ACCV2002

The 5th Asian Conference on Computer Vision, 23--25 January 2002, Melbourne, Australia.

Real-time Lip Synchronization Based on Hidden Markov Models

Ying Huang* [1] Stephen Lin+ Xiaoqing Ding* Baining Guo+ Heung-Yeung Shum+

*Dept. of Electrical Engineering
Tsinghua University

+Microsoft Research, China

Abstract

We propose a novel method of lip synchronization by re-using training video as much as possible when an input voice is similar to training voice sequences. Initially, face sequences are clustered from video segments, then by making use of sub-sequence Hidden Markov Models, we build a correlation between speech signals and face shape sequences. From this re-use of video, we can decrease the discontinuity between two consecutive output faces and obtain accurate and realistic synthesized animations. Our method can synthesize faces from input audio in real-time without noticeable delay.

Since acoustic feature data calculated from audio is directly used to drive our system without considering its phonemic representation, our method can adapt to any kind of voice, language or sound.

1. Introduction

Movement of the lips and chin during speech is an important component of facial animation. Although the acoustic and visual information of the speakers have vastly different characteristics, they are not completely independent since lip movements must be synchronized to speech. Using voice as the input, lip synchronization finds the correlation between lip movements and speech signals. This technique can be used in many applications such as video-phone, live broadcast, long-distance education, and movie dubbing.

In the last ten years, much work has been done in the area of face synthesis and lip synchronization. Techniques based on the methods of Vector Quantification (VQ) [1], Neural Networks [2,3,4], Hidden Markov Models [5,6,7] and Linear Predictive Analysis [8] have been proposed to map speech to lip movements. Most of the systems are based on a phonemic representation (phoneme or viseme). For example, Video Rewrite [9] re-orders existing video frames based on recognized phonemes. Since different people speak in different tones, considerable information will be lost in a phoneme-based approach. Moreover, the phonemic representation for different languages is also different. Brand introduces a method of generating full facial animation directly from audio signals, which is based on HMMs [6]. Although this method has achieved reasonable results, its animation is simplified by use of a mean face configuration with only 26 learned states.

Restricted by algorithm efficiency, all the aforementioned systems cannot support real-time face synthesis. Recently, several methods have been proposed towards this end. Goff et al. described the first prototype of the analysis-synthesis of a speaking face running in real-time [10]. They used five anatomical parameters to animate the lip model adapted to speech with a 200ms delay between audio and video. Huang and Chen implemented a real-time audio-to-visual mapping algorithm that maps the audio parameter set to the visual parameter set using a Gaussian Mixture Model and a Hidden Markov Model [11], but no delay data was mentioned. Morishima presented a real-time voice-driven talking head with a 64ms delay [12]. He converted the LPC Cepstrum parameters into mouth shape parameters by a neural network trained by vocal features.

A primary reason for the delays in the previous real-time algorithms is that future video frames need to be processed to ensure reasonable accuracy in synthesis. Our proposed method circumvents this problem with the use of video sequences. When acoustic data is determined to correspond to a given video sequence, strong future information is available to promote synthesis accuracy. Additionally, there are no discontinuities between consecutive faces in training videos, so we capitalize on this characteristic by re-using video as much as possible when the input voice is similar to voice sequences used in training. We implement this idea by building a map from the audio signals to short sequences of the training video using Hidden Markov Models. If the number of short sequences is more than 100, our animation can be composed of hundreds of different face configurations, and therefore most details of lip and chin movements during speech can be shown in our synthesized result.

[1] This work was done while Ying Huang worked at Microsoft Research China as an intern.

Compared to [1-8], our non-real-time approach obtains more accurate and realistic animation. Since acoustic feature data calculated from audio is directly used to drive our system, unlike [9] our method can adapt to any kind of voice, including non-speech voices. In particular, voices pronounced by different people in different languages can drive our system satisfactorily. The output is a sequence of face shapes that can be used to drive 2D, 3D, or image-based talking heads. By adjusting some parameters, our method can be used in the application of real-time face synthesis, where each face can be synthesized within the time interval between two consecutive frames (40ms for 25Hz animation). Although the performance of this real-time system is slightly lower than that of the non-real-time system, the results are nevertheless satisfactory. Compared to [10-12], our method can synthesize more realistic animations, has higher efficiency, exhibits no delay between audio and video, and adapts better to different kinds of voices.

The remainder of the paper is organized as follows. First, our approach is outlined in Section 2, and then our system is described in detail in Section 3. We present our results in Section 4. Lastly, the conclusion is given in Section 5.

2. Our approach

Our approach to lip synchronization is designed for real-time execution with highly continuous video. To produce better continuity of the generated lip motion, we utilize and re-use sequences from training video since consecutive frames naturally form a smooth progression. These face sequences, empirically chosen to be five frames in length, and their associated acoustic feature vector sequences are clustered from a training video to form 128 representative sequences.

The five frames of a face sequence comprise fifteen different sub-sequences: 1, 2, 3, 4, 5, 1-2, 2-3, 3-4, 4-5, 1-2-3, 2-3-4, 3-4-5, 1-2-3-4, 2-3-4-5 and 1-2-3-4-5. For each sub-sequence, we create a HMM which is initialized using its acoustic feature vector sequence. To train a sub-sequence HMM, we search through training videos for other sequences whose face shapes, defined by feature points, are similar to the five face shapes of this sequence, then use their corresponding acoustic data for training.

However, differences in faces and viewing environments among the training videos may lead to errors in face shape estimation, which is described in Section 3.1. Because of these differences, many face sequences in the videos are unused in training these HMMs, and consequently, the acoustic data used for training does not include all voice sequences present in the training videos. If an input voice differs from those in the training sequences, some distortions may appear in the exported face sequence. To reduce this error, we consider face states as well as face sequences. Face states are representative face shapes clustered from a training video, and are handled like unit-length sequences. By introducing face states into our algorithm, a broader range of voice data is modelled because while many five-frame sequences from a training video are discarded, individual frames are all usable for face state HMMs. Training a HMM for each state using all training data gives a method for handling atypical acoustic data that may arise from unmodelled vocal sequences or unusual intonations.

To incorporate such kinds of disparate HMMs into a single framework, Rabiner [18,19] proposed a high performance connected digit recognition system using the level building algorithm. For recognizing digits from an input voice, the level building algorithm computes the digit string with the maximum probability. Instead of using level building for speech recognition, we apply it to lip synthesis. We additionally introduce a search range parameter that limits the length of the observed sequence. In our search for the face shape sequence that best matches the input acoustic vector, the probabilities of both the face state HMMs and face sequence HMMs are calculated by the Viterbi algorithm, and the product of the segment probabilities gives the path probability. The face state or face shape sequence that has the greatest probability is then exported. A detailed description of this method is presented in the following section.

3. System overview

In the training phase, sequences of the training video including synchronized speech are prepared. With the vocal and facial data obtained from the training video, we create face states and face sequences, and then train their corresponding HMMs, which are used for face synthesis.

In the synthesis phase, our system computes vocal data from the input audio and exports face shapes synthesized by the combination of face state HMMs and face sequence HMMs. In the output phase, we prepare a contour image of a head as the background, deleting the eyes, nose, lips and chin. With the assumption that eye movement is independent of lip movement while speaking, we model eye action as independent eye blinking and eyeball movement. Finally, we copy the eyes and facial contour to the background and export the combined image.

3.1. Signal Processing

To obtain useful vocal data, we calculate the Mel-Frequency Cepstrum Coefficients (MFCC) [14, 15]. These coefficients are known to be useful for speech recognition and robust to variations among speakers and recording conditions.

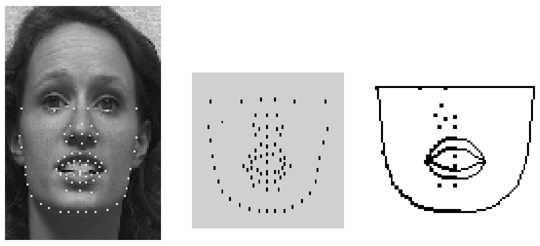

Figure 1. Three steps in our system to obtain facial data.

In our training video, the woman is accustomed to leaving her mouth open when she is not speaking, so we must separate silent frames and voice frames, or else the training result will be affected by this uncertainty. We develop a simple method based on the energy of each frame to perform this task. An energy histogram of 80 seconds of audio is first computed. This histogram exhibits two peaks, one indicating the energy center of the silent frames and one indicating the energy center of the voice frames. We take the average of the two peaks to separate these types of frames, and only the voice frames are used for training and synthesis. A face shape with a closed mouth is considered to be a silent frame.

To obtain facial data, we use the eigenpoints algorithm [16] to label the face and to identify the mouth and its shape, as shown in Figure 1A. Then we smooth the control points and enforce symmetry in Figure 1B. Figure 1C displays the output points that show the shape of the mouth, nose and chin. The upper line marks the lower eyelids, which we assume to be stationary during speech.

In our system, a PAL video sequence (25fps) is used as training data. The audio sampling rate is 44100Hz with 16-bit resolution. We calculate 12-dimensional MFCC coefficients and one energy parameter every 10ms, and map one image frame to four vocal frames.

3.2. Hidden Markov Models

Using the techniques mentioned above, 2000 face shapes and 8000 acoustic vectors are computed from 80 seconds of training video. After excluding all the frames labeled as silent, we obtain about 1500 short sequences, from which we cluster 128 sequences. Each sequence contains not only five face shapes, but also 20 acoustic vectors. The distance between two sequences for clustering is composed of the distance between their face shapes and the distance between their acoustic vectors. From each of these sequences, fifteen sub-sequences are used for HMM training. In addition, sixteen face shape states are also clustered from all the face shapes and are used for HMM training as well. The sequence length, the number of sequences and the number of face states are all experimentally determined. With this approach, we can not only synthesize faces from input audio in real-time, but also obtain accurate and realistic animation. For processing the voice data, we must calculate the average 12-dimensional MFCCs and the energy parameter of the input audio. These parameters can vary with different recording conditions and people, so we subtract them from the input acoustic feature vectors. Then we expand the 13-D vectors to 27-D [17] composed of five acoustic feature groups: Energy (E, ΔE, ΔΔE), MFCCs 1-5,MFCCs 6-12, ΔMFCCs 1-6, ΔΔMFCCs 1-6.

We utilize a left-right discrete HMM with four states. Five HMM models are created for each face state or sub-sequence according to the five acoustic feature groups. Since our system maps an image frame to four vocal frames, we assign at least four quantized vectors of the vocal frames to the observed sequences of HMMs. In the synthesis stage, five probabilities are computed by the Viterbi algorithm. The product of the five values is the output probability of that face state or sub-sequence.

3.3. Training

The state HMMs and sequence HMMs are trained separately. We generate face state indices from the face shapes in the training videos. Then the state HMMs are trained using the Baum-Welch algorithm by mapping one face state to the corresponding acoustic feature vectors.

We initialize the sub-sequence HMMs of each sequence using its acoustic feature vectors. The sub-sequence HMMs should adapt to not only the acoustic vectors of the sequence, but also to the acoustic vectors of other similar sequences in the training video. For short sequences in the training video where face shapes are similar to the five face shapes of a clustered sequence, the corresponding acoustic feature vectors will be included as observed data of the 15 sub-sequence HMMs of that sequence. In this way, the sequence HMMs cover a wider range of voices, and the more than 600 face shapes in the sequences are enough to generate realistic animations.

3.4. Synthesis

The input audio is composed of sound frames, separated by noise frames. A level building structure is then created for each segment The maximal probability and the optimal face sequence of each path are calculated by the function *ProcessPath*. This function and its variables are as follows:

PrevS:	the previous face sequence number.
PrevT:	If *PrevS* is –1, it is the previous face state number. Otherwise, it is the previous face shape number of the sequence. If *PrevT* is –1, the previous frame is silent.
S:	the current face sequence number.

T:	If *S* is –1, it is the current previous face state number. Otherwise, it is the current face shape number of the sequence.
n:	the current line number.
N:	the total number of lines in the path.
P:	the maximal probability of the path
StateP:	the maximal probability of the face states
StateN:	the face state that has the maximal probability
SeqP:	the maximal probability of the sub-sequences
L:	the length of the current line

function *ProcessPath*( )

```
PrevS ← –1
PrevT ← –1
P     ← 1
FOR   n = 1   To   N   DO
{
    Calculate the probabilities of the 16 face states
    StateP ← the maximal probability
    StateN ← the optimal face state
    IF   PrevS ≠ –1   THEN
    {
        SeqP ← the probability of the sub-sequence
            (PrevT+1) – (PrevT+2) – ··· – (PrevT+L)
            of the sequence PrevS
        IF   SeqP ≥ StateP   THEN
        {
                T   ← PrevT + L
                S   ← PrevS
                P   ← P * SeqP
                GOTO Jump
        }
    }
    Calculate the probabilities of the sub-sequence
                1 – 2 – ··· – L
        of the 128 sequences
    SeqP ← the maximal probability
    S    ← the optimal sequence
    IF   SeqP ≥ StateP   THEN
    {
        T  ← L
        P  ← P * SeqP
        GOTO Jump
    }
    S  ← –1
    T  ← StateN           [Export the best face state]
    P  ← P * StateP
Jump:
    PrevS ← T
    PrevT ← S
}
RETURN P
```

Finally, we export the face sequence of the path with the greatest probability.

Sometimes sound frame segments are very long, causing our program to search a large number of paths. To improve the efficiency, we limit the level number of the structure to 10 and divide long segments into short parts. The initial values of *PrevS* and *PrevT* are set as the last values of the previous part.

3.5. Real-time Face Synthesis

If the search range and the maximal level of the level building structure are set to 1, our method can be used in real-time face synthesis applications.

Before performing real-time synthesis, we first do some initialization. Our system will ask the user to input several seconds of voice audio, from which an energy histogram is formed and an energy threshold is computed.

In the real-time synthesis phase, if the energy of the input frame is less than this threshold, we will consider it a silent frame, and a face shape with a closed mouth will be assigned to it. We also calculate the average MFCC coefficients and the energy parameter of the initialization voice and subtract them from the input acoustic data before synthesis. Then we use the function *ProcessPath* mentioned in Section 3.4 to synthesis the face shape. Each time we process 40ms of input voice and return only one face shape. The values of *S* and *T* of the previous frame are also used to calculate the current S and *T*.

4. Results

In this section, we provide experimental results of our algorithm for both non-real-time and real-time face synthesis. To reiterate the requirements of our system, we utilize 9600 HMMs for the face sequences (5 acoustic features x 15 subsequences per face sequence x 128 face sequences) and 80 HMMs for the face states (5 acoustic features x 16 face states). The input to each subsequence HMM is four vocal frames for each subsequence element, and to each face state HMM is four vocal frames, where each vocal frame consists of a 12-D MFCC vector plus one energy parameter. The output for each face subsequence or face state is the product of the five probabilities given by its five HMMs. The system output is the face shape of the subsequence or state that has the highest probability.

4.1. Non-real-time Face Synthesis

We recorded eleven segments of videos that are about 80 seconds long. The face states and sequences were then clustered from the segment with the best tracking of face points by the eigenpoint technique. After initialization, the

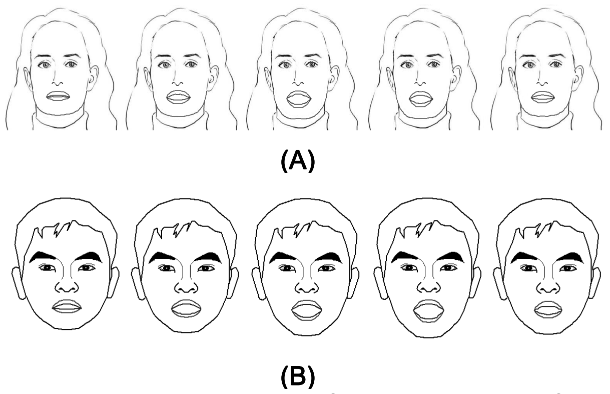

Figure 2. Two examples of the synthesized face sequence after smoothing. Two contour images are prepared as the background to be combined with the result.

face state HMMs and face sequence HMMs are trained using 20000 face shapes and 80000 acoustic feature vectors extracted from ten segments of videos. The remaining segment is then used to test our algorithm. Video discontinuities occur between consecutive frames in the following instances: one is a silent frame and the other is a voice frame, both are associated with face shape states, one comes a face shape state and the other from a face sequence, and each is from a different sequence.

To reduce the magnitude of discontinuities, both previous and subsequent faces are used to smooth the current face by coefficient averaging. On the other hand, closed mouths must be protected from being smoothed when plosions (/b/, /p/) are pronounced. Therefore, we appropriately adjust coefficients in different cases to find a best match between the original and synthesized faces.

Two examples of synthesized output frames are given in Figure 2. In Figure 3B and C, we compare the lip heights of the synthesized faces with the original ones when the system inputs several seconds of a person's voice. The slopes of the two curves are similar in most cases. At the same time, our curve matches the input sound wave and phonemes accurately. Although the two curves still have great differences in some cases, most of these cases occur in low energy frames such as silent frames (points a, b, c). How to build an appropriate model for low energy frames is still an open question.

In Figure 3B, three types of symbols indicate the model that is used to synthesize the face shape of a frame. Except for the silent frames, most sound frames are synthesized by the sequence HMMs. Although only a few frames are synthesized by the state HMMs, the face state HMMs are necessary for two reasons. First, the state HMMs are trained from all kinds of voices, while the sequence HMMs are trained only from videos in which the face shape sequences are similar to one of the clustered sequences. Since the state HMMs model a broader range of voices, they are needed when the input voice is much different from those used to train the sequence HMMs, such as at point d. Second, as described in the function *ProcessPath*, the maximal probability returned by the state HMMs serves as a standard value to judge whether the face shape synthesized by the sequence HMMs is accurate or not.

We invited different people to try our system. The synthesized result matched their voice accurately, and the animations seem very realistic. Although we trained our system using the English voice of a woman, the system can adapt to different kinds of languages, including Chinese. Other sounds such as laughs and catcalls can also drive our system.

4.2. Real-time Face Synthesis

Using the same model and the test video as in Section 4.1, we tested our real-time program. In real-time synthesis, only previously seen faces are used to smooth the current face, and we also adjust the coefficients for different cases to find a best match between the original and synthesized faces. In this way we not only obtain continuous animations, but also protect closed mouths from being smoothed when plosions are pronounced.

With the audio used in Figure 3, we have found that although there is slightly greater difference between the lip heights of the synthesized faces and the original lip heights, our result matches the input audio well.

The synthesis time for each 40ms speech segment is less than 22ms on a 733 MHz Pentium III PC. Therefore, people cannot detect any delay between input audio and synthesized video.

5. Conclusion

In this paper, we have introduced a novel method of lip synchronization based on Hidden Markov Models. By re-using training video sequences as much as possible, we can obtain realistic and accurate synthesized animations. By using the acoustic feature data calculated from audio to generate animations, our method can adapt to all kinds of voices. By adjusting some parameters, we can synthesize face shapes in real-time.

Although we tested our method with 2D faces, our method is also adaptable to 3D and image-based faces. More work will be done to find a best match between the synthesized animation and the input audio.

References

[1] S. Morishima and H. Harashima, "A media conversion from speech to facial image for intelligent man-machine

interface", *IEEE Journal on Selected Area in Communications*, 9(4), 1991.

[2] S. Curinga, F. Lavagetto, and F. Vignoli, "Lip movement synthesis using time delay neural networks", *Proc. EUSIPCO96*, 1996.

[3] F. Lavagetto, "Converting speech into lip movements: a multimedia telephone for hard of hearing people", *IEEE Transactions on Rehabilitation Engineering*, Vol. 3, No. 1, 1995, pp. 90-102.

[4] S. Curinga, R. Pockaj, F. Vignoli, C. Braccini, and F. Lavagetto, "Application of Synthetic Lip Motion to Hybrid Video Coding", *Int. Workshop on "Synthetic - Natural Hybrid Coding and 3D Imaging" (IWSNHC3DI'97)*, September 5-9, 1997, Rodhes, pp. 187-191.

[5] R. Rao and T. Chen, "Using HMM's for audio-to-visual conversion", *IEEE '97 Workshop on Multimedia Signal Processing*, 1997.

[6] M. Brand, "Voice Puppetry", *Proc. ACM SIGGRAPH'99*, 1999.

[7] E. Yamanoto, S. Nakamura, and K. Shikano, "Lip movement synthesis from speech based on hidden Markov models", *Proc. Int. Conf. On automatic face and gesture recognition, FG '98*, pages 154-159, Nara, Japan, 1998. IEEE Computer Society.

[8] S. Kshirsagar, and N. Magnenat-Thalmann, "Lip synchronization using linear predictive analysis", *Proceedings of IEEE International Conference on Multimedia and Expo*, New York, August 2000.

[9] C. Bregler, M. Covell, and M. Slaney, "Video Rewrite: Driving visual speech with audio", *Proc. ACM SIGGRAPH '97*, 1997.

[10] B. LE Goff, T. Guiard-marigny, M. Cohen, and C. Benoit, "Real-time analysis-synthesis and intelligibility of talking faces", *2nd International Conference on Speech Synthesis*, Newark(NY), September 1994.

[11] Fu Jie Huang, and Tsuhan Chen, "Real-time lip- synch face animation driven by human voice", *IEEE Multimedia Signal Processing Workshop*, Los Angeles, California, 1998.

[12] S. Morishima, "Real-time talking head driven by voice and its application to communication and entertainment", *Proc. AVSP 98, International Conference on Auditory-Visual Speech Processing*.

[13] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition", *Proceedings of the IEEE*, 1989,77(2):257-286.

[14] S. B. Davis, and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences", *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-28:357--366, August 1980.

[15] J. Picone, "Signal Modeling Techniques in Speech Recognition", *Proceedings of the IEEE*, 1993.

[16] M. Covell, and C. Bregler, "Eigenpoints", *Proc. Int. Conf. Image Processing*, Lausanne, Switzerland, Col. 3, pp. 471-474, 1996.

[17] Eric Chang, Jianlai Zhou, Shuo Di, Chao Huang, and Kai-Fu Lee, "Large vocabulary Mandarin speech recognition with different approaches in modeling tones", *International Conference on Spoken Language Processing*, Beijing, October 16-20, 2000.

[18]L. R. Rabiner, J. G. Wilpon, and B. H. Juang, "A segmental *k*-means training procedure for connected work recognition based on whole word reference patterns", *AT&T Tech. J.*, vol. 65, no. 3, pp. 21-31, May/June 1986.

[19] L. R. Rabiner, J. G. Wilpon, and F. K. Soong, "High Performance Connected Digit Recognition Using Hidden Markov Models", *IEEE Transaction on Acoustics, Speech and Signal Processing*, vol. 37, no. 8, August 1989.

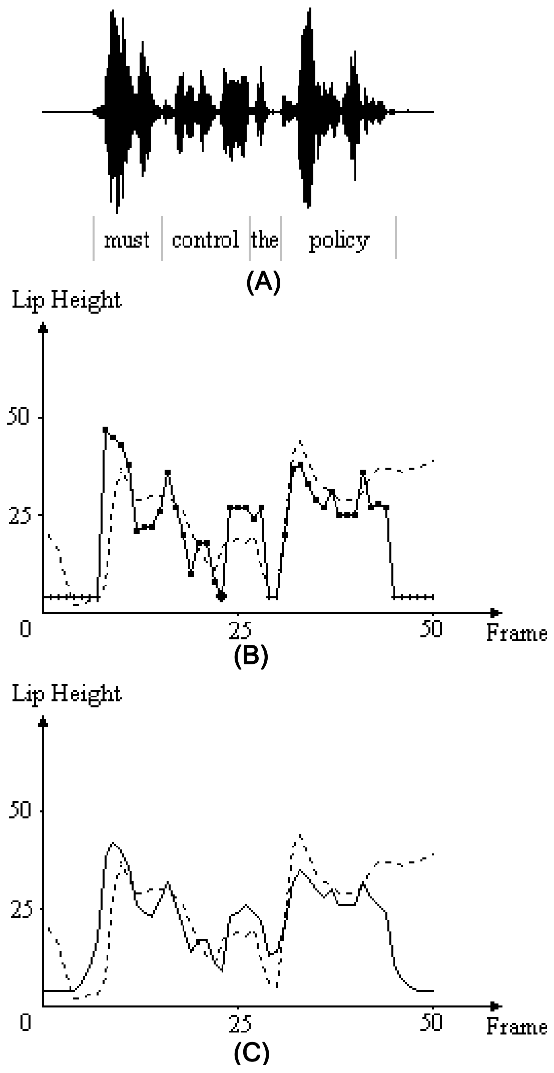

Figure 3. Comparison between the lip heights of the original faces and synthesized faces before smoothing (Figure B) and after smoothing (Figure C) when the system inputs the pronounced phrase "*must control the policy*" (Figure A).

In Figure B, the three kinds of signs indicate the HMMs used to synthesize the face of the current frame:

"+": silence

"▪ ": the face sequence HMMs

"●": the face state HMMs