MODELING BODY LANGUAGE FROM SPEECH IN NATURAL CONVERSATION

A RESEARCH REPORT
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Sergey Levine
Principal Adviser: Vladlen Koltun
Secondary Adviser: Sebastian Thrun
May 2009

Abstract

This report presents a framework for studying the relationship between speech prosody and motion of people in conversation. Several probabilistic models for the relationship between prosodic features in speech and the dynamics of body language are evaluated. These models are then applied to the task of synthesizing new body language animations for a novel utterance. The models are evaluated based on their ability to synthesize those aspects of motion that are hypothesized to correspond well to prosody. Inference is performed both offline, with full knowledge of the entire utterance, and online, with only the content of the utterance up to the current time step available for inference. Online inference of motion parameters allows for synthesis of body language animations from live speech input.

Acknowledgements

First, I would like to thank Vladlen Koltun and Sebastian Thrun for their help and advice over the course of this project. Their patience, ideas, and support were invaluable, and this project would not have been possible without them.

I would also like to thank PhaseSpace Motion Capture for generously lending their facilities for the motion capture session, as well as Sebastián Calderón Bentin for his part as the main motion capture actor.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

Human communication makes extensive use of body language to help convey ideas, express emotion, and add emphasis to important words or phrases [37, 26]. However, despite the importance of body language to natural communication, the precise relationship between motion and speech prosody is poorly understood [32]. In this report, I address the problem of finding and modeling correspondences between prosodic features in speech and the dynamics of the accompanying body language. These correspondences are of interest for several useful applications.

Characters in networked virtual environments and video games often lack the sort of expressive body language that characterizes human communication. Automatic synthesis of these body language animations would benefit significantly from a better understanding of the relationship between speech prosody and body language. Existing methods already address the problem of automatically animating characters from *text* [12, 42], but the problem of animating such characters directly from speech has not received as much attention and has been limited to animation of the face, head motion, or a limited gesture "vocabulary" [8, 46, 45, 14]. The poor understanding of the relationship between speech prosody and gesture presents a serious challenge to any such system.

Another related area is recognition of gesture, which has numerous potential uses for

improving human-computer interfaces, but has not been readily adopted due to poor recognition quality and reliability. Previous work has shown that gesture recognition can be improved with the incorporation of prosodic features in co-occurring speech [27]. While this report does not address gesture recognition directly, I discuss possible applications of the proposed models to gesture recognition in Section 8.1.

To demonstrate the usefulness of the proposed probabilistic models, I present a system that synthesizes appropriate body language animations for a live speech signal, without the ability to look ahead in the utterance. This addresses the issue of animating characters not just for offline animation, but also for interactive applications involving live human speakers.

In constructing the model, the analysis is restricted to the dynamics of gestures and body language, rather than form. As discussed in the next section, the form of a gesture often depends on the semantic meaning of the co-occuring speech, while the dynamics are more closely related to vocal prosody. Semantic meaning is technically challenging to extract, and even more difficult to predict in an online setting. On the other hand, the dynamics of gesture, which we can obtain from prosody, establish our perception of the rhythm and tone of the motion, and can add significantly to the believability of a character [32].

1.2 Gesture, Body Language, and Speech

Perhaps the most widely accepted classification for gestures was proposed by McNeill [37]. McNeill himself later revised this taxonomy [38] to be less discrete, but the original classification remains the more popular framework. This classification separates gestures into four categories: iconic gestures, which represent concrete objects by presenting their likeness, metaphoric gestures, which represent abstract concepts, often presented as the likeness of an appropriate *metaphor* (such as the “cup of knowledge”), deictic gestures, which represent spatial relationships and often manifest themselves as pointing, and beats, which are formless flicks meant to emphasize key phrases or signal the introduction of new ideas.

McNeill suggests that gesture and speech are the result of a single cognitive process

that plans speech and gesture simultaneously to deliver the desired communication [37]. Unfortunately, McNeill does not address the relationship between speech and motion in detail. In discussing beat gestures, McNeill notes that "beats tend to have the same form regardless of content" and are often used to convey emphasis. Other types of gestures often accompany the introduction of new ideas or concepts into a narrative. This suggests that body language may be closely associated with those features of speech that signal emphasis and the introduction of new topics.

Vocal prosody is the natural candidate for such features, since it has been previously employed to automatically segment speech by topic and detect emphasis [56, 51] and corresponds well to the rhythm of both speech and accompanying motions [32]. Prosody is conveyed primarily by the pitch (fundamental frequency), intensity, and duration of speech [47].

Another advantage of employing vocal prosody is that it carries much of the emotive content of speech [1, 52]. This emotional dimension tends to be completely absent from the raw text of the utterance, but is important for lending body language, and human motion in general, the appropriate style [15, 24].

Therefore, while there is no direct connection between motion *form* and prosody [32], the correspondences to motion *dynamics* are complex and extensive. In Chapter 3, I discuss the various motion parameters I investigated for correspondences to prosody, and in Chapter 7, I apply these synthesized parameters to the task of body language synthesis for a novel utterance.

1.3 Related Work on Gesture Synthesis

A number of methods have been devised that synthesize either full-body or facial animations from a variety of inputs. Such methods often aim to animate Embodied Conversational Agents (ECAs), which operate on a pre-defined script or behavior tree [10], and therefore allow for concurrent planning of synthesized speech and gesture to ensure co-occurrence. Often these methods rely on the content author to specify gestures as part of the input, using a concise annotation scheme [21, 29]. New, more complete annotation schemes for gestures are still being proposed [28], and there is no clear consensus on how

gestures should be specified. Some higher-level methods also combine behavioral planning with gesture and speech synthesis [11, 43], with gestures specified as part of a scripted behavior. However, all of these methods rely on an annotation scheme that concisely and completely specifies the desired gestures. Stone et al. [54] avoid the need for annotation of the input text with a data-driven method, which re-arranges pre-recorded motion capture data to form the desired utterance. However, this method is limited to synthesizing utterances made up of pre-recorded phrases, and does require the hand-annotation of all training data. Like Stone et al., I also approach gesture synthesis with a data-driven method, but focus instead on aspects of gesture that can be effectively decoupled from the meaning of speech and do not depend on explicit annotations. Since the system must scale gracefully to large amounts of training data, it also employs automatic segmentation to avoid the need for any manual processing.

Several methods have been proposed that operate on arbitrary input, such as text. Cassell et al. [12] propose an automatic rule-based gesture generation system for ECAs using natural language processing, while Neff et al. [42] use a probabilistic synthesis method trained on hand-annotated video. However, both of these methods rely on concurrent generation of speech and gesture from text. Text does not capture the emotional dimension that is so important to body language, and neither text communication nor speech synthesized from text can produce as strong an impact as real conversation [23].

Animation directly from voice has been explored for synthesizing facial expressions and lip movements, generally as a data-driven approach using some form of probabilistic model. Bregler et al. [9] propose a video-based method that reorders frames in a video sequence to correspond to a stream of phonemes extracted from speech. This method is further extended by Brand [8] by retargeting the animation onto a new model and adopting a sophisticated hidden Markov model for synthesis. Hidden Markov models are now commonly used to model the relationship between speech and facial expression [31, 60]. Other automatic methods have proposed synthesis of facial expressions using more sophisticated morphing of video sequences [19], physical simulation of muscles [53], or by using hybrid rule-based and data-driven methods [4].

Although speech-based synthesis of facial expressions is quite common, it is rarely extended to the synthesis of gestures. Since facial expressions are dominated by mouth

movement, the majority of speech-based systems rely on phoneme extraction and select individual facial poses at each frame. However, a number of methods have been proposed that use speech prosody to model expressive human motion beyond lip movements. Albrecht et al. [2] use prosody features to drive a rule-based facial expression animation system, while Chuang and Bregler [14] use a data-driven, pitch-based system to augment a facial animation with appropriate head motion. Incorporating a more sophisticated model, Sargin et al. [45] use prosody features to directly drive head orientation with an HMM. Although these methods animate only head orientation from prosody, Morency et al. [40] suggest that prosody may also be useful for predicting gestural displays.

1.4 Related Work on Gesture Recognition

Gestures are fundamentally a multi-modal entity, and are closely coupled to the accompanying speech [37]. In fact, recognition of gestures by both machines and human observes can be improved significantly by considering the accompanying speech [18]. While the majority of work on gesture recognition has focused on recognizing gestures purely from the motion of the body, a number of works have addressed the relationship between speech and gesture. Co-occuring speech, including prosody, has been used to improve gesture recognition for human-computer interfaces. Speech has been used to improve recognition accuracy by modulating the inferred probability of a gesture occurrence from keywords [49] or prosody [27]. It has also been used to create a richer user interface that combines gestural and speech input, with heuristics used to align gestural displays with paired utterances [39].

While these approaches to gesture recognition attempted to analyze co-occuring speech, others have incorporated more sophisticated machine learning techniques to improve recognition rates. Gesture recognition is generally accomplished with a generative latent model, such as a hidden Markov model (HMM) [59]. However, hidden conditional random fields (HCRFs) have recently been applied to the problem with some success [58], suggesting that a discriminative rather than a generative model may be appropriate. I consider both generative and discriminative models for the task of modeling relationships between speech and motion, and address the strengths and weaknesses of each approach in Chapter 5.

Chapter 2

Training Data

2.1 Motion Capture

In order to train the probabilistic models discussed in Chapter 5, I use a training set consisting of 12 minutes of motion capture data from a real person in conversation with accompanying speech, as well as a test set consisting of 7 minutes of motion capture data from an unrelated conversation. Both scenes are excerpted from an extemporaneous conversation, ensuring that the body language is representative of real human interaction. The motion capture was instructed to avoid extraneous, non-communicative movement, such as scratching, but no instructions were given as to the topic of the conversation. The training data was excerpted from a conversation on politics, while the test set is excerpted from a conversation on acting, in order to ensure that the performance of the models is not artificially inflated by similarity in the conversation topics.

The skeleton used in the system contained 14 joints: three in each arm (shoulder, elbow, wrist), three in each leg (hip, knee, foot), a pelvis, an abdomen, and a neck. The orientations for these joints were computed from marker positions using the Autodesk MotionBuilder software package.

2.2 Motion Segmentation

Many of the parameters that will be discussed in Chapter 3 operate over coherent segments of motion. In the context of body language, it is natural to identify coherent gesture subunits as segments. Much of the existing work on motion segmentation attempts to identify highly dissimilar motions within a corpus of motion capture data, such as walking and sitting [3, 41]. Perceptually distinct gesture subunits are not as dissimilar, and require a more sensitive algorithm. Gesture units consist of the "pre-stroke hold," "stroke," and "post-stroke hold" phases [37, 26], which provide natural boundaries for segmentation of gesture motions. Such phases have previously been used to segment hand gestures [35].

To segment the motion capture data into gesture unit phases, boundaries are placed between strokes and holds using an algorithm inspired by [20]. Since individual high-velocity strokes are separated by low-velocity pauses, we may detect these boundaries by looking for dips in the average angular velocities of the joints. For each frame of the motion capture data, let ω_j be the angular velocity of joint j, and let u_j be a weight corresponding to the perceptual importance of the joint. This weight is chosen to weigh high-influence joints, such as the pelvis or abdomen, higher than smaller joints such as the hands or feet. We can then compute a weighted average of the angular velocities as following:

$$z = \sum_{j=1}^{14} ||\omega_j||^2 u_j.$$

A stroke is expected to have a high z value, while a hold will have a low value. Similarly, a pause between a stroke and a retraction will be marked by a dip in z as the arm stops and reverses direction. Therefore, segment boundaries are inserted when this sum crosses an empirically determined threshold. To avoid creating small segments due to noise, the algorithm only creates segments which exceed either a minimum length or a minimum limit on the integral of z over the segment. To avoid unnaturally long still segments during long pauses, an upper bound is placed on segment length.

Chapter 3

Motion Parameters

3.1 Desired Properties of Motion Parameters

In order to model some aspect of body language in terms of vocal prosody, it is first necessary to select which aspect of body language to model. In Section 1.2, I note that prosody is unlikely to provide information about the form of individual motions, but is informative about their dynamics, since it is a good indicator of emphasis, rhythm, and pacing. Therefore, we would like to select parameters that capture dynamics without being overly dependent on form.

Ideally, these parameters would, taken together, encompass all of the information that we can obtain about motion from prosody. It is also desirable to use parameters that can be computed automatically, in order to avoid the need for costly hand-annotation of training data and to enable the use of extremely large training sets. It is also desirable to select parameters that are invariant across different speakers, so that the system can generalize well to synthesis and recognition tasks for a variety of individuals. However, normalization of motion parameters across speakers is outside of the scope of this report and is left for future work. In the next section, I examine several possible sets of candidate parameters from the current literature on gesture and movement analysis.

Effort	dynamic qualities of movement
Space	attention to the surroundings
Indirect	spiraling, flexible, wandering
Direct	straight, undeviating, channeled
Weight	attitude to the movement impact
Light	buoyant, weightless, easily overcoming gravity
Strong	powerful, forceful, having an impact
Time	lack or sense of urgency (*not* necessarily speed)
Sustained	leisurely, lingering
Sudden	hurried, urgent, quick
Flow	amount of control and bodily tension
Free	uncontrolled, abandoned, unable to stop
Bound	controlled, restrained, rigid

Table 3.1: *LMA qualities associated with Effort. The qualities are often described as continuous-valued, ranging between the two elements [61, 13].*

3.2 Candidate Parameters

Perhaps the most well-established system for annotating human motion is Laban Movement Analysis (LMA), which has received considerable attention from the animation community [61, 62, 33]. The Effort and Shape factors provide a standardized method for describing the *style* of a motion. Style has been applied to animation by interpolating between samples with different LMA factors [33], and there has been work on automatically extracting Effort parameters from motion capture data [62]. Effort describes motion in terms of four qualities, each of which consists of a pair of opposing elements. Each motion can be described by an assignment to each quality on a scale between the two elements. The qualities are Space, Weight, Time, and Flow, and the corresponding elements are Indirect/Direct, Light/Strong, Sustained/Sudden, and Free/Bound. Table 3.1 summarizes the qualities of Effort and the corresponding elements. In general, selecting the Effort parameters of a specific motion requires annotation by a Laban Movement Analysis expert, although it may be possible to extract the parameters automatically using a neural network trained on directly computable features [62].

Although LMA is the most well-established system for describing the dynamics of a

motion, it is not designed specifically for body language. Other methods have been proposed that deal more directly with gesture or body language. One of the earliest attempts at describing gesture dynamics was made by David Efron in his analysis of the body language of ethnic groups in New York City [17]. Efron described gesture in terms of the physical extent of the stroke and an adjective to describe the manner in which the stroke was performed (i.e., its dynamics). This manner could be, for example, "sinusoidal" or "angular." Efron was the first to present a large body of quantitative data using this somewhat ad-hoc annotation scheme.

More recently, there has been a renewed effort to produce a concise and complete gesture annotation scheme for the purpose of synthesizing gesture animations. Kipp et al. proposed a scheme for annotating gesture with the explicit intent to capture *form* [28]. In discussing possible ways to extend their annotation system to describing gesture dynamics, Kipp et al. propose focusing on form at several key points during the gesture (beginning, middle, and end). This sort of annotation scheme is overly reliant on form and therefore inappropriate for the proposed application. A more style-centric system is proposed by Hartmann et al. [22]. This system describes the style of gesture dynamics in terms of "fluidity," and "power," which correspond to the smoothness of the position curve and the magnitude of the acceleration, respectively. Additional parameters, like spatial and temporal extent, are used to describe the overall properties of a gesture.

In attempting to automatically extract LMA Effort parameters from motion, Zhao and Badler propose yet another set of motion features [62]. Their proposed neural-network-based classifier uses the temporal extent of a motion, displacement, mean velocity, acceleration, curvature, and torsion of the trajectory of the hand, as well as shoulder swivel angles, wrist angles, and sternum height. These parameters are all easily computable from motion capture data and, according to Zhao and Badler, are sufficient to infer LMA Effort parameters. Therefore, we can conclude that this set of computable parameters is sufficient to capture all of the dynamics information carried by LMA Effort. Since they also encompass most of the parameters described by Hartmann et al. [22], the final set of motion parameters is based on these computable features.

Since these motion parameters are generally meaningful only over an entire motion segment, the system uses the gesture unit phase segments described in Section 2.2 as the

segments for parameter extraction.

3.3 Automatic Parameter Extraction

The final set of parameters used in the model is based on the parameters selected by Zhao and Badler for inferring LMA Effort parameters [62]. Temporal extent, displacement, velocity, acceleration, and curvature are used. These parameters will be denoted $\bar{T}, \bar{D}, \bar{V}, \bar{A}$ and $\bar{C}$ respectively.

As suggested by Zhao and Badler, the positions of the hands are used for computing all of the parameters [62]. All features except for temporal extent are presented on a logarithmic scale to better reflect the perceptual similarity between more extreme values: for example, two very fast motions may appear more similar than two slower motions even when the difference in the magnitude of the mean velocity between the faster segments is greater.

Temporal extent is computed simply by using the start and end times of the current segment, t_1 and t_n:

$$\bar{T} = t_n - t_1.$$

Displacement is computed as the maximum extent of the displacement from the segment's initial pose, similar to the measure of displacement proposed by Efron [17]. The maximum displacement is converted to a logarithmic scale as discussed above:

$$\bar{D} = \log || \max_{i \in [2,n]} p_i - p_1 ||.$$

Where p_i is the configuration of the relevant joint positions at the i^{th} time step, given as a six-dimensional vector for the upper and lower body, with three dimensions for each joint, and a three-dimensional vector for the head.

We can compute the velocity and acceleration at a given time step i by using a finite differences approximation as following:

$$v_i = \frac{p_i - p_{i-1}}{t_i - t_{i-1}},$$

$$a_i = \frac{v_i - v_{i-1}}{t_i - t_{i-1}}.$$

With acceleration and velocity computed at each time step, $\bar{V}$, $\bar{A}$, and $\bar{C}$ may be computed as the log of the average values of velocity, acceleration, and curvature:

$$\bar{V} = \log \left|\left| \frac{1}{n} \sum_{i=2}^{n} v_i \right|\right|,$$

$$\bar{A} = \log \left|\left| \frac{1}{n} \sum_{i=2}^{n} a_i \right|\right|,$$

$$\bar{C} = \log \left|\left| \frac{1}{n} \sum_{i=2}^{n} v_i \times a_i \right|\right|.$$

These five parameters are used in the remainder of the report as the motion features synthesized from speech prosody.

Chapter 4

Audio Processing

4.1 Prosody Extraction

The training data for the probabilistic models includes both motion capture data and the accompanying audio track. The system extracts prosody features from the audio track using components of the Praat speech analysis tool [6]. Pitch is extracted using the autocorrelation method, as described in [5], and intensity is extracted by squaring waveform values and filtering them with a Gaussian analysis window.

Pitch and intensity vary considerably over a single syllable due to the phonetics of the utterance. In order to extract useful prosody information, the prosody features must be filtered over a sufficiently large window to reduce the effect of phonetics on feature values. A natural choice for this window is the syllable, since fluctuations in prosody due to phonetics are usually confined to a single syllable, while lower-frequency fluctuations are more often due to sentence structure and emphasis – factors which are likely to influence motion. Therefore, the system segments the audio stream by syllables in order to then average feature values over these segments.

Pitch and intensity have previously been used to drive expressive faces [14, 25] and have been observed to correlate well to emotional state [48]. Therefore, pitch and intensity are the basis of the prosody features used by all of the synthesis methods described in this report. Specifically, intensity, pitch, and the rate of change of pitch are averaged over syllables and used as features in all of the presented probabilistic models. Pitch will be denoted

F_0, the rate of change of pitch $\frac{dF_0}{dt}$, and intensity I. It should be noted that these features are not invariant across speakers and channels, and therefore would need to be normalized on a per-speaker and per-channel basis in order for the system to be truly portable. While this is not technically challenging, it may require a "calibration period" during online synthesis to compute a speaker's normalization parameters.

4.2 Syllable Segmentation

For syllable segmentation, I employed a simple algorithm inspired by Maeran et al. [34], which identifies peaks separated by valleys in the intensity curve, under the assumption that distinct syllables will have distinct intensity peaks. The peak of a syllable is detected as the highest peak within a fixed-size window, and segment boundaries are inserted at the lowest trough between two peaks. Segment boundaries are also inserted at voiced/unvoiced transitions, and when the intensity falls below a fixed silence threshold.

For online processing during synthesis, the signal can be segmented progressively in real time. In this case, the algorithm identifies the highest peak within a fixed radius, and continues scanning the intensity curve until the next local maximum is found. If this maximum is at least a fixed distance away from the previous one, the previous peak is identified as the peak of the syllable terminating at the lowest point between the two peaks.

Chapter 5

Probabilistic Gesture-Speech Models

5.1 Overview

In order to synthesize motion parameters from speech prosody features, an appropriate model must be selected that respects the dependencies between the two sets of parameters. Several models are evaluated to determine the one that produces the best results.

The problem of motion parameter synthesis can be formally stated as follows: given an input sequence $x = (x_1, x_2, ..., x_T)$ of speech prosody features, we must construct the output sequence $y^{(s)} = (y_1^{(s)}, y_2^{(s)}, ..., y_T^{(s)})$ of motion parameters that most closely resembles the correct output sequence $y^{(g)}$. Metrics for evaluating the similarities between $y^{(s)}$ and $y^{(g)}$ are discussed in Section 6.1. As discussed in Section 3.3, the vector y_t consists of the motion parameters and may be written $y_t = (\bar{T}, \bar{V}, \bar{D}, \bar{A}, \bar{C})^T$. Similarly, the vector x_t consists of the speech prosody features discussed in Section 4.1 and may be written $x_t = (F_0, \frac{dF_0}{dt}, I)^T$.

We first assume that $y_t^{(s)}$ depends on the sets $C_x(t) = \{x_t, x_{t-1}, ..., x_{t-c_x+1}\}$ and $C_y(t) = \{y_{t-1}^{(s)}, y_{t-2}^{(s)}, ..., y_{t-c_y}^{(s)}\}$, where $C_x(t)$ is the input context of size c_x and $C_y(t)$ is the output context of size c_y. Since we would like the model to handle live input, we cannot rely on any context from the future. The possible values c_x and c_y indicate what type of model is most appropriate. For example, if $c_y = 0$, then each time step may be considered as a separate regression problem, which can be solved with any of the popular regression techniques. If $c_y = 1$, y satisfies the Markov property and, if c_x is small, can be modeled with a

conditional random field (CRF) [30] or maximum-entropy Markov model (MEMM) [36]. If $c_y > 1$ or c_x is large, we can attempt to model the formation of $y^{(s)}$ with a hidden process q. If trained correctly, the hidden process allows the model to carry forward the relevant context about past states. For example, if $C_y(t)$ provides n bits of relevant context, then a hidden variable q_t with 2^n possible values can provide the same information content. Since q_t carries all of the relevant context from the past, the value of q_{t+1} depends only on q_t and $y^{(s)}_{t+1}$. This formulation motivates an output-input remapped hidden Markov model (HMM) [8]. It should be noted, however, that complex or long-range relationships may be difficult to learn, so a simple HMM may not always be sufficient.

In the following sections, I describe the technical details of several probabilistic models which may be appropriate for synthesizing $y^{(s)}$. I then present numerical results that allow us to compare the relative performance of each model, and discuss what their performance implies for the values of c_x and c_y. I also discuss various techniques for further improving the quality of the synthesized sequence.

5.2 Remapped and Jointly-Trained HMMs

A common approach to synthesizing animation from speech is to model the relationship using a hidden Markov model (HMM) [8, 31, 60, 45]. The hidden Markov model models the signal y as being generated by a sequence of hidden states q, which themselves constitute a Markov processes, in which q_{t-1} carries all of the necessary context to select q_t based only on the observations at time step t. This structure affords efficient training and inference, as described in Rabiner's HMM tutorial [44]. The model can be adapted to synthesizing an output signal y from some input signal x by training the model on the output signal using the Expectation Maximization (EM) algorithm [44] and *remapping* the observations into the input signal by estimating $P(x_t|q_t)$ [8]. The HMM is evaluated with both a discrete and a continuous observation model. The discrete observation model clusters the input and output into a fixed number of clusters, while the continuous model uses a Gaussian distribution to represent observation probability.

Remapping assumes that the output and input sequences have roughly the same structure [31], which may result in poor performance if this is not the case. An alternative model

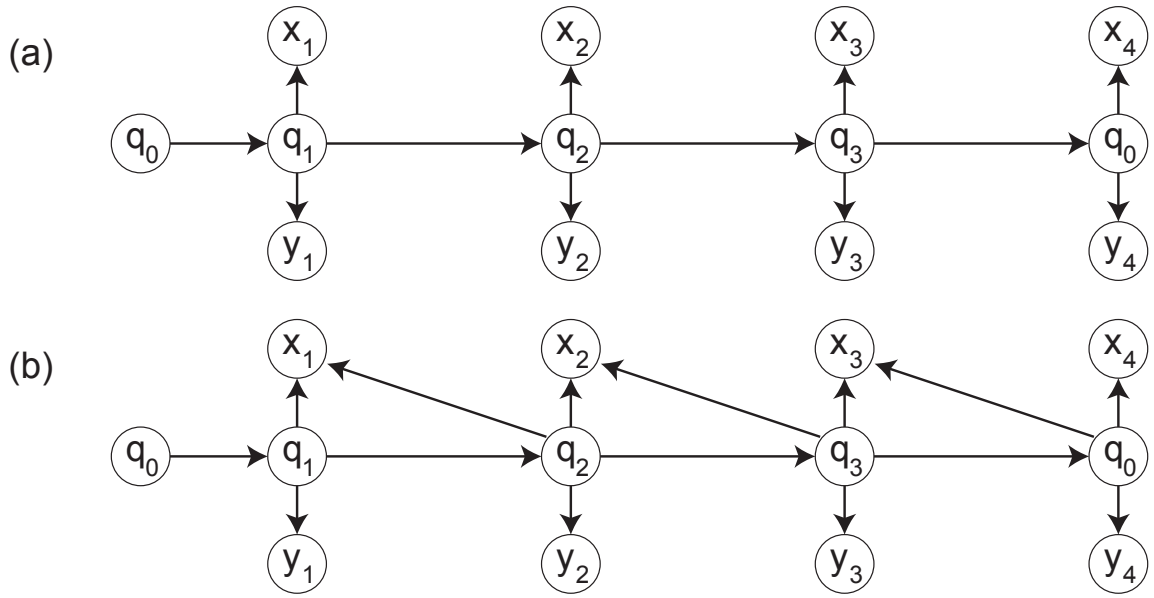

Figure 5.1: *The jointly-trained HMM is shown in (a). When $c_x > 1$, dependencies are introduced between future hidden states and past observations, as shown in (b). The resulting v-structure produces a dependency between past and future hidden states, since the input is always observed.*

is one in which the hidden process q generates both the output signal y and the input signal x, and the model is trained with the concatenated vector $(y, x)^T$ used as the observation. The greater number of parameters makes it more difficult to properly infer the hidden structure of the process without excessive overfitting and falling into local optima. However, it can potentially produce better results when dealing with very different signals. I will refer to this model as the jointly-trained HMM.

One issue with either approach is that, while it should in theory effectively synthesize y for all $c_y > 1$, it requires in practice a very large amount of training data and many hidden states to model a complex processes. In terms of the discussion in Section 5.1, as the number of bits n of required context grows, the number of hidden states grows exponentially. One way to remedy this problem is to provide some context directly to each hidden state, by using $(x_t, x_{t-1}, ..., x_{t-d})$ as the input observation. As shown in Section 6.2, this does improve the synthesized result. However, this approach is not entirely sound, because it

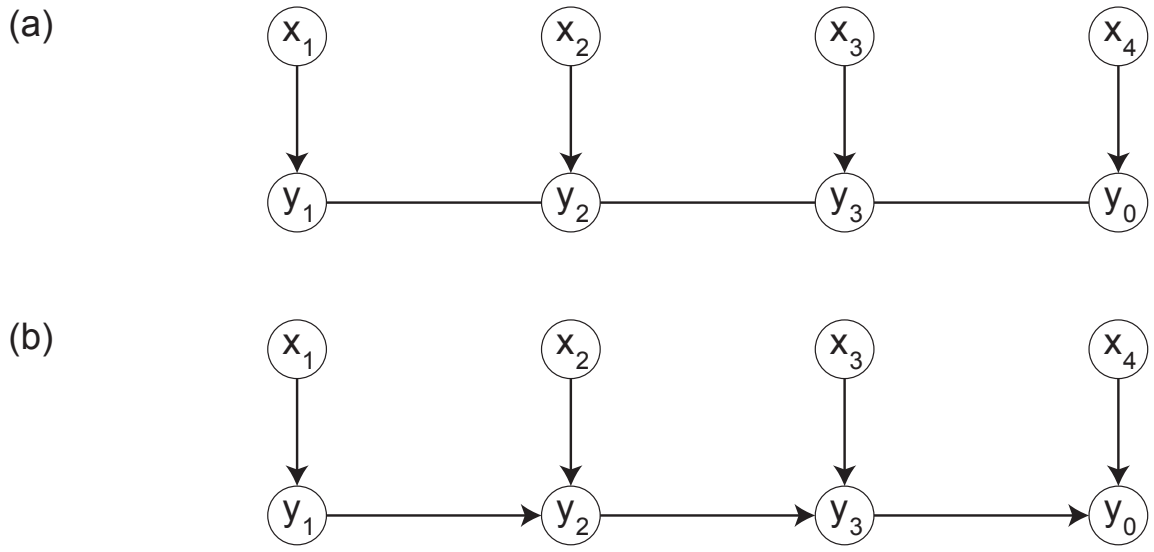

Figure 5.2: *The graphical structure of the CRF is shown in (a). Since the CRF is a partially undirected model, maximum likelihood estimation of the parameters requires global optimization and is costly [55]. A related, full-directed model is shown in (b).*

creates conditional dependencies between non-adjacent hidden states (see Figure 5.1), violating the Markov property. A discriminative model such as a CRF or MEMM does not have this problem, because it makes no independence assumptions between observations [30]. The impact of this problem on synthesis quality is evaluated in Section 6.4.

5.3 Conditional Random Fields

Conditional random fields (CRFs) provide a good method for synthesizing y if $c_y = 1$ and c_x is small – that is, if the current value of the synthesized parameter depends only on the previous value and a small time window of prosody features. The CRF is a discriminative model that estimates the probability of a transition between adjacent values y_{t-1} and y_t, conditional on the current input observation x_t. The graphical representation of a CRF is given in Figure 5.2, along with a closely related directed model, the maximum entropy Markov model (MEMM) [30, 36].

The implementation of the CRF model used for synthesizing body language motion

parameters is based on Sutton and McCallum's CRF tutorial [55]. The discrete CRF implementation uses features $f_{ij}(y, y', x) = 1_{\{y=i\}}1_{\{y'=j\}}$ and $f_{io}(y, y', x) = 1_{\{y=i\}}1_{\{x=o\}}$. These features allow us to most accurately compare the CRF to a discrete-valued HMM, to determine the importance of hidden states for modeling body language and, by extension, to determine whether $c_y > 1$. As for the HMM, the input and output are discretized by clustering.

The continuous CRF implementation uses a Gaussian observation model, with features $f_{ij}(y, y', x) = 1_{\{y=i\}}1_{\{y'=j\}}$, $f_{ip}(y, y', x) = 1_{\{y=i\}}x_p$, and $f_{ipq}(y, y', x) = 1_{\{y=i\}}x_p x_q$. We can show that these features are sufficient for representing a Gaussian observation model with the following derivation: let λ be the parameters of the CRF, such that λ_{ij} is the parameter paired with f_{ij}. Let $\mathcal{L}(\lambda)$ represent the likelihood of the parameters and $Z(\lambda)$ the partition function:

$$\begin{aligned}\mathcal{L}(\lambda) &= \frac{1}{Z(\lambda)}\prod_{t=1}^{T}\Phi(y_t, y_{t-1}, x_t) \\ &= \frac{1}{Z(\lambda)}\prod_{t=1}^{T}\exp\left(\lambda_{y_t,y_{t-1}} + \sum_p x_p\lambda_{y_t,p} + \sum_{p,q} x_p x_q \lambda_{y_t,p,q}\right)\end{aligned}$$

Letting $P^{(y_t)}$ represent the precision matrix corresponding to the observation model for y_t, and $\mu^{(y_t)}$ the mean, we can substitute $\lambda_{y_t,p,q} = -\frac{1}{2}P^{(y_t)}_{pq}$ and $\lambda_{y_t,p} = \sum_q \mu^{(y_t)}_q P^{(y_t)}_{pq}$:

$$\begin{aligned}\mathcal{L}(\lambda) &= \frac{1}{Z(\lambda)}\prod_{t=1}^{T}\exp\left(\lambda_{y_t,y_{t-1}} + \sum_{p,q}\mu^{(y_t)}_q P^{(y_t)}_{pq}x_p - \frac{1}{2}\sum_{p,q}P^{(y_t)}_{pq}x_p x_q\right) \\ &= \frac{1}{Z(\lambda)}\prod_{t=1}^{T}\exp(\lambda_{y_t,y_{t-1}})\exp\left(-\frac{1}{2}\sum_{p,q}P^{(y_t)}_{pq}x_p x_q - \mu^{(y_t)}_q P^{(y_t)}_{pq}x_p - \mu^{(y_t)}_p P^{(y_t)}_{pq}x_q\right) \\ &= \frac{\prod_{t=1}^{T}\exp(\lambda_{y_t,y_{t-1}})\exp\left(-\frac{1}{2}(x_t - \mu^{(y_t)})^T P^{(y_t)}(x_t - \mu^{(y_t)})\right)}{Z(\lambda)\prod_{t=1}^{T}\exp\left(-\frac{1}{2}\mu^{(y_t)T}P^{(y_t)}\mu^{(y_t)}\right)}\end{aligned}$$

Since the partition function ensures that the likelihood is normalized, the observation model for each state y_t is precisely proportional to a Gaussian distribution with covariance $(P^{(y_t)})^{-1}$ and mean $\mu^{(y_t)}$. All remaining implementation details for training and inference

with CRFs are standard and discussed in detail in Sutton and McCallum's tutorial [55].

As discussed in the previous section, discriminative models such as CRFs and MEMMs can handle richer feature sets than HMMs because the structure of the model does not assume any independence relationships between the features. This allows the CRF to take into account a window of c_x prosody observations without violating independence assumptions. In fact, the size of the temporal window is limited only by the need to minimize the number of degrees of freedom in the model, making it ideal when c_x is small and $c_y = 1$.

Unfortunately, the lack of hidden states makes CRFs and MEMMs ineffective for modeling a process in which $c_y > 1$. Introducing a temporal window over the outputs is impractical, because the resulting structure creates long-term dependencies between unobserved states and makes efficient inference very difficult. Models with hidden variables can handle this situation better. One model that combines discriminative learning with hidden states is the hidden conditional random field [58].

5.4 Implementation Details for Latent Models

Traditionally, the training of hidden Markov models involves estimating a prior distribution over the hidden states [44]. This prior is then used to assign initial values to the forward probabilities α_0. The prior is usually estimated as the probability of the first expected hidden state in the training sequence. However, this is not the correct estimate for synthesis of motion parameters from prosody because there are no restrictions on when the signal begins or ends. Therefore, the first hidden state in the sequence is not distinguished in any way from subsequent states, and the prior distribution for state Q_i should be estimated directly as $P(Q_i)$, rather than $P(q_0 = Q_i)$.

An additional approach for improving the training algorithm for a hidden model is to select an appropriate prior distribution over the *parameters* that would be most helpful in inferring the correct hidden structure. Brand [7] proposed an *entropic* prior as the best prior distribution for correctly inferring the hidden state of a Markov process. This prior causes parameters that add little *information* to the model to be less likely, forcing uncertain parameters towards 0 or 1. Formally, this prior may be expressed as $P(\theta) \propto e^{-H(\theta)}$ for parameter θ, where $H(\theta)$ is the entropy of θ. In practice, entropic learning tends to drive

the transition probability into "unnecessary" hidden states towards zero, creating a sparser transition matrix and allowing these states to be trimmed. Details on the implementation of entropic learning are discussed by Brand [7]. In Section 6.3 I evaluate the possible benefits of an entropic prior over the standard uniform prior.

5.5 Inference

Offline inference in dynamic Bayesian networks can be performed efficiently with the Viterbi algorithm [44]. However, in order to generate motion parameters for real-time synthesis of gestures from live speech, the output must be synthesized from the input stream before the entire input stream is observed. Since the Viterbi algorithm must backtrack through the entire sequence to select the most probable path, it is not appropriate for online inference.

We evaluate a number of methods for online inference in Section 6.5. The simplest method for doing online inference is to select, at each time step, the state q_t that is most probable given the previously selected state q_{t-1}. We can express the probability distribution over current states given the previous state with a vector of *direct* probabilities γ_t, given by

$$\gamma_t(i) = P(q_t = i|q_{t-1}, x_t).$$

This probability decomposes to $P(q_t|q_{t-1})P(x_t|q_t)$ for the HMM.

Using γ_t ensures that the synthesized state sequence is coherent, but it fails to take into account past context when selecting the current state. Essentially, this method always assumes that the previously selected state was *correct*, and makes no provisions for correcting past mistakes. To alleviate this problem, we can instead select the state that is most probable at the current time step given the *distribution* over q_{t-1}, without regard for which state was actually selected. The current most likely state may be obtained from the vector of forward probabilities α_t, as described by Rabiner [44]:

$$\alpha_t(i) = P(q_t = i|x_1, ..., x_t) = \sum_j \alpha_{t-1}(j)P(q_t = i|q_{t-1} = j, x_t).$$

However, simply selecting the most probable state according to α_t could create improbable transitions between states, since the most likely state might oscillate between multiple probable "timelines." Instead, we can also consider a hybrid approach that uses the forward probabilities to steer the synthesis away from globally unlikely states while still relying on the direct probabilities to select probable transitions. This is done by using a combined distribution $\eta\alpha_t \times \gamma_t$, where η is a normalizing factor and $\times$ is a pointwise product. Using this combined distribution may allow the model to synthesize a path that is both probable given the current observed input and avoids improbable transitions between states.

Chapter 6

Evaluation

6.1 Comparing Time Series

We can evaluate the effectiveness of a probabilistic model for synthesizing motion parameters by directly comparing a synthesized time series to the ground truth of the motions accompanying the test utterance. A common metric for comparing time series in expressive animation synthesis is the squared error measure of the divergence between the ground truth vector $y_t^{(g)}$ and the synthesized vector $y_t^{(s)}$ [8, 31], given as

$$\text{Err}(y_t^{(g)}, y_t^{(s)}) = \frac{(y_t^{(g)} - y_t^{(s)})^T (y_t^{(g)} - y_t^{(s)})}{(y_t^{(g)} + y_t^{(s)})^T (y_t^{(g)} + y_t^{(s)})}.$$

However, this metric ignores differences in the frequency of the two time series, which can be extremely important for establishing the correct rhythm for gesture. This rhythm is better captured by computing the cross-correlation between the two series, with means $\mu^{(g)}$ and $\mu^{(s)}$ and variances $\sigma^{(g)}$ and $\sigma^{(s)}$, given as

$$\text{CrossCorr}(y_t^{(g)}, y_t^{(s)}) = \frac{(y_t^{(g)} - \mu^{(g)})(y_t^{(s)} - \mu^{(s)})}{\sigma^{(g)}\sigma^{(s)}}.$$

In general, cross correlation is a more useful metric since it captures more precisely how well the synthesized parameter captures the patterns in the ground truth signal. However, cross correlation ignores possible differences in bias, which the squared error metric does

capture. Therefore, both metrics are used to evaluate the proposed models, though high cross-correlation is considered more important.

While the motion parameters are selected in such a way as to reduce ambiguity, gesture formation is inherently a very ambiguous process. Therefore, landmarks in the synthesized time series may be synthesized correctly but still not align well with the ground truth series due to ambiguities in timing. If we use the previously described metrics, such a misalignment would often be penalized more than if the landmark had been missed completely: consider, for example, a ground truth series where $y_t^{(g)} = 0$ for $t \neq i$, and $y(g)_i = 1$. If the synthesized series has $y_t^{(s)} = 0$ for $t \neq i+1$ and $y_{i+1}^{(s)} = 1$, the resulting squared error would be higher, and the resulting cross correlation lower, than if we simply had $y_t^{(s)} = 0$ for $\forall t$, even though the two time series are extremely similar.

To remedy this problem, we can gently warp the synthesized series using dynamic time warping (DTW) to align it to the ground truth before computing a score. In the previous example, this approach would align the two peaks and correctly recognize the two time series as extremely similar. The DTW alignment can be computed efficiently with a monotonically increasing objective function using dynamic programming [16]. Given a measure of the error between any two points in the two series $y_i^{(g)}$ and $y_j^{(s)}$, given as $C(i,j)$, we can construct a $T \times T$ DTW table D, such that $D(i,j)$ is the total error between optimally aligned sequences $y_1^{(g)}, y_2^{(g)}, ..., y_i^{(g)}$ and $y_1^{(s)}, y_2^{(s)}, ..., y_j^{(s)}$. The recurrence for computing $D(i,j)$ is

$$D(i,j) = C(i,j) + \min \begin{cases} D(i-1, j-1) \\ D(i-1, j) \\ D(i, j-1) \end{cases}$$

Since the sequences have fixed start and end points, we want the perfect alignment to align the first and last time step correctly. Therefore, we can initialize $D(0,0) = C(0,0)$, $D(0,j) = C(0,j) + D(0,j-1)$, and $D(i,0) = C(i,0) + D(i-1,0)$. Once D is computed in $O(T^2)$ time, the optimal alignment sequence may be recovered by backtracking along the lowest-cost path from $D(T,T)$.

Since the error measure C is always positive, the algorithm naturally favors shorter paths. However, large deviations in the warped solution are more likely to be caused by

real differences between the synthesized and ground truth series. Therefore, we restrict the maximum difference between i and j to 3 time steps, or 1 second. This is equivalent to setting $D(i,j) = \infty$ for $|i-j| > 3$. To compute the DTW alignment between the synthesized and ground truth series, the monotonically increasing error function $\text{Err}(y_t^{(g)}, y_t^{(s)})$ may be used.

In conclusion, two metrics are used to evaluate the quality of a synthesized parameter series $y^{(s)}$ against a ground truth series $y^{(g)}$: the squared error measure $\frac{1}{T}\sum_{t=1}^{T}\text{Err}(y_t^{(g)}, y_t^{(s)})$ and the cross-correlation measure $\frac{1}{T}\sum_{t=1}^{T}\text{CrossCorr}(y_t^{(g)}, y_t^{(s)})$. The cross correlation measure indicates how closely the variations in the ground truth series are reflected in the synthesized series, and the squared error measure indicates to what degree the actual values are similar. In the case in which the cross-correlation is perfect, the squared error indicates the bias of the synthesized series. In the following sections, all results are presented with and without dynamic time warping.

6.2 Analyzing the Input Context and Structure

We first determine the appropriate size of the input window c_x by comparing the performance of the proposed models with various input window sizes. While a larger window allows the model to capture more of the input context, the number of degrees of freedom increases quadratically in the size of the window for a Gaussian observation model and exponentially for a discrete observation model.

Table 6.1 presents offline synthesis performance using the metrics discussed in Section 6.1. Each model is trained on 12 minutes of training data and tested on a 7-minute novel speech signal. The signals are discretized in time with 3 time steps per second, in order to take the largest possible steps that do not frequently overlap multiple syllables or important motion parameter features. The jointly-trained and remapped HMMs use 48 hidden states, determined empirically to produce good results. The discrete CRF uses 64 input clusters and 32 output clusters. For most of the models, only the $\bar{T}$ parameter is not synthesized best with a 3-step window. However, the quality of the synthesis of $\bar{T}$ is generally erratic and low, which suggests that this parameter correlates poorly to vocal prosody. The high correlation and low error of the other parameters with a 3-step window indicates that this

	c_x	CrossCorr $\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$	CrossCorr with DTW $\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$	Err $\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$	Err with DTW $\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$
Remapped	1	0.00	0.31	0.47	0.29	0.25	0.06	0.40	0.54	0.38	0.34	0.13	0.44	0.03	0.44	0.52	0.12	0.43	0.03	0.43	0.51
	2	-0.02	0.31	0.44	0.31	0.28	0.06	0.41	0.52	0.41	0.38	0.12	0.44	0.03	0.44	0.52	0.11	0.42	0.03	0.42	0.51
	3	-0.02	0.33	0.45	0.33	0.30	0.09	0.45	0.54	0.45	0.43	0.11	0.44	0.03	0.43	0.52	0.10	0.42	0.03	0.42	0.50
	4	0.03	0.29	0.39	0.29	0.27	0.17	0.40	0.49	0.41	0.39	0.11	0.45	0.03	0.45	0.54	0.10	0.43	0.03	0.42	0.51
	5	-0.08	0.20	0.29	0.21	0.18	0.07	0.36	0.43	0.36	0.35	0.11	0.48	0.03	0.47	0.57	0.09	0.44	0.03	0.43	0.53
Joint	1	0.08	0.25	0.38	0.20	0.16	0.18	0.37	0.48	0.31	0.29	0.10	0.46	0.03	0.46	0.55	0.08	0.45	0.03	0.44	0.53
	2	0.00	0.14	0.18	0.15	0.09	0.17	0.40	0.41	0.41	0.37	0.08	0.53	0.03	0.51	0.61	0.07	0.48	0.02	0.46	0.56
	3	0.01	0.22	0.32	0.23	0.17	0.19	0.42	0.48	0.44	0.41	0.08	0.48	0.03	0.47	0.57	0.07	0.45	0.02	0.42	0.54
	4	0.03	0.19	0.30	0.17	0.15	0.21	0.40	0.50	0.37	0.37	0.08	0.47	0.03	0.50	0.60	0.06	0.41	0.02	0.45	0.56
	5	0.04	0.15	0.22	0.19	0.15	0.24	0.40	0.43	0.40	0.40	0.09	0.54	0.03	0.51	0.60	0.07	0.49	0.02	0.46	0.55
Discrete CRF	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.06	0.99	0.05	0.95	0.55	0.06	0.99	0.05	0.95	0.55
	2	0.05	0.08	0.17	0.08	0.07	0.21	0.29	0.33	0.28	0.28	0.08	0.44	0.04	0.47	0.50	0.07	0.41	0.03	0.43	0.45
	3	-0.03	0.20	0.26	0.22	0.20	0.14	0.38	0.41	0.40	0.39	0.09	0.51	0.04	0.52	0.45	0.08	0.46	0.03	0.46	0.38
	4	-0.10	0.21	0.31	0.21	0.19	0.10	0.40	0.47	0.38	0.38	0.09	0.52	0.03	0.51	0.45	0.08	0.46	0.03	0.47	0.38
	5	0.01	0.15	0.18	0.15	0.12	0.13	0.28	0.32	0.29	0.27	0.10	0.46	0.04	0.47	0.47	0.09	0.43	0.03	0.44	0.43

Table 6.1: *This table presents the performance of various models with various input window sizes c_x on the 7-minute test set. All models are trained on 12 minutes of training data and evaluated with the metrics discussed in Section 6.1. For details on the parameters, see Section 3.3*

window size give the best balance between dimensionality and representative power.

The discrete CRF performs slightly better with a 4-step window, although the difference is small. This suggests that the drop in synthesis quality for a window of size 4 in the HMMs is due to the assumption of observation independence discussed in Section 5.2. Since CRFs are discriminative models, they do not suffer from this problem.

In addition, the results in Table 6.1 indicate that a remapped HMM consistently outperforms the jointly-trained HMM. This suggests that there is enough similarity in the structure of the two signals. As discussed in Section 5.2, the remapped HMM is expected to outperform the jointly-trained HMM when the signals have similar structure because the jointly-trained HMM must optimize over more parameters and is therefore more prone to overfitting and local optima.

6.3 Entropy Minimization

Entropy minimization is a technique for applying an entropic prior to the parameters of the HMM, as discussed in Section 5.4. This technique attempts to preserve the most *informative* transitions and drive all other transitions towards 0. Brand hypothesizes that such a prior would improve an HMM's ability to carry information forward in time through the hidden states [7].

Table 6.2 compares the performance HMMs trained with and without the entropic prior. The models trained with the entropic prior do not outperform the conventionally trained HMMs. In fact, their results are, in general, slightly worse. This indicates that the entropy minimization method is not effective for improving the discovery of the hidden structure in the signal.

6.4 Determining the Importance of Hidden States

As discussed in Section 5.1, the choice of model depends heavily on whether or not $c_y > 1$. For longer-range dependencies in the output, a model with hidden variables is preferred, while a simpler model with the Markov property for the outputs may be used if $c_y \leq 1$. To

Model	CrossCorr					CrossCorr with DTW					Err					Err with DTW				
	$\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$	$\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$	$\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$	$\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$
Discrete Joint HMM	-0.00	0.23	0.24	0.24	0.22	0.14	0.44	0.43	0.42	0.41	0.09	0.53	0.03	0.52	0.61	0.08	0.48	0.03	0.46	0.55
Discrete Remapped HMM	0.01	0.29	0.38	0.30	0.27	0.20	0.44	0.51	0.45	0.44	0.11	0.42	0.03	0.48	0.48	0.09	0.38	0.03	0.44	0.44
CRF with Discrete Features	-0.03	0.20	0.26	0.22	0.20	0.14	0.38	0.41	0.40	0.39	0.09	0.51	0.04	0.52	0.45	0.08	0.46	0.03	0.46	0.38
Continuous Joint HMM	0.01	0.22	0.32	0.23	0.17	0.19	0.42	0.48	0.44	0.41	0.08	0.48	0.03	0.47	0.57	0.07	0.45	0.02	0.42	0.54
Cont. Entropic Joint HMM	-0.04	0.21	0.27	0.24	0.19	0.13	0.40	0.42	0.42	0.39	0.08	0.53	0.03	0.52	0.60	0.07	0.49	0.02	0.47	0.56
Continuous Remapped HMM	-0.02	0.33	0.45	0.33	0.30	0.09	0.45	0.54	0.45	0.43	0.11	0.44	0.03	0.43	0.52	0.10	0.42	0.03	0.42	0.50
Cont. Entropic Remapped HMM	-0.00	0.31	0.41	0.34	0.29	0.12	0.42	0.50	0.43	0.40	0.11	0.45	0.03	0.44	0.53	0.10	0.44	0.03	0.42	0.51
CRF with Continuous Features	0.04	0.12	0.12	0.02	0.09	0.15	0.21	0.20	0.13	0.19	0.11	0.44	0.05	0.56	0.48	0.10	0.40	0.05	0.53	0.43

Table 6.2: *This table compares the performance of various models in offline synthesis. All continuous HMMs are trained with* 48 *hidden states, and all discrete models are trained with* 32 *output clusters and* 64 *input clusters. Discrete HMMs are trained with* 32 *hidden states. All models use a time window of* 3 *steps, or* 1 *second.*

determine whether hidden states are useful, I conducted comparisons between the CRF and HMM models, which are presented along with other results in Table 6.2.

The poor performance of the continuous CRF may be attributed to the poor representational power of the Gaussian distribution, which allows the input distribution for each output cluster to contain only a single peak. Since CRFs model distributions of the exponential family, multi-modal distributions are inherently very difficult to represent continuously. The HMM does not suffer from this problem, because multiple hidden states may represent the same output but with a different input distribution. Therefore, the discrete models provide a fairer comparison.

The performance of the discrete CRF and the jointly-trained discrete HMM is fairly similar, but the remapped HMM outperforms the CRF with a considerable margin. This suggests that the signal does in fact possess some meaningful hidden structure that a model with hidden states can exploit. Therefore, the HMM, or another model with latent states, is appropriate for modeling the process.

6.5 Online Synthesis

Finally, we must consider the usefulness of the model for synthesis of motion parameters from a live speech signal, using the inference algorithm described in Section 5.5. Table 6.3 presents the results for online synthesis, produced by feeding the input into the model progressively and selecting the best output at each time step. Results are shown with selection of the most likely state according to the vector of forward probabilities α_t at each time step, using $\arg\max_i \alpha_t(i)$, according to the direct probabilities γ_t, using $\arg\max_i \gamma_t(i)$, and according to the hybrid inference heuristic presented in Section 5.5, using $\arg\max_i \alpha_t \times \gamma_t$.

From the results in Table 6.3, online synthesis achieves results that are comparable to those achieved by offline synthesis with the Viterbi algorithm. This suggests that only the preceding prosody information is necessary to select current motion parameters.

Predictably, using the forward probabilities alone produces the best results in terms of a direct comparison to the ground truth parameters. The hybrid approach produces better results than the direct probabilities alone for the remapped HMM but performs worse on the jointly-trained HMM, suggesting that it does not provide a substantial improvement

	Model	CrossCorr $\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$	CrossCorr with DTW $\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$	Err $\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$	Err with DTW $\bar{T}$	$\bar{V}$	$\bar{D}$	$\bar{A}$	$\bar{C}$
Viterbi	Continuous Joint HMM	0.01	0.22	0.32	0.23	0.17	0.19	0.42	0.48	0.44	0.41	0.08	0.48	0.03	0.47	0.57	0.07	0.45	0.02	0.42	0.54
	Continuous Remapped HMM	-0.02	0.33	0.45	0.33	0.30	0.09	0.45	0.54	0.45	0.43	0.11	0.44	0.03	0.43	0.52	0.10	0.42	0.03	0.42	0.50
α_t	Continuous Joint HMM	0.01	0.16	0.25	0.19	0.12	0.21	0.43	0.46	0.45	0.41	0.08	0.51	0.03	0.48	0.59	0.06	0.46	0.02	0.43	0.55
	Continuous Remapped HMM	-0.01	0.27	0.36	0.26	0.23	0.20	0.48	0.52	0.48	0.48	0.11	0.47	0.03	0.46	0.55	0.09	0.44	0.02	0.42	0.51
γ_t	Continuous Joint HMM	-0.07	0.19	0.27	0.18	0.15	0.12	0.43	0.47	0.38	0.40	0.09	0.51	0.03	0.48	0.59	0.07	0.46	0.02	0.43	0.55
	Continuous Remapped HMM	0.08	0.21	0.24	0.19	0.19	0.14	0.35	0.36	0.34	0.35	0.09	0.53	0.03	0.51	0.59	0.08	0.50	0.03	0.47	0.56
$\alpha_t \times \gamma_t$	Continuous Joint HMM	0.01	0.14	0.20	0.17	0.11	0.21	0.37	0.40	0.38	0.36	0.08	0.51	0.03	0.48	0.59	0.06	0.46	0.02	0.43	0.55
	Continuous Remapped HMM	-0.02	0.19	0.25	0.19	0.14	0.20	0.40	0.43	0.41	0.39	0.09	0.51	0.03	0.50	0.60	0.07	0.47	0.02	0.45	0.56

Table 6.3: *This table compares the performance of various models in online synthesis, using a variety of selection heuristics. The performance of the models in offline synthesis with the Viterbi algorithm is presented for comparison.*

over the direct probability method. The coherence enforced by the direct probabilities or the hybrid method is difficult to measure numerically, and a closer analysis of the tradeoffs between forward probabilities and the hybrid or direct method is left for future work.

6.6 Conclusion

Analysis of the evaluation results provides us with considerable insight into the type of model that is most appropriate for synthesizing body language motion parameters from vocal prosody. We can conclude that a small time window of about 3 time steps, or 1 second, provides a good tradeoff between dimensionality and representational power. Furthermore, the structure of the input and output signals is similar enough that a traditional remapping technique [8] is able to capture the structure of the hidden process.

Analysis of the results for each parameter shows that synthesis of $\bar{T}$ is frequently poor and inconsistent. This suggests that $\bar{T}$ does not correlate well with vocal prosody. Since the length of the desired gesture is important to select the correct animation segment during animation synthesis, an alternative parameter or method could be explored in future work to provide this information.

In considering the importance of hidden states, we observed that the both the continuous and discrete HMMs generally outperform CRFs with a comparable number of degrees of freedom. A CRF that is better able to represent continuous multi-modal distributions may perform better, but the comparison of the discrete models indicates that some hidden structure is in fact present in the signal, and a model with latent variables is therefore more appropriate.

Chapter 7

Gesture Synthesis

7.1 Overview

One possible application of the body language model is synthesis of appropriate gestures for a novel speech input. I present a gesture synthesis engine that can produce appropriate animations for a live novel utterance in real time. This makes the method suitable for synthesizing animations for human characters in interactive environments, where the input speech is recorded through a microphone. This speech is segmented into syllables progressively, as discussed in Section 4.2, and each new syllable is fed to the body language model as soon as it is observed. The model synthesizes appropriate motion parameters using the inference algorithm described in Section 5.5, and predicts future parameters for a fixed number of time steps using the transition model. It is necessary to make predictions several time steps into the future in order to select appropriate motions for live, online synthesis that are well synchronized with speech.

A motion planner uses the synthesized parameters to select appropriate animation segments to insert into the animation stream, and these segments are then blended with previous frames to create a smooth, continuous animation.

7.2 Motion Planner

The motion planner takes as input the predicted values of the motion parameters over some number of future time steps and selects an appropriate animation from a pre-recorded library of motions. This library can be obtained from the same motion capture training set as is used to train the model, but this need not necessarily be the case. For example, the motions can be taken from a specific individual to lend the synthesized animation a personalized style, or they can be taken from a different training set that does not have accompanying audio.

The library consists of animation segments produced using the algorithm described in Section 2.2. For each segment, the library contains the actual frames of the corresponding animation as well as the motion parameters. These parameters are used by the motion planner to select the segment that best fits the synthesized parameters. Specifically, the objective of the motion planner is to select the segment that follows the previous segment smoothly and fits the predicted parameters.

To evaluate how well an animation follows from another segment, a successor distance matrix S is pre-computed. Define the distance between two animation segments i and j, $D(i, j)$, as described in Section 7.3. The successor matrix is then defined as

$$S_{i,j} = D(i, preceding(j)) + D(following(i), j),$$

where $preceding(j)$ is the segment preceding j in the training data, and $following(i)$ is the segment following i.

Since motion parameters do not change over the course of a single segment, we can compare how well a prospective segment fits the predicted parameters by comparing the average value of the synthesized parameters over the length of the segment to its actual parameter values. Given future predictions $y_t, y_{t+1}, ..., y_{t+n}$, this comparison can be performed efficiently by pre-computing a vector P defined as

$$P_i = \sum_{j=t}^{t+i} y_j.$$

For a candidate segment starting at time step t or length ℓ_s, with parameters Y_s, we can evaluate the difference between the segment parameters and the synthesized parameters, $C(Y_s, \ell_s, t)$, as

$$C(Y_s, \ell_s, t) = \left|\left| Y_s - \frac{P_{t+\ell_s}}{\ell_s} \right|\right|^2 .$$

For each subsequent predicted segment, given that the previous segment ended at time t', we can compute the cost as

$$C(Y_s, \ell_s, t') = \left|\left| Y_s - \frac{P_{t'+\ell_s} - P_{t'}}{\ell_s} \right|\right|^2 .$$

Once the parameter cost $C(Y_s, \ell_s, t)$ and succession cost $S_{i,j}$ are computed, we can express the total cost of inserting the segment as a weighted sum of these two quantities. The weights reflect a tradeoff between how smooth the resulting animation is and how faithfully it conforms to the synthesized parameters, and by extension the input speech.

7.3 Distance Between Animation Segments

The distance function for comparing two animation segments should produce large values for segments that are perceptually dissimilar and small values for segments that appear to constitute the same gesture. The objective is to ensure that transitions only occur between gesture unit phases that should logically follow each other. Since the motion planner treats the motion parameters separately, it is not necessary for the distance function to take these parameters into account. In fact, it is best for the distance function to capture precisely the opposite aspect of motion – those elements of form that are neglected by the motion parameters and therefore must be dealt with completely by the motion planner.

Segments are compared according to a three-part distance function that takes into account length, starting pose, and a fixed-size "summary" of the dynamics of the segment. The "summary" represents some measure of velocity and maximum and average displacement along each of the axes for each hand, organized into three six dimensional vectors. Each vector is rescaled to a logarithmic scale, since larger gestures can tolerate greater variation while remaining perceptually similar. Using six-dimensional vectors serves to

de-emphasize the unused hand in one-handed gestures: if instead a three-dimensional vector for each hand was rescaled, small motions in the unused hand would be weighted too heavily on the logarithmic scale relative to the more important large motion of the active hand. To compare two summaries, the sum of the distances between the three vectors is used. Together with the difference in the starting pose and segment length, this "summary" provides a perceptually accurate method to compare two animation segments.

7.4 Constructing the Animation Stream

Once the next animation segment is selected, it must be blended into the animation stream to create a smooth transition. Although linear blending is generally sufficient for a perceptually smooth transition [57], it requires each motion to have enough extra frames at the end to accommodate a gradual blend, and simply taking these frames from the original motion capture data may introduce extra, unwanted gestures. For many of the motions, the blend interval would also exceed the length of the gesture, requiring many gestures to be linearly blended simultaneously.

Instead, the system employs a velocity-based blending algorithm, which keeps the magnitude of the angular velocity on each joint equal to that of the desired animation, and adjusts joint orientations to be as close as possible to the desired pose within this constraint. For a new rotation quaternion r_t, previous rotation r_{t-1}, desired rotation d_t, and the derivative in the source animation of the desired frame $\triangle d_t = d_t \bar{d}_{t-1}$, the orientation of a joint at frame t is given by

$$r_t = \text{slerp}\left(r_{t-1}, d_t; \frac{\text{angle}(\triangle d_t)}{\text{angle}(d_t \bar{r}_{t-1})}\right),$$

where "slerp" is the quaternion spherical interpolation function [50], and "angle" gives the angle of the axis-angle representation of a quaternion. I found that using world-space rather than parent-space orientation in the desired and previous pose produces more realistic results and preserves the "feel" of the desired animation, even when the starting pose of the current gesture is different from the current frame.

7.5 Examples of Synthesized Motions

A number of animations were created using the proposed system for novel voice input, using the online and offline synthesis methods discussed in Section 5.5 together with the motion planner described in this chapter. The animations may be viewed on a website, located at `http://www.stanford.edu/~svlevine/vids/svlmsreport.htm`.

The synthesized animations are presented, alongside with the original motion capture recording for each utterance. The presented animations show, in order, the original motion capture accompanying each utterance, a synthesized animation generated from the ground truth motion parameters of the motion capture sequence to evaluate the motion planner, a synthesized animation generated from parameters synthesized offline with the Viterbi algorithm, and synthesized animations from parameters synthesized with each of the three online synthesis techniques described in Section 5.5.

Chapter 8

Discussion and Future Work

8.1 Potential Applications and Future Work

The system presented in this report is able to synthesize compelling gesture animations from live speech. This makes it useful not just for conventional, offline animation, but also for animation of player avatars in networked virtual environments from live speech. As discussed in Section 1.1, synthesis of compelling body language is almost completely absent from such virtual conversation today and has the potential to greatly enhance them.

In addition to producing believable animations directly from the training data, the modular design of the system allows the motion planner to be decoupled from the probabilistic model. A distinct library of gestures may be used that is different from that used to train the probabilistic model. In fact, the motion planner can conceivably utilize a gesture library from a different individual for a unique, personalized style, or even an artist-created gesture library for animating a stylized or non-humanoid character.

Additional development of the motion planner component could remedy some of the shortcomings of the current motion planner. One possible improvement is the introduction of stochastic selection of the appropriate gesture by using the output mean and covariance of the currently active hidden states as a distribution over the candidates. This would avoid repetitive gestures caused by the limited representational power of a finite number of hidden states. On the other hand the motion planner could also be made more responsive by synthesizing new gestures to fit the desired motion parameters by interpolation, as

demonstrated by Torresani et al. [33], or by altering gestures after they are selected as new information becomes available.

In addition to its uses for synthesizing body language animations, the probabilistic models discussed in this report may also be useful for selecting regions of interest for gesture detection, in a manner similar to that proposed by Kettebekov et al. [27].

8.2 Limitations

Although the method described in this report can efficiently produce body language with compelling rhythm and emphasis, it is limited in its ability to produce meaningful motions. The motion parameters are, by design, as divorced from semantics as vocal prosody. In fact, one way to view the system is as a "style machine": a method for selecting an appropriate style for an animation. When viewed in this way, it is clear that the gesture synthesis engine must consist of two components – a "style machine" and a "gesture selector." In the proposed method, the "gesture selector" operates by selecting gestures that most closely match the desired style, without regard for their semantic meaning (if any). A clear improvement to the system would be to implement a semantically-driven "gesture selector" that would select gestures that fit the content of the utterance within the constraints of the specified style (or vice versa).

A more fundamental issue with the proposed method is one that afflicts any system for online gesture synthesis: it must synthesize gestures from information that is already available to the listener, thus limiting its ability to provide supplementary details. While there is no consensus in the linguistics community on just how much information is conveyed by gestures [32], a predictive speech-based real-time method is unlikely to impart on the listener any more information than could be obtained from simply listening to the speaker attentively, while real gestures often provide supplementary information that is not present in speech [37]. Therefore, while there are clear benefits to immersiveness and realism from compelling and automatic animation of human-controlled characters, such methods cannot provide additional details without some form of additional input.

8.3 Conclusion

In this report, I presented a new method for synthesizing parameters of expressive motion from vocal prosody, and demonstrated how these parameters can be used to animate a character with appropriate body language for a specific utterance. A number of models were evaluated to select the one that is most appropriate for synthesizing these motion parameters, and the relationship between speech and motion was explored in detail to discover the temporal dependencies between the two processes. The proposed system is able to generate appropriate gesture animations from live speech input, which allows it to efficiently produce the sort of compelling body language that is almost completely absent from present-day virtual conversations.

Bibliography

[1] Ralph Adolphs. Neural systems for recognizing emotion. *Current Opinion in Neurobiology*, 12(2):169–177, 2002.

[2] Irene Albrecht, Jörg Haber, and Hans peter Seidel. Automatic generation of non-verbal facial expressions from speech. In *Computer Graphics International*, pages 283–293, 2002.

[3] Jernej Barbič, Alla Safonova, Jia-Yu Pan, Christos Faloutsos, Jessica K. Hodgins, and Nancy S. Pollard. Segmenting motion capture data into distinct behaviors. In *Proceedings of Graphics Interface 2004*, pages 185–194, 2004.

[4] Jonas Beskow. *Talking Heads - Models and Applications for Multimodal Speech Synthesis*. PhD thesis, KTH Stockholm, 2003.

[5] Paul Boersma. Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound. In *Proceedings of the Institute of Phonetic Sciences*, volume 17, pages 97–110, 1993.

[6] Paul Boersma. Praat, a system for doing phonetics by computer. *Glot International*, 5(9-10):314–345, 2001.

[7] Matthew Brand. Structure learning in conditional probability models via an entropic prior and parameter extinction. *Neural Computation*, 11(5):1155–1182, 1999.

[8] Matthew Brand. Voice puppetry. In *SIGGRAPH '99: ACM SIGGRAPH 1999 Papers*, pages 21–28, New York, NY, USA, 1999. ACM.

[9] Christoph Bregler, Michele Covell, and Malcolm Slaney. Video rewrite: driving visual speech with audio. In *SIGGRAPH '97: ACM SIGGRAPH 1997 Papers*, pages 353–360, New York, NY, USA, 1997. ACM.

[10] Justine Cassell. Embodied conversational interface agents. *Communications of the ACM*, 43(4):70–78, 2000.

[11] Justine Cassell, Catherine Pelachaud, Norman Badler, Mark Steedman, Brett Achorn, Tripp Becket, Brett Douville, Scott Prevost, and Matthew Stone. Animated conversation: rule-based generation of facial expression, gesture & spoken intonation for multiple conversational agents. In *SIGGRAPH '94: ACM SIGGRAPH 1994 Papers*, pages 413–420, New York, NY, USA, 1994. ACM.

[12] Justine Cassell, Hannes Högni Vilhjálmsson, and Timothy Bickmore. Beat: the behavior expression animation toolkit. In *SIGGRAPH '01: ACM SIGGRAPH 2001 Papers*, pages 477–486, New York, NY, USA, 2001. ACM.

[13] Diane Chi, Monica Costa, Liwei Zhao, and Norman Badler. The emote model for effort and shape. In *SIGGRAPH '00: ACM SIGGRAPH 2000 Papers*, pages 173–182, New York, NY, USA, 2000. ACM.

[14] Erika Chuang and Christoph Bregler. Mood swings: expressive speech animation. *ACM Transactions on Graphics*, 24(2):331–347, 2005.

[15] Marco de Meijer. The contribution of general features of body movement to the attribution of emotions. *Journal of Nonverbal Behavior*, 13(4):247–268, 1989.

[16] John R. Deller, Jr., John G. Proakis, and John H. Hansen. *Discrete Time Processing of Speech Signals*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1993.

[17] David Efron. *Gesture, Race and Culture*. The Hague: Mouton, 1972.

[18] Jacob Eisenstein and Randall Davis. Visual and linguistic information in gesture classification. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses*, page 30, New York, NY, USA, 2006. ACM.

[19] Tony Ezzat, Gadi Geiger, and Tomaso Poggio. Trainable videorealistic speech animation. In *SIGGRAPH '02: ACM SIGGRAPH 2002 Papers*, pages 388–398, New York, NY, USA, 2002. ACM.

[20] Ajo Fod, Maja J. Mataric, and Odest Chadwicke Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12:39–54, 2002.

[21] Björn Hartmann, Maurizio Mancini, and Catherine Pelachaud. Formational parameters and adaptive prototype instantiation for mpeg-4 compliant gesture synthesis. In *Proceedings on Computer Animation*, page 111, Washington, DC, USA, 2002. IEEE Computer Society.

[22] Björn Hartmann, Maurizio Mancini, and Catherine Pelachaud. Implementing expressive gesture synthesis for embodied conversational agents. In *Gesture in Human-Computer Interaction and Simulation*, pages 188–189. Springer Berlin/Heidelberg, 2005.

[23] Carlos Jensen, Shelly D. Farnham, Steven M. Drucker, and Peter Kollock. The effect of communication modality on cooperation in online environments. In *Proceedings of CHI '00*, pages 470–477, New York, NY, USA, 2000. ACM.

[24] Deborah Zaitchik Joann Montepare, Elissa Koff and Marilyn Albert. The use of body movements and gestures as cues to emotions in younger and older adults. *Journal of Nonverbal Behavior*, 23(2):133–152, 1999.

[25] Eunjung Ju and Jehee Lee. Expressive facial gestures from motion capture data. *Computer Graphics Forum*, 27(2):381–388, 2008.

[26] Adam Kendon. *Gesture*. Cambridge University Press, New York, NY, USA, 2004.

[27] S. Kettebekov, M. Yeasin, and R. Sharma. Prosody based co-analysis for continuous recognition of coverbal gestures. In *Multimodal Interfaces, 2002. Proceedings. Fourth IEEE International Conference on*, pages 161–166, 2002.

[28] Michael Kipp, Michael Neff, and Irene Albrecht. An annotation scheme for conversational gestures: how to economically capture timing and form. *Language Resources and Evaluation*, 41(3-4):325–339, December 2007.

[29] Stefan Kopp and Ipke Wachsmuth. Synthesizing multimodal utterances for conversational agents: Research articles. *Computer Animation and Virtual Worlds*, 15(1):39–52, 2004.

[30] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[31] Yan Li and Heung-Yeung Shum. Learning dynamic audio-visual mapping with input-output hidden markov models. *IEEE Transactions on Multimedia*, 8(3):542–549, June 2006.

[32] Daniel Loehr. *Gesture and Intonation*. PhD thesis, Georgetown University, 2004.

[33] Chris Bregler Lorenzo Torresani, Peggy Hackney. Learning motion style synthesis from perceptual observations. In *Neural Information Processing Systems (NIPS) 19*, 2007.

[34] O. Maeran, V. Piuri, and G. Storti Gajani. Speech recognition through phoneme segmentation and neural classification. *Instrumentation and Measurement Technology Conference, 1997. IMTC/97. Proceedings. 'Sensing, Processing, Networking.', IEEE*, 2:1215–1220, May 1997.

[35] Anna Majkowska, Victor B. Zordan, and Petros Faloutsos. Automatic splicing for hand and body animations. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 309–316, 2006.

[36] Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. Maximum entropy markov models for information extraction and segmentation. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 591–598, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[37] David McNeill. *Hand and Mind: What Gestures Reveal About Thought*. University Of Chicago Press, 1992.

[38] David McNeill. *Gesture and Thought*. University Of Chicago Press, November 2005.

[39] Madoka Miki, Chiyomi Miyajima, Takanori Nishino, Norihide Kitaoka, and Kazuya Takeda. An integrative recognition method for speech and gestures. In *IMCI '08: Proceedings of the 10th international conference on Multimodal interfaces*, pages 93–96, New York, NY, USA, 2008. ACM.

[40] Louis-Philippe Morency, Candace Sidner, Christopher Lee, and Trevor Darrell. Head gestures for perceptual interfaces: The role of context in improving recognition. *Artificial Intelligence*, 171(8-9):568–585, 2007.

[41] Meinard Müller, Tido Röder, and Michael Clausen. Efficient content-based retrieval of motion capture data. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, volume 24, pages 677–685, New York, NY, USA, 2005. ACM.

[42] Michael Neff, Michael Kipp, Irene Albrecht, and Hans-Peter Seidel. Gesture modeling and animation based on a probabilistic re-creation of speaker style. *ACM Transactions on Graphics*, 27(1):1–24, 2008.

[43] Ken Perlin and Athomas Goldberg. Improv: a system for scripting interactive actors in virtual worlds. In *SIGGRAPH '96: ACM SIGGRAPH 1996 Papers*, pages 205–216. ACM, 1996.

[44] Lawrance R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb 1989.

[45] M. E. Sargin, E. Erzin, Y. Yemez, A. M. Tekalp, A.T. Erdem, C. Erdem, and M. Ozkan. Prosody-driven head-gesture animation. In *ICASSP '07: IEEE International Conference on Acoustics, Speech, and Signal Processing*, April 2007.

[46] M.E. Sargin, O. Aran, A. Karpov, F. Ofli, Y. Yasinnik, S. Wilson, E. Erzin, Y. Yemez, and A.M. Tekalp. Combined gesture-speech analysis and speech driven gesture synthesis. *IEEE International Conference on Multimedia and Expo*, pages 893–896, July 2006.

[47] Klaus R. Scherer, Rainer Banse, Harald G. Wallbott, and Thomas Goldbeck. Vocal cues in emotion encoding and decoding. *Motivation and Emotion*, 15(2):123–148, 1991.

[48] Klaus R. Scherer, Rainer Banse, Harald G. Wallbott, and Thomas Goldbeck. Vocal cues in emotion encoding and decoding. *Motivation and Emotion*, 15(2):123–148, 1991.

[49] R. Sharma, J. Cai, S. Chakravarthy, I. Poddar, and Y. Sethi. Exploiting speech/gesture co-occurrence for improving continuous gesture recognition in weather narration. pages 422–427, 2000.

[50] Ken Shoemake. Animating rotation with quaternion curves. In *SIGGRAPH '85: ACM SIGGRAPH 1985 Papers*, pages 245–254, New York, NY, USA, 1985. ACM.

[51] Elizabeth Shriberg, Andreas Stolcke, Dilek Hakkani-Tür, and Gükhan Tür. Prosody-based automatic segmentation of speech into sentences and topics. *Speech Communication*, 32(1-2):127–154, 2000.

[52] Marc Shröder. *Speech and Emotion Research: An overview of research frameworks and a dimensional approach to emotional speech synthesis*. PhD thesis, Phonus 7, Research Report of the Institute of Phonetics, Saarland University, 2004.

[53] Eftychios Sifakis, Andrew Selle, Avram Robinson-Mosher, and Ronald Fedkiw. Simulating speech with a physics-based facial muscle model. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 261–270, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[54] Matthew Stone, Doug DeCarlo, Insuk Oh, Christian Rodriguez, Adrian Stere, Alyssa Lees, and Chris Bregler. Speaking with hands: creating animated conversational characters from recordings of human performance. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 506–513, New York, NY, USA, 2004. ACM.

[55] Charles Sutton and Andrew Mccallum. *Introduction to Conditional Random Fields for Relational Learning*. MIT Press, 2006.

[56] Jacques Terken. Fundamental frequency and perceived prominence of accented syllables. *The Journal of the Acoustical Society of America*, 89(4):1768–1777, 1991.

[57] Jing Wang and Bobby Bodenheimer. Synthesis and evaluation of linear motion transitions. *ACM Transactions on Graphics*, 27(1):1:1–1:15, Mar 2008.

[58] Sy Bor Wang, A. Quattoni, L.-P. Morency, D. Demirdjian, and T. Darrell. Hidden conditional random fields for gesture recognition. volume 2, pages 1521–1527, 2006.

[59] Andrew D. Wilson, Student Member, Aaron F. Bobick, Ieee Computer Society, and Ieee Computer Society. Parametric hidden markov models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:884–900, 1999.

[60] Jianxia Xue, J. Borgstrom, Jintao Jiang, Lynne E. Bernstein, and Abeer Alwan. Acoustically-driven talking face synthesis using dynamic bayesian networks. *IEEE International Conference on Multimedia and Expo*, pages 1165–1168, July 2006.

[61] Liwei Zhao. *Synthesis and Acquisition of Laban Movement Analysis Quantitative Parameters for Communicative Gestures*. PhD thesis, University of Pennsylvania, 2001.

[62] Liwei Zhao and Norman I. Badler. Acquiring and validating motion qualities from live limb gestures. *Graphical Models*, 67(1):1–16, 2005.