Lip Synchronization: from Phone Lattice to PCA Eigen-projections using Neural Networks

Samer Al Moubayed [1], *Michael De Smet* [2], *Hugo Van hamme* [2]

[1] Centre of Speech Technology, KTH, Stockholm, Sweden
[2] ESAT-PSI, K.U.Leuven, Leuven, Belgium

sameram@kth.se, michael.desmet@esat.kuleuven.be, hugo.vanhamme@esat.kuleuven.be

Abstract

Lip synchronization is the process of generating natural lip movements from a speech signal. In this work we address the lip-sync problem using an automatic phone recognizer that generates a phone lattice carrying posterior probabilities. The acoustic feature vector contains the posterior probabilities of all the phones over a time window centered at the current time point. Hence this representation characterizes the phone recognition output including the confusion patterns caused by its limited accuracy. A 3D face model with varying texture is computed by analyzing a video recording of the speaker using a 3D morphable model. Training a neural network using 30 000 data vectors from an audiovisual recording in Dutch resulted in a very good simulation of the face on independent data sets of the same or of a different speaker.

Index Terms: lip synchronization, speech recognition, phone lattice, 3D morphable models, principal component analysis, audio visual speech.

1. Introduction

Facial and visual information is in many situations an invaluable complement to the acoustic speech signal, especially in environments where the speech signal could be disturbed by noise [1]. A fundamental component of audiovisual interaction is the visual motion of the lips according to the accompanying speech.

Some approaches to lip synchronization are based on visemes, which are the visual counterpart of the phone. The approach of mapping phones to visemes has been tried [2,3], where every phone was mapped to one or more representations of the viseme and morph-smoothing techniques were applied on the output vectors. This method has failed simulating the viseme–viseme transitions, and co-articulation. Other approaches used neural networks to map basic speech features to face parameters. For example in [4], a neural network was used to map cepstral vectors to face parameters.

We are tackling the lip-sync problem from a new corner. At the speech end, an Automatic Phone Recognizer (APR) was used, and a phone transition lattice was extracted. This architecture has the advantage that know-how for obtaining characteristics such as noise robustness and speaker independence can be reused straightforwardly from automatic speech recognition technology. Additionally, lexical and language constraints can be imposed on the lattice [5], [6]. The use of a lattice instead of a single recognition result allows taking the uncertainty of the APR into account. At the facial end, we present an approach which requires re-training for each new speaker, but results in a highly accurate representation. The talking head is a detailed 3D reconstruction of the speaker's face, and the animation mimics his personal talking style.

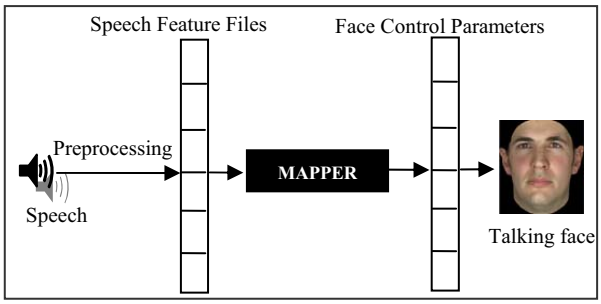

Figure 1: *Data flow of lip synchronization.*

2. Speech processing

Figure 1 shows the architecture of our system. For extracting features from the speech signal, the phone recognizer of [5] is applied, a state-of-the-art LVCSR (large vocabulary continuous speech recognizer) that gives around 75% phone recognition accuracy using only phone-level N-grams as a language model. An MLP-based mapper will subsequently transform the speech feature vector into the facial parameters.

2.1. Phone lattice recognition

The HMM-based phone recognizer produces what is called a phone lattice or a phone probability network (Figure 2) and uses context-dependent phone models [6]. In a phone lattice, the sum of all arc probabilities at each time point equals one. Every arc represents a phone, and carries the probability of the occurrence of this phone between two time points specified by the start node and end node of the arc.

In this experiment, we did not search for the best path in the lattice to recognize a phone string (using Dynamic Programming or the Viterbi algorithm), but we introduced the whole phone lattice as a speech feature. This representation enabled us to represent the uncertainty of the recognizer and use it as speech features, instead of taking the best path and losing the information which doesn't fit with the best path.
The phone lattice holds on its arcs the posterior probabilities [7] of the phones they represent. In order to reduce the dimension of the feature vectors to make the processing faster, enable real-time simulation and reduce the data requirements to model the mapper, all silence rows are summed up to one silence row, as well as all phones that map to the same visual articulation (viseme), a many-to-one mapping similar to the one in [8] was applied. This reduces the size of the feature vector up to 50%.

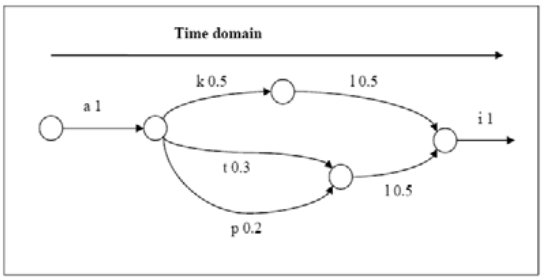

Figure 2: *A phone lattice – every arc represents a phone, and carries a posterior probability of this phone between the start and end time of this arc. The sum of all the arc probabilities at each time point is one.*

To extract time-point feature vectors, the lattice was transformed to a matrix, where every column contains the probabilities of every phone at that time point (Figure 3). The resulting matrix is sparse, since the output of the recognizer gives zeros to most of the probabilities of the phones at every time point.

2.2. Language model

For speech recognition, we only used an Automatic Phone Recognizer (APR). Other than a phone level trigram, no Language Model (LM) was used to enhance the recognition accuracy.

The main reasons for not using a word-level LM are the following.

1) The lip movements do not depend on the language or the words the speaker is uttering, but mainly on the sounds. Even if the speaker is pronouncing Out Of Vocabulary (OOV) words, the lips should move in the right way for the observer.

2) Language models are domain dependent. If we want to benefit from using the LM, we have to constrain the framework only to the domains those LMs are trained to cover.

3) Without language models, recognition errors stay localized. When a phone is recognized wrongly, the lips will move in the wrong way only for this phone trajectory length. LMs would result in incorrectly recognized words, hence making the error at the phone level visually spread to a whole word or even beyond.

In case the language domain can be restricted, it might be interesting to constrain the phone lattice lexically and with a language model, but our tests do not include such constraints.

2.3. Context dependent articulation

Face articulation does not only depend on the phone or the viseme being uttered, but also on transitions between a phone and the phones before and after it,. We therefore predict the current mouth position based on the past, present and future speech. Hence, the feature vector at the *k*th frame should include the *N* previous frames (history) and the *M* next frames (future) i.e. it spans the speech range [*k*-*N*, *k*+*M*]. Therefore, we represent the current speech frame by concatenating all the feature vectors in that range. In this representation, if P is the number of the phones in a language, then the size of the vector for each frame is P * (1+*M*+*N*) and the sum over all components in the input vector at any time point is 1+*M*+*N*.

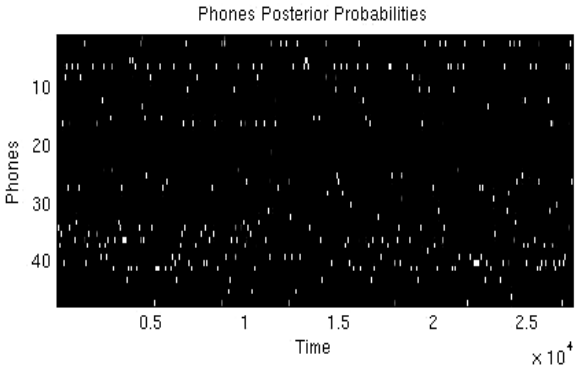

Figure 3: *Matrix form of a lattice shown as a gray-scale image. A column is a time point, and each row is a phone identity; black cells are zero probabilities and white cells are one.*

3. Video processing

For generating realistic lip movements of a speaker, 2D (image based) methods as well as 3D (model based) methods have been proposed. The main advantage of the 2D approach [9] is that it can be trained using only a video of the person speaking. However, this constrains the artificially generated lip movements to the same viewpoint and illumination as seen in the training video. The 3D approach [3] on the other hand, is able to reproduce lip movements under arbitrary pose and lighting conditions. The drawback is that for realistic, person-specific lip movements, 3D recordings of the face during speech are required. We present a texture based approach, which combines the ease of training of 2D with the versatility of 3D.

3.1. Analysis of the speaker's face

Our method requires a static 3D model of the face in a neutral expression (mouth closed). This could be obtained using a 3D scanning device, but such equipment is expensive and usually not available. We therefore employ the technique of [10], which generates a 3D model of the speaker's face by fitting a 3D Morphable Model (3DMM) to one or more images. The images used for generating the 3D model were eight frames taken from a training video. In addition to computing the 3D shape of the face, texture extraction techniques are employed to generate a combined texture map for the neutral face (Figure 4).

Once the geometry and texture of the neutral face are obtained, we use this model as a template to track the 3D rigid motion and illumination of the speaker's face throughout the training video. The process is complicated by the presence of non-rigid motion in the eyebrows, eyes, mouth, lower jaw and neck region. Instead of manually defining the regions of the face where non-rigid motion occurs as in [9], we adopt the strategy of [11], which automatically identifies regions that do not correspond well with the model as outliers. This also provides robustness in the presence of occlusions, such as a sheet of paper moving in front of the speaker's face, which occurred several times during the training video.

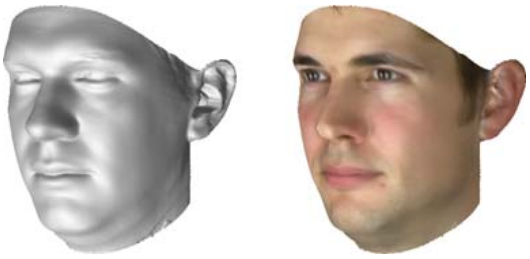

Figure 4: *3D model of the speaker's face obtained by fitting a 3DMM to eight frames of a training video. The texture on the right is the result of texture extraction.*

3.2. Texture analysis of the lip movements

After tracking, we apply texture extraction to each video frame. This results in a representation that is almost insensitive to changes in viewpoint and illumination (Figure 5). Since we are only interested in changes in appearance that are due to speech, we discard the forehead, the eyes, the ears, and part of the cheeks. The remaining region is analyzed using PCA, yielding a mean texture and several eigentextures. After projection onto the PCA eigenspace, each texture can be represented by a feature vector containing its eigencoefficients, which will be predicted from the audio. Using the feature vectors at each time point, the original texture animation can be reconstructed to an arbitrary degree of accuracy depending on the number of principal components retained. A complete face texture can be generated by combining the animated PCA region with the eyes, forehead and ears using straightforward blending techniques.

4. Data recordings

The complexity of function estimation depends mainly on the type of data supplied, the generality, the size, and the accuracy.

An audiovisual recording of 20 minutes of read Dutch speech was made by a speaker not included in the acoustic database used for training the APR. The video sampling rate was 25 fps (frames per second), and the audio was 100 fps. The speech was analyzed and the feature vectors were extracted with a variable context window size ranging from zero frames up to N=M=5. Since the video sampling rate was different from that of the speech, the speech lattice was subsampled to match the video sampling rate. The video was analyzed according to the method described in Section 3, resulting in a vector of texture coefficients for each frame.

5. Training

We trained a Multi Layer Perceptron (MLP) as a general function approximator to predict the face texture eigencoefficients from the digitized matrix of the phone lattice. Assuming that there is no discontinuity in the face parameters we want to estimate (the changes on the face are continuous), an MLP (multilayer perceptron) with one hidden layer should be adequate to estimate the targets.

Figure 5: *The texture representation shows a high degree of invariance w.r.t. changes in viewpoint and illumination. Top row: clips from the original video. Bottom row: corresponding results of texture extraction. For ease of interpretation, the extracted textures have been rendered according to a frontal view of the face model.*

6. Experiments and results

Different neural networks were trained to predict the 10 texture eigencoefficients from the K*(1+M+N) inputs, with the number of neurons in the hidden layer ranging from 10 to 50, with a step size of 10. From the recorded data, 60% was used for training, 20% for validation, and 20% for testing. As a result of the training, increasing the number of hidden neurons from 10 to 20 and then to 30, decreased the Mean Square Error (MSE) on the validation set for every 10 neurons added. When increasing the number of neurons beyond 30, the training became slow, and the gain in terms of MSE became unrecognizable.

As for the context window size shown in Table 1, the MSE decreased (slightly) when increasing the context window size from zero to three (history and future). For larger windows, the size of the training set became too small to enable the network to generalize.

Visually, the test data showed an improving estimation of parameters with increasing context size, and became jerky with a context of five. The quality of lip synchronization was very good with silence, as well as long vowels and consonants (Figure 6).

New test data was introduced to the network, a recording of a female voice at a lower signal-to-noise ratio than for the male speech. The framework was able to generalize to give a very similar quality compared to the test over the original recording conditions.

7. Discussion

Using the proposed speech feature vectors, we expect that with longer recordings, covering all tri-phone sequences, and by applying smoothing techniques on the output vectors of the network, the accuracy of the output can be improved further.

Context Size	MSE
0	0.074
1	0.075
2	**0.063**
3	0.063
4	0.075
5	0.090

Table 1: *The mean square error on the development set while training a 30 hidden neurons MLP as a function of the context window size (in both directions), with 300 epochs of training as a maximum.*

On the other hand, the proposed parameterization of the texture as linear combinations of eigentextures complicates the function estimation problem. This is because a straightforward linear interpolation between mouth configurations in texture space does not produce realistic results for intermediate mouth shapes. Hence, plausible paths between visemes are inherently nonlinear in our parameterization. We suspect that applying optical flow based techniques to the texture component may significantly improve the quality of the results. Such an approach may also reduce the amount of training material that is required for adapting the model to a new speaker's face.

It is worth mentioning here that even in its current form, our system can be automated enough to enable training at the user end, using only one camera and a user manual.

8. Conclusions

Using an MLP, we were able to predict PCA eigen-coefficients of the texture of a 3D face model from posterior phone probabilities generated by an automatic speech recognizer using a trigram phone model. Using the right parameters, network size, context size, and data size, this approach can offer very good lip-sync quality using real faces. Using this method, people can build their own 3D model at home, with textures simulating the user's own face and articulation style.

9. Acknowledgement

This research was partly funded by IWT-Vlaanderen. The authors also thank Peter Karsmakers for his willingness to record the training data.

10. References

[1] W. H. Sumby and I. Pollack, "Visual contribution to speech intelligibility in noise," *J. Acoust. Soc. Am.*, vol. 26, no. 2, pp. 212–215, March 1954.

[2] T. Ezzat and T. Poggio, "MikeTalk: A talking facial display based on morphing visemes," In *Proc. of the Computer Animation Conference '98*, Philadelphia, PA, June 1998, pp. 96–102.

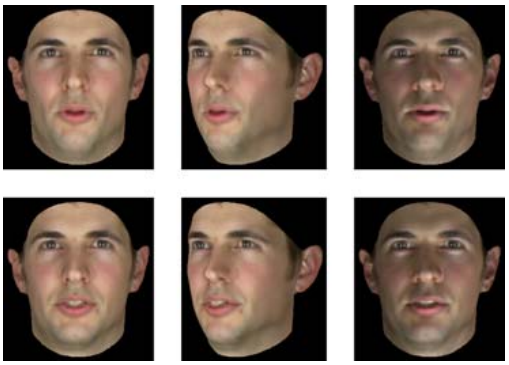

Figure 6: *Lip synchronization results for a novel speech signal. The middle and right columns illustrate our method's capability to generate results for novel viewpoints and illumination conditions.*

[3] P. Vanroose, G. A. Kalberer, P. Wambacq, and L. Van Gool, "From speech to 3D face animation," In *Proc. of the 23rd Symposium on Information Theory in the Benelux*, Louvain-la-Neuve, Belgium, May 2002, pp. 255–260.

[4] D. W. Massaro, J. Beskow, M. M. Cohen, C. L. Fry and T. Rodriguez, "Picture my voice: Audio to visual speech synthesis using artificial neural networks," In *Proc. AVSP '99*, Santa Cruz, CA, August 1999, pp. 133–138.

[5] K. Demuynck, T. Laureys, D. Van Compernolle and H. Van hamme, "FLaVoR: A flexible architecture for LVCSR," In *Proc. European Conference on Speech Communication and Technology*, Geneva, Switzerland, September 2003, pp. 1973–1976.

[6] K. Demuynck, D. Van Compernolle and H. Van hamme, "Robust Phone Lattice Decoding," In *Proc. International Conference on Spoken Language Processing*, Pittsburgh, PA, September 2006, pp. 1622–1625.

[7] F. Wessel, R. Schluter, K. Macherey and H. Ney, "Confidence measures for large vocabulary continuous speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 3, pp. 288–289, March 2001.

[8] Patrick Lucey, Terrence Martin and Sridha Sridharan. 2004. *Confusability of Phonemes Grouped According to their Viseme Classes in Noisy Environments*. Presented at Tenth Australian International Conference on Speech Science & Technology, Macquarie University, Sydney, 8th-10th December, 2004.

[9] T. Ezzat, G. Geiger, and T. Poggio, "Trainable videorealistic speech animation," In *Proc. of ACM SIGGRAPH 2002*, San Antonio, TX, July 2002, pp. 388–398.

[10] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3D faces," In *Proc. of ACM SIGGRAPH '99*, Los Angeles, CA, August 1999, pp. 187–194.

[11] M. De Smet, R. Fransens, and L. Van Gool, "A generalized EM approach for 3D model based face recognition under occlusions," In *Proc. CVPR*, New York, NY, June 2006, vol. 2, pp. 1423–1430.