation* (BSS). Independent Component Analysis is one of the most effective techniques of this kind.

We now describe the mathematics of ICA. Suppose we observe n linear mixtures $x_1, x_2, \ldots, x_n$ of n independent sources $u_1, u_2, \ldots, u_n$,

$$x_j = \sum_{i=1}^{n} a_{ji}u_i, 1 \leq i, j \leq n,$$

or in matrix notation

$$\mathbf{x} = \mathbf{A}\mathbf{u}, \tag{7}$$

where the matrix of coefficients, $\mathbf{A}$, is called mixing matrix. This formulation is different from the notation in the cocktail-party problem since we drop the time index t. In the ICA model, we view each mixture x_i as well as each independent source u_i as a random variable. The observed values $x_i(t)$, for example, the microphone signals in the cocktail-party problem, are thus samples of the associated random variable.

Given a set of observed random variables, x_i, the goal of ICA is to estimate the independent components, u_i, and the mixing matrix, $\mathbf{A}$. This is achieved by maximizing the statistical independence between the source variables. In this context, statistical independence means non-gaussianity [Hyvarinen et al. 2001].

5.1.2 *Facial Motion Decomposition.* Applying ICA to recorded facial motion is straightforward. The motion is represented as a set of time series, $\{x_i(t), 1 \leq i \leq 3n,\}$, that represent the time-varying

Euclidean coordinates of n motion markers. Each frame of these time series can be considered as a sample of a multidimensional random variable x_i. We can then directly apply ICA on this set of random variables, x_i, using Equation (7) and obtain a set of independent components u_i. There are many implementations of ICA [Hyvarinen et al. 2001]. For our experiments, we use *FastICA* [FastICA].

To apply ICA to our entire set of N recorded motions, $\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_N$, we first concatenate them into a single motion $\mathbf{M}$, $\mathbf{M} = [\mathbf{M}_1|\mathbf{M}_2|\ldots|\mathbf{M}_m]$. We then apply ICA on the combined motion $\mathbf{M}$ to obtain the mixing matrix $\mathbf{A}$ and n independent components $u_1, u_2, \ldots u_n$, or in matrix form $\mathbf{u}$. In this context, $\mathbf{u}$ can be represented as

$$\mathbf{u} = [\mathbf{u}^1|\mathbf{u}^2|\ldots|\mathbf{u}^N], \tag{8}$$

where $\mathbf{u}^i$ is the corresponding independent components for motion $\mathbf{M}_i$, $1 \leq i \leq N$.

Note that ICA usually uses principle component analysis during a preprocessing step to reduce the dimension of the original data. In this step, we experimentally determine how many components to keep in order to preserve the subtleties of the motion. For most of our experiments, keeping enough components to cover 95%–98% of the variance proved to be sufficient.

5.1.3 *Semantic Meaning of Independent Components.* In the beginning of Section 5, we assumed that expressive speech motion is the result of two independent processes, one that controls speech content and another that controls emotion. Applying ICA to the motions of our training database produces a mixing matrix $\mathbf{A}$ and a set of n independent components $u_1, u_2, \ldots u_n$. We now need to classify each component,u_i, into one of the following two subsets: speech components $\mathbf{s}$ and emotion components $\mathbf{e}$.

For a pair of motions $(\mathbf{M}_p, \mathbf{M}_q)$, $1 \leq p, q \leq m$ that have the same speech content but different emotion, the corresponding independent components are (u_i^p, u_i^q). Since the two motions have the same content, then the differences they exhibit are due to the difference in emotion. We visually and numerically compare each pair of independent components (u_i^p, u_i^q). Those that differ significantly form the set of emotion components $\mathbf{e}$.

After identifying all emotion components, we can determine which of the remaining components are related to speech. We reconstruct a facial motion $\mathbf{M}_i'$ from a single component, u_i as follows:

$$\mathbf{M}_i' = \mathbf{A} \begin{bmatrix} 0 \\ \ldots \\ 0 \\ \mathbf{u}_i \\ 0 \\ \ldots \\ 0 \end{bmatrix}. \tag{9}$$

We then compare the reconstructed motion $\mathbf{M}_i'$ with the original motion $\mathbf{M}$. If $\mathbf{M}_i'$ captures most of the original mouth motion, then component u_i is mostly related to the motion of the mouth. Since the motion of the mouth is primarily related to speech, so is component u_i.

The details of how to classify an independent component can be found in Cao et al. [2003]. After the classification, a motion $\mathbf{M}^i$ in emotion space i is split into two subsets of independent components, $\mathbf{s}$ and $\mathbf{e}$. Subset $\mathbf{s}$ is associated with speech motion, subset $\mathbf{e}$ is associated with emotion.

5.2 Learning a Mapping Between Emotion Spaces

Given a motion $\mathbf{M}^i$ in emotion space i, we want to develop a mapping that will map the motion $\mathbf{M}^i$ to emotion space j. Using the decomposition mentioned earlier, we can reduce this mapping to a mapping at a frame time t between emotion components only: $\mathbf{e}^j(t) = \mathbf{F}_{i,j}(\mathbf{e}^i(t))$. To learn this mapping, we will use the m sentences in our database that have been spoken with all five emotions. For each pair

of emotion spaces, (i, j), we have $(\mathbf{M}_k^i, \mathbf{M}_k^j), 1 \leq k \leq m$ pairs of corresponding motions. To learn the emotion mapping from emotion space i to emotion space j, we first decompose the training motion pairs $\{(\mathbf{M}_1^i, \mathbf{M}_1^j), (\mathbf{M}_2^i, \mathbf{M}_2^j), \ldots, (\mathbf{M}_m^i, \mathbf{M}_m^j)\}$ using ICA as previously described. Thus, for each pair of motions $(\mathbf{M}_k^i, \mathbf{M}_k^j),\ 1 \leq k \leq m$, we obtain a pair of corresponding speech-emotion components: $\{\mathbf{s}_k^i, \mathbf{e}_k^i\}, \{\mathbf{s}_k^j, \mathbf{e}_k^j\}$.

At this stage, we need to find correspondences between the emotion components of the two motions, $\{\mathbf{e}_k^i, \mathbf{e}_k^j\}$. Since the motions $\mathbf{M}_k^i$ and $\mathbf{M}_k^j$ have the same speech content, the speech components $\mathbf{s}_k^i$ and $\mathbf{s}_k^j$ are similar. However, they might have timing differences. These differences occur because, when a sentence is pronounced with a different emotion, specific phonemes might occur at different points in time. To compensate for these timing discrepancies, we time-shift the emotion components using time correspondences learned from the speech components through Dynamic Time-Warping [Sankoff and Kruskal 1983]. By doing this, we make sure that the emotion components $\mathbf{e}_k^i$ and $\mathbf{e}_k^j$ are aligned in time. Note that computing the time-aligning function from the emotion components would not work since they can be quite different.

We now have a set of corresponding pairs of time-aligned emotion components between emotion spaces i, j: $\{(\mathbf{e}_1^i, \mathbf{e}_1^j), (\mathbf{e}_2^i, \mathbf{e}_2^j), \ldots (\mathbf{e}_m^i, \mathbf{e}_m^j)\}$. The number of frames within emotion components $\mathbf{e}_k^i$ and $\mathbf{e}_k^j$ is T_k. Therefore, we have a total of $T = \sum_{k=1}^{m} T_k$ corresponding samples $(\mathbf{e}_k^i(t), \mathbf{e}_k^j(t)),\ 1 \leq k \leq m,\ 1 \leq t \leq T_k$. For simplicity, we denote these correspondences as $(e_r^i, e_r^j),\ 1 \leq r \leq T$. We can now use a number of approaches to fit a continuous function over these correspondences and produce the desired mapping.

Radial Basis Functions [Buhmann 2003] is an intuitive and straightforward approach that fits a continuous model to a set of discreet samples. In our case, the RBF mapping function $\mathbf{F}$, is as follows:

$$\mathbf{e}^j(t) \;=\; \mathbf{F}_{i,j}(\mathbf{e}^i(t)) = \mathbf{e}^i(t) + \sum_{r=1}^{T} \mathbf{c}_{i,j}^r \phi(\|\mathbf{e}^i(t) - e_r^i\|), \tag{10}$$

$$\phi(x) \;=\; \left\{ \begin{array}{c} e^{-\frac{x}{d}} \\ \text{or} \\ e^{-\frac{x^2}{d}} \end{array} \right\}, \tag{11}$$

where $\mathbf{c}_{i,j}^r$ are the radial basis function coefficients, ϕ is the radial basis function, and d is the distance between the two closest samples.

To compute the coefficients $\mathbf{c}_{i,j}^r$ in Equation (10), we constrain the mapping $\mathbf{F}_{i,j}$ to interpolate the T corresponding samples $(e_r^i, e_r^j),\ 1 \leq r \leq T$ as follows:

$$\begin{aligned} e_1^j \;&=\; \mathbf{F}_{i,j}(e_1^i) \\ &\vdots \\ e_T^j \;&=\; \mathbf{F}_{i,j}(e_T^i). \end{aligned}$$

We solve this linear system for the coefficients, $c_{i,j}^r$, using LU decomposition [Press et al.]. To reduce the size of the system T, we cluster the samples. In our implementation, we use K-Means clustering to cut down the number of samples to 250 clusters. When the coefficients are computed, we can use Equation (10) to map arbitrary sentences from emotion space i to emotion space j as shown in Figure 6.

Our RBF-based machine learning approach accurately fits the training set. To examine how well it performs with input data that is not in the training set (generalization), we perform the following cross-validation experiment.

Instead of using all pairs of corresponding motions, $(\mathbf{M}_k^i, \mathbf{M}_k^j), 1 \leq k \leq m$, to learn the mapping between emotion spaces i and j, we only use the first $m - 1$ pairs. After the mapping function is

Fig. 7. Emotion mapping visual cross-validation. A neutral motion (first row) is mapped to a happy motion (second row) using the proposed approach. A recorded happy motion with the same speech content is shown in row three for visual comparison.

learned, we use the last pair $(\mathbf{M}_m^i, \mathbf{M}_m^j)$ to test the effectiveness of the mapping. We map motion $\mathbf{M}_m^i$ to $\mathbf{M}_m'^j$ using the learned RBF-mapping function. The resulting motion $\mathbf{M}_m'^j$ and the original motion $\mathbf{M}_m^j$ are in the same emotion space j and have the same speech content. By comparing these two motions, we can validate the accuracy of our emotion mapping process.

A visual comparison between five frames of motion $\mathbf{M}_m'^j$ (second row) and motion $\mathbf{M}_m^j$ (third row) is shown in Figure 7. To compare the two motions numerically, we first normalize the motions by fitting them into a unit cube. Then, we use Root Mean Square (RMS) distance to measure the average distance between corresponding frames. For the case shown in Figure 7, the RMS distance between the two motions is 0.0117. In conclusion, our emotion mapping approach works effectively.

6. EMOTION CLASSIFICATION

An important feature of our system is its ability to automatically detect the emotional state of the input speech signals which enables it to automatically synthesize appropriate facial motion. Our emotion detection approach is based on supervised machine learning techniques and, in particular, classification.

For our purposes, we use a feature vector that includes the mean, variance, RMS error of the speech signal, and the first 500 coefficients of the signal's real cepstrum. We compute the feature vectors of our entire database of utterances and produce a training set that can be used with a variety of classification methods. We use a Support Vector Machine classifier [Burges 1998] and, in particular, the publicly available Libsvm software [Chang and Lin 2001a]. Libsvm supports multiclass classification and integrates a number of different techniques including C-SVC and nu-SVC [Chang and Lin 2001b].

Table I. Emotion Classification Results

Results on the training set

#cepstrum coeff.	50	100	500	1000
Linear Kernel	94.06%	99.17%	100.0%	100.0%
Radial Kernel	91.29%	96.40%	95.71%	96.27%

Results using half of the data for training and half for testing

#cepstrum coeff.	50	100	500	1000
Linear Kernel	88.95%	94.75%	96.96%	99.17%
Radial Kernel	85.63%	91.43%	90.88%	91.16%

Fig. 8. Face mesh: wireframe, gouraud shaded, and textured.

One of the most important features of Libsvm is that it can compute the probabilities with which a given utterance belongs to each of the emotional classes [Wu et al. 2003].

Experiments. The training set is scaled so that feature values range from -1 to 1. Scaling the data has been proved to increase the accuracy of the classification result. Libsvm allows us to use linear, polynomial, and radial kernels. Our experiments show that the linear and radial kernel $K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma|\mathbf{x}-\mathbf{y}|^2) = \exp(-|\mathbf{x}-\mathbf{y}|^2/2\sigma^2)$ perform the best. The default value for the parameter γ produces satisfactory results.

Table I provides a summary of our results. The accuracy of the classifier is above 95% for linear and radial kernels. Increasing the number of cepstrum coefficients in the feature vector increases the accuracy of the results. For our purposes, a linear kernel and 500 cepstrum coefficients are sufficient.

7. RENDERING

Our 3D face model was created from high quality photographs of the performer. Given three photographs (front facing, 45 degree right and 45 degree left), we used the photogrammetric technique introduced by Pighin et al. [1998] to construct a believable texture-mapped model of his face 8. The main advantage of this approach over a face scanner is that it provides a much higher resolution facial texture.

We map our motions on this face model using scattered data interpolation [Joshi et al. 2003]. Each marker point is associated with a location on the mesh as shown in Figure 8. This mapping is defined for a frame of motion that corresponds to the expression of the modeled mesh (a neutral face in our experiments). This frame provides us with a "mapping at rest", we call the *rest frame*. Each frame of the motion can subsequently be considered as a displacement from the rest frame. We apply this displacement to the mesh points associated with the markers. To extend the displacement to the whole face, we use a scattered data interpolation technique, Radial Basis Functions [Buhmann 2003]. This technique has already been used for modeling and animating faces [Pighin et al. 1998; Noh et al. 2000]. It provides a smooth interpolating function that extends the motion, specified at a finite set of locations

Fig. 9. Changing the emotion of an utterance from frustrated (top) to happy (bottom).

on the mesh, to the whole face. This smooth interpolation is inadequate in the parts of the face that are physically disconnected such as the upper and lower lips. We get around this problem by segmenting the face into a set of regions that reflect independent controls. We decided experimentally on the following set of regions: lower jaw, upper jaw, right eyelid, left eyelid, and upper face (forehead and eyebrows). Each region is associated with a subset of the markers that solely determines its motion. The different regions are smoothly blended at their boundaries to ensure a smooth deformed facial mesh.

8. RESULTS

We tested the effectiveness of our method with a number of experiments. Figure 1 demonstrates the ability of the system to produce a high-quality rendered animation of the synthesized expressive facial motion. Figure 10 shows snapshots of a novel synthesized motion. Our system produces a high-quality synthesized motion that follows the speech signal closely. Figure 9 demonstrates the ability of the system to map motion between different emotional states. In the example, a frustrated motion is mapped to a happy motion. Our system is capable of handling both short and long sentences and a variety of emotional states. The longest sentence we have synthesized to date is 62 seconds.

9. CONCLUSION AND FUTURE WORK

In this article, we present a novel approach that, given an input utterance, can automatically synthesize matching and expressive facial motion. Our approach organizes a large set of recorded motions and associated speech into a novel data structure, the Anime Graph. Lip-synching with accurate coarticulation is solved by the proposed constrained search of the Anime Graph. We design our algorithms such that the fidelity of the recorded data is preserved at every stage of the motion synthesis process.

We also introduce a novel approach that maps facial motion between different emotion spaces. Our approach is based on the decomposition proposed by Cao et al. [2003] that separates facial motion into content and emotion components. Using sentences from our database that share content but differ in emotion, we create a large set of correspondences that serve as samples of our mapping model. Using scattered data interpolation, we turn these samples into a powerful mapping model that can map any facial motion to the desired emotion space.

We are planning to deploy this animation system in various educational software where a synthetic mentor is needed. These educational tools impose another requirement on our system: real-time

Fig. 10. A long synthesized utterance with emotion variation.

performance. Efforts are under way to make this technique efficient enough for real-time motion synthesis and to interface it with a real-time speech synthesis system.

Despite the quality of the results, there is a lot of room for improvement. First, we would like to incorporate a visual attention model that controls the gaze of the speaking avatar. Lee et al. [2003] describes an interesting first attempt toward this problem. We are also investigating ways of producing motion in the absence of an input speech signal, for example, during a long pause. Synthesizing expressive head motion is also at the top our investigation list.

We realize that manipulating the emotional content of the motion without changing the content of the speech is not optimal. We are studying this problem in the wider context of combined facial motion and speech synthesis. We believe that there is much to be gained by studying and modeling these two highly correlated signals (motion and speech) in the same framework.

The main limitation of our approach is its dependency on motion and speech data. The quality of the output can only be as good as the motions in the training set. In addition, although our algorithm is automatic, building the database requires manual processing. In particular, during the data processing stage, we sometime have to manually fix the phoneme, labeling. To date, we have not found a completely reliable automatic phoneme-labeling software. Our contribution, however, is not in the field of phoneme labeling but in facial motion synthesis. Moreover, because it is costly to gather a second database of three-dimensional face motion, we were not able to test our approach with a second face. However, the techniques described in this article do not depend on the particular person that is being modeled. We believe that they would apply to any human face since human faces share a lot of similarities. In addition, we can retarget the facial motion produced by our method to other face models using a motion retargeting approach such as Pyun et al. [2003].

REFERENCES

ALBRECHT, I., HABER, J., AND PETER SEIDEL, H. 2002. Speech synchronization for physics-based facial animation. In *Proceedings of the International Conference on Computer Graphics, Visualization, and Computer Vision (WSCG'02)*. 9–16.

BRAND, M. 1999. Voice puppetry. In *Proceedings of ACM SIGGRAPH 1999*. ACM Press/Addison-Wesley Publishing Co. 21–28.

BREGLER, C., COVELL, M., AND SLANEY, M. 1997. Video rewrite: Driving visual speech with audio. In *SIGGRAPH 97 Conference Proceedings*. 353–360.

BROOK, N. AND SCOTT, S. 1994. Computer graphics animations of talking faces based on stochastic models. In *the International Symposium on Speech, Image Processing, and Neural Networkds*.

BUHMANN, M. D. 2003. *Radial Basis Functions : Theory and Implementations*. Cambridge University Press, Cambridge, UK.

BURGES, C. 1998. A tutorial on support vector machines for pattern recognition. *Data Mining Knowl. Disc. 2*, 2, 955–974.

CAO, Y., FALOUTSOS, P., AND PIGHIN, F. 2003. Unsupervised learning for speech motion editing. In *Proceedings of Eurographics / ACM SIGGRAPH Symposium on Computer Animation*. 225–231.

CASSELL, J., PELACHAUD, C., BADLER, N., STEEDMAN, M., ACHORN, B., BECKET, W., DOUVILLE, B., PREVOST, S., AND STONE, M. 1994. Animated conversation: Rule-based generation of facial expression, gesture and spoken intonation for multiple conversational agents. In *Proceedings of ACM SIGGRAPH 1994*.

CHANG, C.-C. AND LIN, C.-J. 2001a. *LIBSVM: A Library for Support Vector Machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

CHANG, C.-C. AND LIN, C.-J. 2001b. Training nu-support vector classifiers: Theory and algorithms. *Neural Computation*, 2119–2147.

CHUANG, E., DESHPANDE, H., AND BREGLER, C. 2002. Facial expression space learning. In *Proceedings of Pacific Graphics*.

COHEN, N. AND MASSARO, D. W. 1993. Modeling coarticulation in synthetic visual speech. In *Models and Techniques in Computer Animation*, N. M. Thalmann and D. Thalmann, Eds. Springer–Verlang, 139–156.

EKMAN, P. AND FRIESEN, W. 1978. *Manual for Facial Action Coding System*. Consulting Psychologists Press Inc., Palo Alto, CA.

EZZAT, T., GEIGER, G., AND POGGIO, T. 2002. Trainable videorealistic speech animation. In *Proceedings of ACM SIGGRPAH 2002*. ACM Press, 388–398.

FASTICA. Helsinki University of Technology, Laboratory of Computer Information Science, Neural Networks Research Centre. Available at `www.cis.hut.fi/projects/ica/fastica/`.

HOUSE OF MOVES INC. Diva software. Available at www.moves.com/moveshack/diva.htm.

HYVARINEN, A., KARHUNEN, J., AND OJA, E. 2001. *Independent Component Analysis*. John Wiley & Sons.

HYVARINEN, A. AND OJA, E. 2000. Independent component analysis: Algorithms and applications. *Neural Networks 13*, 411–430.

INTERNATIONAL COMPUTER SCIENCE INSTITUTE, BERKELEY, CA. Rasta software. Available at www.icsi.berkeley.edu/Speech/rasta.html.

JOSHI, P., TIEN, W. C., DESBRUN, M., AND PIGHIN, F. 2003. Learning controls for blend shape based realistic facial animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 187–192.

KALBERER, G. A., MUELLER, P., AND GOOL, L. V. 2002. Speech animation using viseme space. In *Vision, Modeling, and Visualization VMV 2002*. Akademische Verlagsgesellschaft Aka GmbH, Berlin, Germany. 463–470.

KOVAR, L., GLEICHER, M., AND PIGHIN, F. 2002. Motion graphs. In *Proceedings of ACM SIGGRAPH 2002*. ACM Press, 473–482.

KSHIRSAGAR, S. AND MAGNENAT-THALMANN, N. 2003. Visyllable based speech animation. In *Proceedings of Eurographics 2003*.

LEE, J., CHAI, J., REITSMA, P. S. A., HODGINS, J. K., AND POLLARD, N. S. 2002. Interactive control of avatars animated with human motion data. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 491–500.

LEE, S. P., BADLER, J. B., AND BADLER, N. I. 2003. Eyes alive. In *Proceedings of ACM SIGGRAPH 2003*. ACM Press, 637–644.

LEE, Y., TERZOPOULOS, D., AND WATERS, K. 1995. Realistic modeling for facial animation. In *SIGGRAPH 95 Conference Proceedings*. ACM SIGGRAPH, 55–62.

LEWIS, J. 1991. Autmated lip-sync: Background and techniques. *J. Visualiz. Comput. Animat. 2*, 118–122.

LI, Y., WANG, T., AND SHUM, H.-Y. 2002. Motion texture: A two-level statistical model for character motion synthesis. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 465–472.

LIEN, J., COHN, J., KANADE, T., AND LI, C. 1998. Automatic facial expression recognition based on FACS action units. In *Proceedings of the 3rd IEEE International Conference on Automatic Face and Gesture Recognition*. 390–395.

MASUKO, T., KOBAYASHI, T., TAMURA, M., MASUBUCHI, J., AND TOKUDA., K. 1998. Text-to-visual speech synthesis based on parameter generation from hmm. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP'98)*.

NOH, J., FIDALEO, D., AND NEUMANN, U. 2000. Animated deformations with radial basis functions. In *ACM Symposium on Virtual Realisty Software and Technology*. 166–174.

PARKE, F. 1975. A model for human faces that allows speech synchronized animation. *J. Comput. Graph. 1*, 1, 1–4.

PELACHAUD, C. 1991. Realistic face animation for speech. Ph.D. thesis, University of Pennsylvania.

PIGHIN, F., HECKER, J., LISCHINSKI, D., SZELISKI, R., AND SALESIN, D. 1998. Synthesizing realistic facial expressions from photographs. In *SIGGRAPH 98 Conference Proceedings*. ACM SIGGRAPH, 75–84.

PRESS, W. H., FLANNERY, B. P., TEUKOLSKY, S. A., AND VETTERLING, W. T. *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. Cambridge University Press, Cambridge, UK.

PYUN, H., KIM, Y., CHAE, W., KANG, H. W., AND SHIN, S. Y. 2003. An example-based approach for facial expression cloning. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 167–176.

SAHNI, S. 1999. *Data Structures, Algorithms, and Applications in C++*. McGraw-Hill Publishing Co., New York, NY.

SAISAN, P., BISSACCO, A., CHIUSO, A., AND SOATTO, S. 2004. Modeling and synthesis of facial motion driven by speech. In *European Conference on Computer Vision*. 456–467.

SANKOFF, D. AND KRUSKAL, J. B. 1983. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. CSLI Publications, Stanford University, Stanford, CA.

SPEECH GROUP, CARNEGIE MELLON UNIVERSITY. Festival software. Available at www.speech.cs.cmu.edu/festival.

TENENBAUM, J. B. AND FREEMAN, W. T. 1999. Separating style and content with bilinear models. *Neural Computat. J. 12*, 1247–1283.

VASILESCU, M. A. O. AND TERZOPOULOS, D. 2003. Multilinear subspace analysis of image ensembles. In *Conference on Computer Vision and Pattern Recognition*.

WATERS, K. 1987. A muscle model for animating three-dimensional facial expression. In *SIGGRAPH 87 Conference Proceedings*. ACM SIGGRAPH, Vol. 21. 17–24.

WU, T., LIN, C.-J., AND WENG, R. C. 2003. Probability estimates for multi-class classification by pairwise coupling. In *Proceedings of Neurol Information Processing System (NIPS'03)*.

Received July 2004; revised December 2004; accepted April 2005