2005

Real-time viseme extraction

Edward A. Kmett

Real-time viseme extraction

Abstract

With the advance of modem computer hardware, computer animation has advanced leaps and bounds. What formerly took weeks of processing can now be generated on the fly. However, the actors in games often stand mute with faces unmoving, or speak only in canned phrases as the technology for calculating their lip positions from an arbitrary sound segment has lagged behind the technology that allowed the movement of those lips to be rendered in real-time. Traditional speech recognition techniques requires the entire utterance to be present or require at least a wide window around the text to be matched to allow for higher level structure to be used in determining what words are being spoken. However, this approach, while highly appropriate for recognizing the sounds present in an audio stream and mapping those to speech, is less applicable to the problem of "lip-syncing" in real time. This paper looks at an alternate technique for applying multivariate statistical techniques to lip-sync a cartoon or model with an audio stream in real time, which requires orders of magnitude less processing power than traditional methods.

Degree Type

Open Access Senior Honors Thesis

Department

Mathematics

Keywords

Animation (Cinematography), Computer animation

REAL-TIME VISEME EXTRACTION

By

Edward A. Kmett

A Senior Thesis Submitted to the

Eastern Michigan University

Honors Program

In Partial Fulfillment of the Requirements for Graduation

with Honors in Mathematics

Approved at Ypsilanti, Michigan, on this date 5-1-05

1. Abstract

With the advance of modern computer hardware, computer animation has advanced leaps and bounds. What formerly took weeks of processing can now be generated on the fly. However, the actors in games often stand mute with faces unmoving, or speak only in canned phrases as the technology for calculating their lip positions from an arbitrary sound segment has lagged behind the technology that allowed the movement of those lips to be rendered in real-time. Traditional speech recognition techniques requires the entire utterance to be present or require at least a wide window around the text to be matched to allow for higher level structure to be used in determining what words are being spoken. However, this approach, while highly appropriate for recognizing the sounds present in an audio stream and mapping those to speech, is less applicable to the problem of "lip-syncing" in real time. This paper looks at an alternate technique for applying multivariate statistical techniques to lip-sync a cartoon or model with an audio stream in real time, which requires orders of magnitude less processing power than traditional methods.

2. Background Linguistics

2.1. Phonemes and Visemes

Speech in the English language consists of approximately 45 distinct sounds, called *phonemes*. This model is being applied to English language speech, however, it could be generalized to the entire International Phonetics Association's IPA phoneme set if there were sufficient training data available on foreign languages. However, training for such a larger data set would result in a worse fit for the case at hand, and no such gigantic repository of phonetic information is available (see §6.1 for an idea of the scale of such a project).

Disney animators have long relied on a dozen lip positions they call *visemes* to animate cartoon characters (Lander). This technique and terminology has since moved on to become largely standardized, as generations of animators have been using it to hand-translate sounds from voice actors into lip positions.

Extensions to the Disney model have been proposed, the international lip-reading association uses 18, and an extended list of visemes were generated in Ezzat and Poggio, which have improved characteristics, but the underlying idea is the same and the categorization techniques here work regardless of extensions made to the underlying viseme set.

It is difficult to classify phonemes given an audio sample. Simplistic approaches, such as directly examining the samples over a short time segment fail because people speak at different rates and different pitches. Because of varied pitch and the fact that the information in a given time segment is contained in the changes in the frequencies present more than in the actual frequencies present, we need to turn to more sophisticated tools to extract information that will help tease out what phonemes are present in a snippet of audio.

2.2. Mel Frequency Scale

The human eardrum does not perceive sound in a linear fashion, as many people believe. The Hertz scale, which is used traditionally does not model human perception, but instead models the underlying physical reality. For high frequencies, your perception of the sound is not at loud as the actual sound, since the shell-shaped cochlea in your ear attentuates the sound in a non-linear way.

An alternative model, which is very good at capturing the way that human hearing works, is called the *Mel frequency scale*. Below 1000 Hertz, the Mel scale matches up with the Hertz scale, above that point it tapers off logarithmically rather than linearly.

3. Background Mathematics

3.1. Random Variables

A random variable in mathematics can be intuitively viewed as a quantity that is measured in connection with a random experiment.

Let Ω be our sample space consisting of all possible outcomes for the experiment, let w be the outcome of a particular experiment and let $R : \Omega \rightarrow \mathbb{R}$ be a function. Then the process of measuring R(w) is how one obtains the value of the random variable for a particular sample.

There are many distributions of this form that emerge in practice, but one of the most important and oft-recurring ones is the *normal distribution*, which generates the Gaussian (bell) curve familiar to educators.

The normal distribution is of note because it emerges as a consequence of the Central Limit Theorem (DeGroot). In short this theorem states that the sample mean of a large random sample of independent identically distributed random variables with finite mean and variance has an approximately normal distribution. Consequently, when you are measuring something involving many random variables that function as independant parts, the distribution of the result is approximately normal. This result motivates many of the uses of the normal distribution in probability and statistics. In particular it is this observation which motivates our later assumption that we can approximate each phoneme with a Gaussian mixture.

A random variable, X, with a normal distribution is completely described by two parameters, its mean, μ, and its variance σ^2.

Note that $\mu = E(X)$, where E(X) is the expected value of X.

3.2. Multivariate Methods

The discussion thus far has assumed that we are talking about the distribution of only one random variable, or if we are talking about more than one random variable that they are independent. In other words, we have been able to assume that the value taken on by one of the variables does not affect the distribution of the values that the other variable can assume.

However, when dealing with real world data, typically there is a fair amount of cross-talk between your measurements. For example. if you are measuring meteorological data, changing temperatures are likely to affect barometric pressure, so these two measurements are probably not perfectly independant.

In order to capture this, one introduces the concept of covariance. The covariance of two random variables, X and Y, with means, μ_X and μ_Y respectively, is given by

$$Cov(X,Y) = E[(X - \mu_X)(Y - \mu_Y)]$$

Cov(X,Y) gives you a way to measure the linear association of X and Y. As the number of variables

starts to increase, tracking them all individually and assigning them names in this manner starts to become tedious so we turn to linear algebra for more concise terminology. Rather than assign distinct letter names to each variable, we use a single variable name, X, and use numerical subscripts to identify the variable.

So we let $X_1, X_2, \ldots X_n$ represent n distinct random variables. In a similar manner, we define the covariance matrix, C, as follows:

$$c_{ij} = \mathrm{Cov}(X_i, X_j)$$
$$C = (c_{ij})$$

The purpose of this definition is to allow us to use linear algebra to manipulate our data.

3.3. Cepstral Analysis

The finite one-sided Z transform of a signal, $x_1, \ldots x_n$, can be defined by

$$Z\{x\} = X(z) = \sum_{i=0}^{n} \frac{x_i}{z^i}.$$

The Z transform maps a function of a subset of the integers into a complex valued function and has many uses in the field of analysis. Like many transforms, this operation is invertible. In digital signal processing the Z transform and its inverse are used primarily to define complex *cepstrum* of a signal as follows

$$\hat{x} = Z^{-1}\{\log(Z\{x(n)\})\}$$

The cepstrum is used in many problems involving the *convolution* of a signal with a filter to remove noise, strip off echoes, or generally smooth the signal. The process of convolution is analagous to using one signal to smooth out the values of another.

One of the many reasons, cepstral analytic techniques are important in signal analysis because the mapping between the original signal and the cepstra of the signal is a homomorphism that maps convolution onto addition ("Interlude-Cepstral Analysis"), much like the Fourier transform maps convolution onto multiplication. This transforms the very complicated process of convolution on the original signal into simple pointwise addition.

The above formulation for the complex cepstrum of a signal is unfortunately difficult to calculate quickly. However, since we are dealing with a real-valued discrete signal with reasonable parameters, we can use a much faster mechanism for calculating the real cepstrum of the signal based on the fast Fourier transform, $\mathcal{F}$:

$$\hat{x} = \mathcal{F}^{-1}\{\log(\mathcal{F}\{x\})\}$$

The resulting real-valued cepstrum provides a good indicator of where the energy is in the signal, and the logarithmic scale is well suited to human perception. The real cepstrum is an approximation of the contents of the complex cepstrum and unlike the complex cepstrum cannot be transformed back to the input signal. Since this application does not require this capability, this limitation is harmless for our

purposes.

4. The Problem

4.1. General Issues

The problem at hand is to be able to take a real-time stream of audio data and transform it as quickly as possible into lip positions that can then be used to lip-sync a model to the data. However, in order to render this problem concrete, a number of parameters must be fixed.

Once the problem is defined, it decomposes into two problems, a pre-process involving training a model of English language speech and a real-time component which must be capable of quickly categorizing audio data according to the model generated by the first step.

4.2. Specific Decisions

We will assume that the data is arriving in the computer as a 16kHz 8-bit μ-law encoded audio stream. This level of quality should be familiar to the average individual, as this has approximately the same clarity as a telephone conversation across a modern phone switch.

In the interest of transparency we need to be able to generate a response in under 200 milliseconds. If the process takes longer than this and then there is an unnatural and noticable delay between the occurence of speech and lip motion.

To meet this goal, we subdivide the audio into approximately 20 millisecond long time segments of 512 samples each. This quantity is chosen to help meet the delay constraints above, and yet be a power of two, to allow the fast Fourier transform (FFT) to be performed on each segment efficiently. See Folland for a description of the fast Fourier transform algorithm.

We will be examining data from both the current time slice and the previous time slice in order to construct our fit. Since we hear phonemes, not visemes, the model uses the data to fit a phoneme to the audio data, and then maps the phoneme onto the viseme that represents the lip position used to make that particular sound.

Since we are only interested in identifying where the energy is in the signal and do not need to be able to transform the cepstrum back into the input signal we are able to work with the real cepstrum. Once we have obtained the real cepstrum of the signal, we cluster the resulting coefficients into twelve evenly distributed bands. The choice of sampling rate and the characteristics of the human ear work out nicely to give us a set of twelve coefficients that can be used to help identify the sound present in a given segment, with similiar response characteristics to that of a human ear. These coefficients are traditionally known as Mel Frequency Cepstral Coefficients (MFCCs) and are used often in speech recognition tasks.

However, as mentioned earlier, phonemes are characterized not only by the frequencies present in the time segment but the changes in frequencies. In traditional speech recognition techniques this leads to the usage of triphones and hidden Markov models to try to get higher level information about the audio stream to help in the decoding of the sample (Cole). However, such techniques require considerably more processing power and a wider sampling window than can be brought to bear in the real-time processing window alloted to our classification task.

Instead of trying to look both forward and backwards through the data, we elect to only look

backwards. Since we already had to perform the calculation for the MFCCs for the previous time slice, we recycle them momentarily for the evaluation of the current time slice, and calculate twelve additional Δ-MFCCs, representing the changes in the frequencies present from the previous audio sample. This allows us to recognize phonemes that are not really sounds at all, but more related to the cessation of sound. An example is the glottal stop in the middle of the exclamation, "uh-oh".

This gives us 24 values to examine in the classification of our data set, and a large number of elements to train in our model in order to categorize sound segments containing them. From here on out we will simply pack these 24 values into a vector of random variables, $\mathbf{x} = (x_1, \ldots, x_{24})$.

5. Modeling a Solution

Before we can train our model, we have to explore how it is going to be represented internally by a Gaussian mixture model.

A Gaussian mixture consists of a mean vector, $\boldsymbol{\mu}$ and a covariance matrix, which for reasons to be explained momentarily will be stored inverted as C^{-1}.

For each phoneme we store both an index into the table of visemes, and a Gaussian mixture, Φ_i, consisting of mean, $\boldsymbol{\mu}_i$, and inverse covariance matrix, C_i^{-1} which we use to categorize incoming samples.

Since we are dealing with English, we have to deal with the 45 traditional phonemes, plus half a dozen more pseudo-phonemes representing various sounds that humans make that are not conventional phonemes, such as coughing and breathing sounds.

The next problem is how to determine an effective measure of distance between a sample, $\mathbf{x}$, and a particular trained mixture Φ_i.

Since $\boldsymbol{\mu}_i$ and $\mathbf{x}$ are both vectors in $\mathbb{R}^{24}$, the obvious distance metric is traditional Euclidean distance. Unfortunately, Euclidean distance has a number of shortcomings for our purposes. We have no real reason to suspect that the individual aspects of our sample have variances that are even of the same order of magnitude. Furthermore, we may have strong covariance between the various parts of $\mathbf{x}$ that Euclidean distance would just ignore.

We need an alternate distance metric that takes into account the covariance between different coefficients and their relative variances. Once we have this better distance metric, we can take a given sample and measure it against each mixture in turn, so that the phoneme which most closely fits the sample can be selected.

Thankfully, P.C. Mahalanobis defined a distance measure with these very properties back in 1936 (Lande). The square of this *Mahalanobis distance* between a vector, $\mathbf{x}$ and our ith phoneme is given by the formula:

$$M_i^2(\mathbf{x}) = (\mathbf{x}-\boldsymbol{\mu}_i)^T C_i^{-1} (\mathbf{x}-\boldsymbol{\mu}_i)$$

$M_i(\mathbf{x})$ has some very nice statistical properties. The most important for our purposes is that if $\mathbf{x}$ is ε standard deviations distant from Φ_i, then $M_i(\mathbf{x}) = \varepsilon$.

Since we only care about relative ordering, we can use the square of the Mahalanobis distance, rather

than the distance itself to avoid the overhead of calculating the square root.

To select the best match, given a fully trained model, iterate over the set of phonemes, calculating the Mahalanobis distance to each phoneme's Gaussian mixture, and select the one with the lowest distance for the current time segment. This simple approach can be smoothed out by using a spline based fit between the top 2 or 3 candidates if necessary to allow smoother transitions, or this can be carried out further down in the graphics pipeline as part of a more general animation smoothing kernel.

6. The Real World

6.1. Training

The final hurdle is the training of the model with actual human voices so that it can be used to recognize speech. The complexity of this particular hurdle is easy to underestimate.

We have 51 pseudo-phonemes in a table, each with a 24-valued vector, $\boldsymbol{\mu}_i$, and a 24x24 matrix, C_i^{-1}. This means that in order to train the phoneme model, we have to train 51*24*25 = 30,600 different values based on training data obtained from human speech which has been laboriously hand-tagged to identify the phonemes present in each snippet of time.

Even if we assumed that we could train a given variable independantly with just 50 samples of a particular phoneme, and if we assume that all of the sounds in the English language were equially likely we would still need over 8 hours of continuous human speech tagged as previously described in order to train the data set to work in an optimal fashion.

By the time you factor in the relative frequencies of different phonemes, and the various inefficiencies of the Jacobi iteration used to train the matrix, you need approximately 48 hours of continuous audio to train the matrix as described above. Jacobi iteration is a standard iterative technique to approximate a solution to a system of linear equations, which is necessary in our case because the large number of samples involved in training our model preclude other, more precise, methods.

So it becomes necessary to introduce a simplifying assumption. If we can assume that each C_i is a diagonal matrix then our complexity drops by a factor of 24. This reduces the tagged audio requirement to 2 hours, at the expense of weakening the accuracy of our Mahalanobis distance metric. In practice this is not a major problem for this particular application, because phonemes that map to the same viseme tend to be close to one another in our search space; in the event that our less accurate metric results in a wrong phoneme being selected, it typically selects one that maps to the same or a similar viseme.

This simplifying assumption also has implications for the run time performance of the algorithm. We enjoy a similar factor of 24 reduction in overhead there, reducing the final classification of a given sample vector to approximately 2500 multiplications and 50 additions. This renders the algorithm appropriate for use when the majority of the processing power is being devoted to other things, such as rendering a 3D scene.

6.2. Implementation

The data set used to test the implementation of the algorithm was the 1993 DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus, which consists of recordings of 630 speakers from 8 major dialects of American English. The availability of this dataset motivated many of the choices

made about sampling rates. Furthermore its limited size forced the decision to diagonalize C^{-1}.

The source code from this project has been contributed to the GNU Open Mind Speech initiative, available at http://freespeech.sourceforge.net/. The Open Mind Speech initiative is an effort to develop a freely available library for speech recognition and related technologies.

7. Conclusion

By approaching the tools used for speech recognition from a novel direction, we have managed to apply a mixture of signal processing tools and multivariate statistical techniques to the problem of viseme extraction. The result draws on techniques from many areas of research including linguistics, analysis, statistics, and computer science.

The one commercial vendor who provides a comparable product retails their solution for over $10,000 USD at the time of this writing. By contributing this research to a public project, it is hoped that real-time lip-syncing will find broader acceptance and application. A public-domain implementation is particularly of interesting at this time because of the increasing use of broadband internet connections and "walkie-talkie"-like voice communication in games. It is only a small step to integrate this technology in existing interactive multiplayer 3D games so that you can actually watch the lips move on the animated faces of your fellow players while they talk.

References

Cole, Ron (Ed. in Chief) Survey of the State of the Art in Human Language. (1996) Dec 10, 2004. <http://cslu.cse.ogi.edu/HLTsurvey/HLTsurvey.html>

DeGroot, Morris and Schervish, Mark. *Probability and Statistics: Third Edition.* (2002) Addison-Wesley.

Ezzat, Tony and Poggio, Tomaso. "Videorealistic talking faces: a morphing approach." (1997) AVSP-1997, 141-144

Folland, Gerald B. *Fourier Analysis and its Applications*. (1992) Brooks/Cole Publishing. p253

Garofolo, John et al. "TIMIT Acoustic-Phonetic Continuous Speech Corpus." *NIST Speech Disc CD1-1.1.* 1993. Dec 10, 2004. <http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S1>

"Interlude-Cepstral Analysis" *Signals and Systems Documentation.* Wolfram Research. (2004) Dec 10, 2004. <http://documents.wolfram.com/applications/signals/CepstralAnalysis.html>

"IPA: Alphabet." International Phonetic Association (1996) Dec 10, 2004. <http://www.arts.gla.ac.uk/ipa/ipachart.html>

Lande, Unni. *Mahalanobis Distance*. Mar 24, 2003. Sep 25. 2004. <http://biologi.uio.no/fellesavdellinger/finse/spatialstats/Mahalanobis%20distance.ppt>

Lander, Jeff. "Read My Lips – Facial Animation Techniques," Gamasutra. Apr 06, 2000. Sep 25. 2004. <http://www.gamasutra.com/features/20000406/lander_01.htm>

"Mel scale," Wikipedia. Oct 13, 2004. Dec 10, 2004. <http://en.wikipedia.org/wiki/Mel_scale>

Valin, Jean-Marc. *Open Mind Speech – Free Speech Recognition for Linux*. Universite de Sherbrooke. Dec 10, 2004. Dec 10, 2004. <http://freespeech.sourceforge.net>