

A Talking Face Driven by Voice using Hidden Markov Model*

GUANG-YI WANG, MAU-TSUEN YANG, CHENG-CHIN CHIANG AND WEN-KAI TAI

Department of Computer Science and Information Engineering

National Dong Hwa University

Hualien, 974 Taiwan

E-mail: mtyang@mail.ndhu.edu.tw

In this paper, we utilized Hidden Markov Model (HMM) as a mapping mechanism between two different kinds of correlated signals. Specifically, we developed a voice-driven talking head system by exploiting the physical relationships between the shape of the mouth and the sound that is produced. The proposed system can be easily trained and a talking head can be efficiently animated. In the training phase, the Mel-scale Frequency Cepstral Coefficients (MFCC) were analyzed from audio signals and the Facial Animation Parameters (FAP) were extracted from video signals. Then both audio and video features were integrated to train a single HMM. In the synthesis phase, the HMM was used to correlate a completely novel audio track to a FAP sequence for face synthesis with the help of Facial Animation Engine (FAE). The experiments demonstrated the effects of the proposed voice-driven talking head on both man and woman, with two kinds of styles (speaking and singing) and using three kinds of languages (Chinese, English and Taiwanese). The possible applications of the proposed system are computer aided instruction, online guide, virtual conference, lip synchronization, human computer interaction and so on.

Keywords: talking head, audio-to-visual mapping, HMM, FAP, lip synchronization

1. INTRODUCTION

Due to the recent advance of computer technologies and high-speed networks, online real-time interaction becomes possible in our daily life. More and more people communicate with each other or interact with online avatars through the internet. The newest version of the well-known chat software, such as MSN, YM, and ICQ, can not only exchange texts but also real voice and live video. Nevertheless, the network bandwidth still imposes a limitation on possible applications. Typically, it is required to transfer about 30K bytes/sec for a clear audio track, and 3M bytes/sec for a video with good quality. When there are more than two people involved in a discussion, the requirement of bandwidth will be increased multiplicatively. If the quality of the network cannot stably fulfill this requirement, it is common for people to encounter troublesome problems like delayed video, skipping or unclear images.

To solve these problems, we proposed a simple and flexible audio-to-visual mapping system based on HMM. With the help of this mapping scheme, the transmitter can send only real voice, then the receiver can synthesis a virtual talking head that is syn-

Received August 16, 2005; accepted January 17, 2006.

Communicated by Jhing-Fa Wang, Pau-Choo Chung and Mark Billingham.

* This research was supported in part by the National Science Council of Taiwan, R.O.C., under grant No. NSC 95-2221-E-259-027-MY2.

chronized with the received audio signals. It is also possible to extend the system to synthesize a real talking face with the availability of an initial face image.

The proposed system can be divided into two phases. In the training phase shown in Fig. 1 (a), the Mel-scale Frequency Cepstral Coefficients (MFCC) were calculated from audio signals and the Facial Animation Parameters (FAP) were extracted from video signals. Then both audio and video features were combined to train a single HMM. In the synthesis phase shown in Fig. 1 (b), the HMM was used to generate an optimal state sequence for arbitrary new audio track. Then the state sequence was mapped to a FAP sequence that was applied to animate a 3-D talking head.

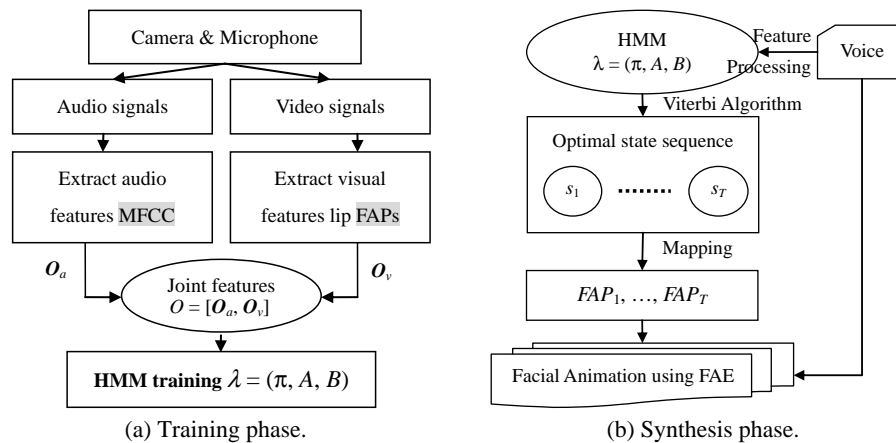


Fig. 1. The flowchart of the proposed system.

The possible applications of the proposed voice-driven talking head are computer aided instruction, online guide, virtual conference, lip synchronization, human computer interaction and so on. The remaining of this paper is organized as follows. Section 2 surveys the related researches. Section 3 discusses the audio and visual processing for feature extraction. Section 4 explains the training of the HMM for audio-to-visual mapping. Section 5 presents the synthesis of the voice-driven talking head. Section 6 shows the results of experiments. Finally, the paper is concluded in section 7.

2. RELATED WORKS

Facial animation can be synthesized from texts, images or voices. In the first category, Waters *et al.* [14] synthesized the appropriate mouth configurations using text as an input. Similarly, Tamura *et al.* [12] produced auditory speech and lip motion from arbitrary text using a tri-phone model. In the second category, Gao *et al.* [6] performed vision-based analysis to extract the movements of facial features from image sequence. Then the facial movements were used to drive face animation. This approach is intuitive but it is very difficult to track facial feature points accurately in dynamic illumination environment without pasting markers in user's face. In the third category, Bregler *et al.*

[3] animated a face by combing video segments that were obtained from a huge video database. Each video segment, called a *viseme*, was corresponding to and time-aligned with a basic unit of voice, called a *phoneme*. Since this approach was based on phonemes, phoneme-level transcription was required for training. Thus the technologies of speech analysis/recognition were involved and computation overhead was unavoidable. On the other hand, face can be directly animated by voice based on the physical relationships between the shape of the mouth and the sound that is produced. In this viewpoint, the main challenge of a realistic voice-driven talking head is to perform audio-to-visual mapping that does not really need to recognize the spoken words or sentences.

Many approaches for audio-to-visual mapping have been proposed including the Vector Quantization (VQ) [10], Artificial Neural Network (ANN) [5], Gaussian Mixture (GM) [4], Multidimensional Morphable Model (MMM) [17] and Hidden Markov Model (HMM) [16]. The VQ and ANN approaches are both based on frame-by-frame mappings that do not take account of intra-phoneme phenomenon caused by surrounding phoneme contexts, called coarticulation. Besides, the distinct number of output levels often leads to a staircase-like response. The GM approach can provide a smoother estimate of visual parameters but the coarticulation effect and temporal correlation are still not considered. The MMM approach takes coarticulation into account but the MMM synthesis requires intensive computation and is impractical for interactive applications. The HMM [11] is a very powerful statistical method to characterize the observed data samples of either a discrete or continuous time series. It provides an efficient way to build parametric models of time-varying data sequences. The model size of HMM does not relate to the quantity of data and it can predict the state of data with time relation.

Aleksic and Williams [1, 15] proposed an audio-to-visual mapping mechanism to synthesize lip motion to fit the speech. Two separate HMMs were trained based on audio and visual inputs respectively. A third HMM was trained to correlate the audio HMM with the visual HMM. Brand *et al.* [2] trained their HMM using only visual parameters. They assumed that the visual and audio HMMs share the same initial probability and state transition matrix. The audio information was only used for the determination of the optimal state sequence but did not contribute to improve the quality of the HMM. Chen *et al.* [4] trained their HMMs using joint audio-visual observation parameters. Both audio and visual parameters were modeled by a single HMM with better confidence and more accuracy.

This paper mainly focuses on the voice driven facial animation system using HMM. However, unlike most HMM approaches [1, 4, 15, 16], a single unified HMM was trained in the proposed system to circumvent the tedious work of training several HMMs, each for a phoneme or a word. In addition, unlike Brand's method [2], the proposed system trained the HMM by integrating both audio and visual input signals to better reflect the combing states of phonemic and facial patterns. Moreover, MPEG-4 compliant Facial Animation Parameters (FAPs) were applied to represent features for both training inputs and synthesis outputs to enhance the usability. The details of the implementation were discussed and a combination of experiments was made for both man and woman, either speaking or singing, using three kinds of languages: Chinese, English, and Taiwanese. The proposed system can be easily trained and a talking head can be efficiently animated for any completely novel audio track.

3. EXTRACTION OF AUDIO AND VISUAL FEATURES

For the purpose of HMM training, features should be extracted from audio and video input data. The audio and video input data were acquired using microphone and camera respectively. These input signals were individually analyzed to obtain audio and visual features. Then the two kinds of features were combined to form a joint feature vector.

3.1 Audio Feature Extraction

Mel Frequency Cepstrum Coefficients (MFCC) [7] are dominant features used for speech analysis and recognition. To obtain MFCC, a Discrete Fourier Transform (DFT) spectrum of each speech frame is warped in frequency through a Mel-scale transformation and warped in amplitude using a logarithmic transformation. Suppose $MFCC_t^i$ is the i -th MFCC parameter at time instant t . These MFCC parameters include a log-energy and a series of Mel-scale Cepstrum parameters. For simplicity, we only considered the first four parameters in our experiments. Suppose T is the length of observation sequence, the audio observation vector $O_{a,t}$ at time instant t is defined as follows:

$$O_{a,t} = \begin{bmatrix} MFCC_t^1 \\ MFCC_t^2 \\ MFCC_t^3 \\ MFCC_t^4 \end{bmatrix}, 1 \leq t \leq T.$$

3.2 Video Feature Extraction

Facial Animation Parameter (FAP) is a facial animation standard in MPEG-4. Totally, there are sixty eight parameters that are divided into ten groups. Each FAP represents the movement of a feature point on a human face compared to a neutral face. The neutral face refers to an expressionless face that can be obtained in the first few frames in the input video. The most important features corresponding to speech are lip movements that belong to group eight in the definition of FAP.

Fig. 2 (a) shows ten feature points around outer lip. Among them, four feature points are marked as 8.1, 8.2, 8.3 and 8.4. Table 1 lists four FAPs that are related to these feature points. The first field is the FAP identity number, the second field simply describes the meaning of the FAP, the third field represents the unit of displacement in FAP, the fourth field determines the direction of positive FAP, and the FAP group number is in the fifth field.

The basic unit of the FAP displacement is called Facial Animation Parameter Units (FAPUs). Each FAP could have its own FAPU. As shown in Fig. 2 (b), two FAPU are used by FAPs around the lip. $MW0$ represents the horizontal distance between two mouth corners. $MNS0$ indicates the vertical distance between the horizontal line of nostrils and the horizontal line connecting the mouth corners. Instead of using $MNS0$, a smaller FAP unit $MNS = MNS0/1024$ is defined to obtain sub-pixel accuracy.

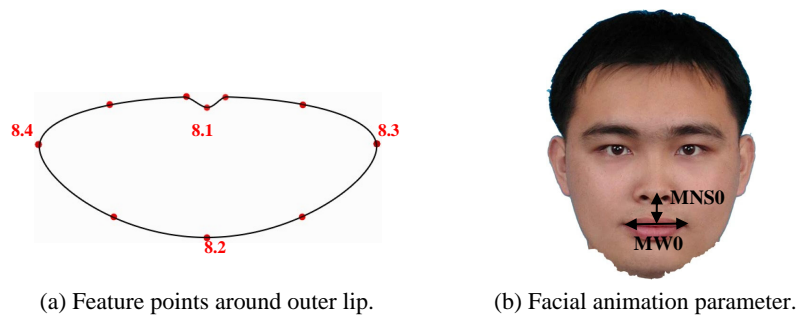


Fig. 2. Facial animation parameters (FAP).

Table 1. Facial animation parameters (FAP).

FAP#	FAP Description	Units	Positive Motion	Group
51	Vertical displacement at 8.1	<i>MNS</i>	Down	8
52	Vertical displacement at 8.2	<i>MNS</i>	Up	8
59	Vertical displacement at 8.4	<i>MNS</i>	Up	8
60	Vertical displacement at 8.3	<i>MNS</i>	Up	8

Visual features can be obtained automatically using a vision-based technique with three steps. First, we pasted four fluorescent markers on the feature points located on the outer lip as shown in Fig. 3 (a). The speech of the person with markers was captured using a PC camera in a dark room. The markers were usually quite clear in the captured images as shown in Fig. 3 (b). Second, we found contours in each frame using an edge detection algorithm. The area inside each closed contour is calculated as shown in Fig. 3 (c). If the area is under a threshold, the contour is considered as noise and removed. Third, a circle-fitting algorithm was performed on each contour; the center coordinate of each ellipse was output as a candidate of feature points. The candidate with the minimum y -coordinate was considered as feature point 8.1 and the candidate with maximum y -coordinate was taken as feature point 8.2. Similarly, the candidate with the minimum x -coordinate was considered as feature point 8.4 and the candidate with maximum x -coordinate was taken as feature point 8.3.

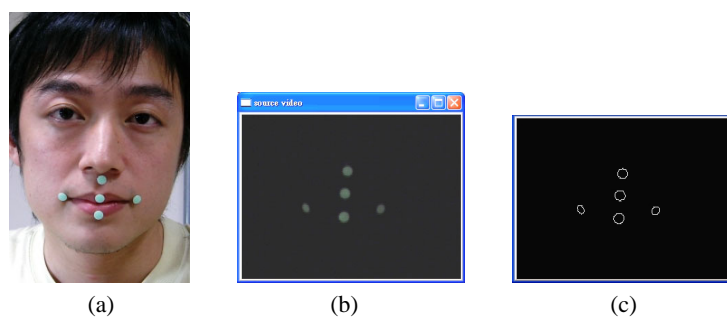


Fig. 3. The FAPs can be calculated by detecting four fluorescent markers around the lip.

Finally, the values of FAPs in each frame can be calculated using the locations of the detected feature points. In a neutral face, we assume the value of each FAP is zero. Suppose fap_t^i represents i -th FAP value at time t , and 8.1_t^y indicates the y -coordinate in the feature point 8.1 at time t . Table 2 shows the equations for the computation of each FAP. We assume that the first frame in a captured image sequence always contains a neutral face. Thus, the $8.1_{neutral}$ is the same as $8.1_{t=0}$.

Table 2. The computation of FAP using coordinates of feature points around the lip.

FAP #	Value of FAP
fap_t^{51}	$8.1_t^y - 8.1_{neutral}^y$
fap_t^{52}	$8.2_{neutral}^y - 8.2_t^y$
fap_t^{59}	$8.4_{neutral}^y - 8.4_t^y$
fap_t^{60}	$8.3_{neutral}^y - 8.3_t^y$

For simplicity, only four FAPs (FAP# 51, 52, 59 and 60) were used in our experiments. The video observation vector $O_{v,t}$ at time t was defined as follows:

$$O_{v,t} = \begin{bmatrix} fap_t^{51} \\ fap_t^{52} \\ fap_t^{59} \\ fap_t^{60} \end{bmatrix}, \quad 1 \leq t \leq T.$$

The audio and visual features were integrated for HMM training that will be discussed in the next section. A joint observation O including both audio feature O_a and visual feature O_v was defined as follows:

$$O = \begin{bmatrix} O_a \\ O_v \end{bmatrix} = \begin{bmatrix} o_{a,1} & o_{a,2} & \cdots & o_{a,T} \\ o_{v,1} & o_{v,2} & \cdots & o_{v,T} \end{bmatrix}.$$

4. TRAINING OF HIDDEN MARKOV MODEL

The goal of this section is to find the HMM parameters providing the joint audio-visual observations O . Both audio and visual input features were integrated in the proposed system to train a single unified HMM. Each state in the HMM is related to a typical combination of facial and phonemic pattern. Suppose N is the number of states and T is the length of observation sequence in a HMM. Let s_t represents the state at time t and o_t indicates the observation at time t . The parameters of HMM can be specified as $\lambda = (A, B, \pi)$ in that the first parameter $A = \{a_{ij}\}$ is a state transition probability distribution where a_{ij} is the transition probability from state i to state j .

$$a_{ij} = P(s_t = j | s_{t-1} = i), \sum_{j=1}^N a_{ij} = 1, a_{ij} \geq 0, \forall 1 \leq i, j \leq N.$$

The second parameter $B = \{b_i(o)\}$ is an observation probability distribution where $b_i(o)$ defines the i -th row in B and represents the probability of observation o providing the system is in state i at time t . Since our observations are continuous, a Gaussian Mixture is used to model $b_i(o)$ as follows:

$$b_i(o) = P(o_t = o | s_t = i) = \sum_{k=1}^M c_{ik} N(o, \mu_{ik}, U_{ik}), \forall 1 \leq i \leq N,$$

where M is the number of components in the Gaussian mixture. The c_{ik} is the mixture weight for the k -th mixture in state i . The $N(o, \mu_{ik}, U_{ik})$ is a Gaussian function with mean vector μ_{ik} and covariance matrix U_{ik} for the k -th mixture component in state i . The third parameter $\pi = \{\pi_i\}$ is an initial state probability distribution where π_i is the probability of starting the system in state i .

The Baum-Welch algorithm [11] is an Expectation-Maximization (EM) method that can find λ such that $P(O | \lambda)$ is locally maximized using an iterative procedure. Four variables are used for the re-estimation purpose in the Baum-Welch method. First, the forward variable $\alpha_t(i)$ represents the probability of the partial observation from the beginning to t and being in state i at time t , given the model is λ . The $\alpha_t(i)$ is defined as

$$\alpha_t(i) = P(o_1, o_2, \dots, o_t, s_t = i | \lambda), 1 \leq t \leq T, 1 \leq i \leq N - 1,$$

and can be computed inductively as follows:

$$\begin{aligned} \alpha_0(i) &= \pi_i b_i(o_0), 1 \leq i \leq N, \\ \alpha_{t+1}(j) &= \left[\sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(o_{t+1}), 1 \leq t \leq T - 1, 1 \leq j \leq N. \end{aligned}$$

Second, the backward variable $\beta_t(i)$ indicates the probability of the partial observation sequence from time $t + 1$ to the end, given being in state i at time t and the model is λ . Similarly, the $\beta_t(i)$ is defined as

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \dots, o_T | s_t = i, \lambda), 1 \leq t \leq T, 1 \leq i \leq N - 1,$$

and can be computed inductively as follows:

$$\begin{aligned} \beta_T(i) &= 1, 1 \leq i \leq N, \\ \beta_t(i) &= \sum_{j=1}^N a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), 1 \leq t \leq T - 1, 1 \leq i \leq N. \end{aligned}$$

The third variable $\xi_t(i, j)$ is the probability of being in state i at time t and in state j at time $t + 1$, given the model is λ and the observation sequence is O .

$$\xi_t(i, j) = P(s_t = i, s_{t+1} = j | O, \lambda) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}.$$

The final variable $\gamma_t(i)$ is defined as the probability of being in state i at time t , given the model is λ and the observation sequence is O . For simplicity, we assume the number of components in the Gaussian mixture for the observation distribution is one. As a result, $\gamma_t(i)$ can be calculated by summing $\xi_t(i, j)$ over j as follows:

$$\gamma_t(i) = P(s_t = i | O, \lambda) = \sum_{j=1}^N \xi_t(i, j) = \left[\frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^N \alpha_t(j)\beta_t(j)} \right].$$

The Baum-Welch algorithm re-estimates the λ^{new} in each iteration. The λ^{new} is put in place of λ and the HMM parameters are continuously updated until convergence. The HMM training algorithm can be summarized as follows:

- Step 1:** Initialize $\lambda = (A, B, \pi)$.
- Step 2:** Compute $\alpha_t(i), \beta_t(i), \gamma_t(i), \xi_t(i, j)$.
- Step 3:** Re-estimate $\lambda = (A, B, \pi)$.
- Step 4:** Return to step 2 if $P(O | \lambda)$ increases.

In order to make reliable estimates of the HMM parameters, multiple observation sequences were used for HMM training in our implementation. Assuming that each observation sequence is independent from each other observation sequence, the re-estimation equations of HMM parameters can be easily modified to accommodate multiple observation sequences.

Since the Baum-Welch method only finds local maximum, the accuracy of the trained HMM is affected by the quality of the initialization of the parameters. Experiments have shown that either uniform or random values are adequate for A and π . Thus, all elements in A and π are set to $1/N$ in our implementation. However, experiments have also shown that a better initialization is necessary for B , especially in the case of a continuous HMM. A k -means clustering approach was used in our implementation to initialize the matrix B . The i -th row in matrix B represents the observation probability distribution in state i , and can be specified by two parameters: the mean μ_i and the covariance matrix U_i of the observations in the i -th cluster. The k -means clustering algorithm in that $k = N$ can be summarized as follows:

- Step 1:** Assign N random values to μ_1, \dots, μ_N .
- Step 2:** For each observation o , compute the distance from it to the means of every clusters. Then classify the observation to the cluster with minimum distance. The index of the cluster with minimum distance to the observation o can be defined as: $Cluster(o) = \arg \min_{1 \leq i \leq N} \|o - \mu_i\|$.

Step 3: Compute the new mean in each cluster: $\mu_i^{new} = \frac{\sum_{Cluster(o)=i} o}{\sum_{Cluster(o)=i} 1}$.

Step 4: Set $\mu_i = \mu_i^{new}$ and repeat steps 2 and 3 until the clustering is converged. Then the mean μ_i and covariance matrix U_i of the i -th cluster can be used as the initial parameters of the i -th row in B .

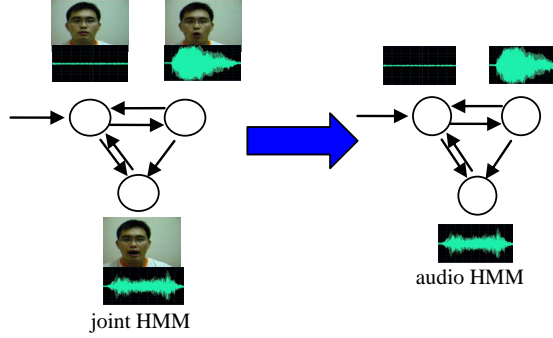


Fig. 4. The audio HMM can be extracted from the joint HMM with the same structure (π and A), but different observation probability distribution (B).

The joint HMM consists of an audio HMM and a visual HMM. These two HMMs are the same at structure (π and A), but different in observation probability distribution (B). As shown in Fig. 4, we can disassemble the audio HMM and visual HMM from the joint HMM. Suppose the mean vector and covariance matrix of B_i in the joint HMM are as follows:

$$\mu_i = \begin{bmatrix} \mu_i^a \\ \mu_i^v \end{bmatrix}, U_i = \begin{bmatrix} U_i^{aa} & U_i^{av} \\ U_i^{va} & U_i^{vv} \end{bmatrix}.$$

Then the mean vector and covariance matrix of B_i in the audio HMM can be extracted as $\mu_i = \mu_i^a$ and $U_i = U_i^{aa}$.

5. SYNTHESIS OF TALKING HEAD

After finishing the HMM training, we can find the corresponding FAPs for any new audio track to synthesize a talking head. The synthesis phase consists of three steps. First, the audio features were extracted from the input audio signals using the same method described in section 3.1. Second, the extracted audio observations were imported to the audio HMM to find an optimal audio state sequence. The audio HMM was disassembled from the joint HMM as mentioned at the end of section 4. Finally, the state sequence was converted to a sequence of FAP that can be used to drive a talking head.

The Viterbi algorithm [11] is a dynamic programming method that can be used to find an optimal state sequence for any given observation sequence. Two variables are introduced in the Viterbi algorithm. The $\delta_t(i)$ is the highest probability along a path ending in state i at time t accounting for the first t observation. The computation of $\delta_t(i)$ is similar to those of $\alpha_t(i)$ described in section 4 except that the function of the summation

is replaced by the maximization. To keep track of the path with the highest probability, a variable $\psi_t(i)$ is defined as the argument j that maximizes $\delta_{t-1}(j)a_{ji}$ for each time t and state i . The Viterbi algorithm to find the optimal state sequence is summarized as follows:

Step 1: Initialization

$$\delta_1(i) = \pi_i b_i(o_1), 1 \leq i \leq N,$$

$$\psi_1(i) = 0.$$

Step 2: Recursion

$$\delta_t(i) = \max_{1 \leq j \leq N} [\delta_{t-1}(j)a_{ji}b_i(o_t)], 2 \leq t \leq T, 1 \leq i \leq N,$$

$$\psi_t(i) = \arg \max_{1 \leq j \leq N} [\delta_{t-1}(j)a_{ji}], 2 \leq t \leq T, 0 \leq j \leq N.$$

Step 3: Termination

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)],$$

$$q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)].$$

Step 4: Path Backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*), T-1 \geq t \geq 1.$$

The output of the Viterbi algorithm is $\{q_i^*\}$ that reveals the optimal state sequence with the highest probability. The next step is to convert the optimal state sequence to a FAP sequence for face animation. In other words, given a state i and an audio observation o_a , we need to find the best corresponding visual feature o_v . As shown in Fig. 5 (a), $N(a, v, \mu_i, U_i)$ is a Gaussian surface that can be found in the parameters of the joint HMM. The intersection of the surface $N(a, v, \mu_i, U_i)$ and the plane $a = o_a$ is a curve that can be specified as $N(a, v, \mu_i, U_i | a = o_a)$. Suppose the peak of this curve locates at the point P' , then the v -coordinate of P' , denoted as o_v , is the set of visual features that is most likely to occur. Fig. 5 (b) shows the mapping from an overhead point of view. The mapping can be expressed as follows:

$$o_v = \arg \max_v N(a, v, \mu_i, U_i | a = o_a).$$

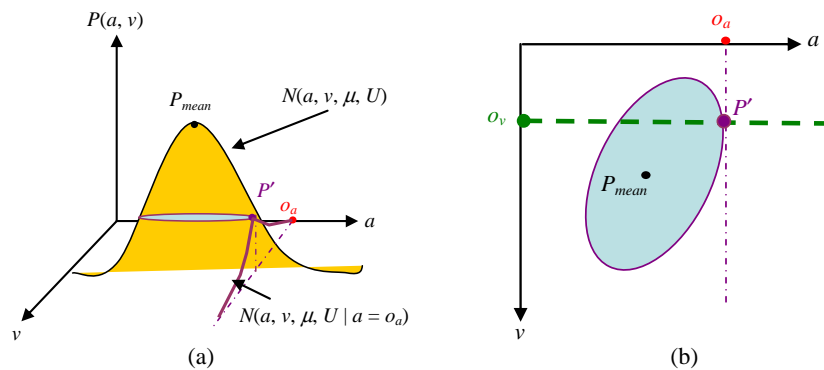


Fig. 5. Given a state i and an audio observation o_a , we need to find the best corresponding visual feature o_v .

With the help of this state-to-FAP mapping, a set of visual features $o_{v,t}$ can be obtained as the FAPs for each time t . Then the FAP sequence $O_v = [o_{v,1} o_{v,2} \dots o_{v,T}]$ can be used to animate a talking head. In order to make the animation more realistic, each FAP is temporally smoothed using a Gaussian filter to suppress jittering effect. Facial Animation Engine (FAE) [8] is a handy tool for driving facial animation by FAP and is utilized in our implementation. As the last step in the proposed system, both input audio signals and synthesized FAPs are imported to FAE to generate a realistic talking head as shown in Fig. 6.

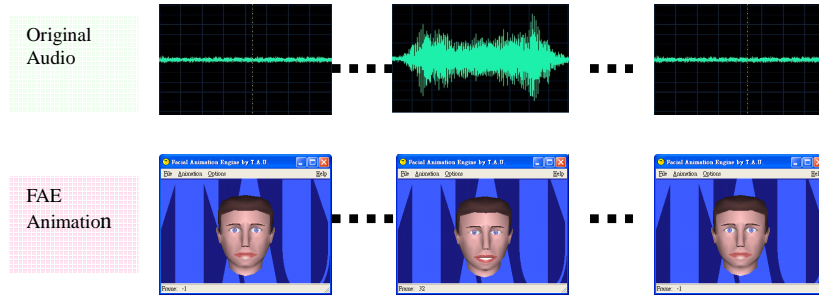


Fig. 6. The input audio signals and the output FAE talking head.

6. EXPERIMENTAL RESULTS

A directional microphone was used to record the audio signals and an off-the-shelves PC camera was used to record visual signals. The sampling rate of the audio signals was 12000 Hz and each audio sample occupied 16 bits. The sampling rate of the visual signals was 15 frames per second (fps) and the image resolution was 320×240 pixels. As a result, a four seconds observation segment consisted of 48000 audio samples and 60 image frames. The underlying hardware platform of our system was a PC with Intel P4 2.8GHz CPU. Our implementation took advantage of the Intel Integrate Performance Primitives (IPP) [13] and Open-source Computer Vision (OpenCV) library for the speedup purpose.

Two persons (**M**: man and **W**: woman) were involved in our experiments. Each of them recorded six test sequences using three kinds of languages (**C**: Chinese, **E**: English and **T**: Taiwanese) with two kinds of styles (**Sp**: speaking and **Si**: singing). Two sets of data (training data and testing data) were captured separately. For simplicity, the number of states was sixteen in each HMM, the dimension of both audio and visual features was four, and the number of Gaussian mixture in each state is one. The error of the synthesis was measured as the Mean Squared Error (MSE) that was defined as follows:

$$MSE = \frac{\sum_{t=1}^T \sum_{j=1}^{NOFAP} (FAP_{tj}^{truth} - FAP_{tj}^{synthesis})^2}{T \times NOFAP} \times 100\%,$$

where T is the length of observation data (number of frames) and $NOFAP$ is the number

of FAP in each frame. The FAP_{ij}^{truth} is the ground truth of the j -th FAP in the t -th frame in the captured video, and the $FAP_{ij}^{synthesis}$ is the synthesis output of the j -th FAP in the t -th frame.

In the first experiment, six types of HMMs (three languages multiplied by two styles) were trained using the training data, and six types of input audio signals in the testing data were used to synthesize the output FAP for the same person. Each column in Table 3 compares the synthesis errors of different kinds of audio inputs using the same trained HMM. Each row compares the synthesis errors of the same audio input using different trained HMMs.

Table 3. The first experiment: man-training and man-synthesis.

Man training \ Man synthesis	C Sp	C Si	E Sp	E Si	T Sp	T Si	Avg.
C Sp	0.16	4.76	1.51	2.12	2.55	3.72	2.47
C Si	2.70	2.10	3.02	6.57	3.09	5.93	3.90
E Sp	1.50	4.80	1.05	3.28	2.18	4.42	2.87
E Si	2.03	4.63	2.38	2.30	3.74	2.49	2.93
T Sp	2.53	6.27	3.20	5.73	0.80	4.97	3.91
T Si	2.25	6.37	2.25	4.82	3.25	3.05	3.67
Avg.	1.86	4.82	2.23	4.13	2.60	4.10	3.30

Table 4. The second experiment: man-training and woman-synthesis.

Man training \ Woman synthesis	C Sp	C Si	E Sp	E Si	T Sp	T Si	Avg.
C Sp	1.39	8.79	1.45	2.25	2.09	1.86	2.97
C Si	2.47	7.58	2.32	4.83	2.34	7.56	4.52
E Sp	5.06	3.27	4.15	5.36	4.17	2.34	4.06
E Si	5.45	4.79	3.55	8.55	3.78	4.43	5.09
T Sp	3.08	6.37	2.85	4.67	2.51	4.29	3.96
T Si	4.17	5.90	3.65	5.38	3.43	7.28	4.97
Avg.	3.60	6.12	2.99	5.18	3.06	4.63	4.26

In the second experiment, six types of HMMs (three languages multiplied by two styles) were trained for a person (man) using the training data, and six types of input audio signals in the testing data were used to synthesize the output FAP for another person (woman). Each column in Table 4 compares the synthesis errors of different kinds of audio inputs using the same trained HMM. Each row compares the synthesis errors of the same audio input using different trained HMMs.

In the third experiment, twelve kinds of input audio signals (two sexes, three languages and two styles) were tested to synthesize FAP using three HMMs trained by different languages (Chinese, English and Taiwanese). Table 5 compares the synthesis errors of these combinations. Each row uses the same kind of input audio signals and each column uses the same type of trained HMM.

Table 5. The third experiment: the comparison of synthesis errors of three HMMs trained using different languages.

Training model Synthesis data	Chinese	English	Taiwanese	Avg.
M C Sp	0.98	1.67	1.93	1.53
M C Si	4.09	3.25	7.27	4.87
W C Sp	1.58	3.47	1.85	2.30
W C Si	2.43	4.45	4.11	3.67
M E Sp	2.25	3.25	2.81	2.77
M E Si	2.89	2.51	4.53	3.31
W E Sp	5.95	2.26	4.90	4.37
W E Si	9.58	1.84	8.07	6.49
M T Sp	3.35	2.84	5.12	3.77
M T Si	2.43	2.82	4.02	3.09
W T Sp	3.77	3.93	4.47	4.06
W T Si	2.83	4.19	2.81	3.28
Avg.	3.51	3.04	4.33	3.63

Table 6. The fourth experiment: the comparison of synthesis errors of two HMMs trained by people with different sex.

Training model Synthesis data	Man	Woman	Avg.
M C Sp	1.79	2.31	2.52
M C Si	6.90	2.72	5.83
M E Sp	2.61	1.83	2.87
M E Si	5.24	2.53	4.37
M T Sp	5.34	1.97	4.71
M T Si	2.05	3.45	3.91
W C Sp	1.83	0.95	1.98
W C Si	3.97	2.60	3.95
W E Sp	4.80	2.45	4.72
W E Si	7.88	3.65	7.44
W T Sp	5.39	2.72	4.65
W T Si	8.62	2.97	5.79
Avg.	4.70	2.51	4.33

In the fourth experiment, twelve kinds of input audio signals (two sexes, three languages and two styles) were tested to synthesize FAP using two HMMs trained by a man and a woman respectively. Table 6 compares the synthesis errors of these combinations. Each row uses the same kind of input audio signals and each column uses the same type of trained HMM.

In the fifth experiment, twelve kinds of input audio signals (two sexes, three languages and two styles) were tested to synthesize FAP using two HMMs trained by

Table 7. The fifth experiment: the comparison of synthesis errors of HMMs trained using different styles.

Training model Synthesis data	Speak	Sing	Avg.
M C Sp	1.48	1.62	1.55
M E Sp	3.09	2.35	2.72
M T Sp	3.40	4.34	3.87
W C Sp	1.18	1.74	1.46
W E Sp	4.29	5.10	4.70
W T Sp	2.83	3.62	3.22
M C Si	5.52	5.69	5.61
M E Si	3.69	3.62	3.66
M T Si	3.27	3.40	3.33
W C Si	3.50	2.16	2.83
W E Si	5.78	7.07	6.43
W T Si	4.57	2.32	3.44
Avg.	3.55	3.59	3.57

Table 8. The final experiment: the comparison of synthesis errors of HMMs trained using different number of states.

Number of States	4	8	16	32	64
MSE	7.1	6.7	2.2	1.6	7.1

speaking and singing respectively. Table 7 compares the synthesis errors of these combinations. Each row uses the same kind of input audio signals and each column uses the same type of trained HMM.

In the last experiment, several HMMs were trained using different number of states. Table 8 compares the synthesis errors of these HMMs with different number of states. The results show that the error is minimal when the number of states is around 32. Generally speaking, the facial animation is very natural if the MSE value is smaller than three, and disturbance can be observed if the MSE value is greater than five. Fig. 7 demonstrates four synthesized faces corresponding to four different states of a HMM trained by a man singing a Chinese song.

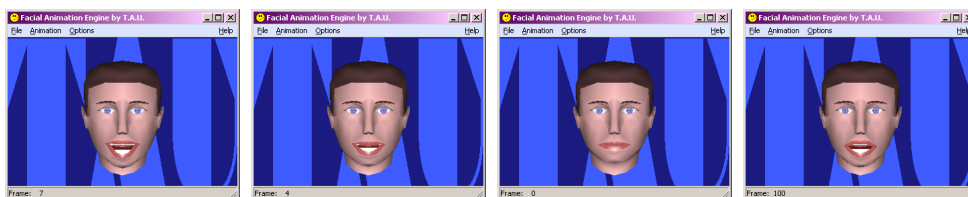


Fig. 7. Four synthesized faces corresponding to four different states in a HMM.

Since the HMM was trained using joint visual-audio parameters in our system, it is inherently speaker dependent. This explains that the average error in Table 4 (man-woman experiment) is larger than the average errors in Table 3 (man-man experiment). Also, the average error in Table 6 (sex experiment) is larger than the average errors in Table 5 (language experiment) and Table 7 (style experiment). Moreover, in Tables 3, 5, 6 and 7, the errors in the diagonal entries (marked as gray blocks in the tables) are generally smaller than those in the other parts of the table. This means that it would be perfect to directly synthesize FAPs from a HMM that is trained by the same speaker using the same language with the same style. However, it is a tedious process to train and maintain so many individual HMMs. In fact, it is possible to alleviate this problem by training one large HMM that can synthesize acceptable outputs for all groups, with the price of using more input data and processing time for HMM training.

As a case study, we have tried to apply the proposed audio-driven talking head to a 3-D Virtual English Classroom (VEC3D) [9] in that a person can not only hear another person's speech but also look at his animated talking face. Once a general HMM is trained in advance, a participant only needs a PC and a microphone to join the VEC3D through the internet. The proposed mapping scheme can animate the talking head of his avatar to synchronize with his voice using the trained HMM. Experiments show that teacher and students can understand each other better and feel more comfortable with the help of the animated talking head. Fig. 8 shows a screen shot of the VEC3D system.



Fig. 8. A screen shot of the virtual English classroom (VEC3D) system.

7. CONCLUSIONS

We proposed an audio-to-visual mapping mechanism without the need of phoneme-level transcription and speech recognition. A single HMM was trained that can be used to synthesize talking head for arbitrary novel audio track. Experiments were made to demonstrate the results of the voice-driven talking face for both man and woman, either speaking or singing, using three kinds of languages: Chinese, English and Taiwanese.

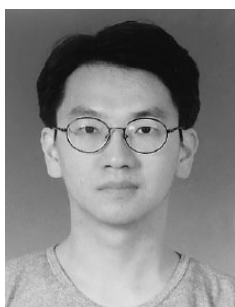
The proposed mapping system is general and flexible in that both input and output can be raw signals. In fact, it can be applied for transformation between any two correlated signals. For example, it is possible to generate dance animation from music, to determine facial expressions from voice, or to predict human emotion from human behavior, and so on. It is also possible to extend the proposed method to identify the source of voice for speaker/language identification.

REFERENCES

1. P. Aleksic and A. Katsaggelos, "Speech-to-video synthesis using MPEG-4 compliant visual features," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 14, 2004, pp. 682-692.
2. M. Brand, "Voice puppetry," in *Proceedings of the ACM SIGGRAPH*, 1999, pp. 21-28.
3. C. Bregler, M. Covell, and M. Slaney, "Video rewrite: driving visual speech with audio," in *Proceedings of the ACM SIGGRAPH*, 1997, pp. 353-360.
4. T. Chen and R. Rao, "Audio-visual integration in multimedia communication," in *Proceedings of the IEEE*, Vol. 86, 1998, pp. 837-852.
5. Y. Chen, W. Gao, Z. Wang, and L. Zuo, "Speech driven MPEG-4 based face animation via neural network," in *Proceedings of the 2nd IEEE Pacific Rim Conference on Multimedia*, LNCS 2195, 2001, pp. 1108-1113
6. W. Gao, Y. Chen, R. Wang, S. Shan, and D. Jiang, "Learning and synthesizing MPEG-4 compatible 3-D face animation from video sequence," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 13, 2003, pp. 1119-1128.
7. X. Huang and R. Reddy, *Spoken Language Processing*, Prentice Hall, 2001.
8. F. Lavagetto and R. Pockaj, "The facial animation engine: toward a high-level interface for the design of MPEG-4 compliant animated faces," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 9, 1999, pp. 277-289.
9. Y. Lin, M. Yang, and Y. Shih, "VEC3D: 3D virtual English classroom for second language learning," in *Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies*, 2005, pp. 906-908.
10. S. Morishima and H. Harashima, "Speech-to-image media conversion based on VQ and neural network," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 1991, pp. 2865-2868.
11. L. Rabiner, "A tutorial on hidden Markov model and selected applications in speech recognition," in *Proceedings of the IEEE*, Vol. 77, 1989, pp. 257-286.
12. M. Tamura, S. Kondo, T. Masuko, and T. Kobayashi, "Text-to-audio-visual speech synthesis based on parameter generation from HMM," in *Proceedings of the European Conference on Speech Communication and Technology*, 1999, pp. 959-962.
13. S. Taylor, *Intel Integrated Performance Primitives*, Intel Press, 2003.
14. K. Waters and T. Levergood, *DECface: An automatic Lip Synchronization Algorithm for Synthetic Faces*, Technical Report, No. CRL 93/4, Digital Equipment Corporation, Cambridge Research Laboratory, 1993.
15. J. Williams and A. Katsaggelos, "An HMM-based speech-to-video synthesizer," *IEEE Transactions on Neural Networks, Special Issue on Intelligent Multimedia*, Vol. 13, 2002, pp. 900-915.
16. E. Yamamoto, S. Nakamura, and K. Shikano, "Lip movement synthesis from speech based on hidden Markov model," *Speech Communication*, Vol. 26, 1998, pp. 105-115.
17. T. Ezzat, G. Geiger, and T. Poggio, "Trainable videorealistic speech animation," *ACM Transactions on Graphics*, Vol. 21, 2002, pp. 388-398.



Guang-Yi Wang (王光一) received the master degree in Computer Science and Information Engineering from the National Dong Hwa University, Taiwan in 2005. He is currently an engineer in the Utechzone Vision Automation Service Technology, Taiwan and is charged to inspect integrated chips with computer vision techniques.



Mau-Tsuen Yang (楊茂村) received the B.S. degree in Applied Mathematics from National Tsing Hua University, Taiwan in 1992, and the Ph.D. degree in Computer Science and Engineering from the Pennsylvania State University, U.S.A. in 2000. He is currently an Associate Professor of the Dept. of Computer Science and Information Engineering at National Dong Hwa University, Hualien, Taiwan. His research interests are computer vision, virtual reality, and image based rendering.



Cheng-Chin Chiang (江政欽) received his Ph.D. degrees in Computer Science and Information Engineering from National Chiao Tung University, R.O.C., in 1993. His research interests include neural networks, pattern recognition and human-machine interface. Dr. Chiang received the honor of Best Long-Term Doctorial Thesis Award from Acer Corporation in 1993 and the honor of Outstanding Young Engineer Award from Chinese Institute of Engineers in 2000.



Wen-Kai Tai (戴文凱) was born on January 29, 1965, in Hualien, Taiwan, R.O.C. He received his M.S. and Ph.D. degrees in Computer Science from National Chiao Tung University in 1989 and 1995, respectively. He is currently an Associate Professor of the Department of Computer Science and Information Engineering at National Dong Hwa University, Taiwan, R.O.C. His research interests include texture synthesis, visibility, and gaming.