Laughing, Crying, Sneezing and Yawning: Automatic Voice Driven Animation of Non-Speech Articulations *

Darren Cosker†
Department of Computer Science
University of Bath

James Edge‡
Centre for Vision, Speech and Signal Processing
University of Surrey

Abstract

In this paper a technique is presented for learning audio-visual correlations in non-speech related articulations such as laughs, cries, sneezes and yawns, such that accurate new visual motions may be created given just audio. Our underlying model is data-driven and provides reliable performance given voices the system is familiar with as well as new voices. We demonstrate how performance accuracy in voice driven animation can be related to maximizing the models likelihood, and that new voices with similar temporal and spatial audio distributions to that of the model will consistently provide animation results with the lowest ground truth error. By exploiting this fact we significantly improve performance given voices unfamiliar to the system.

Introduction

Facial animation has a wide range of applications, including video game and movie characters, medical visualisation, psychological stimuli, online avatars and virtual guides. Highly realistic characters, such as those seen in movies, require teams of expert artists and animators and involve months of manual effort. Perhaps the most popular technique employed to generate facial animations for movies and video games is blend-shape animation [Fordham 2003]. This allows an animator fine control over a facial model in order to add subtle nuances or correct some unwanted motion. Performance driven animation is also a popular animation technique [Pighin and Lewis 2006]. This involves tracking a performers facial movements (e.g. using motion-capture) and then mapping them onto a facial model. However, it is very uncommon for such a method to not be followed up with some manual intervention, e.g. by adjusting blendshape weights on a model. High quality animation, such as that seen in movies, is always the result of several techniques and a large proportion of manual effort. There is therefore always an interest in the development of new techniques that can increase productivity and reduce development time.

The idea of being able to automatically generate a facial animation from speech is therefore a highly attractive proposition. Given such a technique, an actors voice track could be used to automatically animate a facial model, including lip-synching and facial expression. This has advantages over e.g. performance driven animation which additionally involves physically recording an actors performance using a capture system.

Speech driven animation also has great potential in online video games, such as World of Warcraft [Entertainment]. In this case, the voice of a person speaking to their friend may be translated onto their virtual avatar. This would far improve current online games where the avatars of two people conversing by voice do not show any sign of motion or interaction, and instead are quite wooden.

Another important aspect to consider in voice driven animation is the many auditory gestures that are not related to speech, but may be considered as non-verbal or non-linguistic articulations. These include e.g. laughing, crying, sneezing and yawning. Consider again the example of an online game, where now the two people share a joke and begin to laugh into their head-sets. In this case, it would add a new dimension of interaction if the virtual avatar where also to laugh, or perform whatever non-speech related articulation the player is currently exhibiting.

An important aspect of any such system is also that it should be robust to the sound of different peoples articulations, such that it should be able to generate appropriate actions given voices it has not heard before. This is a difficult problem in automatic audio driven synthesis.

In this paper, the subject of automatically animating virtual characters using non-speech related articulations is considered. A technique is presented for learning audio-visual correlations in example laughs, cries, sneezes and yawns, such that accurate new visual motions may be created given just audio. The technique is data-driven in that the underlying model is trained from captured motions and audio segments from performers as opposed to a simulated model. We also address the generation of appropriate visual actions using voices the system is not familiar with. We show how our approach inherently allows a degree of speaker independence, and that an added classification step can further improve robustness given new voices used for animation.

Prior Work

Automatic animation from speech has been an area of much activity. Two general approaches are usually considered: phoneme driven animation or animation from raw audio recordings. Cohen and Massaro [Cohen and Massaro 1993] described an approach whereby an audio track is first transcribed using phonemes and then the phonemes are translated to corresponding visemes (visual counterpart to a phoneme) using a rule based mapping. Viseme dominance functions then control the extent to which a viseme is articulated. More recently, Ezzat *et al* [Ezzat et al. 2002] presented video-realistic phoneme driven animations that are almost perceptually indistinguishable from real speakers. This result demonstrates the potential accuracy of phoneme driven methods. Bregler *et al* [Bregler et al. 1997] also produced highly video-realistic results using triplets of phonemes (triphones) as the input to a prelabelled video-triphone database. Sumedha *et al* [Kshirsagar and Magnentat-Thalmann 2003] achieve realistic results by employing vi-syllables as the input to their system. Cao *et al* [Cao et al. 2005], while using standard phonemes as an input for lip-synching, also use a classifier to determine the emotional state of the voice, thus accompanying mouth animation with a suitable facial expression. The advantage of phoneme and phoneme variant based approaches is that phoneme to visual mappings can be pre-defined as a set of rules, making a high degree of reliability possible. Phoneme labels are also speaker-independent, meaning that any persons voice can be labelled and used as the input to a system. The drawback of course is that any audio input needs to be annotated with phonemes before an animation can be produced, and although research activity exists in producing automatic phoneme transcription tools,

[*] Appears in the Proceedings of CASA 2009

[†] e-mail: D.P.Cosker@cs.bath.ac.uk

[‡] e-mail:J.Edge@surrey.ac.uk

reliable results can still only be obtained using semi-automatic or manual means.

A second approach to voice driven animation is to attempt to generate the animation automatically from the raw audio. Gutierrez-Osuna *et al* [Gutierrez-Osuna et al. 2005] used a K-Nearest Neighbour (KNN) classifier to relate short audio speech segments with corresponding visual ones, while Brand *et al* [Brand 1999] introduced a Hidden Markov Model (HMM) approach to learn time-dependent correlations between speech and visual segments. Cosker *et al* [Cosker et al. 2004] produced near-video realistic animation by applying appearance model [Cootes et al. 2001]parameters as visually rich features in a HMM synthesis framework.

The main challenge in attempting to automatically generate visual parameters from speech is to learn the complex many-to-many mappings between the signals. Another challenge is then to make such a system speaker independent, such that it can generate new animations from voice identities it has not heard before. Brand *et al* [Brand 1999] and Cosker *et al* [Cosker et al. 2004] both suggest building separate models for each identity. However, these models typically require several minutes of training footage where facial features in each frame are accurately tracked – and this can be a difficult process. The challenge of encoding speaker independence into a more feasible framework is therefore still an open one.

The large body of previous audio-driven facial animation research is mostly concerned with generating lip-synching from speech. Work on detecting emotion in the voice may also be applied to facial animation. Cao *et al* [Cao et al. 2005] achieve this using classification based approaches that relate underlying audio features – such as voice-pitch – with speaker mood. The resulting corresponding facial expression then accompanies head movement and/or lip-synching.

A separate but equally important animation problem from generating lip-synching or mood is that of creating non-speech related articulations – such as laughing, crying, sneezing and yawning – automatically from speech. Given the importance of such articulations in adding life to an animation, it is surprising how little research has been carried out in this area. The growth of online video-games where people converse with their voice also opens the door to some important new applications of this research – as in the avatar interaction examples described in the introduction. This is in addition to the potential increase in animator productivity given a system for producing full lip-synching and non-speech related articulations from a performers voice.

In an early study, Cosker *et al* [Cosker et al. 2008] began to explore approaches to learn person specific mappings between some common non-speech related actions and facial motion. More recently, DiLorenzo *et al* [DiLorenzo et al. 2008] proposed a parametric physical chest model that could automatically be animated from recorded laughs. To date, these studies are among the only ones to begin addressing audio driven animation of non-speech related motions such as laughing and crying, and there is still much research to be carried out in this area.

In this paper we propose a data-driven HMM based method for learning correlations between non-speech related audio signals and visual facial parameters. Unlike the work of DiLorenzo *et al* these signals are observed from recorded motions of real performers as opposed to a pre-defined physical model. We concentrate on several common non-speech related actions – laughing, crying, sneezing and yawning. As previously described, a major challenge when using automatic audio driven systems is that of achieving reliable performance given a variety of voices from new people. We demonstrate our approach in a number of speaker-independent synthesis experiments, and show how animation error in voice driven animation has a relation to the proximity of audio distributions for different people and well as similarities between their temporal behaviour. By exploiting these facts we consistently improve synthesis given voices from new people. We implement this improvement using a pre-synthesis classification step. In sum, our approach potentially increases the reusability of such a model for new applications (e.g. online games), and can reduces the need to retrain the model for new identities. Our study is the first to explore voice driven synthesis of non-speech actions using a data-driven approach, as well as the first to examine speaker independent animation using such models.

Overview

Our approach initially requires example audio-visual performances of the action of interest for training: e.g. several laughs, cries, sneezes or yawns. A HMM framework then encodes this audio-visual information. The framework may be trained using any number of desired non-speech action types. Training the framework on a single persons voice limits it in the number of new vocal identities it can reliably synthesise new animations for. However, we show that by increasing this basis by just a small number of new identities we can significantly improve animation for new peoples voices. This is because given new identities the known audio distribution of the framework increases, and speaker independence is related to this distributions ability to approximate input distributions. We propose a classification based scheme to improve system robustness to new voices. Figure 1 provides a high level overview of this procedure, including a potential application.

Audio-Visual Data Acquisition

Our entire experimental data set consists of four participants (2 male and 2 female) captured performing approximately 6-10 different laughs, cries, sneezes and yawns using a Qualysis optical motion-capture system. We captured audio simultaneously at 48KHz. We placed 30 retro-reflective markers on each person in order to capture the visual motion of their face while performing the different actions (see Figure 2). The system captured the movement of the markers at 60 frames-per-second. The motion-capture therefore provides our visual input for training and initial output for animation. During training, our model learns correlations between the recorded audio and the visual motion-capture data. The algorithms output is a new set of 3D motion vectors – 30 in total corresponding to the positions of the motion-capture markers used in training. We then use an Radial Basis Function (RBF) mapping [Lorenzo et al. 2003] to animate a 3D facial model using these 3D motion vectors – this provides the output of the system. Note that any 3D head model may be animated during this last step depending on the application.

The motion capture movements of each person contain head motion as well as the facial changes caused by e.g. a laughing action. It is important to model these variations separately so that any model does not confuse a head motion with a facial expression change [Cootes et al. 2001]. The motion capture data set was therefore aligned to a common coordinate system using a least-squares alignment procedure [Umeyama 1991]. This eliminates variations in translation, rotation and scale caused by changes in head pose.

This provided us with a data set $\mathbf{X} = \{\vec{x}_1, \dots \vec{x}_N\}$ containing the different actions of all four people , where N is the total number of motion-capture vectors, and $\vec{x}$ is a vector of 3D points relating to the positions of each of the 30 retro-refelctive facial markers, i.e. $\vec{x}_i = [x_1, y_1, z_1, \dots, x_{30}, y_{30}, z_{30}]$. Note that in our notation we refer to $\vec{x}_i$ as a column vector.

The physical structure of each persons face is very different. There-

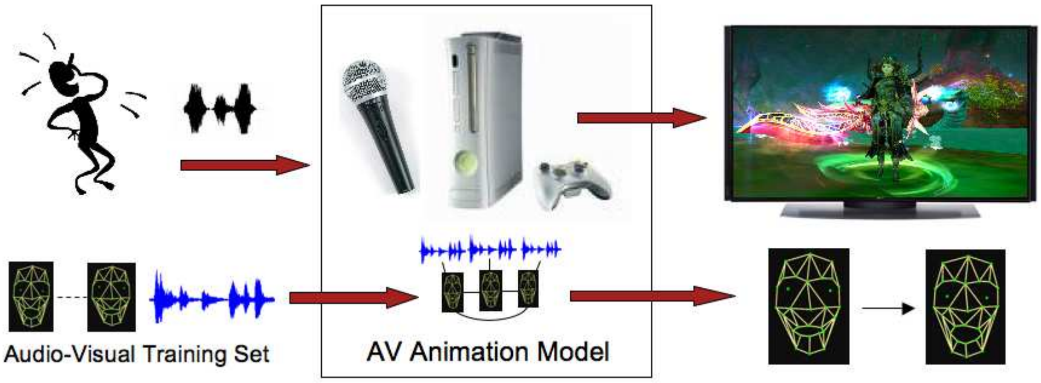

Figure 1: *Given non-speech related inputs for a person (e.g. laughing), these may be transformed into corresponding 3D facial animations. A pre-trained model may sit on a system and given existing knowledge of just a small set of people voices can show remarkable robustness to a wide range of new voice inputs.*

fore, when two different people laugh the shape of the face at the neutral, in-between, and peak positions may vary greatly. Since it is the basic facial movement performed during e.g. a laugh that we are interested in – and not comparative differences in facial identity during different laughs – then it is necessary to attempt to eliminate differences due to facial identity from our training set. We do this normalisation as follows: we first pick one of the performers to serve as the common facial identity, and then calculate the mean 3D motion vector $\vec{x}_o^{norm}$ for this person. We now wish to make the identities of the motion-capture data for the other performers the same as for this chosen person. To do this, we take the entire motion capture for a performer, calculate its mean 3D motion vector $\vec{x}_o^{new}$, and then calculate the offset $\vec{x}_o^{diff} = \vec{x}_o^{norm} - \vec{x}_o^{new}$. We wish to make the mean vector $\vec{x}_o^{new}$ as close to $\vec{x}_o^{norm}$ as possible. If $\vec{x}_o^{norm} = \vec{x}_o^{new}$ then we declare the identity for the new performer as being the same as the one chosen as the common facial identity. To do this we apply $\vec{x}_o^{diff}$ to each 3D motion vector for the new performer and recalculate $\vec{x}_o^{new}$. At each iteration, $\vec{x}_o^{diff}$ should decrease and $\vec{x}_o^{new}$ should converge with $\vec{x}_o^{norm}$. We terminate once $\vec{x}_o^{diff}$ is below a sufficiently small threshold. This procedure is then repeated for the remaining performer motion-capture data such that all motion-capture data for all performers is aligned with the performer chosen to have the common facial identity. After this procedure, the data set $\mathbf{X} = \{\vec{x}_1, \ldots \vec{x}_N\}$ – which consists of all the laugh, cry, sneeze and yawn data for all four performers – is now aligned to a common coordinate frame, and has a normalised identity.

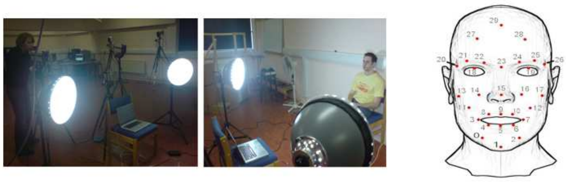

Figure 2: *Motion Capture Set-Up: 30 retro-reflective markers are placed on a performer.*

Audio-Visual Parameterisation and Visualisation

It is widely accepted that the space of facial expressions can be approximated by a low dimensional expression manifold [Chuang et al. 2002]. This is convenient, since it means that we can model and animate faces in a more efficient low dimensional space without a significant loss in generality. The motion capture data currently lies in a 30 dimensional space, and in terms of efficiency both in terms of modeling and processing it would be convenient to map this space to a lower dimensional one. Given an appropriate mapping, this lower dimensional space would then efficiently represent all the important variation in our facial data set.

Principle Component Analysis (PCA) provides a convenient way to reduce the dimensionality of a data set without loss of generalisation. Perhaps the most popular examples of the use of PCA for dimensional reduction of facial data are Active Appearance Models [Cootes et al. 2001] and 3D Morphable Models [Blanz and Vetter 1999]. In both cases, facial data is shown to be well approximated by a lower dimensional parameter space. In a similar way we use PCA to reduce the dimensionality of our data set. We first calculate the covariance matrix for $\mathbf{X}$, and then perform a Singular Value Decomposition (SVD) of the covariance matrix to find its eigenvectors $\mathbf{U}$ and eigenvalues $\vec{u}$. We may now represent $\vec{x}$ as follows:

$$\vec{x} = \vec{x}_o + \mathbf{U}\vec{v} \tag{1}$$

where $\vec{x}_o$ is the mean of $\mathbf{X}$. The eigenvectors $\mathbf{U}$ are ordered in terms of how much of the total variation in $\mathbf{X}$ they represent. In order to reduce the dimensionality of our data set, we remove the eigenvectors from $\mathbf{U}$ representing the smallest amounts of variation. This can be done without harm to generalisation since the lower energy eigenvectors typically represent noise or visibly insignificant facial variations [1]. The reduced dimensional vector $\vec{v}$ may then be calculated by projecting $\vec{x}$ onto the reduced basis $\mathbf{U}$

$$\vec{v} = \mathbf{U}^T(\vec{x} - \vec{x}_o) \tag{2}$$

We retain enough eigenvectors to approximate 95% of the total original variation in the motion capture training set. Projecting the

[1] Of course, one must still be careful to observe how much variation is taken from the model since removal of eigenvectors will eventually lead to a degraded overall model

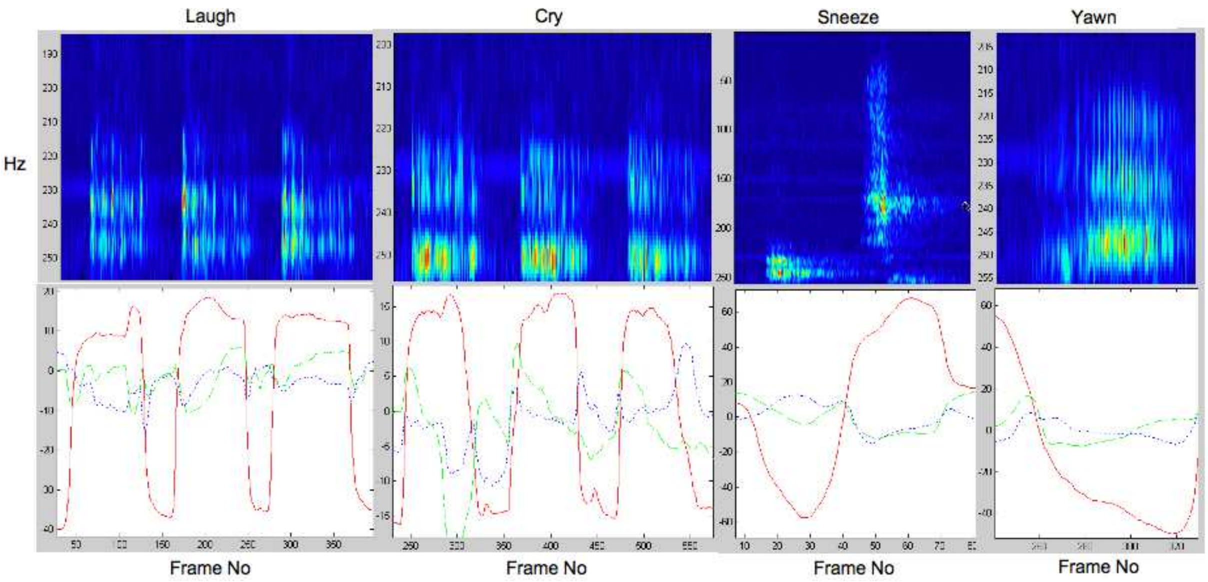

Figure 3: *Audio signals are represented using Spectrograms, and visual parameters using elements of $\vec{v}$ relating to the three eigenvectors encoding the most visual variation. Red, Green and Blue trajectories represent temporal changes in the first, second and third elements of $\vec{v}$ respectively*

.

entire set of motion capture vectors $\mathbf{X}$ onto $\mathbf{U}$ in equation [2] gives $\mathbf{V} = \{\vec{v}_1, \ldots, \vec{v}_N\}$.

As well as efficiently parameterising the visual features, it is equally important to select a set of suitable audio features. Several types of feature extraction techniques are employed in speech processing, and most are based on representing the audio signal as a combination of different frequency components. We do not go into depth about the extraction of such features here, and instead refer the reader to [Deller et al. 1999]. Given the array of different audio processing techniques available it is important to select the most appropriate. A Canonical Correlation Analysis (CCA) can be performed to measure the correlation between different types of signal, and we used this to compare different types of audio features to corresponding visual features. We compared the correlation between our visual parameters and audio features produced using Linear Predictive Coding (LPC), Mel-Frequency Cepstral Coefficients (MFCC) and Relative Spectral Transform - Perceptual Linear Prediction Coefficients (RASTA-PLP). We found that the strongest overall correlation was between MFCC audio features and our visual parameters.

We wished to have a corresponding set of MFCCs for each visual parameter. Since audio was sampled at 48KHz, and video at 60FPS, this meant that each set of MFCCs corresponded to a 800Hz audio window approximately 16 milliseconds in width. This audio sampling window and feature set compares favourably with those used in studies exploring recognition of crying [Green et al. 1998] and laugh structure analysis [Szameitat et al. 2007]. Our rates also compare with those used in speech recognition tasks, i.e. between 10 and 20 millisecond windows [Deller et al. 1999]. To represent our audio feature set we use the notation $\mathbf{A} = \{\vec{a}_1, \ldots, \vec{a}_N\}$, where $\vec{a}$ is a column of MFCCs. Spectrograms offer a useful way of visualising example vocal sounds [Deller et al. 1999]. A spectrogram represents how audio frequency density varies with time. Figure 3 shows example spectrograms for a laugh, cry, sneeze and yawn. By observing these spectrograms we can distinguish some regular patterns and structure in the sounds. This is consistent with literature on laugh and cry analysis [Szameitat et al. 2007; Green et al. 1998]. Having structural consistencies across different non-speech sounds is important when attempting to learn relationships with the corresponding visual signal. Figure 3 shows visual parameters with corresponding audio features for some different articulations. Clear structural consistencies are evident when examining such correspondences – suggesting that the relationship between the audio and visual signals for non-speech related articulations is one that can be reliably modelled.

Modelling Audio-Visual Relationships

Observing audio-visual signals for different non-speech related articulations reveals evidence of a temporal structure. Speech literature also provides structural evidence for these articulations – e.g. laughs are commonly based on fricative-vowel template [Szameitat et al. 2007], where the fricative often consists of an aspirated *‘h’* sound and the vowel an *’ah’* sound. Work on cry analysis also reveals common acoustic behaviours [Green et al. 1998], while computer based sneeze and yawn classification has also been proved possible with temporal models [Temko et al. 2006].

Given the evidence of temporal patterns inherent in our data we therefore decided to model this behaviour using HMMs [Rabiner 1989]. A HMM consists of a hidden Markov state process and an observable sequence of events resulting from this process. The term Markov describes the fact that the future state of the system is dependent on the present system state. With an audio signal we may state that the future sound of the audio will depend on the current sound, i.e. the *’h’* of a laugh may be succeeded by a *’ah’* sound. This may be described as a Markov process, and a HMM will tell us the probability of the *’h’* preceding the *’ah’* based on observations from the training data.

The hidden process in the HMM is that which ultimately generates the observable sequence. In the analogy of speech production, if the observable sequence is sound then the hidden sequence may be thought of as the mouth that generated it. However, by default a HMM is trained solely on one type of data, and this data represents its hidden process and observable sequence. It is therefore necessary to modify the basic HMM to model this.

We first consider a traditional HMM trained using visual data. Let us consider this data to be a set of example non-speech sounds from $\mathbf{V}$. For details on HMM training using the Expectation-Maximisation algorithm the reader is referred to [Rabiner 1989]. After training, the HMM may be represented using the tuple $\lambda_v = (\mathbf{Q}, \mathbf{B}, \pi)$, where **Q** is the state transition probability distribution, **B** is the observation probability distribution, and π is the initial state distribution. In our model, each of the K states in a HMM are represented as a Gaussian mixture $G_v = (\mu_v, \sigma_v)$, where μ_v and σv are the mean and covariance. Each state therefore represents the probability of observing a visual vector.

A HMM may be used for three general problems: (1) determining the probability that a specific HMM generated an observation, (2) determining the hidden state sequence responsible for an observation, and (3) determining the HMM parameters given an observation. The second problem may be solved using the Viterbi algorithm [Rabiner 1989]. The Viterbi algorithm uses dynamic programming to find an optimal path (e.g. a state sequence) through a temporal state space given an observation sequence. The observation sequence is considered as being generated by the direct result of moving through the state space, so the Viterbi algorithm attempts to infer the state sequence most likely to have generated the observation. It is not feasible to consider every possible state sequence that could have generated the observation, so therefore dynamic programming is employed.

Therefore, given a set of visual parameters (i.e. observation), we may estimate the underlying state sequence. However, we wish to slightly modify the problem such that we may estimate the visual state sequence given an *audio* observation instead. This is our animation goal, i.e. automatic animation of visual parameters given speech. We can do this by remapping the visual observations to audio ones using the learned HMM parameters, i.e. for each G_v we wish to calculate the distribution $G_a = (\mu_a, \sigma_a)$ based on the audio **A** corresponding to the visual vectors **V** used in HMM training. Thus, we calculate

$$\mu_a^j = \frac{\sum_{t=1}^{N} \gamma_t(j).\vec{a}_t}{\sum_{t=1}^{N} \gamma_t(j)} \tag{3}$$

$$\sigma_a^j = \frac{\sum_{t=1}^{N} \gamma_t(j).(\vec{a}_t - \vec{\mu}_M^j)(\vec{a}_t - \vec{\mu}_M^j)^T}{\sum_{t=1}^{T} \gamma_t(j)} \tag{4}$$

where $1 \leq j \leq K$. The term $\gamma_t(j)$ is the probability of being in state *j* at time *t*, and may be found by recalculating the Expectation-step of the EM algorithm. Details on how to calculate $\gamma_t(j)$ may be found in [Rabiner 1989].

Creating Visual Parameters from Audio

Using the Viterbi algorithm, we may now estimate the most probable visual state sequence using an audio observation. More formally, we can estimate via the HMM the post probable hidden sequence of Gaussian distribution parameters μ_v and σ_v corresponding to the observation sequence of MFCC vectors $\mathbf{A}_{P1,L}$. One way of displaying an animation at this stage is to simply render the sequence of Gaussian means μ_v. However, this will result in an inaccurate animation. Instead, we consider what visual parameters $\vec{v}_t$ may be displayed at each state, and then attempt to select the best one by again considering how probable that visual parameter is given the observation sequence.

We therefore first partition the visual parameter distribution used to train the HMM into distinct regions based on the proximity of a visual parameter to each gaussian. Using μ_v and σ_v, we calculate the Mahalanobis distance between each observation $\vec{v}_i$ and each of the K states and assign a visual parameter to its closest state. This results in K partitions of the parameter training set, and given an audio observation we may now state that the visual parameter to display at time t given $\vec{a}_t$ is taken from the visual parameter partition associated with the state at time t.

We now define some new notation. We term the audio sequence being used to create the new animation as $\mathbf{A}^{new} = \{\vec{a}_1^{new} \dots \vec{a}_T^{new}\}$, and define the sequence of states it generates as $S = \{s_1 \dots s_T\}$, where T is the length of the input audio sequence $\mathbf{A}^{new}$. Using $\mathbf{A}^{new} = \{\vec{a}_1^{new} \dots \vec{a}_T^{new}\}$ and $S = \{s_1 \dots s_T\}$, our aim is now to calculate the new animation parameters $\mathbf{V}^{out} = \{\vec{v}_1^{out} \dots \vec{v}_T^{out}\}$

In order to do this we adapt the Viterbi algorithm such that it provides us with the visual sequence $\mathbf{V}^{out} = \{\vec{v}_1^{out} \dots \vec{v}_T^{out}\}$ most likely to have generated the audio sequence $\mathbf{A}^{new} = \{\vec{a}_1^{new} \dots \vec{a}_T^{new}\}$. We also have the added constraint that the visual parameter displayed at time t must be chosen from the partition of visual parameters associated with state s_t. Figure 4 gives an overview of visual synthesis, and defines it in terms of two levels: High-Level Re-synthesis, and Low-Level Resynthesis. The High-Level stage is concerned with initially selecting the visual state sequence through the HMM given the audio input $\mathbf{A}^{new} = \{\vec{a}_1^{new} \dots \vec{a}_T^{new}\}$. This results in a sequence of visual parameter partitions – one for each time *t*. The Low-Level stage then finds the most probable path through these partitions given the observed audio. In order to use the Viterbi algorithm to solve this low-level stage, we must define two measures: the probability of transitioning between two neighbouring visual vectors, and the probability of observing the audio feature $\vec{a}_t^{new}$ given a possible visual parameter $\vec{v}_t^{out}$. The transition probability is defined using the Mahalanobis distance between two neighbouring visual parameters. The second measure takes advantage of the fact that each visual parameter vector in the current partition also has a corresponding audio vector observed in training, i.e. visual parameter $\vec{v}_i$ in partition s_t also has a corresponding audio vector $\vec{a}_i$. The probability of observing $\vec{a}_t^{new}$ is therefore the Mahalanobis distance between this audio parameter and $\vec{a}_i$.

Once $\mathbf{V}^{out}$ has been calculated this can be converted into an animation sequence of 3D vectors $\mathbf{X}^{out}$ by projecting onto the visual eigenvectors as in [1]. Animations at this point consist of a sequence of 3D points moving in space. Given a set of vectors of this type, there is a multitude of research on methods to use their motion to animate a a vast range of detailed 3D facial models [Lorenzo et al. 2003] – either of human looking characters or of more fantastic ones (e.g. Yoda). Thus, animation vectors of this type have potentially a wide use in video-games (online and offline). We use the method of Edge *et al* [Lorenzo et al. 2003] to animate a variety of different facial models using the vectors $\mathbf{X}^{out}$, and the reader is directed to this paper for implementation details.

Person and Action Identification using HMMs

In a situations such as an online video game, any voice driven animation system should give robust performance for a wide range of potential players voices. The alternative is to train a speaker dependent system for each player, and while this is indeed possible – and may even be practical as long as a player is willing to invest the necessary time – it is more desirable to mimimise the level of person specific training necessary to give person independent robust animation.

Consider the case where a voice driven system contains knowledge

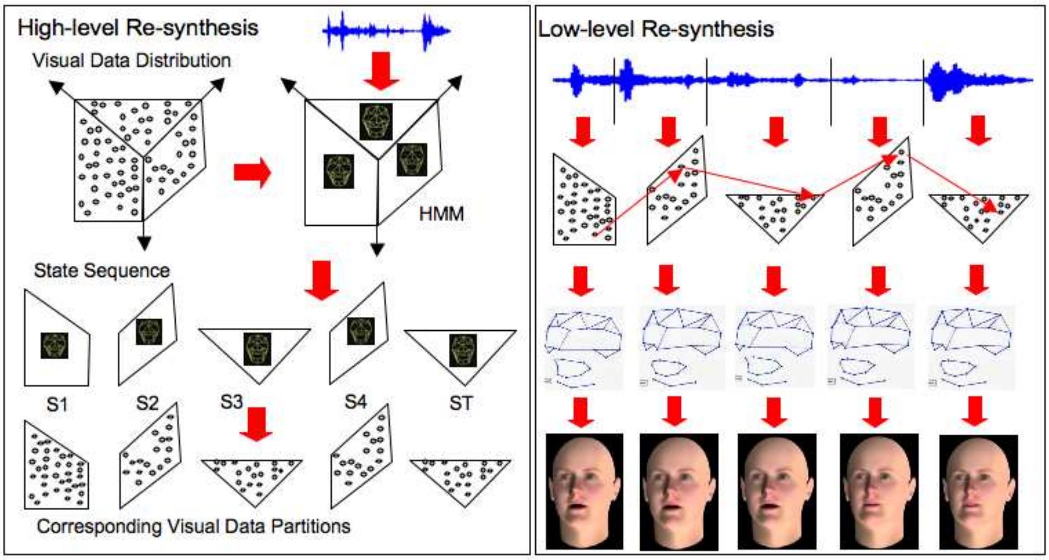

Figure 4: *Animation production may be visualised as a high-level state based process followed by a low-level animation frame generation process.*

of several identities, and this is represented by a set (or basis) of several HMMs – one for each identity. A player will make a non-speech sound – which must first be detected by the system – and then this will be used to synthesise an animation. If the system has not heard the persons voice before then we must synthesise the animation using either (1) a single HMM trained with the knowledge of multiple people, or (2) one of several HMMs where each contains audio-visual data for a specific person. We concentrate on the latter case for now, so our problem is therefore to select one of several HMMs where each encodes information from a specific identity. It turns out that this is equivalent to the general HMM problem number 1 previously described, i.e. determining the probability that a specific HMM generated the observation. Determining this probability may be achieved by estimating the log-likelihood that a HMM could have generated the persons input audio. Calculating the log-likelihood is a simple procedure, but for the sake of space the reader will have to be referred to [Rabiner 1989] for technical details. The log-likelihood calculation takes into account (1)the temporal nature of the input vocal sound, and (2) the overall proximity of the input sound to the distribution used to train the HMM. We show in our results how selecting a HMM with a higher log-likelihood consistently leads to a lower overall animation error.

Experimental Results

Here we perform several experiments to investigate the performance of our approach. The video accompanying this paper provides multiple animation results – including difficult cases of animating a model given a persons voice that the system has not been trained on. However, here we first consider animation using known voices.

Animating with Known Voices

We first consider person and action specific synthesis of animations. We trained audio-visual HMMs for a range of specific non-speech actions – laughing, crying, sneezing and yawning – for each of our four performers. Each HMM was trained using approximately 4 different actions, and approximately 4 more were left out for the test cases. Audio corresponding to the test cases was then used to synthesise new 3D animation vectors which were compared to the motion-capture ground truth. Example animations may be found in the video, and RMS errors in millimeters may be found in Table 1. We trained each HMM using $K = 10$ states, we determined this value by observing the overall average RMS value for different values of K and selected in the best. The overall animation errors are very low, the worst average marker error being 4.55 millimetres. Perceptually, animations are synched to the audio and closely match the ground truth motion.

We next tested combining data from multiple people performing a specific non-speech action inside the same HMM. This assesses the models ability to generalise data for different people within the same model. Again, we left out part of the data for each performer to use as a test-set and calculated RMS errors as shown in Table 2. This time we used a value of $K = 20$. Errors are comparable to those in Table 1, demonstrating that the HMM is capable of combining non-speech data for specific actions across multiple people while still allowing accurate animations to be produced.

Animating with Unknown Voices

We now test the case where the model has no prior knowledge of a persons voice. This is perhaps the most challenging task in audio driven speech synthesis, and is key for applications such as online video games.

For each performer we trained four separate HMMs – one for each action. Given input audio for an action, the HMM with the best log-likelihood was selected for synthesis – thus taking into account match between input audio distribution and those of the trained HMMs. Table 3 shows the results, and Figure 5 gives side-by-side comparisons between ground truth video data of a performer, reconstructed 3D vectors, and an animated 3D facial model. Our results clearly show that a HMM with a higher log-likelihood always gives a lower average error reconstructions error. This shows that a high log-likelihood appears correlated with a low animation error. The RMS errors are in fact comparable with the person and action specific HMM results in Table 1, which is an extremely favorable result

Person		Laugh			Cry			Sneeze			Yawn	
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
P1	0.7	1.42	0.95	0.89	2.3	1.36	0.6	1.5	1.99	1.8	5.1	2.49
P2	2.68	4.7	3.68	1.96	2.42	2.12	3.8	5.6	4.56	1.99	4.13	2.8
P3	0.93	1.49	1.19	1.57	2.25	1.96	0.6	0.92	0.92	ND	ND	ND
P4	1.75	2.16	1.92	1.11	1.4	1.24	1.57	2.52	2	3.74	5.8	4.55

Table 1: *Action Specific HMM animation: Min, Max and Mean RMS errors (millimetres) for average synthesised 3D coordinates versus ground truth 3D coordinates.*

Person		Laugh			Cry			Sneeze			Yawn	
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
P1+P2+P3+P4	1.15	2.76	1.75	1.29	3.61	2.01	1.6	5.96	3.52	1.77	6.15	3.52

Table 2: *Animation with HMMs encoding multiple actions: Min, Max and Mean RMS errors (millimetres) for average synthesised 3D coordinates versus ground truth 3D coordinates.*

since person and action specific modelling should always provide the baseline of best possible performance. Table 3 also shows results given a HMM with the lowest log-likelihood, and in general these show that even the worst matching HMM can provide a reasonable error and good animation results. However, most cases do indeed show that using the wrong HMM for synthesis (i.e. one with the lowest log-likelihood) can lead to substantially higher reconstruction error, e.g. P1 yawn results. The results therefore indicate that using a selection process offering a range of different possible HMM models for synthesis can markedly improve performance. Overall the results show that for the actions selected, a system for non-speech action synthesis can be made quite robust to different voices, and that a multiple HMM selection scheme can further improve robustness and reliability. Another interesting result is how robust the system is to voices from different sexes. We found that given a male voice as an input, 44% of the time the best selected HMM (one with the highest log-likelihood) was one also trained on a male voice, while 70% of the time the worst selected HMM was one trained on a female voice. This appears to show that for male input voices the best synthesis result may equally come from a prior male or female voice trained synthesis model. Results for female input voices are far more clear cut. Given a female input voice, the best selected HMM was one trained on a female voice 83% of the time, and the worst selected HMM was one trained on a male voice 100% of the time. This shows strong evidence that a voice driven animation system for non-speech actions should always have some training using female voices.

Conclusions and Future Directions

In this paper we have presented a data-driven model for automatically generating facial animation using non-speech sounds such as laughing, crying, sneezing and yawning. Animation of these articulations is important in many potential applications, such as online video games where players communicate online via headsets. We have demonstrated our method to naturally encode a strong degree of speaker independence, and have also shown how a 'best-HMM' classification technique can further improve robustness. Using this technique, the overall animation error is comparable to that obtained in person and action specific animation. We find that animations created using female voices will typically rely on HMMs trained specifically using female audio-visual data, whereas male voices will use male or female prior information with an almost equal chance. The implication of this is that any pre-trained system used in a real application should contain both prior male and female audio-visual information. There is a range of future work to be done in voice-based synthesis of non-speech actions, this includes broadening the general type of action that can be animated, incorporating head motion into synthesis, and combining these actions with speech.

References

Blanz, V., and Vetter, T. 1999. A morphable model for the synthesis of 3d faces. In *Proc. of SIGGRAPH*.

Brand, M. 1999. Voice puppetry. In *Proc. of SIGGRAPH*, ACM Press, 21–28.

Bregler, C., Covell, M., and Slaney, M. 1997. Video rewrite: driving visual speech with audio. In *Proc. of SIGGRAPH*, ACM Press, 353–360.

Cao, Y., Tien, W. C., Faloutsos, P., and Pighin, F. 2005. Expressive speech-driven facial animation. *ACM Trans. on Graphics 24*, 4, 1283–1302.

Chuang, E., Deshpande, H., and Bregler, C. 2002. Facial expression space learning. In *Proc. of Pacific Graphics*.

Cohen, M., and Massaro, D. 1993. Modelling coarticulation in synthetic visual speech. In *Proc. of Methods and Techniques in Computer Animation*.

Cootes, T., Edwards, G., and Taylor, C. 2001. Active appearance models. *IEEE Trans. PAMI 23*, 6, 681–684.

Cosker, D., Marshall, D., Rosin, P. L., and Hicks, Y. 2004. Speech driven facial animation using a hidden markov co-articulation model. In *Proc. of IEEE ICPR*, vol. 1, 128–131.

Cosker, D., Holt, C., Mason, D., Whatling, G., Marshall, D., and Rosin, P. L. 2008. Automatic audio driven animation of non-linguistic vocalisations. In *Proc. of CMBBE*.

Deller, J., Hansen, J., and Proakis, J. 1999. *Discrete-Time Processing of Speech Signals*. Wiley-IEEE Press.

Deng, Z., Neumann, U., Lewis, J. P., Kim, T., Bulut, M., and Narayanan, S. 2006. Expressive facial animation synthesis by learning speech coarticulation and expression spaces. *IEEE Trans. on Visualization and Computer Graphics 12*, 6.

DiLorenzo, P., Zordan, V., and Sanders, B. 2008. Laughing out loud: Control for modelling anatomically inspired laughter using audio. *ACM Trans. Graphics 27*, 5.

Entertainment, B. World of warcraft.

Ezzat, T., Geiger, G., and Poggio, T. 2002. Trainable video realistic speech animation. *ACM Trans. on Graphics 21*, 3.

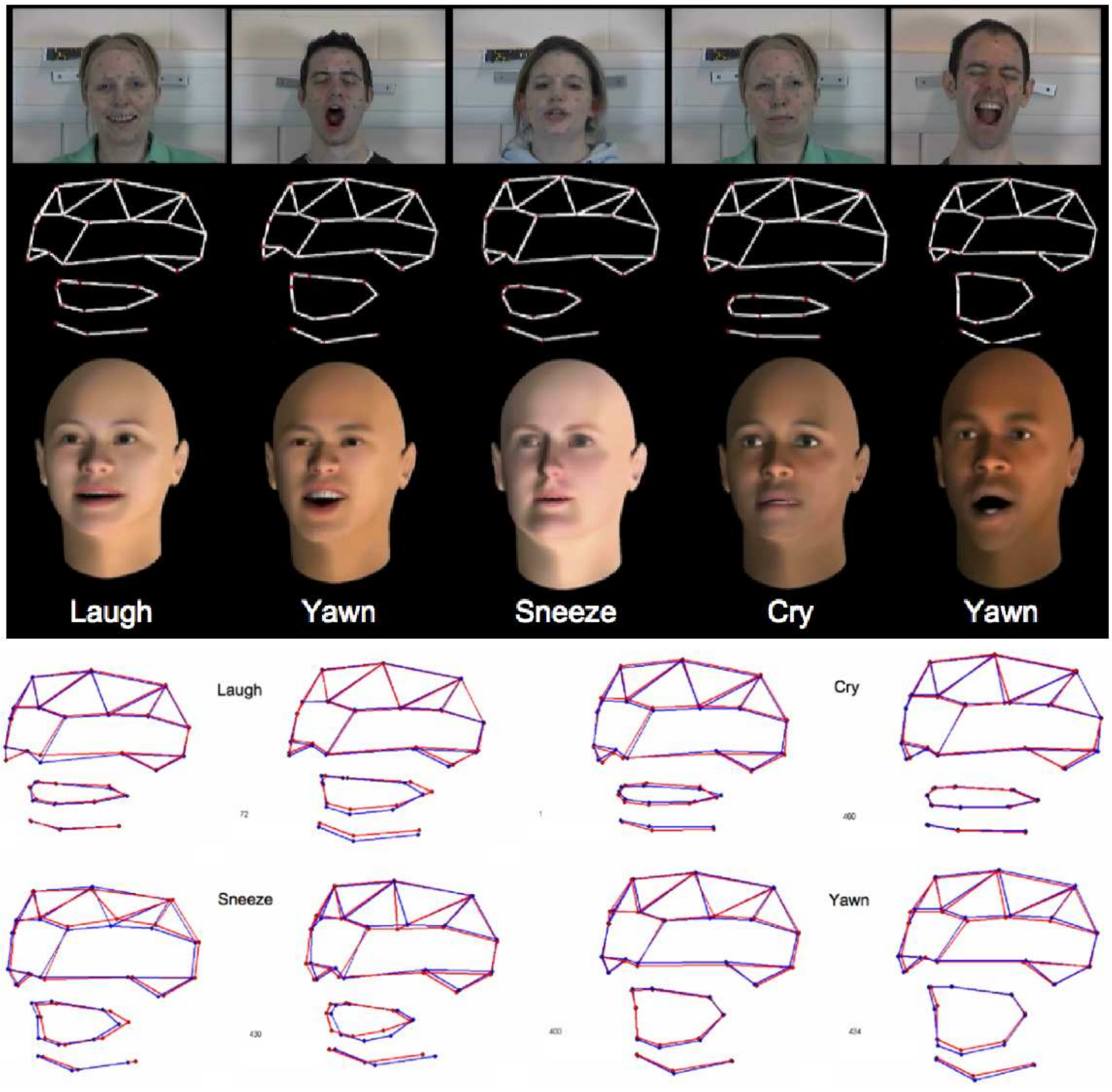

Figure 5: *Example Animation Frames. (Black Region - Top) Ground truth video. (Black Region - Middle) Corresponding 3D Motion vectors automatically synthesised from speech. (Black Region - Bottom) A 3D head model animated using the motion vectors using an RBF mapping technique. (White Region) 3D motion vectors for different non-speech actions compared to ground truth motion capture vectors.*

FORDHAM, J. 2003. Middle earth strikes back. *Cinefex*.

GREEN, J., GUSTAFSON, G., AND MCGHIE, A. 1998. Changes in infants' cries as a function of time in a cry bout. *Child Development 69*, 2.

GUTIERREZ-OSUNA, R., KAKUMANU, P., ESPOSITO, A., GARCIA, O., BOJORQUEZ, A., CASTILLO, J., AND RUDOMIN., I. 2005. Speech-driven facial animation with realistic dynamics. *IEEE Trans. on Multimedia 7*, 1.

KSHIRSAGAR, S., AND MAGNENTAT-THALMANN, N. 2003. Visyllable based speech animation. *Computer Graphics Forum 22*, 2, 631–639.

LORENZO, M. S., EDGE, J., KING, S., AND MADDOCK, S. 2003. Use and re-use of facial motion capture data. In *Proc. of Vision, Video and Graphics*, 135–142.

PIGHIN, F., AND LEWIS, J. 2006. *Performation Driven Facial Animation*. Siggraph Courses.

RABINER, L. R. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. of the IEEE 77*, 2 (February), 257–285.

SZAMEITAT, D., DARWIN, C., SAZMEITAT, A., WILDGRUBER, D., STERR, A., DIETRICH, S., AND ALTER, K. 2007. Formant characteristics of human laughter. In *Proc. of Workshop on the Phonetics of Laughter, Saarbruken*.

TEMKO, A., MACHO, D., AND NADEU, C. 2006. Improving the performance of acoustic event classification by selecting and combining information using the fuzzy integral. *LNCS 3869*, 357–368.

UMEYAMA, S. 1991. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Mach. Intell. 13*, 4, 376–380.

	Laugh		Cry		Sneeze		Yawn	
	B / W Err	B/W Log	**B / W Err**	B / W Log	**B / W Err**	B / W Log	**B / W Err**	B / W Log
P1	**2 / 3.3**	-1033 / -1126	**2.08 / 2.45**	-704 / -851	**2.4 / 2.46**	-749 / -1105	**2.8 / 6.6**	-607 / -1239
P2	**2.3 / 2.5**	-422 / -777	**1.3 / 2.2**	-662 / -1130	**3.4 / 3.66**	-748 / -1006	**3.5 / 5.4**	-1081 / -1281
P3	**1.6 / 2.8**	-763 / -2558	**1.1 / 2.1**	-857 / -3519	**2.4 / 2.9**	-1050 / -2352	ND	ND
P4	**1.7 / 3**	-1085 / -2039	**0.8 / 2.1**	-770 / -1804	**1.5 / 2.7**	-902 / -1627	**1.8 / 2.8**	1073 / 1215

Table 3: *Average 3D vector animation error (millimeters) given best and worst matching (log-likelihood) HMMs. (B/W Err = best/worst error, B/W Log = best/worst log-likelihood)*