

Codebook Based Face Point Trajectory Synthesis Algorithm Using Speech Input[§]

Levent M. Arslan* and David Talkin

Entropic Research Laboratory, Washington, DC 20001 U.S.A.

Technical Report

Abstract

This paper presents a novel algorithm which generates three-dimensional face point trajectories for a given speech file with or without its text. The proposed algorithm first employs an off-line training phase. In this phase, recorded face point trajectories along with their speech data and phonetic labels are used to generate phonetic codebooks. These codebooks consist of both acoustic and visual features. Acoustics are represented by line spectral frequencies (LSF), and face points are represented with their principal components (PC). During the synthesis stage, speech input is rated in terms of its similarity to the codebook entries. Based on the similarity, each codebook entry is assigned a weighting coefficient. If the phonetic information about the test speech is available, this is utilized in restricting the codebook search to only several codebook entries which are visually closest to the current phoneme (a visual phoneme similarity matrix is generated for this purpose). Then these weights are used to synthesize the principal components of the face point trajectory. The performance of the algorithm is tested on held-out data, and the synthesized face point trajectories showed a correlation of 0.73 with true face point trajectories.

Keywords: principal component analysis, 3-D, face synthesis, lip synching, codebook, audio-visual, visual similarity matrix.

Mail All Correspondence To:

Levent M. Arslan
Bogazici University,
Electrical and Electronics Department,
Bebek, 80815, Istanbul, TURKEY.

email: arslanle@boun.edu.tr

Phone: +90 (212) 263 15 00 ext. 1421

FAX: +90 (212) 287 24 65

<http://www.ee.boun.edu.tr>

*: Levent Arslan is now with Bogazici University Electrical and Electronics Engineering Department, Bebek, 80815, Istanbul, Turkey.

[§] Submitted on April 16, 1998. Revised on September 2, 1998.

1 Introduction

Recently there has been significant interest in the area of face synthesis. This topic has numerous applications including film dubbing, computer-based language instruction, cartoon character animation, multimedia entertainment, etc. There is a large effort in developing autonomous software agents that can communicate with humans using speech, facial expression, gestures, and intonation. Katashi and Akikazu (Katashi and Akikazu 1994) employed animated facial expressions in a spoken dialogue system. Other researchers (Cassel et al. 1994, Bertensham et al. 1995, Beskow 1997) used various forms of visual agents that featured animated gestures, intonation, and head movements. Lip synching is another application of wide interest. The Video Rewrite system (Bregler et al. 1997) uses existing footage to automatically create new video of a person mouthing words that she did not speak in the original footage.

In this study, we propose a new algorithm to synthesize three dimensional face point trajectories corresponding to a novel utterance. The general algorithm does not require any text input. However, the performance of the algorithm significantly improves if phonetic information is known a priori. Therefore, throughout this paper the algorithm will be described assuming phonetic information is available. At the end we will describe what the proposed algorithm is like for the case where phonetic information is not available. The most significant contribution in this paper is that it addresses the problem of generating audiovisual speech from the acoustic signal alone. Therefore, it is possible to add the proposed system to acoustics-only synthesizers to complement speech with visual information.

The general outline of the paper is as follows. Section 2 describes the proposed face point trajectory synthesis algorithm. In this section, the formulation and automatic generation of a novel visual phoneme similarity matrix is described as well. Section 3 presents the simulations and performance evaluation. Finally Section 4 discusses the results and future directions.

2 Algorithm Description

The face synthesis algorithm proposed in this paper is an extension of the STASC voice transformation algorithm which is described in Arslan and Talkin (1997). The STASC algorithm modifies the utterance of a source speaker to sound like speech from a target speaker. The acoustic parameters (LSF) are transformed to target speaker acoustic parameters by employing a weighted codebook mapping approach. The generation of audio codebooks in this paper

follows the same methodology as in the STASC algorithm. However, instead of mapping the acoustic parameters to a target speaker’s acoustic space, the proposed algorithm maps the incoming speech into the source speaker’s own acoustic space, and augments it with visual data. The flowchart of the proposed face synthesis algorithm is shown in Figure 1. The algorithm requires two on-line inputs: i) a digitized speech file; and ii) its corresponding phoneme sequence. It also requires two additional inputs which are generated prior to face synthesis during the training stage: i) an audio-visual codebook; and ii) a visual phoneme similarity matrix. First, we will explain how the codebook and the visual phoneme similarity matrix are generated.

INCLUDE FIGURE 1 HERE.

2.1 Audio-Visual Codebook Generation

For the data collection, synchronized speech and face point trajectories must first be recorded from a subject. For this study the point trajectories were recorded using a multi-camera triangulation system yielding 60 samples/sec at a spatial resolution of .254 mm in X, Y, and Z. In the pilot study reported here, 54 points on and around the face were recorded while a single subject uttered approximately 300 TIMIT sentences selected to provide the richest possible phonetic coverage. Projections of the 54 points onto a two dimensional plane are shown in Figure 2. Unfortunately, tongue movement was not included in the dataset. Some of the marked points on the face do not factor into phonetic discrimination since they do not move significantly (e.g., points on the forehead, and on the nose). However, these points were necessary to estimate and factor out head and jaw movements. Speech and electroglottograph (EGG, Glottal Enterprises) data were also digitized via a DAT recorder at 48kHz, then later digitally down-sampled to 16kHz for more compact storage. Principal Component Analysis (PCA) is applied to the trajectories to reduce the $(54 \times 3)=162$ dimensions to 20 (20 principal components account for 99 % of the total variance. This appears to capture all of the visually relevant movement). Actually, the fixed points could have been eliminated prior to PCA, however we let the PCA decide what is important and what is not by including all the points in the face in our analysis.

The acoustic signal is segmented at the sub-phonetic level using a hidden Markov model (HMM) speech recognizer run in forced-alignment mode. In the forced-alignment mode, the recognizer searches the best path through states of the HMM sequence which corresponds to the phonetic translation of the orthographic transcription of the test utterance (Wightman and Talkin 1994). The HMM’s were trained on a larger corpus of speech-only data from the talker,

and thus yield reliable and consistent phonetic segmentations of both the audio and trajectory data (including, of course, the PCA reduced data).

Acoustic features are line spectral frequencies (LSF) which provide a compact representation of the speech signal. They have a number of nice properties which make them attractive among speech researchers especially in the speech coding area. The relation of LSFs to visual features have been investigated by Yehia et. al. (Yehia et al. 1998). It was found that 91% of the total variance observed in the facial motion data could be determined from LSFs by means of simple linear estimators. For the inverse path, i.e., recovery of vocal-tract motion from facial motion, they found that about 80% of the variance observed in the vocal tract can be estimated from the face. For a more detailed treatment of LSFs the reader can refer to Kleijn and Paliwal (1995).

The visual features are principal components of 162 dimensional face point parameter vector. The principal components can be obtained using the Karhunen-Loeve transformation technique (Fukunaga 1990). Since the movements of points on the face are highly correlated, a significant dimensionality reduction can be achieved with minor distortion. Principal components have been used for various applications in image processing and face animation (Bregler et al. 1997). A nice property of principal components is that they represent the directions which correspond to most correlated movements. Therefore, one can modify the weights on principal components to animate the underlying face with realistic motions. For example, the eigenvector with the largest eigenvalue in our analysis corresponded to lower jaw movement. This was expected because the lower jaw corresponds to the most highly correlated set of points in a talking person's face. Therefore, we were able to move the lower jaw trajectories just by modifying the value of the first principal component. Figure 2 shows the face point trajectories corresponding to variation of two principal components. The circles represent the original face points, and the arrows indicate the movement of face points (displacement vector) after adjusting the component coefficient. The second principal component in our analysis corresponded to the movement of the sides of mouth. The influence of adjusting the first and second principal components is shown in Figure 2.

INCLUDE FIGURE 2 HERE.

We use an audio-visual codebook in order to model the acoustic and visual features specific to a speaker. Each codebook entry in our formulation corresponds to a certain context and it consists of an acoustic feature vector and a visual feature vector. Associated with each specific

context are 5 codewords corresponding to uniformly spaced locations in time across the duration of the phoneme. The audio-visual codebook entries are generated as follows. First, the speech data is segmented into phonemes. Next, each phoneme is tagged with a symbol which we refer to as “context-phone” which represents the left and right contexts of the phoneme. Figure 3 illustrates the output of a typical segmentation procedure on the word “whenever”. For example, the phoneme /eh/ is represented as /w_eh_n_eh_v_axr_f/ including three closest neighbors on each side. After the data is tagged this way, each phoneme is labeled with 5 uniformly spaced time locations. The acoustic and visual features corresponding to those 5 locations are then appended to the audio-visual codebook¹. For example, in the figure the time locations of 5 entries that correspond to the /eh/ phoneme are shown at the bottom.

INCLUDE FIGURE 3 HERE.

2.2 Automatic Generation of Visual Phoneme Similarity Matrix

Since in practice the training data will not include all possible context-phones, we need a way of associating unseen context-phones with the audio-visual codebook. In this paper, a novel procedure for automatic selection of the closest context-phone is developed. The criterion that we chose for visual similarity of phonemes is based on Euclidean distance of the principal components of face data. Therefore, initially an average principal component vector is estimated for each phoneme:

$$\mathbf{m}_k = \frac{1}{T_k} \sum_{t=1}^{T_k} \mathbf{P}_{kt}, \quad k = 1 \dots K, \quad (1)$$

where K represents the total number of phonemes present in the language, T_k represents the number of tokens for the k^{th} phoneme, and \mathbf{P}_{kt} represents the t^{th} principal component coefficient vector that is associated with k^{th} phoneme. Then, the Euclidean distance between each phoneme pair is calculated as:

$$\mathbf{D}_{ik} = \|\mathbf{m}_i - \mathbf{m}_k\| \quad i = 1 \dots K, \quad k = 1 \dots K. \quad (2)$$

Finally, a similarity measure is derived from the distances by using:

$$\mathbf{S}_{ik} = e^{-\nu \mathbf{D}_{ik}} \quad i = 1 \dots K, \quad k = 1 \dots K. \quad (3)$$

This formulation assures that similarity values, \mathbf{S}_{ik} , will range between 0 and 1. The constant ν in the equation can be adjusted to control the dynamic range of similarity values appropriately.

In the experiments reported in this study we used a value of 10 for ν . A gray-scale image of the visual similarity matrix derived using this procedure is shown in Figure 4. Darker squares in the figure represent higher levels of similarity. For example, the most similar phoneme to /b/ was identified to be /p/, and the most similar phoneme to /uh/ was identified as /uw/, based on the estimated similarity matrix. In general, it is observed that the entries in the automatically derived matrix agree with intuitive expectations. However, we have not performed subjective tests to verify this statement yet. The top three visually most similar phonemes and their similarity measures are listed in Table 1 for reference.

Next, we formulated a procedure to pick visually most similar context-phones to an unseen context-phone. It has been shown that visual confusability depends highly on the context of a phoneme (Auer et al. 1997). Therefore, we have taken into account the context of a phoneme when selecting the appropriate context-phones in the codebook. We represent the context-phone as $\dots l_3 l_2 l_1 c r_1 r_2 r_3 \dots$, where “ l_n ” represents n^{th} phoneme to the left, “ r_n ” represents n^{th} phoneme to the right, and c represents the center phoneme. The similarity of a test context-phone to each of the context-phones in the codebook can be formulated as:

$$\mathbf{n}_j = \mathbf{S}_{cj} + \sum_{i=1}^C \xi^{-i} \mathbf{S}_{l_{ij}} + \sum_{i=1}^C \xi^{-i} \mathbf{S}_{r_{ij}} \quad j = 1 \dots L \quad (4)$$

where C is the level of context information, L is the total number of context-phones in the codebook, \mathbf{S}_{cj} is the similarity between the center phone of the unseen context-phone and the j^{th} context-phone in the codebook, $\mathbf{S}_{l_{ij}}$ is the similarity between the i^{th} left phoneme of the unseen context-phone and the j^{th} context-phone in the codebook, and $\mathbf{S}_{r_{ij}}$ is the similarity between the i^{th} right phoneme of the unseen context-phone and the j^{th} context-phone in the codebook. Since similarity matrix values range between zero and one, by selecting ξ to be greater than 10 one can assure that center phoneme match will always have the highest precedence in the decision procedure, and as we move away from the center the influence of match will decrease.

INCLUDE FIGURE 4 HERE.

INCLUDE TABLE 1 HERE.

The next section describes the face synthesis process using the visual phoneme similarity matrix and the audio-visual codebook.

2.3 Face Synthesis

In the first step of the face synthesis process, the context-phone which corresponds to the incoming speech frame is compared to available context-phones in the codebook in terms of their visual similarity. Using the similarity metric discussed in the previous section, the top N most similar context-phones are selected in the audio-visual codebook. Next the LSF vector for the incoming speech frame is calculated. LSFs can be estimated by modifying the LPC polynomial, $A(z)$, in two ways: $P(z)$ and $Q(z)$ are obtained by augmenting $A(z)$'s PARCOR (see Deller et al. (1993)) sequence with a +1 and -1 respectively. This results in the following two polynomials which have all their roots on the unit circle

$$\begin{aligned} P(z) &= (1 - z^{-1}) \prod_{k=1,3,5,\dots}^{P-1} (1 - 2 \cos \mathbf{w}_k (z^{-1} + z^{-2})) \\ Q(z) &= (1 + z^{-1}) \prod_{k=2,4,6,\dots}^{P-1} (1 - 2 \cos \mathbf{w}_k (z^{-1} + z^{-2})), \end{aligned} \quad (5)$$

where P is the LPC analysis order, and the angles of the roots, \mathbf{w}_k , are line spectral frequencies.

The acoustic feature vector corresponding to the incoming speech frame is compared to all the LSFs that correspond to the top N context-phones. There will be $5N$ such vectors, since each context-phone is represented with 5 uniformly spaced audio-visual vectors. The incoming LSF vector \mathbf{w} is compared with each LSF vector, \mathbf{L}_i , in the codebook and the distance, \mathbf{d}_i , corresponding to each codeword is calculated. In general, pairs of LSFs characterize a formant frequency (Crosmer 1985), and the bandwidth of a given formant depends on the closeness of the corresponding LSFs. Formants that have narrow bandwidths can have a greater influence on the human perception of acoustic signals than similar formants that have wider bandwidths. The distance calculation takes this perceptual weighting into account, and closely spaced line spectral frequencies which are likely to correspond to sharp formant locations are assigned higher weights (Laroia et al. 1991):

$$\begin{aligned} \mathbf{h}_k &= \frac{1}{\operatorname{argmin}(|\mathbf{w}_k - \mathbf{w}_{k-1}|, |\mathbf{w}_k - \mathbf{w}_{k+1}|)} & k = 1, \dots, P \\ \mathbf{d}_i &= \sum_{k=1}^P \mathbf{h}_k |\mathbf{w}_k - \mathbf{L}_{ik}| & i = 1, \dots, 5N \end{aligned} \quad (6)$$

where $5N$ is the reduced codebook size based on context. However, if the phonetic information is not available $5N$ can be replaced with the whole codebook size $5L$. This way the codebook search is not restricted based on context and therefore face features can be derived based on acoustic information only (i.e., language specific pronunciation dictionaries are not required).

However restricting the codebook search by incorporating phonetic information always increases the performance of the algorithm, and for most practical purposes having phonetic information a priori is a reasonable assumption to make. Therefore, we will continue the discussion assuming we have access to phonetic information. Based on the distances from each codebook entry, an expression for the normalized codebook weights can be obtained as in Arslan et al. (1995):

$$\mathbf{v}_i = \frac{e^{-\gamma d_i}}{\sum_{i=1}^{5N} e^{-\gamma d_i}} \quad i = 1, \dots, 5N \quad (7)$$

This set of weights \mathbf{v} allows us to approximate the original LSF vector \mathbf{w} as a weighted combination of codebook LSF vectors:

$$\hat{\mathbf{w}}_k = \sum_{i=1}^{5N} \mathbf{v}_i \mathbf{w}_{ik} \quad (8)$$

The value of γ in the previous equation is found by an incremental search in the range of 0.2 to 2 with the criterion of minimizing the perceptual weighted distance between the approximated LSF vector $\tilde{\mathbf{w}}$ and the original LSF vector \mathbf{w} . The set of weights \mathbf{v} estimated based on acoustic similarity are used to construct the PCs of face points corresponding to the current speech frame:

$$\hat{\mathbf{p}}(t) = \sum_{i=1}^{5N} \mathbf{v}_i \mathbf{F}_i \quad (9)$$

where \mathbf{F}_i represents the average principal component vector for i^{th} codebook entry. Next, the time sequence of estimated principal component vectors, $\hat{\mathbf{p}}(t)$ is smoothed to provide more natural face point trajectories. The next section describes two smoothing algorithms that we tried.

2.4 Smoothing Principal Components Trajectory

In general the output of the proposed face point trajectory synthesis algorithm is not smooth and natural, although the trajectories pass through appropriate locations. Therefore, we employed a smoothing algorithm to obtain more natural trajectories. We tested two methods for the smoothing of the trajectory: i) triangular windowing; and ii) spline interpolation. In the first method, a triangular averaging window over a 100 ms duration is used to smooth the principal component trajectories. However, in general this method does not result in natural face point trajectories. The second method employs spline interpolation method, and produces more natural trajectories. The most important criterion in the design of the second algorithm was to

be able to preserve the minimum and maximum points in the face trajectory. This was necessary because smoothing using a simple averaging filter resulted in the reduction of the dynamic range of face point movements. The proposed smoothing algorithm is shown in Figure 5. First, a low-pass filter is applied to the principal component vector trajectory. Next, a minimum-maximum picking algorithm is employed to locate the regions where extrema occur. Once the extreme locations are obtained, a short averaging window (50 ms) is applied around the min-max points to the unfiltered principal component vector. A cubic spline interpolation is performed based on min-max points to generate the principal components corresponding to the rest of the data. Finally, the smoothed principal components are used to construct the synthesized face point trajectories.

INCLUDE FIGURE 5 HERE.

3 Evaluations

We used ten minutes of audio-visual training data from a single talker to generate our codebooks and the visual phoneme similarity matrix. Five minutes of data were set aside for testing. The visual data were recorded at a 60 Hz sampling rate. Using the proposed algorithm face point trajectories were synthesized for the test data. In order to test the upper limit on the performance of the algorithm, we resynthesized the training utterances as well. Figure 6 shows an example face trajectory synthesized from one of the test utterances. Here, the middle plot shows the center upper lip point trajectories along the y-axis across time for original (dark dotted curve), synthesized with spline smoothing (dark solid curve), and synthesized with triangular smoothing (light curve). The bottom plot shows the center lower lip point trajectories along the y-axis across time for original (dark dotted curve), synthesized with spline smoothing (dark solid curve), and synthesized with triangular smoothing (light curve). As can be seen from the figure, both synthesis algorithms are approximating the true face point trajectories reasonably well. For example, for the /f/ phonemes in “often” (at time 0.8 sec) and “farm” (at time 2.0 sec) the synthesized lower lip moves upward following the true trajectory. For the /m/ sound in “farm” (at time 2.3 sec) the lip closure is captured as the synthesized lower lip moves upwards and upper lip moves downwards. In fact, the highest error regions correspond to non-speech sections. For speech sections, the performance is significantly better. The reason for larger errors during silence regions can be partly explained by the fact that the face position during

these sections tends to show higher variation because of varying pause length, inhaling, exhaling, lip smacks, throat clearing, etc. Since we used an automatic phonetic segmenter, all the silence regions were labeled the same. Therefore our model was not able to differentiate between any of those events in the face synthesis phase. Future studies might consider better modeling of silence intervals. From the figure it can also be observed that the spline method produces more natural and smooth trajectories when compared to triangular smoothing method. However, it results in relatively larger delays when compared to the triangular smoothing method.

INCLUDE FIGURE 6 HERE.

For the evaluations, we used the correlation coefficient between original and synthesized face point trajectories as the performance criterion. In order to make a fair judgment of the performance we used speech-only frames. The silence frames were identified and disregarded in the evaluations based on energy thresholding. In order not to disregard stop consonants a median filtering over a sufficiently long duration (112 ms) on the energy contour was applied before the energy thresholding. The selected frames are marked with dots in a straight line at the center of the lip trajectory plots in Figure 6. No dots are printed for silence frames. Since the motion of the rest of the face is highly correlated with the lips and jaw, we used upper and lower lip y-axis trajectories to obtain a reference of performance. The average correlation coefficients between face point coordinates for the original and synthesized data are shown in Table 2 both for training and test data. From the table it can be seen that despite the fact that the spline method produces more natural face point trajectories it performs slightly worse when compared to the triangular smoothing method. This result can be explained by the fact that in general the spline method produces relatively larger delays.

In order to determine the optimal number of similar context-phones (N in Equation 6) used in the restricted codebook search, we performed simulations. Figure 7 shows the correlation between synthesized (triangular smoothing) and true face point trajectories as a function of the number of similar context-phones used in the codebook. After 3 context-phones the curve levels off for held-out data. As can be expected the performance on the training data degrades as more context-phones are used.

Our future plans include the development of a more accurate global evaluation measure in terms of its correlation with human judgment.

INCLUDE TABLE 2 HERE.

INCLUDE FIGURE 7 HERE.

4 Conclusion

In this paper, a novel algorithm for face point trajectory synthesis is described. For the modeling phase, an audio-visual codebook is generated based on context-dependent phonetic labels. In addition, automatic generation of a visual phoneme similarity matrix is described. The codebook and the matrix are then used in the synthesis stage to select the most likely codebook entries for a given speech segment and phonetic label.

The performance of the algorithm is tested on held-out data, and the average correlation between synthesized and true face point trajectories was calculated to be 0.73. The correlation coefficient for training data was 0.92 which represents the current upper limit for the proposed algorithm. It is possible to move closer to this limit by increasing the training database size (i.e., by providing wider coverage of context).

The most significant contribution of this paper is the use of acoustics in synthesizing the fine detail face trajectories. The algorithm can be generalized by not restricting the codebook search using phonetic information. In that case, acoustic information alone can be used to determine the codebook weights from the complete audio-visual codebook. The performance may not be as good when compared to algorithm performance using phonetic information, since acoustically confusable phonemes (e.g., /m/ versus /n/) may create problems in the synthesized face in such a scheme. However, this capability may be useful in practical applications such as video conferencing or where language independence is a requirement.

References

- L.M. Arslan, A. McCree, and V. Viswanathan (1995). “New Methods for Adaptive Noise Suppression”. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Vol. 1, pp. 812–815, Detroit, USA.
- L.M. Arslan and D. Talkin (1997). “Voice Conversion by Codebook Mapping of Line Spectral Frequencies and Excitation Spectrum”. In *Proc. EUROSPEECH*, Vol. 3, pp. 1347–1350, Rhodes, Greece.
- E.T. Auer, L.E. Bernstein Jr., R.S. Waldstein, and P.E. Tucker (1997). “Effects of phonetic variation and the structure of the lexicon on the uniqueness of words”. In *Proceedings of the Workshop on Audio-Visual Speech Processing*, pp. 21–24, Rhodes, Greece.
- J. Bertenstam, J. Beskow, M. Blomberg, R. Carlson, K. Elenius, B. Granstrom, J. Gustafson, S. Hunnicut, J. Hogberg, R. Lindell, L. Neovius, A. de Serpa-Leitao, L. Nord, and N. Strom (1995). “The Waxholm system - a progress report”. In *Proceedings of Spoken Dialogue Systems*, Vigsø, Denmark.
- J. Beskow (1997). “Animation of Talking Agents”. In *Proceedings of the Workshop on Audio-Visual Speech Processing*, pp. 149–152, Rhodes, Greece.
- C. Bregler, Michele Covell, and Malcolm Slaney (1997). “Video Rewrite: Visual Speech Synthesis from Video”. In *Proceedings of the Workshop on Audio-Visual Speech Processing*, pp. 153–156, Rhodes, Greece.
- J. Cassel, M. Steedman, N. Badler, C. Pelachaud, M. Stone, B. Douville, S. Prevost, and B. Achorn (1994). “Modeling the interaction between speech and gesture”. In *Proceedings of the 16th Annual Conference of the Cognitive Science Society*, Georgia Institute of Technology, Atlanta, USA.
- J.R. Crosmer (1985). *Very low bit rate speech coding using the line spectrum pair transformation of the LPC coefficients*. PhD thesis, Elec. Eng., Georgia Inst. Technology.
- J. Deller and J. Proakis and J.H.L. Hansen (1993). *Discrete Time Processing of Speech Signals*. Macmillan Series for Prentice-Hall, New York.
- K. Fukunaga (1990). *Statistical Pattern Recognition*. Academic Press, San Diego, Second Edition.
- N. Katashi and T. Akikazu (1994). “Speech dialogue with facial displays”. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pp. 102–109.
- Editors: W.B. Kleijn and K.K. Paliwal (1995). *Speech Coding and Synthesis*. Elsevier, Amsterdam.
- R. Laroia, N. Phamdo, and N. Farvardin (1991). “Robust and Efficient Quantization of Speech LSP Parameters Using Structured Vector Quantizers”. In *Proc. IEEE ICASSP*, pp. 641–644.
- C. Wightman and D. Talkin (1994). *The Aligner User’s Manual*. Entropic Research Laboratory, Inc., Washington, DC.

H. Yehia, P. Rubin, and E. Vatikiotis-Bateson (1997). “Quantitative Association of Orofacial and Vocal-Tract Shapes”. In *Proceedings of the Workshop on Audio-Visual Speech Processing*, pp. 41–44, Rhodes, Greece.

H. Yehia, P. Rubin, and E. Vatikiotis-Bateson (1998) (in press). “Quantitative association of acoustic, facial, and vocal-tract shapes”. *Speech Communication*.

List of Figure Captions

Figure 1: Flow-diagram of the proposed face synthesis algorithm. The algorithm uses a digitized speech file and its corresponding phoneme sequence along with an audio-visual codebook and a visual phoneme similarity matrix to synthesize the principal components trajectory of face points.

Figure 2: The influence of modification of (a):first, (b):second principal components on face point locations.

Figure 3: The labeling and segmentation scheme used in generating codebook entries for the word “whenever”.

Figure 4: The automatically derived visual similarity matrix corresponding to phonemes in American English.

Figure 5: Flow-diagram of the proposed face feature trajectory smoothing algorithm.

Figure 6: The comparison of original and synthesized face point trajectories for held-out data. *Top plot:* speech signal corresponding to “Tornadoes often destroy acres of farm land”. *Middle plot:* center upper lip y-axis trajectories - Dark dotted curve: original, Dark solid curve: synthesized with spline smoothing, Light curve: synthesized with triangular smoothing, Dotted line in the center: speech/silence detection. *Bottom plot:* center lower lip y-axis trajectories - Dark dotted curve: original, Dark solid curve: synthesized with spline smoothing, Light curve: synthesized with triangular smoothing, Dotted line in the center: speech/silence detection.

Figure 7: The influence of the number of similar context phones N incorporated in the codebook on the performance of the proposed algorithm.

List of Table Captions

Table 1: Top three visually most similar phonemes and their similarity measures of phonemes in American English based on the proposed visual similarity metric.

Table 2: Average correlation between original and synthetic face point trajectories (using spline and triangular smoothing methods) during speech-only sections using top 3 visually most similar context-phones.

List of Footnotes

1: If the number of samples in the context-phone is less than 5, then the closest sample to each of the 5 locations is used. Since the sampling frequency of audio and visual feature vectors were determined to be the same (60 Hz), the synchronization between audio and visual data was not an issue.

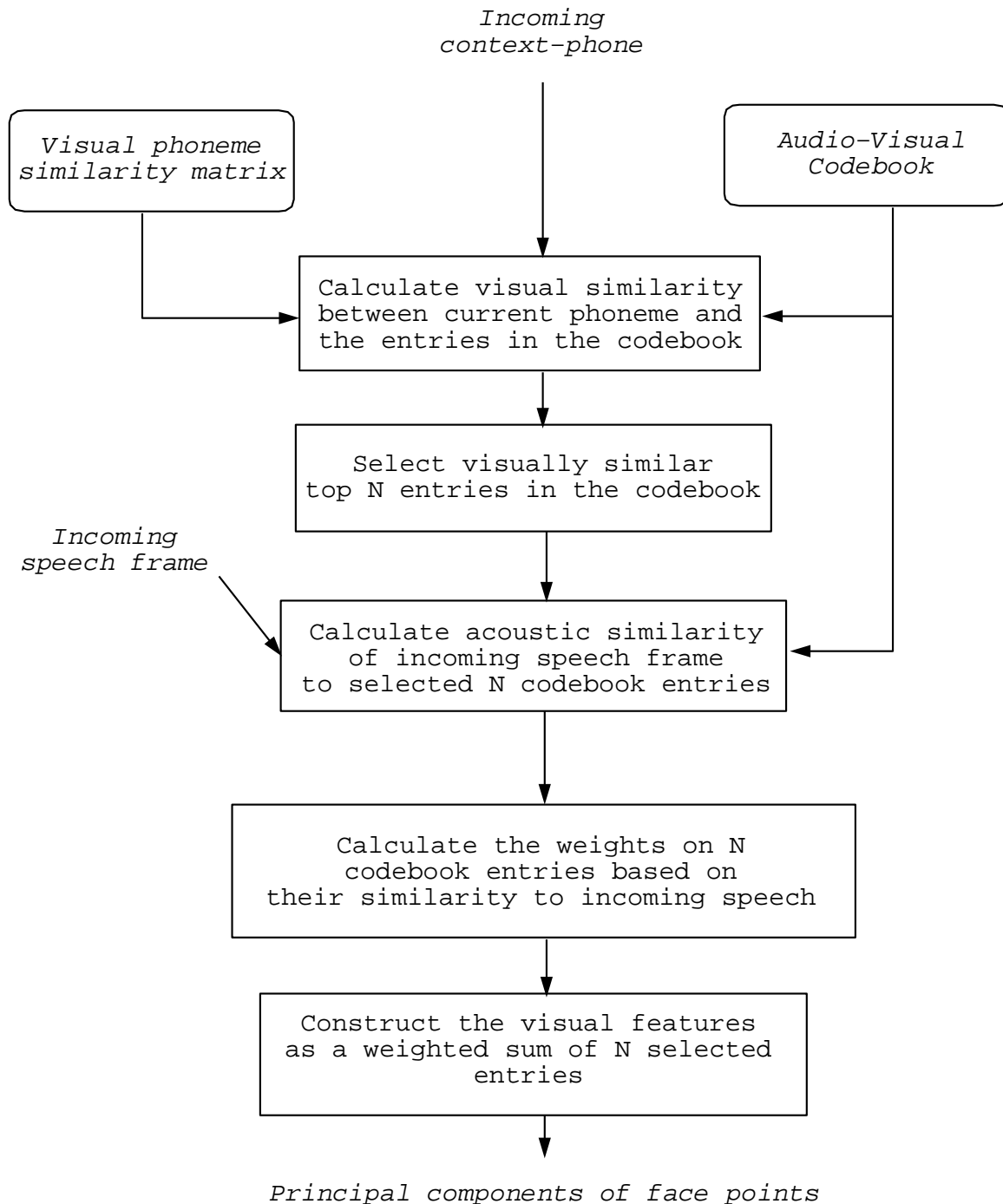


Figure 1: Flow-diagram of the proposed face synthesis algorithm. The algorithm uses a digitized speech file and its corresponding phoneme sequence along with an audio-visual codebook and a visual phoneme similarity matrix to synthesize the principal components trajectory of face points.

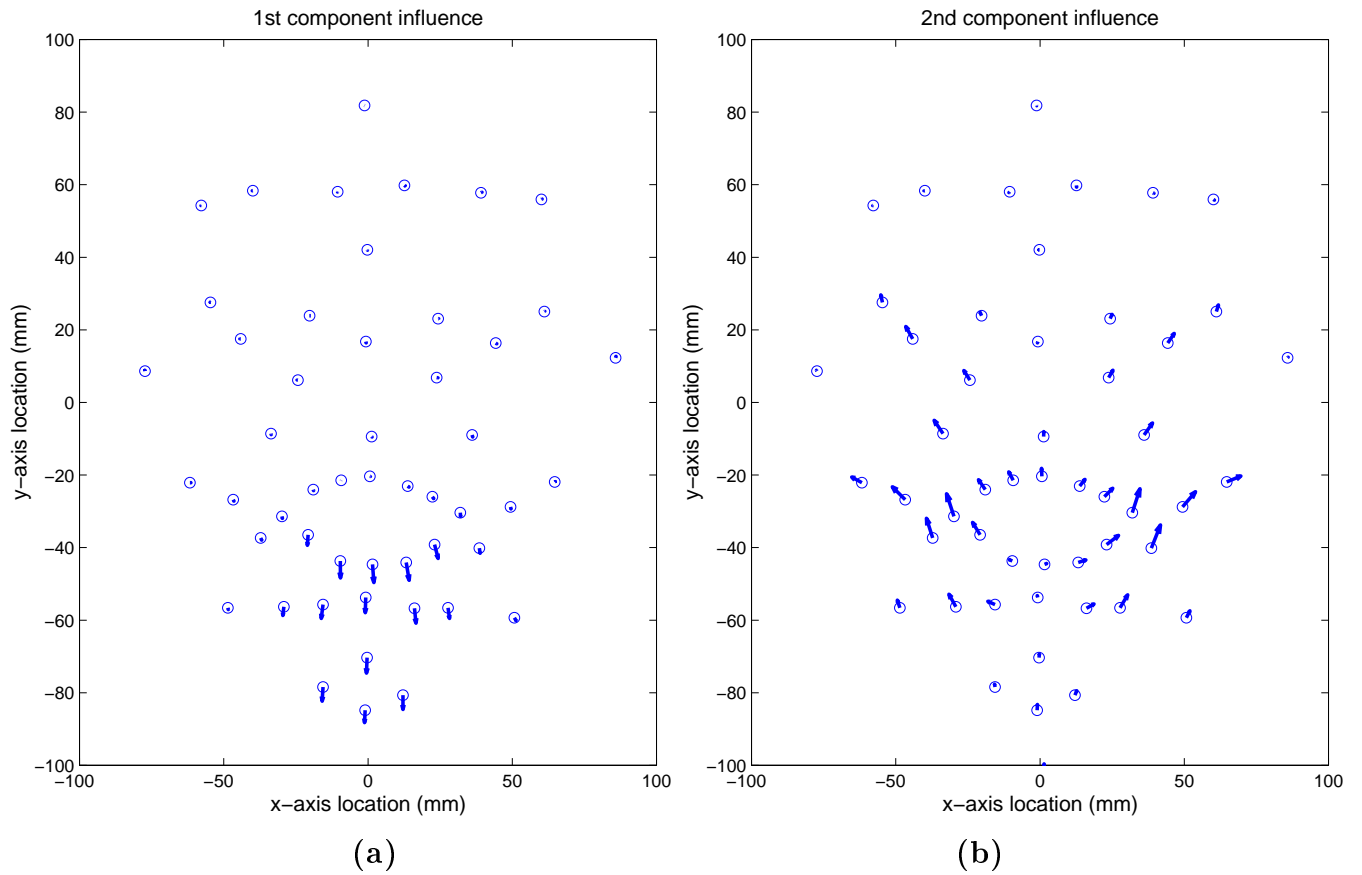


Figure 2: The influence of modification of (a):first, (b):second principal components on face point locations.

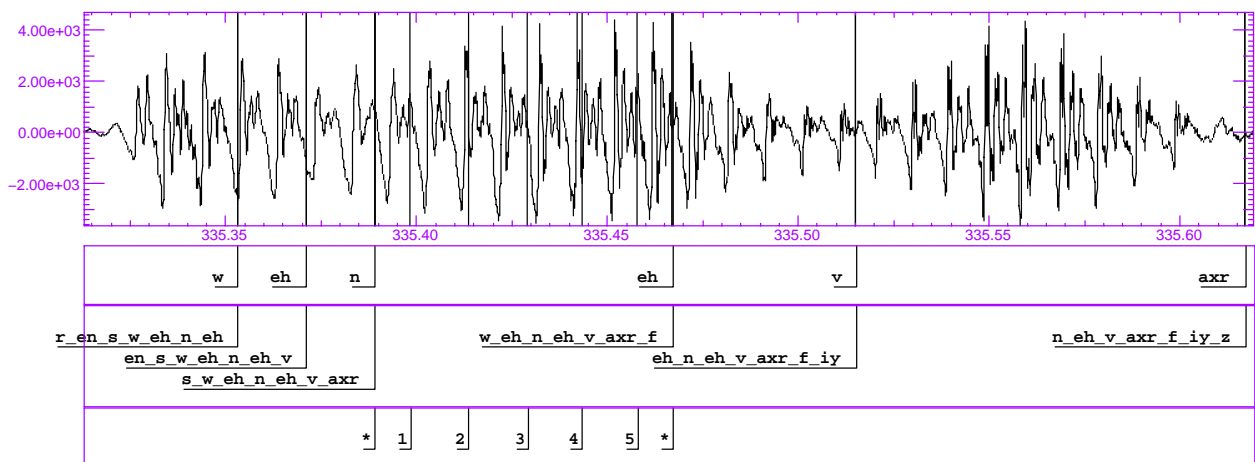


Figure 3: The labeling and segmentation scheme used in generating codebook entries for the word “whenever”.

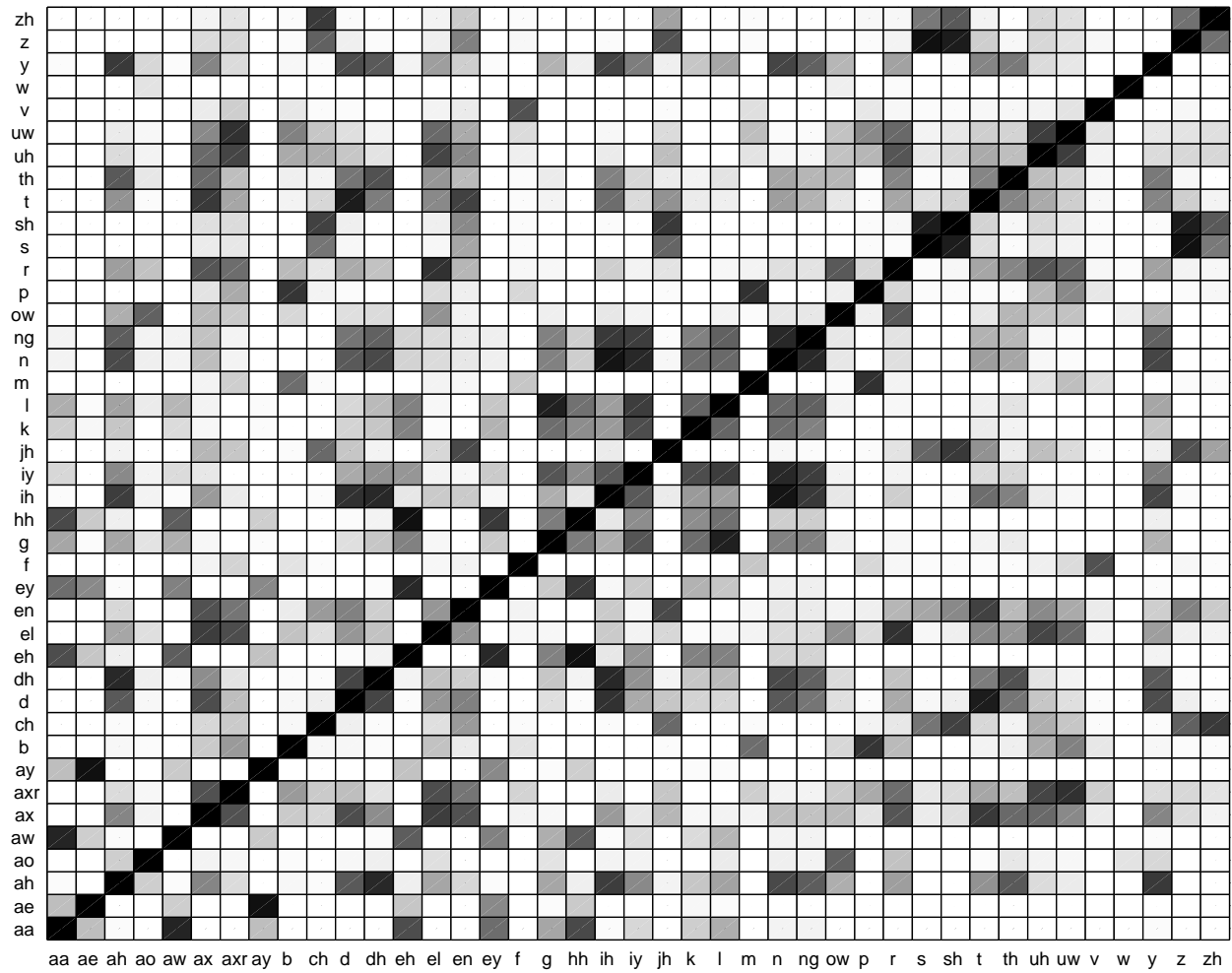


Figure 4: The automatically derived visual similarity matrix corresponding to phonemes in American English.

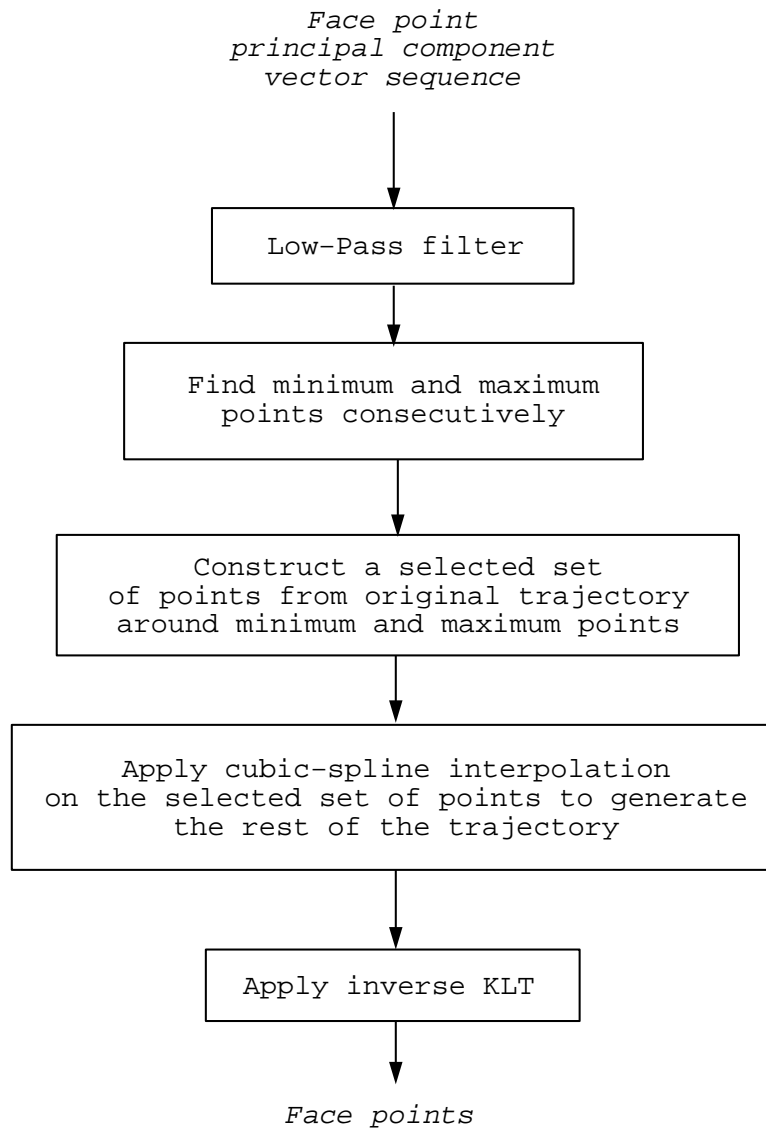


Figure 5: Flow-diagram of the proposed face feature trajectory smoothing algorithm.

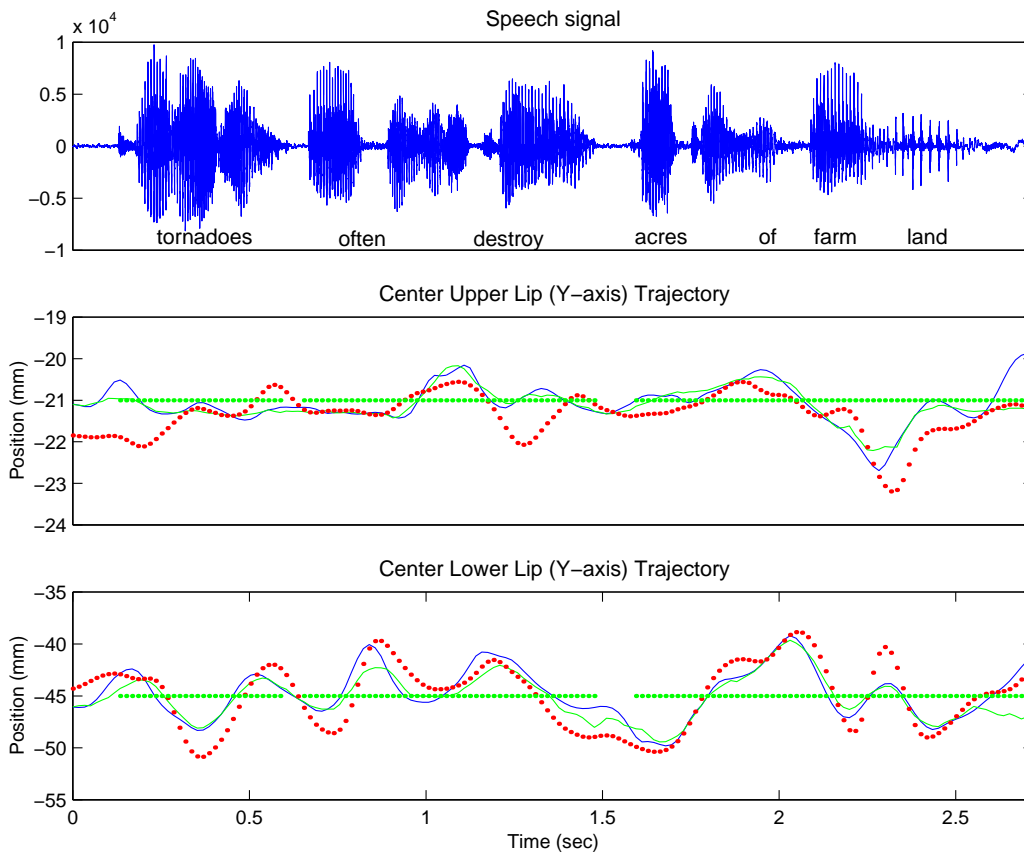


Figure 6: The comparison of original and synthesized face point trajectories for held-out data. *Top plot:* speech signal corresponding to “Tornadoes often destroy acres of farm land”. *Middle plot:* center upper lip y-axis trajectories Dark dotted curve: original, Dark solid curve: synthesized with spline smoothing, Light curve: synthesized with triangular smoothing, Dotted line in the center: speech/silence detection. *Bottom plot:* center lower lip y-axis trajectories Dark dotted curve: original, Dark solid curve: synthesized with spline smoothing, Light curve: synthesized with triangular smoothing, Dotted line in the center: speech/silence detection.

Top Three Visually Most Similar Phonemes						
<i>Phoneme</i>	<i>Closest phoneme</i>		<i>2nd Closest phoneme</i>		<i>3rd Closest phoneme</i>	
aa	aw	0.86	hh	0.71	eh	0.70
ae	ay	0.94	ey	0.46	aa	0.26
ah	dh	0.83	y	0.77	ih	0.75
ao	ow	0.62	r	0.24	ah	0.20
aw	aa	0.86	hh	0.64	eh	0.64
ax	t	0.78	el	0.76	d	0.69
axr	uw	0.81	uh	0.72	el	0.69
ay	ae	0.94	ey	0.47	aa	0.26
b	p	0.78	m	0.58	uw	0.49
ch	zh	0.77	sh	0.74	z	0.61
d	t	0.88	ih	0.81	dh	0.73
dh	ih	0.84	ah	0.83	d	0.73
eh	hh	0.94	ey	0.84	aa	0.70
el	r	0.81	ax	0.76	uh	0.73
en	t	0.75	jh	0.71	ax	0.68
ey	eh	0.84	hh	0.77	aa	0.57
f	v	0.68	m	0.23	axr	0.18
g	l	0.86	iy	0.67	k	0.57
hh	eh	0.94	ey	0.77	aa	0.71
ih	n	0.92	dh	0.84	d	0.81
iy	n	0.83	ng	0.76	l	0.75
jh	sh	0.77	en	0.71	z	0.68
k	iy	0.70	l	0.60	g	0.57
l	g	0.86	iy	0.75	ng	0.62
m	p	0.81	b	0.58	uw	0.26
n	ih	0.92	ng	0.84	iy	0.83
ng	n	0.84	ih	0.77	iy	0.76
ow	r	0.65	ao	0.62	el	0.43
p	m	0.81	b	0.78	uw	0.46
r	el	0.81	uh	0.67	ax	0.67
s	z	0.93	sh	0.88	jh	0.60
sh	z	0.89	s	0.88	jh	0.77
t	d	0.88	ax	0.78	en	0.75
th	dh	0.68	ah	0.65	ax	0.59
uh	uw	0.76	el	0.73	axr	0.72
uw	axr	0.81	uh	0.76	el	0.59
v	f	0.68	axr	0.20	m	0.14
w	ao	0.12	ow	0.07	r	0.02
y	ah	0.77	ih	0.73	n	0.72
z	s	0.93	sh	0.89	jh	0.68
zh	ch	0.77	sh	0.64	z	0.56

Table 1: Top three visually most similar phonemes and their similarity measures of phonemes in American English based on the proposed visual similarity metric.

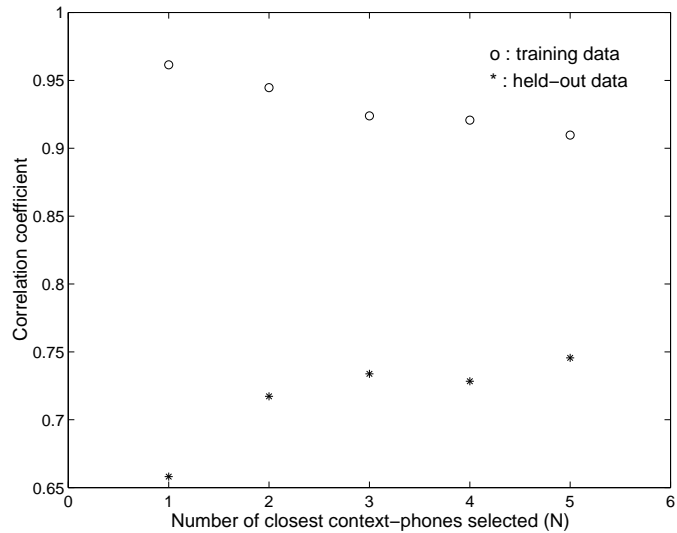


Figure 7: The influence of the number of similar context phones N incorporated in the codebook on the performance of the proposed algorithm.

Face Synthesis Algorithm Performance Evaluation			
<i>Test condition</i>	Whole Face	Lower Lip	Upper Lip
<i>Training Data (Triangular)</i>	0.9239	0.9463	0.9287
<i>Training Data (Spline)</i>	0.9145	0.9388	0.9126
<i>Test Data (Triangular)</i>	0.7338	0.8468	0.7230
<i>Test Data (Spline)</i>	0.6969	0.7721	0.7060

Table 2: Average correlation between original and synthetic face point trajectories (using spline and triangular smoothing methods) during speech-only sections using top 3 visually most similar context-phones.