A Nonlinear Viseme Model for Triphone-Based Speech Synthesis

Robert Bargmann Volker Blanz
Hans-Peter Seidel

MPI–I–2007–4-003 June 2007

Authors' Addresses

Robert Bargmann
Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85
66123 Saarbrücken
Germany

Volker Blanz
Universität Siegen
Hölderlinstr. 3
57068 Siegen
Germany

Hans-Peter Seidel
Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85
66123 Saarbrücken
Germany

Abstract

This paper presents a representation of visemes that defines a measure of similarity between different visemes, and a system of viseme categories. The representation is derived from a statistical data analysis of feature points on 3D scans, using Locally Linear Embedding (LLE). The similarity measure determines which available viseme and triphones to use to synthesize 3D face animation for a novel audio file. From a corpus of dynamic recorded 3D mouth articulation data, our system is able to find the best suited sequence of triphones over which to interpolate while reusing the coarticulation information to obtain correct mouth movements over time. Due to the similarity measure, the system can deal with relatively small triphone databases and find the most appropriate candidates. With the selected sequence of database triphones, we can finally morph along the successive triphones to produce the final articulation animation.

In an entirely data-driven approach, our automated procedure for defining viseme categories reproduces the groups of related visemes that are defined in the phonetics literature.

Keywords

Facial animation, 3D dynamic recording, Dimension reduction, Speech Synthesis, Triphone concatenation.

Contents

1 Introduction

In phonetics, the audio signal of human speech is segmented into phonemes according to the frequency spectra and other acoustic properties. Automated text-to-speech systems map each letter, syllable or word into sequences of phonemes, which are concatenated to a synthetic audio stream. The mapping from letters to phonemes is not unique, but depends on the letters that precede and follow. Moreover, the sound of phonemes depends on the preceding and following phonemes, an effect that is known as coarticulation [8].

Visual text-to-speech systems have defined the viseme as the visual counterpart of a phoneme. However, the visual appearance of visemes varies at least as much as the acoustic signal of a phoneme, and there is no one-to-one mapping between phonemes and visemes. Therefore, a given phoneme occurrence in an audio file can correspond to many viseme candidates, but these visemes might as well be candidates for other phonemes.

In this work, we address the problem of selecting appropriate visemes for given phonemes in a new utterance, and go beyond the widely used standard mapping between visemes and phonemes. Our approach is based on a similarity criterion for visemes, and on a triphone concatenation framework [4] that breaks the speech down into overlapping triplets of subsequent phonemes or visemes. We study 3D articulation data recorded from a real talking person to automatically build a similarity graph for visemes and a triphone database.

Our statistical analysis depends critically on the type of representation of the training data. We use Locally Linear Embedding (LLE) as a non-linear dimension reduction for the mouth-shape space that will give an intuitive representation of the data. In this representation we study the distribution of different viseme clusters according to the phoneme they correspond to in the original recorded sequences. By analyzing these distributions, we are able to find out which clusters are included in other ones and can therefore be used to replace missing data by what is available in the database.

The main idea in this inclusion criterion is that the algorithm tries to avoid unnatural movements, by substituting unavailable visemes only by candidate visemes that come from smaller clusters. However, in special cases when no substitution can be found, the system is forced to look in broader clusters, thus introducing a risk of selecting uncompatible visemes. This risk is measured by a cost function which we minimize when generating animations.

When we synthesize a new animation, the algorithm finds in the recorded database a sequence of triphones that match best to a new audio file. As we do not have all possible triphones in our database, due to the high number of possible combinations of visemes in triphones, we have to find the closest one that fits our needs, based on the replacement rule described above.

We perform the statistical analysis on the 3D positions of a set of marker points. These marker points are identified automatically in a time sequence of 3D scans of a talking face. For the animation of a new utterance, we first select triphones from the database in the required order, base on our replacement rules. These triphones consist of individual frames, some of which are interpolated in the transition phases with those of the previous or next triphone. For each frame, we have a textured 3D scan, and we can interpolate between them within a morphable model framework [1] to create the final 3D animation sequence. For data compression, we perform a PCA on the shape vectors of the 3D scans. Each frame is only stored in terms of the PCA coefficients of the 50 most important components, and these are used to reconstruct the full shape vectors before rendering. Our interpolation scheme retains most of the velocity and the acceleration of the lips from the original triphone instance.

The new contributions of this paper are

- An LLE-based approach for the analysis and representation of viseme distributions,
- A similarity measure for visemes,
- A categorization of visemes that reflects many findings known from phonetics,
- A criterion for substitution of triphones for animation.

All of these elements are integrated in a speech synthesis system.

2 Related Work

This chapter gives a summary of the different approaches for simulating the dynamics of human speech and facial expressions. While performance-driven animation reproduces the motion that is captured from an actor, many applications require an automated system that can be combined with a text-to-speech system to generate animation without any user interaction. The most popular and widely used animation technique is keyframe animation, which has been adopted by a number of systems and commercial production tools. In keyframe animation, the frames in between are filled by blending or morphing the previous and the subsequent keyframes. This may be done using linear interpolation, or with smooth acceleration.

In their model of coarticulation, [8] go beyond keyframe interpolation in order to model the interaction between subsequent phonemes. They define dominance functions that control how the influence of each phoneme slowly increases before the phoneme is actually heard, and how it decreases slowly afterwards. Due to the overlap of dominance functions, phonemes interact and produce a smooth motion sequence.

In a completely different approach, Video Rewrite [4] concatenates short pieces of prerecorded video footage to generate new utterances. The pieces are grouped into triphones, which are triplets of subsequent phonemes, and they are stored in a database. For a new utterance, the algorithm selects triphones and connects them in an overlapping way. Triphones are selected according to two criteria: one is the similarity of the triphone in the database to the triphone required for the animation, and the second term controls smoothness by minimizing the difference between the connected, overlapping ends of the triphones. We extend this approach by presenting a data driven rule for triphone selection by a general statistical analysis of visemes and an application to 3D faces.

Voice Puppetry [3] applies a Hidden Markov Model to learn the dynamics of speech from audio, and transfer this information to a face model.

Ezzat et al [10] have proposed a method that uses regularization for generating a smooth animation. Each viseme is represented by a Gaussian distribution in a high-dimensional vector space based on principal component analysis (PCA). The distribution is centered around the average appearance of the viseme, and varies according to the variation found in a database of samples. For finding the motion trajectory of a new utterance, a regression algorithm computes a smooth curve that passes as close as possible by the centers of the visemes, relative to the variance of the viseme cluster, and is at the same time as smooth as possible. In a refinement step of the training of the system, the viseme clusters are shifted and deformed in order to obtain trajectories that are as close as possible to those in a training set.

Cao et al [5] propose an Anime-Graph, which combines visual and speech information in a single data structure, and use a greedy search algorithm for generating new utterance. Unlike the static visemes, animes capture the entire motion during a phoneme.

Kshirsagar and Magnenat-Thalmann [12] propose *visyllables* as visual elements of speech, instead of visemes or triphones, and they present a system that learns visyllables from marker point data, transfers them to Facial Movement Parameters, and uses these for animation. In speech synthesis, visyllables are stitched together, and boundary mismatches are corrected. Ma et al [13] segment a large corpus of marker data into syllables and concatenate them for new utterances using a Viterbi search algorithm.

Deng and Neumann [9] have analyzed motion capture data with Isomap, a statistical method is that closely related to LLE. They use this for an interactive user interface where the user can explore a database of phoneme measurements, and select motion frames. Dynamic programming is used for building a motion-sequence from the database that reflects the user's selection and that has a predefined emotion and smooth transitions.

In a physics-based simulation, Sifakis et al [15] map marker point positions back to muscle activations, and then decompose a given speech sequences into physical primitives called *physemes*. In speech synthesis, the effects of coarticulation are produced by the physical simulation.

Cao et al [6] separate the parameters for emotion and for speech content in animation sequences using independent component analysis (ICA), and their algorithm allows users to edit these independently. Wang et al [16] use a combination of Locally Linear Embedding (LLE) and a bilinear model within in the LLE manifold to separate style and emotion in sequences of 3D scans. Chuang and Bregler [7] use a bilinear model to statistically factorize and control speech and emotion in image-based motion capture data.

3 Capturing 3D Motion Data

Our corpus of 3D data of a talking face was recorded with a structured-light dynamic 3D scanner at a frame rate of 40 3D-scans per second. Our corpus contained a total of 1933 frames. Data was preprocessed and then brought into correspondence based on an optical flow algorithm similar to Blanz and Vetter [2]. As a result, each 3D frame was represented as a shape vector $\mathbf{S}$ in a morphable model that consists of a collection of all x, y and z coordinates of all n points of the model, combined into a single $3n$-dimensional vector. In a further step we performed a Principal Component Analysis (PCA) on the set of every tenth shape vector. Using all vectors would have been computationally expensive, and the data would have been likely to be highly redundant. The PCA is used for data compression of all frames, allowing us to store only a set of model coefficients for each frame. To reduce the influence of eye movements and edge artifacts, we performed the PCA on reduced vectors that contain only the 3D coordinates of the vertices of the mouth region. The eigenvectors can be mapped back to the entire $3n$-dimensional space based on the fact that each eigenvector is a linear combination of the reduced sample vectors, and the same linear combination can be formed with the original sample vectors. In our system, we used 50 coefficients for the first, most relevant principal components.

In the next chapters, we will use the following notation: phonemes are written with slashes, such as /AH/, /B/ or /K/, and visemes are written with vertical bars, —AH—, —B— or —K—. Triphones, a sequence of three consecutive phonemes or visemes, are written with angle brackets, for instance <AH R V>. /SIL/ corresponds to the “silence phoneme” and —PIT— is the neutral viseme, when the mouth is slightly open and on rest.

4 Data Representation

For the exploration and representation of motion data, we investigated two different statistical methods, PCA and LLE. The PCA performs a linear reduction where the articulation curves can be visualized nicely (in the first 3 dimensions, see Figure 4.1). But the distributions of the different visemes are spread out very far, and the representation did not recover any intuitive structures of our data.

A similar result as with the PCA coefficients of the shape vectors was obtained with the concatenated 3D coordinates of 12 vertices that we selected from the model. These vertices correspond to marker points that we painted on the face around the lips. They were tracked by the optical flow algorithm more precisely than many of the points on unstructured face regions or on the boundaries of the scans.

Please note that the LLE reduction is only used for determining similarities among visemes and by extension to triphones. All processes related to the animation are performed in the PCA space.

Locally Linear Embedding

On the marker point vectors $\mathbf{v} = (x_1, y_1, z_1, ..., z_{12}) \in R^{36}$, we performed a Locally Linear Embedding (LLE) [14] to transform the data in a non-linear way into a low dimensional manifold. This representation shows an intuitive separation of the data, and visemes become easier to classify. In our case, a constant neighborhood of four samples and a target dimensionality of four gave us the expected representation. The price for this non-linear separation is that new, intermediate samples between the original data points are more difficult to generate. Exploiting the locally linear nature of the representation, this is achieved by finding the nearest neighbors in the database and interpolating between these in a linear way. It is straight forward then to map the vectors $\mathbf{v}$ back to the shape vectors $\mathbf{S}$ of the entire face, due to the

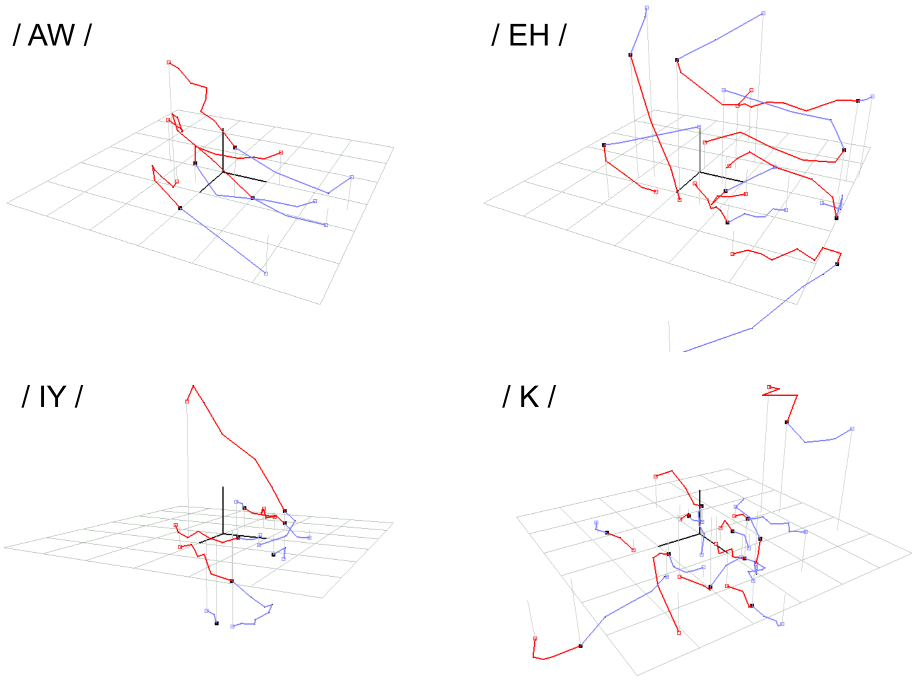

Figure 4.1: The representation in the PCA space of the recorded transition curves around different phonemes. The first frame of the occurrence of a phoneme is marked as a black point. The blue and the red segments are the articulation path to the preceding and the succeeding phoneme respectively. Refer to Chapter 4 for more detailed discussions on the representation of articulation in the PCA space.

fact that we always use linear combinations of individual frames k, so we can transfer the coefficients c_k from one vector space to the other:

$$\mathbf{v} = \sum_k c_k \cdot \mathbf{v}_k, \qquad \mathbf{S} = \sum_k c_k \cdot \mathbf{S}_k. \tag{4.1}$$

An important characteristic of our data, is that because the samples were recorded dynamically, the distribution of the data is not homogeneous; the reason is that the samples follow the articulation curves and mostly, for every sample, its closest neighbors are lying on the same curve. This correlation among the neighbors has a strong impact on the LLE computation. LLE constructs a representation that is based on the spatial relation among the samples, and to get a better representation of the underlying manifold inside the data, we must first select a subgroup of samples that distribute evenly in the original space.To achieve this, we measure the distance between any pair of samples (Euclidean distance in the reduced space of the marker points). By running through all the samples successively, we check whether the sample is closer than a given threshold k to another sample; in this case, the sample is removed from the database (the frame label is mapped to the close sample), otherwise we keep it. With increasing k, neighbors lying of a common curve are slowly removed, until the threshold reaches the average distance that separates different curves, and samples start to disappear much more quickly. If we draw the number of resulting samples in function of k, we notice that when k reaches this critical distance, the function curve decreases rapidly (see Figure 4.2). In our case, we took $k = 2,5$. This procedure makes sure that LLE is not dominated by the individual curves of closely adjacent vectors, but by the intrinsic structure of the set of different mouth shapes in general.

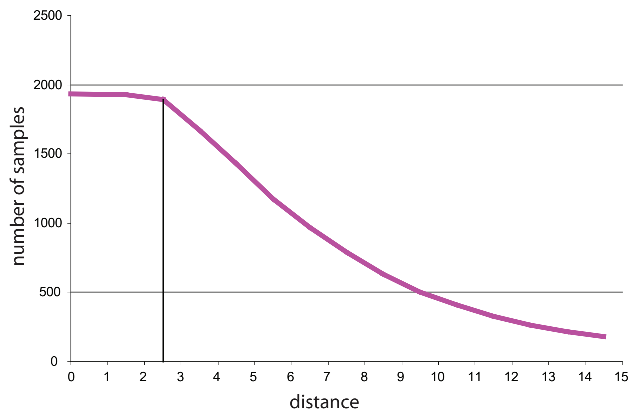

Figure 4.2: To get a more homogeneous distribution of the data, we remove samples that lie too close to the surrounding samples according to a distance threshold k. The number of remaining samples drops steeply when the threshold reaches the critical distance that separates the articulation curves, which is around $k = 2.5$.

From LLE, we obtain the star shaped representation shown in Figure 4.3; the central region of the star holds neutral mouth configurations (slightly open, released muscles), the three direction go respectively towards mouth shapes that are typical for —IY—, {—O—, —UH—} and —AH—. Our algorithm will define phoneme clusters in the LLE space along these three main directions.

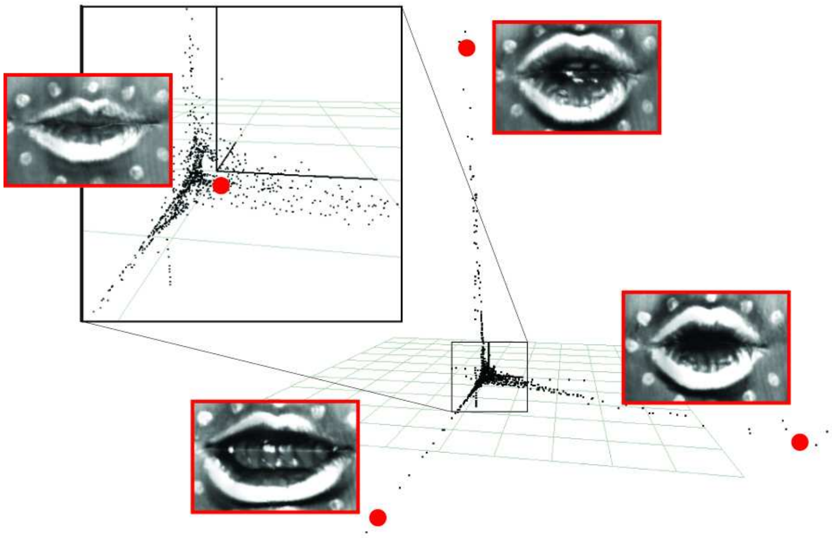

Figure 4.3: A Locally Linear Embedding (LLE) over the recorded data shows an intuitive and intrinsic low dimensional representation. The central region of the star-shaped manifold holds neutral mouth configurations, the three directions correspond to mouth shapes that are typical for —IY—, —O—, and —AH—.

5 A Measure for the Similarity of Visemes

The audio from the recorded corpus is decomposed into a sequence of phonemes using the CMU-SPHINX algorithm [11], which gives us the time interval when each of the phonemes of the utterance is heard, and thus a short sequence of a few frames. We take the first frame of the sequence as a viseme sample, and investigate the distribution of these samples using LLE.

The remaining frames of each phoneme sequence capture the transitions, and will be used later to synthesize new sentences with realistic transition velocities and accelerations. In Chapter 7 we describe how we reconstruct articulations by a concatenation of stored triphones that we morph together. Given the large number of possible phoneme combinations in triphones, it is inevitable to be able to identify similar triphones if an exact match for a desired output is not in the database. We propose to do this based on a LLE-based similarity measure of visemes, and a viseme categorization that defines a substitution rule. By combining valid candidates, we look for available triphones whose visemes are close enough to the desired triphone and that can be used as a substitution. Our algorithm gives a data-driven distance metric between visemes and is extended to distances between triphones.

In the original recorded corpus, all first frames of phoneme sequences are mapped to the LLE representation where they form viseme clusters. Figure 5.2 illustrates several viseme clusters. The different clusters vary not only in the distance they spread along the branches of the star-shaped overall manifold, but also cover only specific branches. We use these two criteria to distinguish between the clusters and to measure their similarities.

The LLE representation can be modelled by a star-shaped manifold composed of three segments (see Figure 5.1). The spread of the clusters along each segment is measured after a chosen quantization; in our case they range from 0 to 3 (we used a scale from 0 to 2 in Figure 5.1). By taking the segments in a fixed order, we are able to describe the distribution of a cluster by

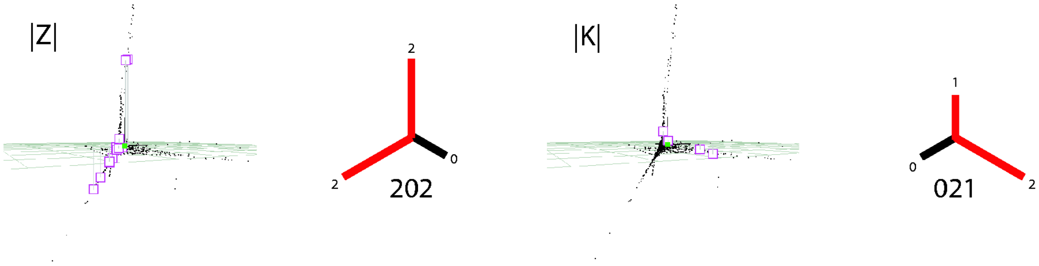

Figure 5.1: Each viseme cluster defines a region on the LLE representation. This distribution is modeled by a three directional star-shaped manifold that describes (on a scale from 0 to 2 in the example above) on each segment, how much the cluster covers this region. By taking the segments in a specific order, we are able to describe the distribution of a cluster by a simple triplet ([202] and [021] in this case).

a triplet that describes how far the cluster extends in each branch, in a fixed order. For example, [202] means that the cluster of this viseme spreads to a distance 2 on the first branch, 0 on the second and 2 on the third branch of the manifold.

This modeling is easily implemented. A viseme from the center of the representation and the three from the extremities of the branches are selected. We compute the projections of the visemes from a cluster onto the axes defined by these branches. The position of the most distant projection along the segments from the center give the measure on the quantization.

In Table 5.1 we give the conversion to our triplet notation for each of the visemes. The categorization in this table is an intrinsic statistical property to the observed shapes of the mouth

PIT	000	P, D, DH, DX, F, G, K, AE, EH, R	111
IX	001	V, CH	112
AXR, AW,	010	AH, UW	113
TH	011	B, AA, W, L	121
OW	021	HH	130
ZH, AO, ER, EY, UH, NG	100	AX	131
Z, AY, OY, Y	101	IY	210
N	102	T	211
JH	103	S	312
M, SH, IH	110	SIL	333

Table 5.1: All phonemes are mapped to a triplet representation with a quantization from 0 to 3. We added the neutral head (a slightly open mouth) as a generalization to all phonemes as PIT with the triplet 000.

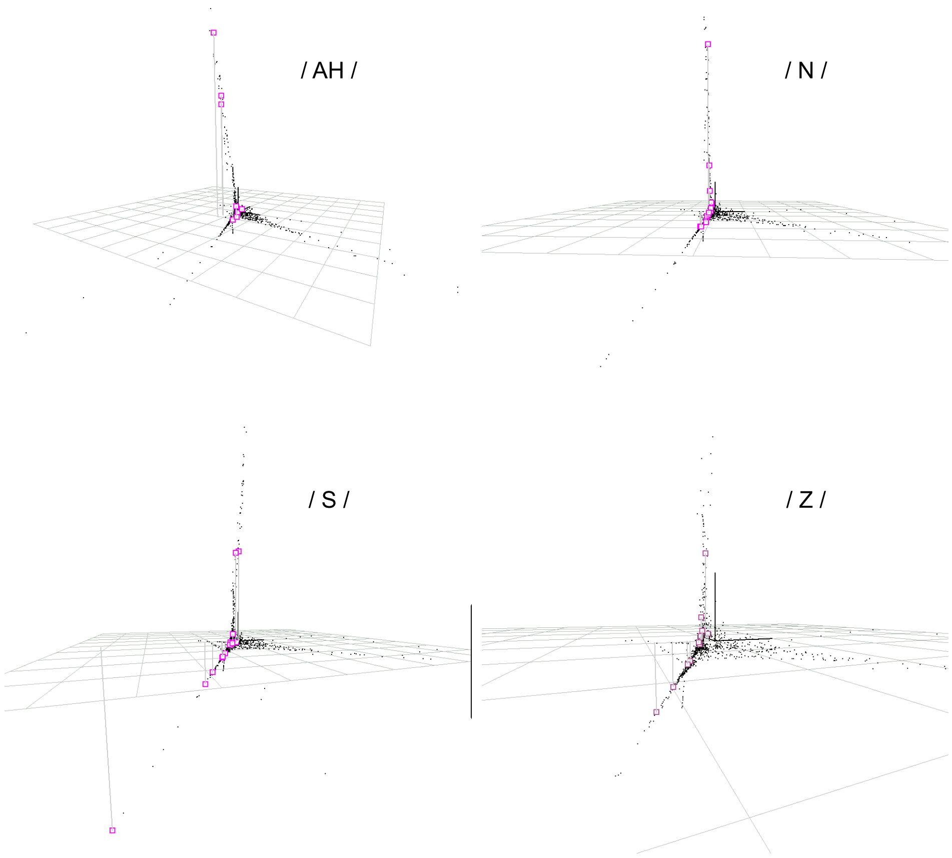

Figure 5.2: The representation in the LLE space of the recorded viseme candidates for the different phonemes. The black points show all the recorded frames from our corpus, the red squares illustrate the distribution of the different viseme clusters. Chapter 5 discusses the distributions of these clusters on the LLE representation and how they allow an analysis of the similarities among visemes.

6 Substitution Rules for Visemes

We can now formulate a rule for replacing visemes by others when we select triphone candidates. Consider a cluster $|B|$ that is included entirely in a larger cluster $|A|$. That means that all samples in $|B|$ could also be considered an instance of viseme $|A|$, but not vice versa. In the animation, we want to avoid unnatural movements, and rather be conservative in the replacement of visemes. Therefore, in our example, an instance of $|B|$ (the smaller cluster) can replace an instance of $|A|$. In the triplet notation, this applies if all digits of the triplet $[B]$ are smaller or equal to their corresponding digits in the triplet $[A]$.

With this simple rule, we can build a graph of allowed substitutions which is used when we look for best matching triphones (Figure 6.1). The nodes hold the triplets values and the edges follow the substitution rule. Following the edges of the graph, all nodes that we cross will be valid candidates. At every jump, at least one of the digits decreases until we reach the triplet 000 (—PIT—). This triplet is a non-viseme cluster that doesn't spread. If we reach PIT without having found a fitting triplet, we need to traverse the graph in the other direction from the point where we started. In this scenario, the cluster is expanded to a cluster that is larger (because at least one of the digits increases), so the viseme candidates are less likely to fit in the original cluster. Therefore, to minimize the extensions in that direction, we define a cost, or a distance metric. Every time a digit increases by one, the cost increases by one. For instance, if we have to go from [121] to [322], the cost is 3. On the other hand, the cost is 0 in the other direction as [322] can be extended with [121] (all digits greater or equal). Note however that in this way, not any pair of triplets are connectable, for instance 202 and 021 (in Figure 5.1); this ensures that we will not get a viseme substitution between two cluster whose distributions have nothing in common. Figure 6.2 shows the two relations between the two connected clusters [121] and [111].

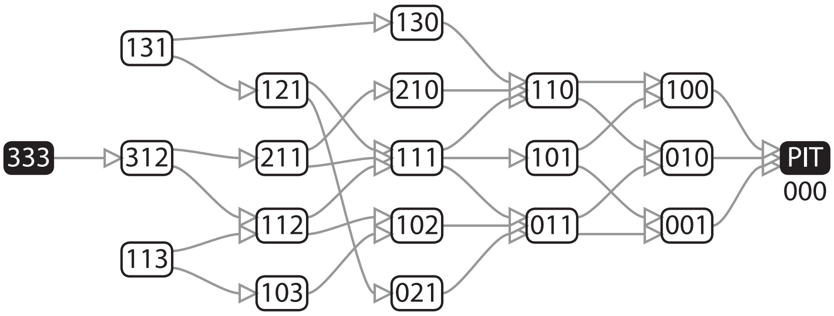

Figure 6.1: The acyclic directed graph defines the valid visemes candidates for substitutions in a triphone that has not been recorded. The graph shows only the first order connections. If no suitable viseme is found, the edges can be taken backwards, but a substitution cost is induced.

The graph will now allow us to find substitutes to triphones that are not available. For instance, suppose we look for the closest triphone to $< |AA|, |P|, |UH| >$. The triphone is first converted in to the triplet notation: $< [121], [111], [100] >$. We can then compare it to all available triphones and compute the distance between all three corresponding clusters of the two triplets when they are connected in the graph. In the total distance, we put more weight on the central viseme than on the two others.

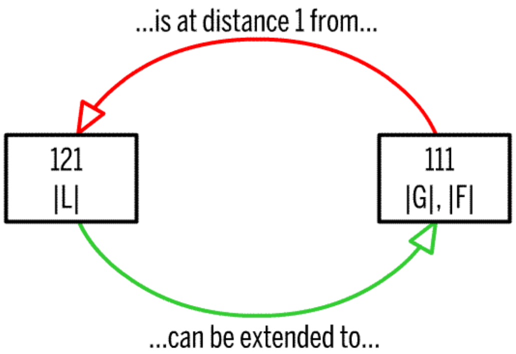

Figure 6.2: By comparing triplets, [121] and [111] in this example, we can tell which cluster can substitute another. This means that we can use the samples from [111] as (conservative) substitutes for [121]. Taken in the reverse direction, the relation gives a distance metric that will define the penalty if, when selecting triphones from the database, we don't find appropriate candidates and have to replace visemes in the opposite direction.

7 Sentence Generation

In the previous chapter, we already outlined our approach to articulation synthesis. The given audio file of a new sentence is decomposed into a sequence of phonemes using the CMU-SPHINX algorithm [11]. In order to account for coarticulation and to generate the most natural transition between the visemes, the visemes of the sequence are transformed into overlapping target triphones. The phonemes in each triphone are mapped to triplets (Table 5.1) and the algorithm searches for the available triphone candidates in the database that could match a given target. For every target triphone, the system generates a list of all possible candidates by verifying that every pair of visemes inside the triphones are connected in the graph and that the candidates are available. To each of the candidate in the lists, a cost is attributed as the sum of the viseme distance costs of each corresponding pair. We call this cost the *local cost* for a substitution triphone candidate.

Because triphones were sampled from different portions of the sequence of training scans, they will not describe a continuous curve and may be far apart in terms of face shape. Therefore, we smoothly blend between triphones while they overlap temporally, using 3D morphing. If two successive triphones lie far apart, the transition will most likely look abrupt. For this reason, it is necessary to collect a list of all possible triphone candidates in the database, and then compute the sequence of triphones that has a minimal morphing distance.

The next section presents how two triphones are morphed, and how the morphing cost is defined. To find the best suited triphone sequence, the sum of all local costs and morphing costs is minimized (Chapter 7.2).

7.1 Concatenation of Triphones

A triphone is composed of a central visemes $|A|$ and two neighbors $|B|$ and $|C|$. The triphone curve holds the position of the three visemes plus the frames for the transitions from $|B|$ to $|A|$ and from $|A|$ to $|C|$. The number of transition frames varies with the different triphones. In the triphone morphing algorithm, it is important that the central viseme remains unaltered. The same holds for the tangent of the motion curve at that point.

Let us assume that the triphone $< B_1A_1C_1 >$ has to be morphed with the next triphone $< B_2A_2C_2 >$, resulting in a sequence $< B_1A_1A_2C_2 >$. Let us also assume for a moment that for each phoneme occurrence we have found a correct viseme candidate. The role of the concatenation process consists in morphing the individual segments. In the example above, the transition from $|A_1|$ to $|A_2|$ has to be computed knowing that

- the number of available frames between $|A_1|$ and $|C_1|$ is not the same as between $|B_2|$ and $|A_2|$;
- the number of frames to be generated for the new animation is dictated by the phoneme sequence in the target audio file;
- variations in the speed of articulation between the two transitions have to be taken into account;
- the tangent of the curve at the central visemes should remain unchanged.

For each frame t in a candidate triphone, we have initially transformed the original shape vector $\mathbf{S}$ into 50 PCA coefficients $\mathbf{a} = (a_1, a_2, ..., a_{50})^T$, so we have a curved trajectory $\mathbf{a}(t)$ in coefficient space that we could map back to shape vectors $\mathbf{S}$ and render on the screen for reproducing the original motion. We use a piecewise linear function to interpolate between the discrete time steps of the original frames. The argument $t \in [t_0, t_1]$ of this function can be substituted by a variable $u \in [0, 1]$, $u = \frac{t-t_0}{t_1-t_0}$ to obtain a time-normalized function $\mathbf{a}(u)$. If the lengths of the transition phases in two subsequent triphones differ, this will be corrected by the substitution of t by u.

Let the transitions from $|A_1|$ to $|C_1|$ and from $|B_2|$ and $|A_2|$ be parameterized by $\mathbf{a}_1(u)$ and $\mathbf{a}_2(u)$, respectively. The new transition function $\mathbf{a}_{new}(u)$ from $|A_1|$ to $|A_2|$ is an interpolation between $\mathbf{a}_1(u)$ and $\mathbf{a}_2(u)$. A linear interpolation between the two curves would be: $\mathbf{a}_{new}(u) = (1-u)\cdot\mathbf{a}_1(u)+u\cdot\mathbf{a}_2(u)$

However, the tangents at the central visemes would be changed. To retrieve the tangents of the curves' extremities, we weight the interpolation coefficient over time using the function

$$s(u) = \frac{u^2}{u^2 + (1-u)^2}$$

and the weighted transition becomes

$$\mathbf{a}_{new}(u) = (1 - s(u))\mathbf{a}_1(u) + s(u)\mathbf{a}_2(u)$$

$\mathbf{a}_{new}(u)$ is then sampled with constant Δu to generate the desired number of frames for the new transition. Figure 7.1 illustrates the result of such a morphing.

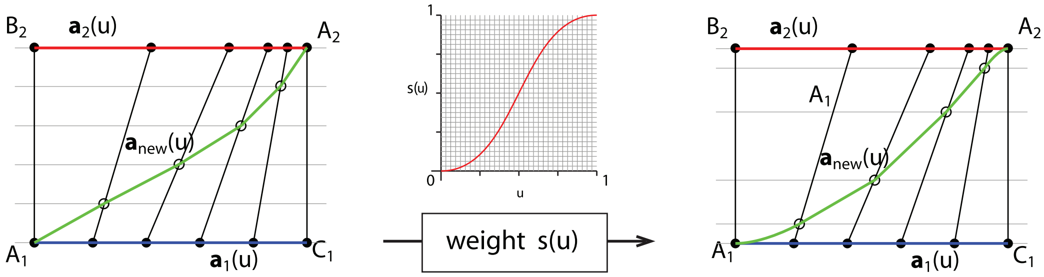

Figure 7.1: When we morph two transitions of two different triphones, we transform the time axis to a normalized variable $u \in [0, 1]$. The curves $\mathbf{a}_1(u)$ and $\mathbf{a}_2(u)$ still preserve the original velocities (spacing of time steps on the red and blue line). The weight factor $s(u)$ for morphing makes sure that the tangents a the starting and the ending point remain. We thus reuse most of the coarticulation information that is in the original data.

For computing the morphing cost θ we take the integral of the distance between the two segments,

$$\theta = \int_0^1 \|\mathbf{a}_2(u) - \mathbf{a}_1(u)\| du$$

7.2 Finding the Optimal Sequence

Given the set of desired target triphones for a new sentence, we find the sequence of database triphones that minimizes both the distance of each triphone candidate to the targeted and the morphing cost that connects them. To compute this solution, the system provides all possible triphone candidates for each triphone in the sentence. A graph is constructed whose edges connect every triphone from these list, to all candidates in the preceding and the next list.

The new sentence is decomposed in a sequence of N triphones. For each desired triphone $i \in 1, .., N$, the system generates a list of all M_i possible triphone candidates $C_{i,j}$. Each of these candidates holds a *local cost* $\theta_{i,j}$ (the distance to the requested target triphone) and the *morphing costs* $\phi_{i,j,k}$ that connect all M_{i-1} preceding candidate nodes k to $C_{i,j}$. Figure 7.2 illustrates one of the triphone candidates with its relation to its neighbors.

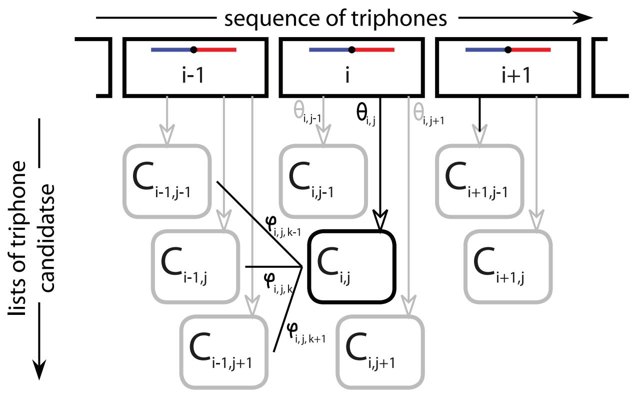

Figure 7.2: The optimal sequence of triphone substitutions is the one that minimizes the distance of its triphones to the requested triphone and the morphing cost connecting them. This solution is found by computing a shortest path algorithm on a graph generated over the triphone candidates.

We want to find the path that minimizes the total cost

$$\Theta_{min} = \sum_{i=0}^{N} \min_{j \in \{0,\ldots,M_i\}} \min_{k \in \{0,\ldots,M_{i-1}\}} (\theta_{i,j} + \phi_{i,j,k})$$

We use a shortest path algorithm over the graph to find the optimal solution. If Θ_{min} is divided by the number of triphones in the sentence, it defines a metric for evaluating the quality of the animation. Finally, the morphing between the selected triphones that was generated when the costs were computed is used to generate the final articulation curve by mapping the coefficients $\mathbf{a}(t)$ to shape vectors $\mathbf{S}$.

8 Results

In this chapter, we first discuss the advantages and the results of the viseme model and the categories that we found, compared with those used in phonetics, and then comment the quality of the final animation (see supplemental video) and the improvements that can be done.

Due to the data compression used in our representation, the viseme representation, the generation of the best fitted new sequence of triphones and the morphing that outputs the PCA coefficients for each new frame are computed in a fraction of a second. Our approach is thus well suited for real time applications.

8.1 Phoneme Similarity

In our process, we started with no *a priori* knowledge of phoneme similarity. However, as it occurs in literature related to mouth animation, different but comparable categories are used based on phonetics theory. For instance, the categories used in [4] groups the phonemes the following way:
the consonants {/CH/,/JH/,/SH/,/ZH/}, {/K/,/G/,/N/,/L/}, {/T/,/D/,/S/,/Z/}, {/P/,/B/,/M/}, {/F/,/V/}, {/TH/,/DH/}, {/W/,/R/}, {/HH/}, {/Y/}, {/NG/}, and all vowels separately, {/EH/}, {/EY/}, {/ER/}, {/UH/}, {/AA/}, {/AO/}, {/AW/}, {/AY/}, {/UW/}, {/OW/}, {/OY/}, {/IY/}, {/IH/}, {/AE/}, {/AH/}.

According to phonetics, phonemes are generated by a combination of structures, mainly of the lips, the tongue, the epiglottis and the larynx (the nasal cavity is only phonetically noticeable as a consequence of the air conducted to the nose). From these four elements, the two first have a direct influence on the shape of the mouth whereas the last two occur at the back of the mouth. The epiglottis will close the aperture to the trachea to pronounce phonemes like /K/ or /G/, whereas the larynx has the control over producing a voiced or an unvoiced phoneme.

Vocals (/AA/, /AE/, /AH/, /AO/, /AX/, /AXR/, /AW/, /AY/, /EH/, /ER/, /EY/, /IH/, /IX/, /IY/, /OW/, /OY/, /UH/, /UW/) are all voiced phonemes with no influence of the tongue though for some have air conducted through the nose like for /ER/ (the nasal phonemes changes when the nose is closed); they only depend on the shape of the lips and thus have a direct influence over the produced viseme; it is then valid to represent them all as distinct visemes. Regarding the consonants, some can be visually grouped in pairs when they are uttered the exact same way, once voiced, once unvoiced. These groups are the (/B/, /P/), (/G/, /K/), (/V/, /F/), (/ZH/, /SH/), (/JH/, /CH/), (/Z/, /S/), (/D/-/DH/-/DX/, /T/). Regarding the rest of the consonants, /M/, /W/, /Y/, /L/, /N/, /R/, /NG/, /TH/ or /HH/ have different tongue positions and lip configurations; voiced or not, they should fall into separate categories as they produce slight differences on the behavior of the mouth. It is one of the strengths of our system that it determines, based on the recorded corpus, which viseme can be approximated by which other one or how far a phoneme is from another.

Considering that we have a relatively small corpus of viseme samples (from 2000 fames, around a fourth are taken as viseme candidates) and that the number of occurrences of the different visemes varies massively (from a single candidate for —UH— to 17 for —D— and 71 for —SIL—), our similarity measure and viseme categories would improve with a larger corpus. Nevertheless, it comes close to the usual categories. According to our generated classification of the visemes (see Figure 6.1), for —P— is a direct extension to —B—, as are —F— to —V—, —ZH— to —SH—, —DH— and —DH— to —T—. —G— and —K— are represented by the same triplet, —Z— is a 3 step away extension to —S—. We have a miss match with —CH— and —JH—, though both can be directly extended the a common child node in the tree ([102]).

Another aspect in the articulation process is that some phonemes have more influence that others. For instance /UH/ forces the mouth to a round shape whereas /K/ can be articulated with almost any shape, especially because the sound is produced at the back of the mouth. This aspect is important to consider when generating animations because it dictates which visemes are to be more accurately approximated. This information is available in our database as is automatically taken into account in our pipeline. —K— or —N— are typical examples of visemes that can be substituted by more clusters.

8.2 Animation

For the synthesis of a novel animation, please see the supplemental video. Most of the transitions between the visemes are smooth. It might however occur that the system was unable to find a good substitution to a triphone without drastically increasing the cost of the neighbor triphones; this explains the quick mouth jumps that happen from time to time. Increasing the size of our database would have a direct influence on the smoothness of the results.

When a long pause occurs in the novel sentence, the pause is synthesized by a triphone than spans many frames. The coarticulation is still learned from short triphones that are stretched over time. Therefore, the transition during the pause is a slow morphing from the last viseme of the previous sentence to the first viseme of the next sequence. This effect can be neutralized by defining a time span limit over which the articulation should go to the neutral mouth configuration (PIT) and restart just before the new sentence.

We would like to point out that the white lipstick on the recorded face was only on the edges of the lips, thus giving the impression that the mouth never closes properly. When the articulation is viewed from the side, one can see that the mouth still closes completely.

9 Conclusion

We presented an automated, data-driven approach for developing a representation of visemes based on a nonlinear statistical analysis. Our representation defines a similarity measure, a hierarchy if viseme categories that we store in a graph structure, and an algorithm for triphone-based speech animation. On a recorded corpus of 3D articulation data, we constructed a system that determines which visemes from different sets of clusters can be valid candidates for other clusters and thus defines a rule for viseme substitution and triphone selection. Our algorithm also contains a measure for the cost of substitutions whenever we make selections that violate our conservative standard substitution rule. The strength of that approach is that we can work with a relatively small corpus and still get large groups of candidates for specific triphones.

For generating a new articulation animation, our system is given a sequence of target triphones for which we want an optimal sequence of database triphones. The system looks for the best suited triphones candidates that minimizes the distance to the desired triphones and that also minimizes the morphing cost at the transitions.

The final animation show smooth, but distinct movements and coarticulation effects, with some limitation due to the size of our corpus. However, the learned coarticulations bring a lot of realism which shows that our approach might bring promising results in the future. We are planning to address the problem of pauses in a sentence by forcing the mouth to return to a rest position. Head movement and eye blinking would also be an interesting way to bring more realism to the animation.

Our viseme similarity measure and the categories of similar visemes are a general concept that can be derived from a wide variety of input data. It would be interesting to investigate how similar the results are that we obtain with different measurement techniques, and if a larger database reveals more structures and a finer distinction of the viseme clusters.

Bibliography

[1] V. Blanz, C. Basso, T. Poggio, and T. Vetter. Reanimating faces in images and video. In P. Brunet and D. Fellner, editors, *Computer Graphics Forum, Vol. 22, No. 3 EUROGRAPHICS 2003*, pages 641–650, Granada, Spain, 2003.

[2] V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. In *Computer Graphics Proc. SIGGRAPH'99*, pages 187–194, 1999.

[3] M. Brand. Voice puppetry. In *Computer Graphics Proc. SIGGRAPH'99*, pages 21–28, Los Angeles, 1999.

[4] C. Bregler, M. Covell, and M. Slaney. Video rewrite:driving visual speech with audio. In *Computer Graphics Proc. SIGGRAPH'97*, pages 67–74, 1997.

[5] Y. Cao, P. Faloutsos, E. Kohler, and F. Pighin. Real-time speech motion synthesis from recorded motions. In *SCA '04*, pages 345–353, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[6] Y. Cao, P. Faloutsos, and F. Pighin. Unsupervised learning for speech motion editing. In *SCA '03*, pages 225–231, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[7] E. Chuang and C. Bregler. Mood swings: expressive speech animation. *ACM Trans. Graph.*, 24(2):331–347, 2005.

[8] M. M. Cohen and D. W. Massaro. Modeling coarticulation in synthetic visual speech. In *Models and Techniques in Computer Animation*, 1993.

[9] Z. Deng and U. Neumann. efase: expressive facial animation synthesis and editing with phoneme-isomap controls. In *SCA '06*, pages 251–260, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[10] T. Ezzat, G. Geiger, and T. Poggio. Trainable videorealistic speech animation. In *Computer Graphics Proc. SIGGRAPH'02*, pages 388–398, San Antonio, 2002.

[11] X. Huang, F. Alleva, H.-W. Hon, M.-Y. Hwang, K.-F. Lee, and R. Rosenfeld. The SPHINX-II speech recognition system: an overview (http://sourceforge.net/projects/cmusphinx/). *Computer Speech and Language*, 7, 2:137–148, 1993.

[12] S. Kshirsagar and N. Magnenat-Thalmann. Visyllable based speech animation. In P. Brunet and D. Fellner, editors, *Computer Graphics Forum, Vol. 22, No. 3 EUROGRAPHICS 2003*, pages 631–639, Granada, Spain, 2003.

[13] J. Ma, R. Cole, B. Pellom, W. Ward, and B. Wise. Accurate visible speech synthesis based on concatenating variable length motion capture data. *IEEE Transactions on Visualization and Computer Graphics*, 12(2):266–276, 2006.

[14] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[15] E. Sifakis, A. Selle, A. Robinson-Mosher, and R. Fedkiw. Simulating speech with a physics-based facial muscle model. In *SCA '06*, pages 261–270, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[16] Y. Wang, X. Huang, C. Lee, S. Zhang, Z. Li, S. Samaras, D. Metaxas, A. Elgammal, and P. Huang. High resolution acquisition, learning and transfer of dynamic 3-d facial expressions. In M.-P. Cani and M. Slater, editors, *Computer Graphics Forum, Vol. 23, No. 3 EUROGRAPHICS 2004*, Grenoble, France, 2004.