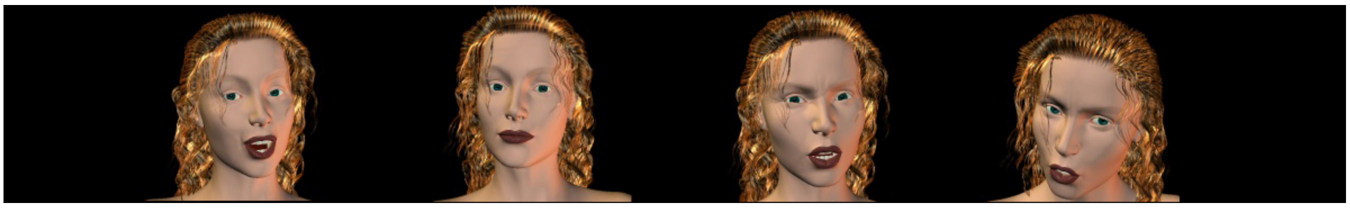
Mood Swings : Expressive Speech Animation

Erika Chuang[*]
Stanford University

Christoph Bregler[†]
New York University



Abstract

Motion-capture based facial animation has recently gained popularity in many applications, such as movies, video games, and human-computer interface designs. With the use of sophisticated facial motions from a human performer, animated characters are far more lively and convincing. However, editing motion data is difficult, limiting the potential of reusing the motion data for different tasks. To address this problem, statistical techniques has been applied to learn models of the facial motion in order to derive new motions based on the existing data. Most existing research focuses on audio-to-visual mapping and reordering of words, or on photo-realistically matching the synthesized face to the original performer. Little attention has been paid to modifying and controlling facial expression, or to mapping expressive motion onto other 3D characters.

This paper describes a method for creating expressive facial animation by extracting information from the expression axis of a speech performance. First, a statistical model for factoring the expression and visual speech is learned from video. This model can be used to analyze the facial expression of a new performance, or modify the facial expressions of an existing performance. With the addition of this analysis of the facial expression, the facial motion can be more effectively retargeted to another 3D face model. The blendshape retargeting technique is extended to include subsets of morph targets that belong to different facial expression groups. The proportion of each subset included in a final animation is weighted according to the expression information. The resulting animation conveys much more emotion than if only the motion vectors were used for retargeting. Finally, since head motion is very important in adding liveness to facial animation, we introduces an audio-driven synthesis technique for generating new head motion.

Keywords: Facial animation, expression, retargeting, motion

1 Introduction

Computer animated characters are now necessary components of computer games, movies, web pages, and various human computer interface designs. In order to make these animated characters lively and convincing, they require sophisticated facial expressions and motions. Traditionally, facial animation has been produced largely by skilled artists using manual keyframe techniques. Although it ensures the best quality animation, this process is slow and costly. While large studios or production houses can afford to hire hundreds of people to make feature films and movies, it is not feasible for low budget or interactive applications.

Recently, a great deal of research has been dedicated to motion capture based and performance driven methods, hoping to produce facial animation more efficiently. However, editing motion capture data is quite difficult, and no completely satisfying solution yet exists. Ideally an artist could freely edit the speech content, the emotional style, or the visual appearance of a character, while retaining the essence of the captured performance.

Most current research has focused on either data driven speech synthesis which changes the speech content and lip motion of a video sequence, or character animation which focuses on visual appearance and methods for retargeting geometric deformations from one face onto another. Relatively few methods exist for editing and retaining the expressive style of facial animation. While existing methods can produce photorealistic results, they focus on changing the content of "what" the character does, but not the style of "how" they do it, leaving the expression of captured data untouched. This paper addresses the need for expressive facial animation by introducing methods for both retargeting and head motion synthesis that have been explicitly designed to incorporate emotion.

Methods for editing content, style and appearance have been largely separated, with few attempts to build a complete systems for editing and retargeting all aspects of a facial animation. One of the keys to this problem lies in the choice of data representation. Content editors seek to preserve the identity of the original actor or actress, so they choose a representation that preserve the appearance of the data, often including complex high-dimensional texture models. Retargeting methods seek to preserve speech content while editing appearance, so they choose a representation such as marker positions that is general and adaptable to multiple face models. Unfortunately sparse models of 3D markers may not contain sufficient information to encode emotion. A complete system will need to choose a data representation that is compatible with these often contradictory goals. To address this need, and highlight the importance of designing algorithms that are compatible with the whole process, we introduce a framework for integrating content, style and appearance into a single editing pipeline.

This paper attempts to resolve the problem of data representation by choosing the middle ground. We neither choose a dense texture model, nor a simple sparse marker model. We instead analyze video texture information and extract concise higher level information that encodes facial expression. This expression information is used to augment a sparse geometric model when retargeting. In addition, our method allows artists to retain control over the process of interpreting various facial expressions. Input geometry is mapped implicitly to the output face models, rather than preserving explicit

[*]e-mail: echuang@graphics.stanford.edu
[†]e-mail: bregler@cat.nyu.edu

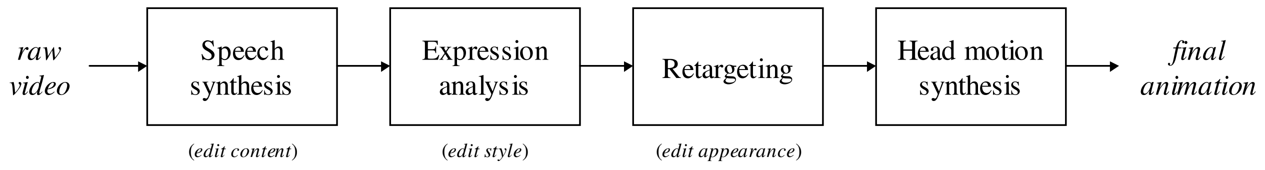


Figure 1: Editing pipeline for expressive speech animation. This work focuses on the last three modules, editing and retargeting expressive animation onto new characters.

geometric similarity.

Figure 1 illustrates the editing pipeline proposed in this work. The input performance is a video sequence of a talking face. The speech content of this sequence could be edited using any of several existing techniques. The resulting video is then analyzed using a statistical bilinear model, to factor emotional style and speech content into two components. This model essentially provides an expression shader with which we can modify the emotional style of the video. We next retarget the video sequence onto a 3D character. By using the extracted emotion vector to augment traditional shape information, our retargeting method preserves emotional style while allowing the characters appearance to be freely edited. Finally, since head motion is an important aspect of expression, we present a data driven synthesis technique that matches the characters emotional state. Since this work emphasizes the importance of expression, we focus on the last three modules, simply passing the input video's speech content directly to the expression synthesis module.

This paper contributes novel techniques for expressive retargeting as well as head motion synthesis. In addition, and perhaps more importantly, we demonstrate how these methods fit in a complete pipeline that allows the creation and editing of compelling facial animations.

2 Related work

Many research areas are relevant to this paper. We will organize them into the following categories based on the flow of the paper: speech content editing, facial expression analysis and modelling, facial motion retargeting, and head motion synthesis.

Statistical techniques has been applied to learn models of speech content in order to derive novel motion based on existing data. For instance, Video Rewrite [Bregler et al. 1997] applied machine learning technique to synthesize a talking face. To ensure a smooth transition between words, a triphone model is used for each viseme. The resulting system enables the same person to say things that he or she has never said before. Voice Puppetry [Brand 1999] learns a probability distribution of facial motion by applying a hidden markov model (HHM) to both the audio signal and the facial motion. Given novel audio input, the algorithm predicts the most likely trajectory of the facial motion. The synthesized motion can be further retargeted to a different face by image warping. Ezzat et al. [Ezzat et al. 2002] improved upon Video Rewrite by learning a set of prototype mouth shapes from the training data. This minimizes the storage required to keep all the video in the database. Many other research projects demonstrate similar types of work e.g. [Cossato 2002; Cohen and Massaro 1993; Kalberer and Gool 2001]. However, the research effort in data-driven facial synthesis has largely ignored emotion, and the resulting speech retains the same neutral expression as the input data.

A great deal of previous research does involves the study of appropriate models of facial expression. Most of the effort has been with regard to tracking and recognition of facial expressions, utilizing the static or short-term dynamics of single-unit facial expressions, such as a smile or a frown [Essa and Basu 1996; Essa and Pentland 1997; Tian et al. 2001]. A popular representation based on muscular movement [Ekman and Friesen 1978] is the Facial Action Coding System (FACS). However, rather than learning and recognition, our goal is editing and synthesis of an expressive talking face, where the long term dynamics of the entire facial configuration are important. A method of separating the contributions of content and emotion is needed. Cao applied Independent Component Analysis (ICA) for representing facial motion data, and extracted out the influence of facial expressions on speech by separating regions and layers that contribute to different factors [Cao et al. 2003]. The authors of this work have previously reported a bilinear model for factorizing the influence of speech content and facial expression [Chuang et al. 2002]. This paper makes use of the bilinear model for factorizing video in section 3, but considerably expands upon our prior method by reporting a complete pipeline for expressive animation including novel methods for retargeting and head motion synthesis.

This work uses a two factor model for expression analysis. Multilinear analysis involving more than two factors has been applied to facial recognition [Vasilescu and Terzopoulos 2003]. Although the method has not yet been applied to expression analysis, it seems a promising future area.

Techniques for retargeting facial motion fall into several categories. The first category uses explicit parameterization, where the parameters usually have values associated with physical dimensions, such as eye opening, eyebrow raising, mouth opening, etc. [Buck et al. 2000; Parke 1982; Ostermann 1998]. The second category of retargeting techniques directly applies the movement of facial marker data to the corresponding features on the target model. For vertices that have corresponding markers, the displacement vectors are simply re-normalized to match that of the target model. A interpolating deformation method is applied to compute the displacement of the intermediate vertices [Litwinowicz and Williams 1994; Noh and Neumann 2001]. A third category of facial retargeting parameterizes facial motion with blendshapes. At each time step, the facial state is a weighted combination of several basic expressions, sometimes called morph targets in commercial animation software [Kouadio et al. 1999; Pighin et al. 1998; Pyun et al. 2003; Joshi et al. 2003; Zhang et al. 2003]. No existing methods have specifically addressed the need to maintain emotional style during the process of retargeting.

We base our retargeting method on blendshapes, since the mapping is implicit rather than explicit, allowing broader and more intuitive artistic control over character appearance. Traditional motion capture systems and explicit retargeting may not be able to effectively capture and parameterize subtleties in facial expressions such as eyes squinting, wrinkles, or shadows cast by small changes in the face geometry. As we will show, implicit mapping allows these subtleties to be analyzed and encoded separately, but still used in the retargeting process. In addition, blendshape animation fits well the traditional mode of how animators work, as many commercial softwares already provides such a tool.

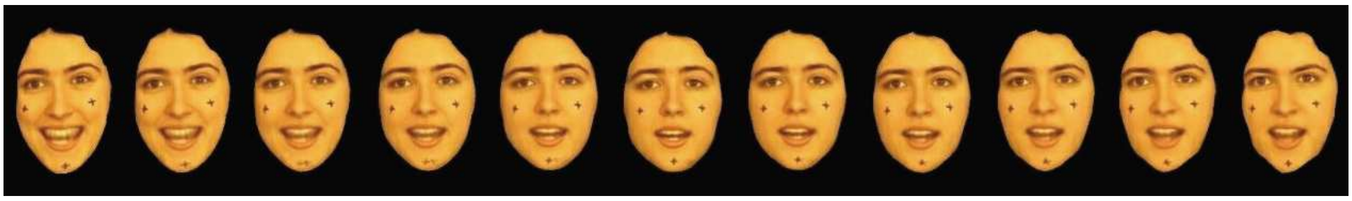


Figure 2: Interpolated facial expressions from happy to neutral, and from neutral to angry. The expression changes gradually, while the speech content maintains the same.

Head motion for animated faces has typically been produced either manually, randomly, or by a set of rule derived from communication studies. Ken Perlin proposed generating random noise for character movement [Perlin 1997]. Although surprisingly effective, viewers are eventually frustrated by the lack of correlation with the facial animation. Head-motion procedures that model speaker turns and interaction with a user have been proposed by several researchers [Cassell et al. 1994; Takeuchi and Nagao 1993; Poggi et al. 2000]. Most recently, [DeCarlo et al. 2002] proposed a new procedural system that enhances non-verbal aspects of speech with head motions. This work successfully models the correct behavior; however, since behavior can only be specified at a higher level, the animated avatars often lack the fine details of motion found in realistic characters. Lastly, some facial synthesis systems use a ”background” sequence, which contains real head motion from a different sequence, and align it with the synthesized facial motion. Unfortunately since the head motion and the face are not correlated, this motion is not always believable. This paper introduces a data driven method for synthesizing head motion that is correlated with the emotional content of the facial animation, and contains the fine details that are necessary for interesting animation. Although the parameterizations and challenges are very different from head motion, we draw inspiration from several recent articulated motion synthesis systems that have explored similar data driven methods [Arikan and Forsyth 2002; Pullen and Bregler 2002; Kovar et al. 2002; Li et al. 2002].

3 Facial expression analysis

In order to create animated characters with expressive facial motion, we need to create a mathematical model of expression. This work builds that model from video training data. Expression analysis gives us a tool to factor the contributions of expressive style and visual speech content. We use a bilinear model for this task [Tenenbaum and Freeman 2000]. Our analysis results in an emotional style vector used during retargeting. Additionally, given an input facial image, we can modify the facial expression to enhance the animation.

3.1 Data representation

We first represent images of a talking face by a statistical model of the shape and color appearance in a similar spirit as in Active Appearance Model [Cootes et al. 2001]. A set of n tracked feature points, $[f1,..fn]$, on the face define the face shape, represented as the x,y positions of the features in the vector $\vec{x}$, where $\vec{x} = [f_{x1}, f_{y1}, f_{xn}, f_{yn}]'$. We apply Principal Component Analysis (PCA) to the data from all the frames in the training set to obtain a representation with reduced dimensionality. Each sample shape can now be approximated using:

$$\vec{x} = \vec{x_0} + \mathbf{P}_s \cdot \vec{b_s}, \tag{1}$$

where $\vec{x_0}$ is the mean shape vector, $\mathbf{P}_s$ is the set of orthogonal modes of variation derived from the PCA, and $\vec{b_s}$ is a set of shape parameters.

For the texture, we first warp all the face images to the average shape, and sample the pixel values in the normalized image, denoted as $\vec{g}$. Similarly to the shape model, we apply PCA to the texture data:

$$\vec{g} = \vec{g_0} + \mathbf{P}_g \cdot \vec{b_g}, \tag{2}$$

where $\vec{g_0}$ is the mean gray-level vector, $\mathbf{P}_g$ is a set of orthogonal modes of variation and $\vec{b_g}$ is a set of color parameters.

The shape and appearance of the face can thus be summarized by the vectors $\vec{b_s}$ and $\vec{b_g}$. The facial configuration vector at any given frame is defined as:

$$\vec{y} = \begin{bmatrix} B_s \cdot \vec{b_s} \\ \vec{b_g} \end{bmatrix}. \tag{3}$$

B_s is a diagonal matrix with weights for the shape parameters, allowing for a difference in units between shape and gray values.

3.2 Model for facial expression

The bilinear model makes a simple assumption that the data is influenced only by two underlying factors. In this case, the factors are the visual speech component and the expression component. Bilinear models are two-factor models with the mathematical property of separability, and the outputs are linear in either factor when the other is held constant. Together, the two factors modulate each other's contributions multiplicatively, which allows rich interactions between them.

Mathematically, a bilinear model can be described as:

$$y_{k,s} = \vec{e_s}' \cdot \mathbf{W}_k \cdot \vec{c}, \tag{4}$$

where $y_{k,s}$ is the k^{th} component of the data vector, shown in equation 3, with a particular style s. $\vec{e_s}$ is the style vector, or expression vector, $\vec{c}$ is the content vector, and $\mathbf{W}_k$ is the basis matrix that describes the multiplicative relationship between the style and the content for the k-th component of the facial vector. For K facial components, we will need $\mathbf{W} = [W_1, ..., W_K]$. In this work, the style is the facial expression, and the content is the visual representation of speech, or viseme. We train our model using video sequences captured with each of three expressions: happy, angry, and neutral. The training algorithm is similar to [Chuang et al. 2002] and we refer the reader to their work for details. Training results in the models basis vectors, $\mathbf{W}$, as well as a set of expression vectors for happy, angry and neutral expressions, $\vec{e_h}, \vec{e_a}, \vec{e_n}$, that are used in the retargeting process.

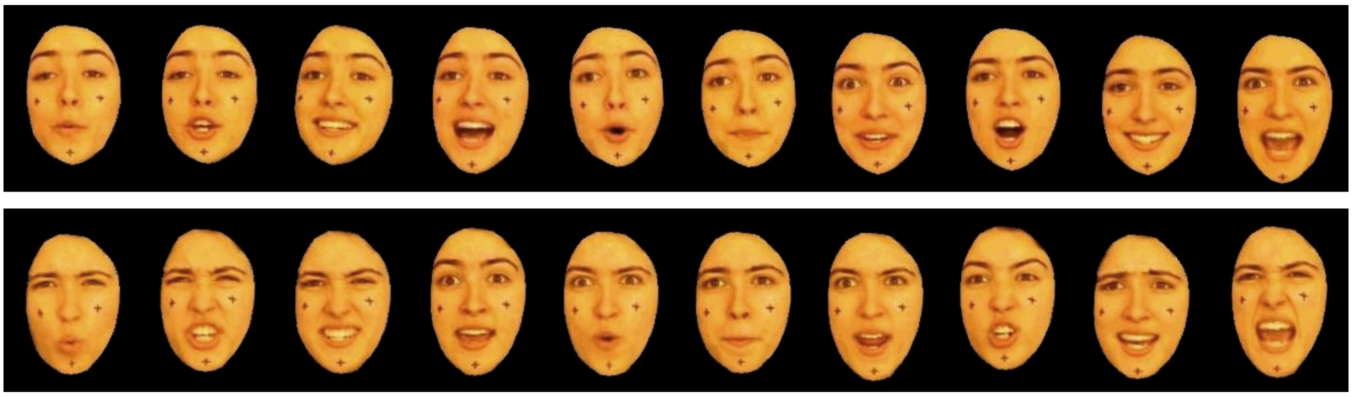


Figure 3: Examples of selected keyshapes for happy and angry expressions.

3.3 Modifying facial expression

The model described above can be used to modify facial expressions. Given a new sequence represented by the facial configuration shown in equation 3, $\vec{y}_{test}$, suppose we would like to analyze it and perform some modifications. Take the bilinear basis vector $\mathbf{W}$ from above, we need to solve for the expression vector $\vec{e}$ and the content vector $\vec{c}$. This can be done by iteratively solving for one factor, while keeping the other factor fixed. Given the input data $\vec{y}_{test}$ and using the average from equation 4 as an initial guess for the content vector $\vec{c}_0$, the expression vector is:

$$\vec{e}' = [[\mathbf{W} \cdot \vec{c}_0]^{VT}]^{-1} \cdot \vec{y}_{test}. \tag{5}$$

Similarly, the content vector is:

$$\vec{c} = [[\mathbf{W}^{VT} \cdot \vec{e}_0{}']^{VT}]^{-1} \cdot \vec{y}_{test}. \tag{6}$$

Here $\mathbf{X}^{VT}$ denotes the vector transpose of $\mathbf{X}$, and $\mathbf{X}^{-1}$ denotes the pseudo inverse of $\mathbf{X}$. We iterate over these two equations until reaching convergence.

This process factors the image sequence, providing the expression and content vectors for each frame. In order to modify the facial expression, the expression vector is simply replaced. By interpolating between the expression vectors obtained during training, we can obtain in-between expressions, such as halfway happy. For instance, the facial vector $\vec{y}_i$ with facial expression between *s1* and *s2* is given as:

$$\vec{y}_i = (\alpha \cdot \vec{e}_{s1} + (1-\alpha) \cdot \vec{e}_{s2}) \cdot \mathbf{W}_i \cdot \vec{c}, \tag{7}$$

where α specifies the amount of each target expression. The value of α should stay fairly close to the range between zero and one, for if it's outside this range, we are extrapolating the facial expressions. Extrapolating too far can produce an invalid expression. Figure 2 shows interpolated facial expressions from happy to neutral, and then to angry. The expressions change gradually, while the content remains the same. Notice that the changes in the locations of the facial features are subtle, the change in appearance is primarily due to the changes in small facial details encoded in the texture.

We now have both a method for modifying facial expressions, and a concise description of emotional state. This description is encoded in the expression vector, which we will make use of during retargeting.

4 Facial expression retargeting

To retarget an expressive talking face to a different face model, we choose to use a blendshape retargeting method. Many commercial animation software systems have built-in methods for blendshape animation using a set of predefined model poses. These basic poses are typically called morph targets. It is common to find commercial products of characters with pre-constructed morph targets built into the character. However, they are intended for making facial animation using manual keyframing, as opposed to for facial motion retargeting. To retarget captured, or synthesized, facial motion onto the pre-constructed morph targets, one would normally have to either build a face model that mimics the appearance of the motion captured source [Pighin et al. 1999], or parameterize the marker data, establish marker correspondences, and re-normalize the displacement, so that the motion can be used on a different face[Kouadio et al. 1999; Chai et al. 2003]. We introduce a method that neither requires building a complex 3D photo-realistic model for each person we want to capture from, nor uses parameters that require physical correspondences between the models. Instead, we select a set of prototype images called keyshapes from the training data, and allow an artist to design a corresponding set of keymorphs.

4.1 Keyshapes selection

An ideal set of keyshapes should cover all the possible variation in the training set, using as few shapes as possible. Ezzat et al. [Ezzat et al. 2002] described a method for automatically selecting prototype images, or keyshapes, by clustering. Instead, we use the data points that have the largest values when projected onto the principle axes found by Principle Component Analysis. We found that this performs slightly better, since clustering sometimes loses the extreme facial poses [Chuang and Bregler 2002]. Starting from the first principle axis, the one that has the largest eigenvalue, the frames with the maximum and the minimum projection onto each axis are chosen as key shapes. To pick k key shapes, we will use the first k/2 axes. Sometimes the extremum for one dimension may be the same or nearly the same data point as extremum for a different dimension. We compare the square difference of the two data points, and if the difference is below some threshold, eliminate the keyshape that was drawn from the lower dimension. The process of keyshape selection is repeated separately for each expression used in the training set for the bilinear model. For example, $\mathbf{B} = \{\vec{B}_1^{(s)}, \vec{B}_2^{(s)}, ..., \vec{B}_{k_s}^{(s)}\}$ represents a set of k_s keyshapes for expression s. There is not any correspondence in a mathematical sense between the keyshapes chosen for different expressions, nor do the number of keyshapes need to be the same for all expressions. However, they do often look similar. Figure 3 shows some examples of keyshapes for the happy and angry data sets. In our experiment, we find that for the purpose of retargeting, it is sufficient to use only the shapes of the face ($\vec{x}$ as described in section 3) for the selection of keyshapes.

4.2 Motion parameterization

Given a new sequence of input images where the facial expressions change over time, we need to first solve for the expression and speech content as described in section 3.3 to get the expression matrix and content matrices. Once we have these matrices, we can synthesize new sequences for each of the basic expressions (happy, angry, etc...). To represent these expression sequences in terms of the keyshapes for that expression as defined in the previous section, we first re-project back out the shape and texture vector using the inverse of equation 1 through 3. Here we only use the shape portion of the facial vector, $\vec{x}$. Each frame $\vec{x}^{(s)}(t)$ of the sequence associated with expression s can be decomposed into a linear combination of key shapes **B**, and weights $w_1, w_2, ..., w_{k_s}$.

$$\vec{x}^{(s)}(t) = \sum_{i=1}^{K_s} w_i(t) \cdot \vec{B}_i^{(s)}, \qquad w_i \geq 0. \tag{8}$$

The weights are found by minimizing the least-squares error subject to a non-negative constraint. Each frame of the input sequence is now represented as a linear combination of keyshapes, with different keyshapes and weights for each expression.

In our representation of **B**, there is no simple relationship between the above equation and the synthesis equation (equation 4) without going through several rounds of matrix transformation. One might wonder why a new basis **B** is needed for retargeting, and whether there is some natural relation between these bases. **W** is a mathematically optimal basis set computed in the context of the bilinear model. It needs to represent the data well, but there is no need for it to have semantic meaning to a human. In contrast, the keyshapes **B** must have semantic meaning since, as will be described in the next section, an artist must produce corresponding keymorphs. However **B** does not need to be “optimal”, it just needs to adequately represent the space of variations. **W** could not be used to replace **B**, since an artist could not interpret **W**. Given the complexity encoded in the bilinear model, we doubt that **B** could be used to replace **W** in a straightforward way. An interesting question for future exploration is whether some other set of basis functions could be found which are both semantically meaningful and suitably powerful, such that they would be appropriate for both domains.

4.3 Keymorphs and motion retargeting

In order to relate a captured sequence of images to our output 3D model we build a new set of morph targets, $\mathbf{G} = [\vec{G}_1^{(s)}, ..., \vec{G}_K^{(s)}]$, that resemble the look of the keyshapes chosen from the training data. These are built using the pre-defined morph targets, and we call these keymorphs. Keymorphs are similar to the set of primitive morph targets defined in the character model. However, the mapping between the keymorphs and the keyshapes are constructed to be one-to-one, whereas no natural mapping exists between keyshapes and the predefined morph targets. A one-to-one mapping allows the vector of weights from the source data to be applied directly to the keymorphs to produce a retargeted animation.

A set of keymorphs is created for each basic facial expression, resulting in several retargeted animations that all say the same thing, each with a different facial expression. Figure 4 shows some pairs of keyshapes and keymorphs.

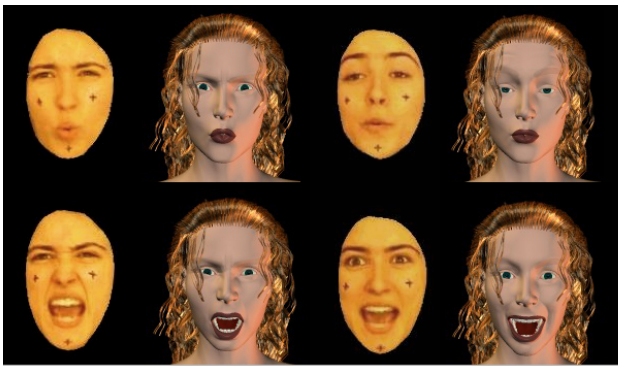


Figure 4: Examples of corresponding keyshapes to keymorphs for different expressions.

Finally, to create an animation with arbitrary or changing facial expression, we blend the retargeted facial animation for each basic expression together using:

$$\vec{H}(t) = \sum_{j=1}^{N} \alpha_j(t) \cdot \sum_{i=1}^{K} w_i^{(j)}(t) \vec{G}_i^{(j)}, \tag{9}$$

where N is the number of basic expressions. The α_j's are the user chosen barycentric coordinates of the desired output expression in terms of the basic expression vectors found during training.

Notice that we do not solve for direct correspondence in a vertex to vertex sense. This is important as it keeps the method flexible and does not limit the types of motion capture source or types of target model. The vector of source data is only defined with respect to source keyshapes, while the vector of target data is only defined with respect to target keymorphs. In the examples shown in this paper, we use 2D facial feature points as data when constructing the source keyshapes, **B**, and blendshape control parameters for the target keymorphs, **G**. We have been equally successful using polygon vertices or NURB control points as the data vector when constructing keymorphs. In principle, keyshapes and keymorphs can be constructed from any measurement as long as it can be linearized.

Note that the semantic notion of ”happy” or ”angry” is defined in both the video based bilinear space (training), and the output character’s 3D blendshape representation. In order to change expressions, we merely need to interpolate between the various semantically defined expressions. In the bilinear space this interpolation occurs on the style (expression) vector, allowing the expression of video images to be changed. In the blendshape space, this interpolation occurs between the keymorph sets defined for each expression, allowing the 3D character’s expression to be changed. It is both meaningful and possible to control the expression in either space.

This method requires manual modeling and therefore it is not completely automatic. We feel that this is beneficial since artists frequently wish to control the precise appearance of the target character. Furthermore, we find that the keyshapes for several expressions are often similar. In these cases, the shape itself is not an adequate descriptor of expression, and automatic methods are unlikely to correctly interpret artistic intent. Artistic judgment is *required* to insure that the keymorphs for each expression are interpreted as desired.

By modeling the keymorphs one might wonder whether the artist has done most of the hard work already, negating the need for a retargeting system at all. However, this is precisely the point of the method. The artist has done most of the hard work, but they have not done most of the tedious work. Using a small number of manually defined keymorphs, the algorithm can process a long video sequence. This would normally require the artist to define hundreds of keyframes manually.

5 Adding head motion

Head motion is a very important part of facial animation. Traditional animators often start by creating the head motion first, and then filling in "less important" details, like lip-sync and other expressions. This work introduces a data-driven approach to synthesizing head motion that is consistent with the characters intended emotion.

Head motion synthesis presents a different set of challenges to facial speech animation. The coupling between phonemes and visemes (the visual appearance of the mouth) is much stronger then that found between audio signal and head motion. Many statistical techniques rely on strong correlations among the different parameters. However in this case there is no deterministic relationship or obvious correlation between head motion and other communication cues. For the same sentence and expression, many different head motions are possible. Nevertheless, there is at least a weak correlation, and randomly generated head motion will not convey the expressiveness desired.

Fortunately, there is evidences that leads us to believe that audio features as simple as pitch contour are correlated with head motions. Recent study in the phonetics and linguistic community suggests that there is anatomical evidence of a coupling between head motion and pitch [Honda 2000]. In addition, empirical study by Yehia et al. [Yehia et al. 2000] also found that the pitch contour in a voice audio signal is highly correlated to head motions.

Our process of head motion synthesis begins by building a database of examples which relate audio pitch to motion. Synchronized motion and audio streams are captured and then segmented and stored in the database. A new audio stream can be matched against segments in the database. A smooth path is found through the matching segments and stitched together to create synthetic head motion. Figure 5 shows the overview of the approach.

5.1 Segmentation

We segment the input audio track using pitch contour information. The same segment boundary is then used for slicing the head motion tracks.

We used a program called Praat [Boersma and Weenink 2003] to process the audio data and extract pitch. Unfortunately this processing is relatively noisy. The top of figure 6 shows a raw pitch signal. We filter the pitch values to consider only those that lie within the normal human range. In addition we interpolate neighboring values to fill in very short periods of missing data. After cleaning the pitch curve, we use voiceless regions (regions with no pitch), to divide the curve into segments. The bottom part of figure 6 shows the same pitch contours now partitioned and cleaned up after the processing. Since head motion continues to change regardless of whether the

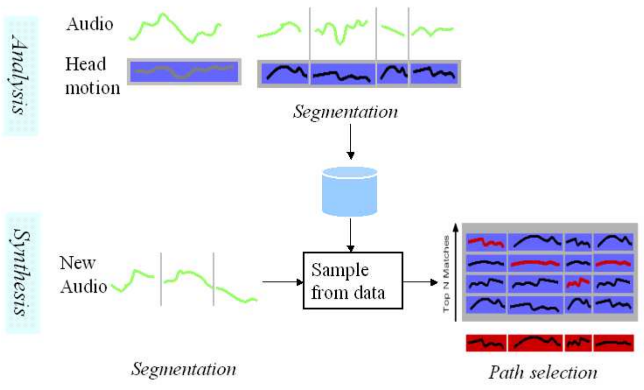


Figure 5: Overview for the head motion synthesis.

audio contains a voiced and voiceless region, a segment is defined as starting from the onset of one pitch segment and ending on the onset of the next pitch segment. Empirically we observe that most of the large head motion happens right after pitch onset, suggesting that this is indeed a good segment boundary.

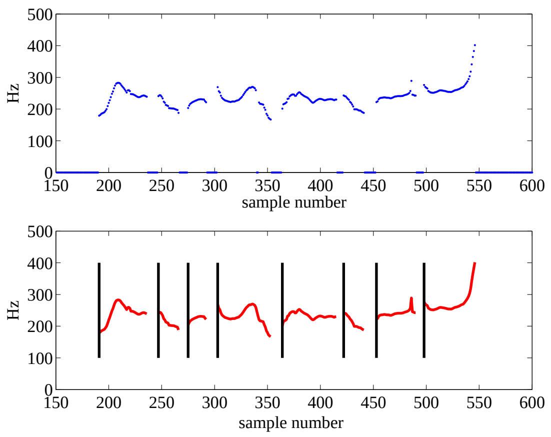


Figure 6: Top: noisy pitch data. Bottom: cleaned up pitch data and segmented for use in the matching process.

5.2 Pitch matching

For a given sequence of new audio pitch segments, we need to find a matching sequence of segments from the database. We define the matching distance in a two-stage process:

In the first stage, we compare pitch for an entire phrase, not just a single segment. This is important because the emotional, idiosyncratic content of speech is often conveyed at a sentence level through pitch phrasing. By matching first at the sentence or phrase level we use only those sentences in our database with expressive style similar to our new audio sequence. Sentences are compared by matching feature vectors derived from the audio data. These features include statistics related to the speech rhythm; the speaking rate, and the average length of the voiced and voiceless regions, as well as simple statistics on the pitch signal, including the minimum, maximum, mean, and standard deviation of pitch values [Dellaert et al. 1996]. These statistics are normalized and used as a feature vector. Euclidian distance is used for calculating the top M sentences that best match the test input. These M sentences represent a subset of the database which has the desired expressive style.

In the second stage, we compare individual pitch segment in the test phrase against the pitch segments in the database subset. The metric consists of similarity in geometric properties of the segments G_d (min val, max val, range, min slope, max slope, curvature, etc.). Each pitch segment is re-sampled to match the average length of all pitch segments. Then root-mean-square difference is used to compute a distance metric to every other pitch segment:

$$P_d = RMS(P_{test} - P_{template}). \tag{10}$$

P_{test} and $P_{template}$ are length normalized pitch contour. To avoid over stretching or shortening of the segments, a second criteria is used to penalize cases where the difference in lengths of the original segments are too large:

$$L_d = \frac{|length(P_{template}) - length(P_{test})|}{length(P_{test})}. \tag{11}$$

A combined distance metric is defined as a weighted sum of these two metrics:

$$D_{total} = b \cdot G_d + c \cdot P_d + (1 - b - c) \cdot L_d. \tag{12}$$

In our experiment, we used $c = 0.3$ and $b = 0.05$ to produce the final animations. Finally, the list of matching segments for each input segment is pruned to retain only the top K choices. These segments are the possible matches which will be stitched together during path searching.

5.3 Path searching

Given N pitch segments in the input audio, and the top K matches for each test segment, each matching segment forms a node (S_t^i) in a Trellis graph, as shown in figure 7. We set the weight of nodes equal to the pitch distance metric as defined in equation 12.

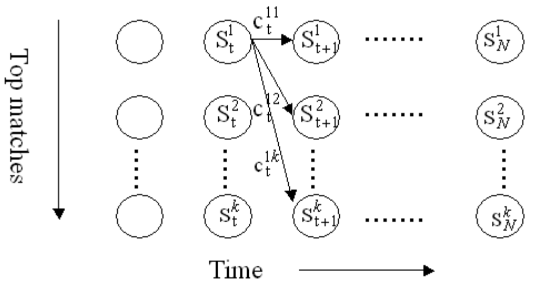


Figure 7: Trellis graph formed by the audio segments in the database which match those of the new sequence.

We need to find a path through the graph that produces a good head motion trajectory. Here we use the head motion data accompanying each matching segment and compute the cost of transitions (C_t^{ij}) from one segment to the next. Transition cost is based on two criteria.

First, in order to produce a smooth head motion, neighboring segments should have matching boundaries in terms of position, P_t^i, velocity, V_t^i, and acceleration, A_t^i. Therefore, the first cost function is:

$$\begin{aligned} H_t^{ij} = \omega_P \cdot (P_t^i - P_{t+1}^j)^2 + \\ \omega_V \cdot (V_t^i - V_{t+1}^j)^2 + \omega_A \cdot (A_t^i - A_{t+1}^j)^2 \end{aligned}, \tag{13}$$

where the values of $\omega_P, \omega_V, \omega_A$ are chosen to normalize the average contributions of each term.

Second, consecutive segments in the original database are highly encouraged, since these result in the most natural motion. Similarly, repeating the same segment is discouraged, since it produces repetitive motion. The second cost function is thus:

$$R_t^{ij} = \{ \begin{array}{ll} 0, & S_t^i \ and \ S_{t+1}^j \ are \ consecutive \\ 2, & S_t^i = S_{t+1}^j \\ 1, & otherwise \end{array}. \tag{14}$$

The complete transition cost is a weighted sum of these two terms:

$$C_t^{ij} = \omega_1 \cdot H_t^{ij} + \omega_2 \cdot R_t^{ij}. \tag{15}$$

We merely set the weight of ω_1 and ω_2 equally, and have achieve good results. It would be interesting to investigate an optimal balance of the various segment and transition weights in the future.

Finally, we apply the Viterbi algorithm to find the best path through the graph [Viterbi 1967].

5.4 Motion Blending

The motion data associated with each segment is joined together to produce the final motion. To achieve this, each motion segment is re-sampled to match the length of the test segment. Although this alters the frequency content of the motion data, the length of the segment is part of the matching criteria, thus the selected segments should have length similar to the test segment. Next, the starting position of each motion segment is moved to match the ending position of the previous segment. This could lead to potential drifting for long sentences. When this occurs, we break long sequences into smaller units, and perform path search for each sub-sentence. In general, we found that breaks between sentences, where the speaker take a breath serve as good boundaries for the subunits. Each subsequence is linearly warped to join the next sub-sequence to correct for the drifting. Finally, the end positions where the segments are connected are smoothed to minimize high frequency artifact caused by mismatch in the velocity.

6 Experiment

Our training data is drawn from three short video sequences including three basic expressions: neutral, angry and happy. Each sequence is around 12 seconds long, but chosen so that the words cover most viseme groups. The facial features are tracked using a technique similar to eigen-tracking [Black and Jepson 1996], and the resulting data is compressed and represented as vectors as shown in equation 3. A bilinear model is then fitted to this data according to equation 4.

Subsets of keyshapes for different facial expressions are selected from the training set using the method described in section 4. Although the required number of keyshapes or basis vectors are often chosen automatically to represent a fixed percentage of the source variation, this still leaves the choice of cutoff percentage as a parameter to be chosen by the animator or system developer. For the examples presented in this section the animator determined that approximately 20 keyshapes adequately represented the range of motion expected, and specified the number of keyshapes directly, since this is more intuitive for artists than specifying the underlying mathematical quantities. It should be noted that there is no requirement that the number of keyshapes be equal for each expression, and artists tended to choose slightly fewer keyshapes for the neutral expression since less variation existed in the data. The keyshapes themselves are determined automatically and separately in each expression set. These keyshapes will be used for decomposing and retargeting facial expressions.

For the output character, we used commercial models that have built-in morph targets, such as mouth opening, eye browsing raising, etc., and sliders for manipulating the blendshape with the morph targets. Keymorphs for the chosen keyshapes are constructed to correspond to each keyshape. Depending on the model, the user's skill, and inclination for perfectionism, this modelling process can take anywhere from minutes to hours.

To capture training data for head motion synthesis, a marker-based motion-capture system from Vicon is used for motion acquisition. We employed seven cameras with 120 Hz visual sampling frequency. Eight markers are placed on a plastic hair-band worn by the actress, and two markers are placed on each of her ear lobes. These points are rigid relative to each other, so we can compute 3D rotations and translations using a technique based on singular value decomposition (SVD) [Arun et al. 1987]. To increase the amount of data, we add to the head motion data by mirror imaging the head motion. We recorded a collection of 67 phrases. Each phrase consists of two or three sentences. The actress was free to move from waist up, and was informed that highly expressive motions were desired, therefore the footage consists of a wide variety of motion.

To show the process of making a new expressive animation, we videotaped an input test sequence about 13 seconds long which consists of the actress saying a completely new set of sentences with neutral facial expression. The test sequence is analyzed using the bilinear model, and sequences with different facial expressions are synthesized. We then decompose the reconstructed sequences into a weighted combination of keyshapes. The resulting weights are used for retargeting. Since we do not have a method for synthesizing expressive audio, we recorded the actress speaking the same text as the input test sequence but with different emotions. The audio is used as input to the head motion synthesis. Finally, the timing of the final animation is warped to match the new speech.

We show sample frames of the final animation in figure 8 and 9. The full animation is in the supplementary video. Notice that the facial expression does not stay constant throughout the whole animation, instead, it seems to be more extreme in some frames and rather neutral in others. This is desirable because a realistically animated character will need to talk and convey emotions at the same time. As one tries to make a certain mouth shape necessary for speaking, the dynamics of the entire facial configuration change depending on the viseme, and therefore it would be unnatural to hold a smile constantly throughout the entire sequence. This variation is crucial to natural looking facial animation.

To see the video of the complete animation, please visit the following URL: http://graphics.stanford.edu/papers/moodswings/.

7 Conclusion and Discussion

This paper presents a method for creating expressive facial animation and retargeting it onto new characters with arbitrary appearance. The example takes expressionless speech performance as input, analyzes the content, and modifies the facial expression according to a statistical model. The expressive face is then retargeted onto a 3D character using blendshape animation. By explicitly maintaining a facial expression vector in our model, we are able to choose the appropriate combination of morph targets during retargeting. The resulting animation is thus much more expressive than would otherwise be possible. Finally, we present a head motion synthesis algorithm which produces expressive head motion that is correlated to the audio signal, making the resulting facial animation more lively and characteristic.

There are many opportunities to improve the techniques presented here. First, the bilinear model we use for analyzing the expression does not include temporal constraints, therefore the output synthesis can sometimes be jittery. A method for incorporating temporal constraints would be valuable. Second, the fact that we completely rely on the implicit mapping from keyshapes to keymorphs can sometimes be problematic, since it relies on the artist to maintain consistency in the mapping. For instance, if eyebrow raising in one source keyshape is mapped to an ear enlarging in the target keymorph, then eyebrow raising in another keyshape should also map to ear enlarging in that keymorph. Either an automated method for ensuring consistency or a method of visual feedback to guide the artist would be useful. Lastly, the head motion synthesis technique presented does not address the dependency of head motion on speech context. For instance, in most cultures, nodding is used for confirmation, and head shaking for negation. A method to incorporate these contextual rules in the algorithm is needed.

Although specific methods and data types were used for the modules described in this paper, the methodology is much more general, and we wish to reiterate our emphasis on building a complete process for creating expressive animation. Changes could be made to almost any module while preserving the pipeline, and we hope to inspire future work by suggesting a couple of possibilities here. We currently rely on 2D video as a source for deriving input shape and emotion vectors. The use of video-rate 3D range images would presumably result in better models, which will address issues relating to robustness against lighting changes and out-of-plane head movement. Furthermore, as with other video synthesis techniques, the facial expression analysis method used in this paper is person specific. Two possibilities can be explored. A more general statistical model, such as a multi-linear model described in [Vasilescu and Terzopoulos 2003] could be used for representing the person's identity as the third dimension. Alternatively, more abstract facial parameters such as those used in the field of facial expression recognition could replace the appearance based facial features used here. This would allow for the modeling of more general features that would represent the facial expressions of all people. It would also be interesting to explore physical simulation for creating the morph targets. In this case, instead of blending the locations of vertices, the output space would be a linear combination of physical parameters. Linking expression to physical parameters may lead to a level of performance beyond what is now possible.

References

ARIKAN, O., AND FORSYTH, D. 2002. Interactive motion generation from examples. In *Proceedings of ACM SIGGRAPH*, ACM Press, 483–490.

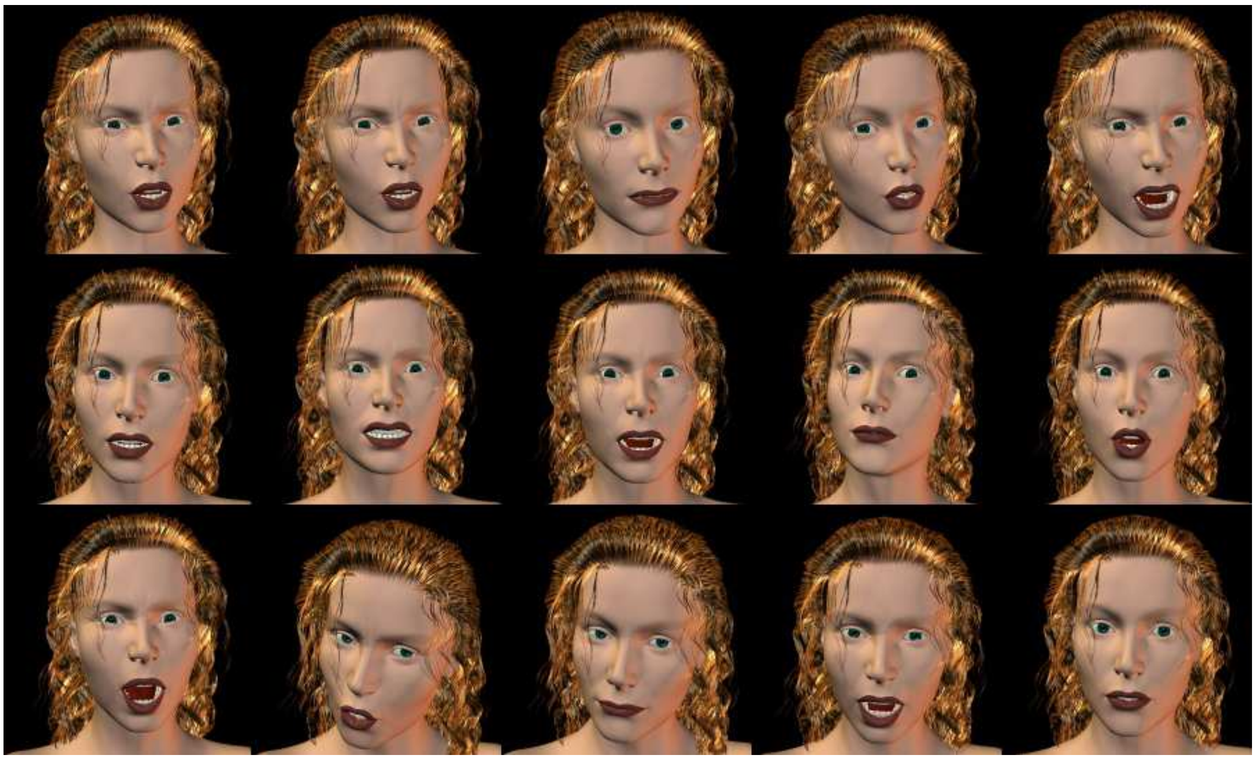


Figure 8: Sample frames of the test sequence retargeted with angry expression.

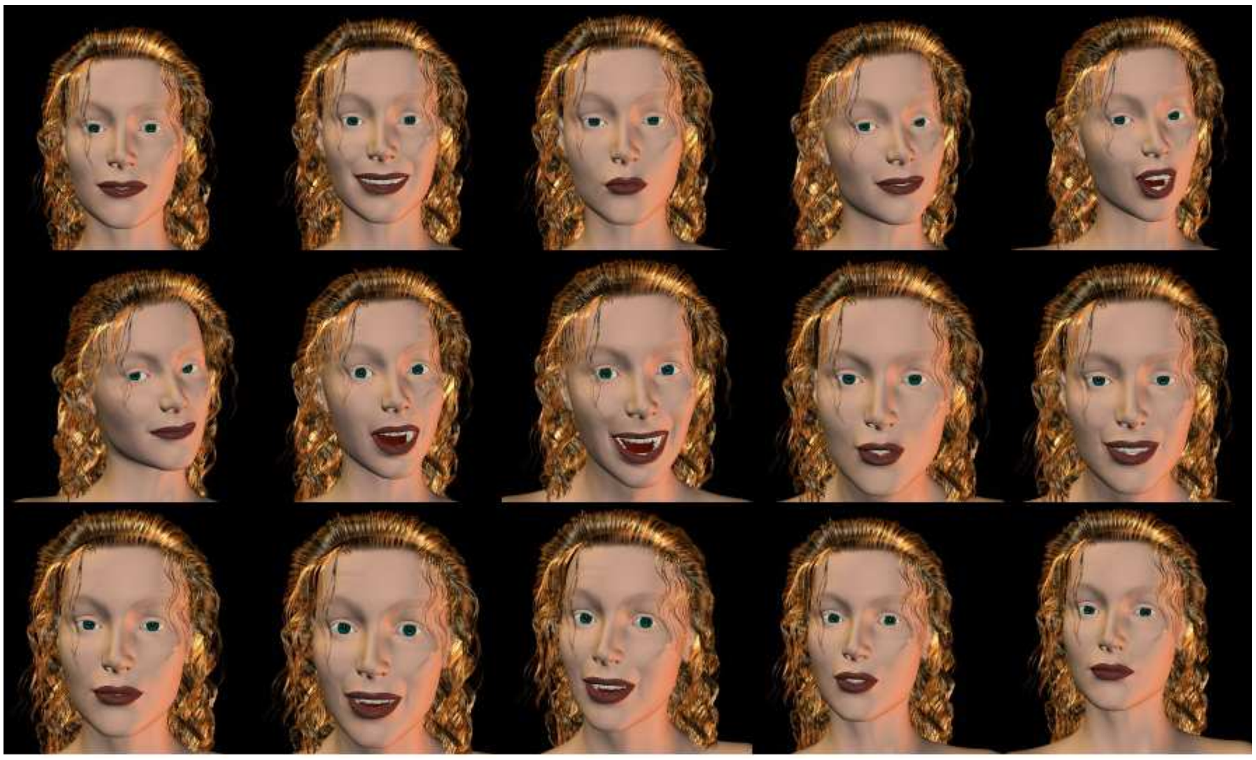


Figure 9: Sample frames of the test sequence retargeted with happy expression.

ARUN, K. S., HUANG, T. S., AND BLOSTEIN, S. D. 1987. Leastsquares fitting of two 3-d point sets. *IEEE Transaction on Pattern Analysis and Machine Intelligence 9*, 5 (Septempber), 698–700.

BLACK, M. J., AND JEPSON, A. D. 1996. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *Proceedings of the European Conference in Computer Vision*, 329–342.

BOERSMA, P., AND WEENINK, D., 2003. Praat: doing phonetics by computer. http://www.praat.org.

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

BREGLER, C., COVELL, M., AND SLANEY, M. 1997. Video rewrite: Driving visual speech with audio. In *Proceedings of ACM SIGGRAPH*, ACM Press/Addison-Wesley Publishing Co., 353–360.

BUCK, I., FINKELSTEIN, A., JACOBS, C., KLEIN, A., SALESIN, D., SEIMS, J., SZELISKI, R., AND TOYAMA, K. 2000. Performance-driven hand-drawn animation. In *Proceedings of Non-Photorealistic Animation and Rendering*, ACM Press.

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

CASSELL, J., PELACHAUD, C., BADLER, N., STEEDMAN, M., ACHORN, B., BECKET, T., DOUVILLE, B., PREVOST, S., AND STONE, M. 1994. Animated conversation: Rule-based generation of facial epression, gesture and spoken intonation, for multiple conversation agents. In *Proceedings of ACM SIGGRAPH*, ACM Press, 413–420.

CHAI, J., XIAO, J., AND HODGINS, J. 2003. Vision-based control of 3D facial animation. In *Proceedings of the Symposium on Computer Animation*, Eurographics Association, 193–206.

CHUANG, E., AND BREGLER, C. 2002. Performance driven facial animation using blendshape interpolation. Tech. Rep. SU-CS-TR-2002-02, Stanford University.

CHUANG, E., DESHPANDE, H., AND BREGLER, C. 2002. Facial expression space learning. In *Pacific Graphics*, IEEE, 68–76.

COHEN, M., AND MASSARO, D. 1993. Modeling coarticulation in synthetic visual speech. In *Computer Animation.*, Springer-Verlag.

COOTES, T. F., EDWARDS, G. J., AND TAYLOR, C. J. 2001. Active appearance models. *IEEE Trans. Pattern Anal. Mach. Intell. 23*, 6, 681–685.

COSSATO, E. 2002. *Sample-Based Talking-Head Synthesis*. PhD thesis, Swiss Federal Institute of Techology, Switzerland.

DECARLO, D., REVILLA, C., STONE, M., AND VENDITTI, J. 2002. Making discourse visible: Coding and animating conversational facial displays. In *Computer Animation*, IEEE Computer Society, 11–16.

DELLAERT, F., POLZIN, T., AND WAIBEL, A. 1996. Recognizing emotions in speech. In *Proceedings of International Conference on Spoken Language Processing*, vol. 3, 1970–1973.

EKMAN, P., AND FRIESEN, W. V. 1978. *Facial Action Coding System: A Technique for the Measurement of Facial Movement.* Consulting Psychologists Press.

ESSA, I., AND BASU, S. 1996. Modeling, tracking and interactive animation of facial expressions and head movements using input from video. In *Proceedings of Computer Animation*, 68–69.

ESSA, I., AND PENTLAND, A. 1997. Coding, analysis, interpretation, and recognition of facial expressions. *IEEE Trans. on Pattern Analysis and Machine Intelligence 19*, 7, 757–763.

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

HONDA, K. 2000. Interactions between vowel articulation and F0 control. In *Proceedings of Linguistics and Phonetics: Item Order in Language and Speech (LP'98)*, Hungary: Kakrolinum Press, Prague, B. D. J. O. Fujimura and B. Palek, Eds., 517–527.

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

KALBERER, G. A., AND GOOL, L. V. 2001. Face animation based on observed 3D speech dynamics. In *Computer Animation*, IEEE Computer Society, 20–27.

KOUADIO, C., POULIN, P., AND LACHAPELLE, P. 1999. Real-time facial animation based upon a bank of 3D facial expressions. In *Computer Animation*, IEEE, 128–136.

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

LI, Y., WANG, T., AND SHUM, H. 2002. Motion texture: a two-level statistical model for character motion synthesis. In *Proceedings of ACM SIGGRAPH*, ACM Press, 465–472.

LITWINOWICZ, P., AND WILLIAMS, L. 1994. Animating images with drawings. In *Proceedings of ACM SIGGRAPH*, ACM Press, 409–412.

NOH, J., AND NEUMANN, U. 2001. Expression cloning. In *Proceedings of ACM SIGGRAPH*, ACM Press, 277–288.

OSTERMANN, J. 1998. Animation of synthesis faces in MPEG4. In *Computer Animation*, IEEE, 49–55.

PARKE, F. I. 1982. Parameterized models for facial animation. *IEEE Computer Graphics and Applications 2*, 9, 61–68.

PERLIN, K., 1997. Layered compositing of facial expression. Proceedings of SIGGRAPH : Sketches and Applications.

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

PIGHIN, F. H., SZELISKI, R., AND SALESIN, D. 1999. Resynthesizing facial animation through 3D model-based tracking. In *International Conference on Computer Vision*, IEEE, 143–150.

POGGI, C., PELACHAUD, F., AND DE ROSIS, F. 2000. Eye communication in a conversational 3D synthetic agent. *Special issue on Behavior Planning for Life-Like Characters and Avatars of AI Communications.*

PULLEN, K., AND BREGLER, C. 2002. Motion capture assisted animation: Texturing and synthesis. In *Proceedings of ACM SIGGRAPH*, 501–508.

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 Symposium on Computer Animation*, Eurographics Association, 167–176.

TAKEUCHI, A., AND NAGAO, K. 1993. Communicative facial displays as a new conversational modality. In *ACM/IFIP INTERCHI*, ACM Press, 187–193.

TENENBAUM, J. B., AND FREEMAN, W. T. 2000. Separating style and content with bilinear models. *Neural Computation 12*, 6, 1247–1283.

TIAN, Y., KANADE, T., AND COHN, J. 2001. Recognizing action units for facial expression analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 23*, 2, 97–115.

VASILESCU, M. A. O., AND TERZOPOULOS, D. 2003. Multilinear subspace analysis for image ensembles. In *Proc. Computer Vision and Pattern Recognition Conference*, 93–99.

VITERBI, A. J. 1967. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Trans. Informat. Theory 13*, 260–269.

YEHIA, H., KURATATE, T., AND VATIKIOTIS-BATESON, E., 2000. Facial animation and head motion driven by speech acoustics. P. Hoole (ed). 5th Seminar on Speech Production: Models and Data, Kloster Seeon, Germany, May 1-4.

ZHANG, Q., LIU, Z., GUO, B., AND SHUM, H. 2003. Geometry-driven photorealistic facial expression synthesis. In *Proceedings of the Symposium on Computer Animation*, Eurographics Association, 177–186.