

HMM Parameter Reduction for Practical Gesture Recognition

Stjepan Rajko and Gang Qian
Arts, Media and Engineering Program
Arizona State University, Tempe, AZ
srajko@asu.edu

Abstract

We examine in detail some properties of gesture recognition models which utilize a reduced number of parameters and lower algorithmic complexity compared to traditional hidden Markov models. We show that the reduced parameter models are comparable to standard HMM-based gesture recognition models in their ability to effectively model gestures, and in some cases superior when training data is limited. We also show that in order to effectively differentiate similar gestures, a gesture recognition model must utilize a large number of states, a scenario which can only be adequately handled by reducer parameter methods to maintain real-time speeds.

1. Introduction

Hidden Markov models (HMMs) are the basis of most gesture recognition algorithms used today. However, traditional HMM-based gesture recognition systems require a large number of parameters to be trained in order to give satisfying recognition results. In particular, an n state HMM requires $\Theta(n^2)$ parameters to be trained for the transition probability matrix, which limits its usability in environments where training data is limited.

Recently, Rajko et al. [6] presented a variation on HMMs which reduces the number of parameters required to infer all transition probabilities to $\Theta(1)$. In addition, their proposed model reduces the computational complexity of the inference algorithm, permitting the number of states used in the model to increase significantly while preserving real-time applicability. In this paper, we compare the properties of a $\Theta(1)$ parameter model similar to the one proposed in [6], a $\Theta(n^2)$ parameter HMM, as well as two new extensions of the proposed parameter-reducing model. One of the new extensions uses $\Theta(n)$ parameters for the transition probabilities, and the other uses $\Theta(1)$ parameters while giving better results than the model adapted from [6].

To evaluate the four models, we focus on a particular scenario in which each model is trained and then tested on

trials of a captured gesture, as well as being tested on a synthesized gesture similar to the captured gesture. For each testing trial, we evaluate how well each model can model the trial by calculating the probability that the gesture was generated by the model. In addition, we examine how well the model can separate the two gestures.

Our experimental results show that the reduced parameter models can perform on par with or better than the standard $\Theta(n^2)$ parameter HMM, especially when training data is limited. Additionally, the reduced parameter models require far less CPU time for training and inference. We have released our implementation of all of the models and experiments in an open source library [5].

1.1. Previous Work

There is an enormous body of work covering the use of HMMs in gesture recognition, as well as in other pattern recognition applications. For a review of the method, we refer to [4, 3]. As our paper offers little in terms of applying gesture recognition to concrete applications, we will omit references to the many publications which discuss applications of gesture recognition (many such references can be found in the publications we cite).

To follow our discussion, the reader should be familiar with the basics of hidden Markov model use in gesture recognition, as well as the use of the expectation-maximization algorithm [2] in training hidden Markov models. In addition, we recommend a basic understanding of the parameter reduction method presented in [6]. In particular, we will make use of the concept of *auxiliary states* from [6]. Such states are essentially non-emitting HMM states, with the novelty lying in how they are used to reduce the computational complexity of the training and inference algorithms.

To cover the broader context of gesture recognition, we should mention scenarios in which HMMs or the related reduced parameter models are not appropriate. Such basic hidden Markov models are most effective in situations where the observations at different times are independent of each other - i.e., each observation only depends on the un-

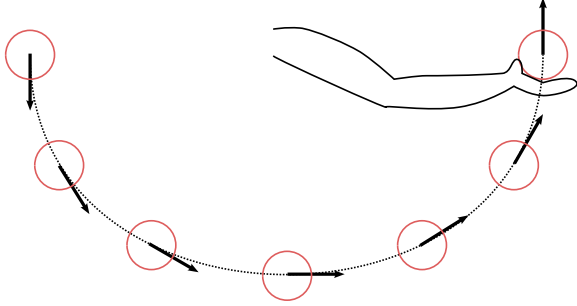


Figure 1. Movement of the example arc gesture.

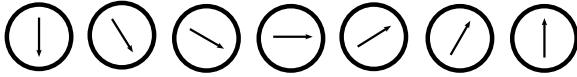


Figure 2. States of the example arc gesture (when modeled by an HMM with 7 emitting states).

derlying hidden gesture state. For scenarios where the observation independence does not hold, other methods have been proposed, such as coupled hidden Markov models, hidden conditional random fields, and maximum entropy hidden Markov models [1, 8, 7]. We have not yet sufficiently investigated how the parameter reduction models presented here relate to such models.

2. Preliminaries

For the purposes of this paper, we will use a straightforward hidden Markov model which has been augmented to support non-emitting states (labeled auxiliary states in [6]). As there are many different notation conventions in use for hidden Markov models, here we will utilize a convention we believe will be easy to remember while reading this paper.

We thereby define our augmented hidden Markov model ($S = \{E, N\}, s_b, s_e, T, O$) by a set of states S , a designated beginning state s_b , a designated ending state s_e , a state transition probability function T , and an observation probability function O .

The augmented HMM behaves essentially the same as a regular HMM, with only a few points of departure. The set of states S is divided into disjoint sets of emitting states E and non-emitting states N . The difference between the two is that when entered, emitting states emit an observation belonging to the observation set \mathbb{O} according to the observation probability function $O : E \times \mathbb{O} \rightarrow [0, \infty)$. The model always begins in the beginning state $s_b \in S$, and until it ends up in the ending state $s_e \in S$ it makes transitions according to the transition probability function $T : (S - s_e) \times S \rightarrow [0, \infty)$. T must also satisfy that the sum of transition probabilities out of any state is 1.

3. Gesture Recognition Models

All of the models described in this paper take a standard HMM approach to modeling gesture. They represent the gesture using a set of states where each state represents a stage of the gesture. For example, consider the gesture shown in Figure 1. The gesture is expressed through the arc-like motion of a tangible object (ball) held in the user’s hand. We can divide the gesture into a number of stages, each represented by the direction of the moving ball. Each stage will correspond to a state in the HMM, as shown in Figure 2.

The four models we use in this paper differ in the way they model the transition probabilities between the states. Since this is the main difference between the models, and this paper focuses on how these differences influence the performance of each model, when we explain the training of the models we will not go into detail regarding other aspects of the model (such as the observation probability distribution O , which is trained equivalently for each of the models and requires $\Theta(n)$ parameters).

Here is a brief overview of the models we will introduce with an indication of how many parameters are needed to determine the transition probabilities. We also introduce the identifiers (in *italics*) we will use to refer to each model in later discussion and experimental results:

- *standard* : A standard HMM model with no backward transitions ($\Theta(n^2)$ parameters)
- *reduced* : A model whose transition probabilities can be defined by 3 parameters for each state ($\Theta(n)$ parameters)
- *constant* : A model whose transition probabilities can be defined by only 3 parameters, and identical to each of the 3 parameters from the *reduced* model having constant values across all states ($\Theta(1)$ parameters)
- *zigzag* : A model whose transition probabilities can be defined by only 4 parameters, and identical to some of the parameters from the *reduced* model following a simple ”zig-zag” pattern across states ($\Theta(1)$ parameters)

3.1. *standard* Model

The *standard* HMM we use is composed entirely of emitting states, except for the starting and ending states s_b and s_e . In a model with n emitting states $E_1 \dots E_n$, we enumerate the states as $S_0 = s_b, S_1 = E_1, S_2 = E_2, \dots, S_n = E_n, S_{n+1} = s_e$. To specify the transition probabilities between each pair of states, the standard HMM uses a transition probability matrix:

$$T(S_i, S_j) = \begin{cases} 0, & \text{if } j < i \text{ or } i = n + 1 \\ t_{i,j}, & \text{otherwise} \end{cases} \quad (1)$$

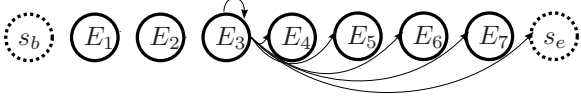


Figure 3. Transitions for a *standard* HMM model with 7 emitting states. Only transitions out of one state (E_3) are displayed for clarity. In general, any state can have a non-zero probability specified for transitions to itself or to any state depicted to its right.

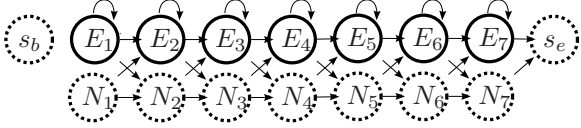


Figure 4. Transitions for the reduced parameter models with 7 emitting states. For simplicity, we omit transitions from the beginning state s_b . The displayed transitions are determined differently for each of the reduced parameter models.

To train the *standard* HMM model, the EM algorithm is used. The model is initialized using reasonable initial values, and then used on a number of training instances of a gesture. Each training instance yields a sequence of states that has most likely produced the observations in the training gesture. The state sequences are then used to update the parameters. For example, we update the value for $t_{i,j}$ using the percentage of times there was a transition from S_i to S_j . The expectation of each transition probability is calculated separately, yielding $\Theta(n^2)$ transition parameters to be trained independently.

3.2. reduced Model

The *reduced* model uses a constant number of parameters for each state to determine transition probabilities between all states. The parameters correspond to the probabilities of transitions depicted in Figure 4.

In this case, we use the following parameters (we highlight in bold *italic* the letter that might help you remember the symbol due to visual similarity):

- τ_i is the probability of **remaining** in an emitting state ($T(E_i, E_i)$)
- η_i is the probability of going to the **next** emitting state ($T(E_i, E_{i+1})$)
- ς_i is the probability of **skipping** at least one emitting state ($T(E_i, N_{i+1})$)
- κ_i is the probability of **skipping** an additional emitting state ($T(N_i, N_{i+1})$)
- ρ_i is the probability of ending a **skip** sequence ($T(N_i, E_{i+1})$)

The parameters pertaining to an individual pair of emitting / non-emitting states is shown in Figure 5. Note that given these parameters for each state, we can construct an entire transition probability matrix that would correspond to

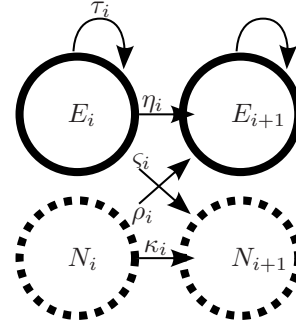


Figure 5. Transition probability parameters for the *reduced* model.

the transition matrix used in the *standard* model (focusing on the transitions between emitting states):

$$T_{standard}(E_i, E_j) = \begin{cases} 0, & \text{if } j < i \\ \tau_i, & \text{if } j = i \\ \eta_i, & \text{if } j = i + 1 \\ \varsigma_i \prod_{k=i+1}^{j-2} \kappa_k \rho_{j-1}, & \text{if } j > i + 1 \end{cases} \quad (2)$$

Since we constrain each state's outgoing probabilities to sum to 1, we note the following constraints:

$$\tau_i + \eta_i + \varsigma_i = 1 \quad (3)$$

$$\kappa_i + \rho_i = 1 \quad (4)$$

Hence, it is sufficient to determine three of the parameters for each state to construct all probabilities of transition coming out of a state.

Training is again performed using the EM algorithm, but now focused on the expectation and maximization of only the 5 parameters for each state. For example, τ_i and η_i are determined much like $T(E_i, E_i)$ and $T(E_i, E_{i+1})$ in the *standard* HMM case. κ_i is determined by looking at all transitions in which states E_i and E_{i+1} were skipped, while for ρ_i we look at transitions in which E_i was skipped but E_{i+1} was not.

3.3. constant Model

The *constant* model uses a total of 5 parameters to determine transition probabilities between all states. It is an adaptation of the model presented in [6]. The states and transitions can again be seen in Figures 4 and 5, but the transition probabilities are determined slightly differently than with the *reduced* model. In particular, we enforce that each of the 5 parameters has the same value for every state - i.e., $\tau_i = \tau$ for some value τ and all i , $\eta_i = \eta$ for some value η and all i , etc.

Thus, we can again can construct the transition probability matrix between emitting states that would correspond to

the *standard* model:

$$T_{standard}(E_i, E_j) = \begin{cases} 0, & \text{if } j < i \\ \tau, & \text{if } j = i \\ \eta, & \text{if } j = i + 1 \\ \varsigma \kappa^{j-i-2} \rho, & \text{if } j > i + 1 \end{cases} \quad (5)$$

Since we constrain each state's outgoing probabilities to sum to 1, we note the following constraints:

$$\tau + \eta + \varsigma = 1 \quad (6)$$

$$\kappa + \rho = 1 \quad (7)$$

Hence, it is sufficient to determine three of the parameters to construct all transition probabilities.

Training is again performed using the EM algorithm, and similar to the training of the *reduced* model. The difference is that the parameters which have individual values for the *reduced* model are now constrained to have identical values for the *constant* model. Thus, for example, to determine the value of τ , we would look at the percentage of times an emitting state (any emitting state) has transitioned to itself (versus transitioning to any other state).

3.4. zigzag Model

The *zigzag* model is a simple example of using a low number of parameters ($\Theta(1)$) while allowing some variation in the values of the 5 parameters across the states. Specifically, the model alternates between two sets of values for τ and η . It is a special case of the *reduced* model where $\tau_i = \tau_{i+2}$ and $\eta_i = \eta_{i+2}$ for all i (i.e., for odd i all τ_i are the same, and for even i all τ_i are the same, and similarly so for η values), while the other 3 parameters have constant values across all states (ς , κ , and ρ). We will denote the two values used for τ_i and η_i by τ_{odd} , η_{odd} , τ_{even} and η_{even} .

This model was inspired by the results we obtained in our experiments on the gesture shown in Figure 1. For that particular gesture, we found that the parameters τ_i and η_i , when graphed over all i , tended to exhibit a "zig-zag" pattern after training. The *zigzag* model was constructed to examine what happens if we build this pattern into the model.

The training of the *zigzag* model is similar to that of the *reduced* and *constant* models. It is identical to *constant* for values ς , κ , and ρ . To determine the values of τ_{odd} , η_{odd} , τ_{even} and η_{even} , suppose we obtain the individual values of τ_i and η_i as we would for the *reduced* model. Let τ_{mean} and η_{mean} be their respective means, and η_{dev} be the expectation of $|\eta_i - \eta_{mean}|$. We set the following:

$$\tau_{odd} = \tau_{mean} + \eta_{dev} \quad (8)$$

$$\eta_{odd} = \eta_{mean} - \eta_{dev} \quad (9)$$

$$\tau_{even} = \tau_{mean} - \eta_{dev} \quad (10)$$

$$\eta_{even} = \eta_{mean} + \eta_{dev} \quad (11)$$

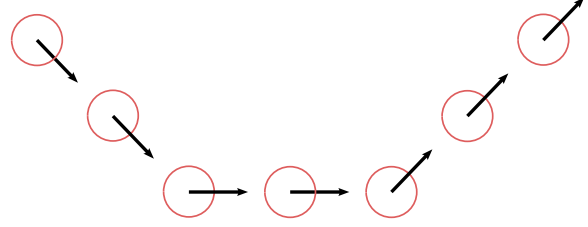


Figure 6. Synthesized gesture similar to the gesture shown in Figure 1, also used for experimental results.

4. Experimental Results

To test the training requirements and gesture recognition effectiveness of each model, we implemented each of the four models and tested it on collected gesture data of a tangible object (ball) moving in a 3D space, as well as on synthesized ball data. For the collected data, the position of the ball was provided by a 6-camera tracking system at 60 frames per second (resulting in ≈ 50 -80 frames per trial).

The gesture we collected was the simple arcing gesture shown in Figure 1. We collected 100 trials of the gesture, which were divided into 30 training and 70 testing trials. In addition, we synthesized 70 testing trials of a gesture similar to the capture gestured, which is shown in Figure 6.

Our experiment was as follows. Each model was trained using a variable number of collected gesture training trials, and tested on the 70 testing trials of the collected and synthesized gesture data. We then measured the likelihood of each testing trial given by each gesture model, resulting in two mean likelihoods for each model (one mean for the collected gesture test trials, and one mean for the synthesized gesture test trials). The likelihood for each test trial was scaled by the number of observations in the test trial (this makes the measurements meaningfully comparable when the number of observations differs). In cases where we used less than 30 training trials to train the models, we repeated the experiment multiple times with different sets of training trials so that all training trials are used at some point. For example, when training with 15 training trials, we divided the training trials into two groups (1-15 and 16-30), and repeated the experiment for each group. In such cases, the mean likelihoods reported are a mean of means over all groups.

We briefly note that the above evaluation strategy is not the typical way in which gesture recognition algorithms are evaluated. Typically, gesture recognition algorithms are given data sets containing various gesture and non-gesture data, and their performance is evaluated according to how well they classify the gestures. In our case, we take a different approach because we are precisely interested in the ability of the different models to effectively model a particular gesture - which can be measured by looking at the likelihoods.

To make the comparison fair, the four models were all initialized to the same transition probabilities before training, and trained with enough iterations of the EM algorithm to reach a maximum. We also implemented the models so they share as much code as possible, adding fairness to the runtime measurements.

We will first present the logarithm of the ratios of the mean likelihood of the collected gesture versus the mean likelihood of the synthesized gesture. This measurement reflects the ability of each model (trained with trials of the collected gesture) to separate the trials of the collected gesture from the trials of the similar synthesized gesture. If the log of the ratio is negative, it means that the synthesized gesture fits the model better. If the log of the ratio is large and positive, it means that the model is able to effectively separate the collected gesture from the synthesized gesture. Note that either behavior might be desirable or undesirable, depending on the scenario. Since the synthesized gesture is quite similar to the collected gesture, it could be argued that each model should recognize trials of the synthesized gesture even when it was trained only on trials of the collected gesture, and behavior to the contrary identified as overfitting. However, if the scenario requires that similar gestures be effectively separated, then it is desirable for the model to do so.

The results indicate that with a low number of states (3), none of the models is able to reliably distinguish the two gestures. With a higher number of states (5), the performance becomes reasonable only with a higher number of training trials. The highest number of states (30 and 60) show the best separation results. This shows that a high number of states is necessary in situations where it is important to reliably distinguish similar gestures from each other. Also, note that in such scenarios, each of the models does about equally well in separating the two gestures.

Next, we look at the CPU runtime costs associated with inference in each model. Results from Figure 7 show the superior runtime performance of the reduced parameter models, which increases only linearly with the number of emitting states (as opposed to quadratically in the case of the *standard* model). Note that with 60 emitting states, the *standard* HMM is reaching the limit of real-time performance, taking 1.5 seconds to process a gesture that takes about the same amount of time.

Finally, we examine the logarithms of mean likelihoods of the collected gesture trials alone. This measurement reflects the ability of each model to discriminate the collected gesture on which it was trained from completely dissimilar gesture models (such as a non-gesture model which might utilize a uniform distribution over all observations).

The results shown in Table 2 indicate that overall, modeling is better with a higher number of emitting states. Also, with a lower number of emitting states (3 and 5), the re-

emitting states	training trials	<i>constant</i>	<i>zigzag</i>	<i>reduced</i>	<i>standard</i>
3	2	-13.4	-13.5	-13.4	-5.4
3	3	-5.5	-5.5	-5.5	-3.8
3	5	-2.8	-2.8	-2.9	-2.0
3	15	-0.8	-0.8	-0.8	-0.6
3	30	-0.7	-0.7	-0.7	0.2
5	2	-12.4	-7.5	-6.4	-12.6
5	3	-5.1	-4.8	-5.2	-8.3
5	5	-4.8	-4.6	-4.8	-4.6
5	15	5.7	5.2	5.3	9.7
5	30	2.8	3.2	4.6	9.3
10	2	4.8	9.5	9.1	8.3
10	3	10.7	12.1	12.0	13.9
10	5	10.5	9.9	11.0	8.3
10	15	12.2	13.7	12.5	12.3
10	30	12.3	12.1	12.2	12.2
30	2	30.3	25.4	30.1	29.8
30	3	38.1	31.1	40.2	40.2
30	5	38.9	39.1	38.2	39.0
30	15	41.9	28.7	34.1	34.4
30	30	35.8	34.1	32.0	33.5
60	2	29.2	30.0	81.1	49.2
60	3	39.7	39.2	50.8	50.9
60	5	44.6	42.2	42.7	45.6
60	15	37.3	38.0	35.4	37.7
60	30	40.0	40.3	37.8	39.0

Table 1. The logarithm of ratios of likelihoods of the collected gesture trials versus similar synthesized gesture trials. Negative numbers indicate higher likelihood for synthesized gesture trials, while positive ones indicate higher likelihood for collected gesture trials. High positive numbers indicate good separation of gesture data from the non-gesture data. Note that a large number of states is necessary to separate the two similar gestures, regardless of the model.

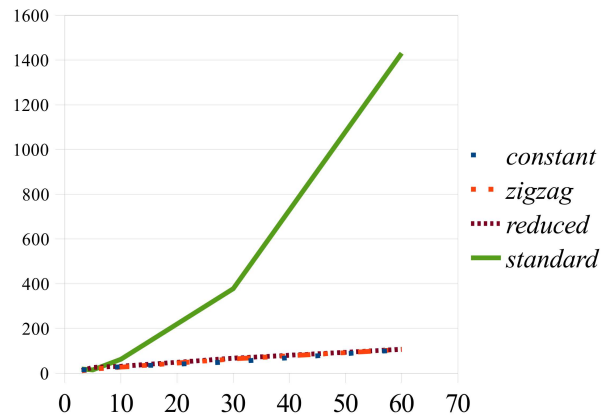


Figure 7. The CPU time (in milliseconds) of inference using each model, averaged over all test cases for a particular number of emitting states. The runtime of the reduced parameter models increases linearly with the number of emitting states, while the runtime of the *standard* HMM increases quadratically.

emitting states	training trials	<i>constant</i>	<i>zigzag</i>	<i>reduced</i>	<i>standard</i>
3	2	-12.7	-12.7	-12.7	-21.4
3	3	-12.8	-12.8	-12.8	-21.4
3	5	-12.7	-12.6	-12.7	-20.9
3	15	-12.6	-12.6	-12.6	-21.1
3	30	-12.7	-12.7	-12.7	-21.0
5	2	-6.1	-6.0	-6.2	-12.4
5	3	-6.2	-6.1	-6.2	-11.6
5	5	-6.3	-6.3	-6.4	-10.6
5	15	-7.5	-7.4	-7.5	-11.0
5	30	-7.3	-7.2	-7.1	-11.2
10	2	1.7	1.5	1.6	-1.0
10	3	-1.2	-1.5	-1.3	-1.3
10	5	-1.9	-2.3	-1.5	-2.6
10	15	-3.1	-3.0	-2.8	-3.5
10	30	-3.2	-3.1	-2.6	-3.1
30	2	-7.1	-10.2	-4.5	-4.5
30	3	-5.1	-1.1	-1.1	-1.0
30	5	-2.8	-0.0	2.4	2.1
30	15	-2.8	-0.2	1.3	0.2
30	30	-3.2	2.0	2.4	1.8
60	2	-26.8	-27.2	-26.5	-26.4
60	3	-6.0	-5.6	-4.6	-4.5
60	5	-1.9	-0.9	0.3	0.2
60	15	-1.9	-0.7	1.3	1.1
60	30	-0.4	0.5	1.8	2.4

Table 2. The log-likelihoods of the collected gesture test trials. Higher values indicate better ability of the model to recognize test trials of the collected gesture after being trained on a number of training trials.

duced parameter models outperform the *standard* HMM model. With a higher number of emitting states, most models perform similarly, except for the *constant* model which lags slightly. The *zigzag* model is closer in performance to the *reduced* model, showing how it is possible to keep $\Theta(1)$ transition probability parameters while improving performance over the *constant* model.

5. Conclusions and Future Work

We have presented a specific analysis of four different gesture recognition models, and came to the following conclusions:

- a large number of emitting states is necessary to distinguish between similar gestures
- only reduced parameter models show acceptable run-time performance with a high number of emitting states
- models with $\Theta(1)$, $\Theta(n)$ and $\Theta(n^2)$ parameters determining transition probabilities (*zigzag*, *reduced*, and *standard*) all show comparable performance results in most scenarios.

Even though the analysis in this paper is focused on a very specific scenario (a relatively simple gesture expressed in a unimodal sensing environment), it shows that the reduced parameter models can be a good tool for practical, real-time gesture recognition applications. We will continue to expand our study to include more variability in the gesture recognition scenarios. Those interested can find examples of other scenarios in the provided implementation [5].

Also, this paper only focuses on parameters for transition probabilities. There are also parameters associated with the observation distribution O . In our implementation, we used $\Theta(n)$ such parameters - a mean and a variance describing a normal distribution for each emitting state. But these parameters could also be reduced - for example by enforcing that all variances should be the same.

Some other directions we are interested in involve the use of basis functions to describe the parameters. This is similar to the *constant* and *zigzag* models, which essentially fit a constant or "zig-zag" function to the values of each of the 5 parameters in the *reduced* model. Other basis functions could be applicable to other scenarios, and an automated method (based on automatic selection algorithms) could determine which basis functions are most appropriate. This would improve on the method we used to come up with the *zigzag* model, which was essentially looking at the trained parameters in the *reduced* model and finding them to look more "zig-zag" the more they are trained.

References

- [1] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. *Computer Vision and Pattern Recognition*, pages 994–999, 1997. **2**
- [2] T. Moon. The expectation-maximization algorithm. *Signal Processing Magazine, IEEE*, 13(6):47–60, 1996. **1**
- [3] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UNIVERSITY OF CALIFORNIA, 2002. **1**
- [4] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. **1**
- [5] S. Rajko. AME Patterns Library [computer software]. <http://ame4.hc.asu.edu/amelia/patterns/>, 2008. **1, 6**
- [6] S. Rajko, G. Qian, T. Ingalls, and J. James. Real-time Gesture Recognition with Minimal Training Requirements and On-line Learning. *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8, 2007. **1, 2, 3**
- [7] C. Sminchisescu, A. Kanaujia, and D. Metaxas. Conditional models for contextual human motion recognition. *Computer Vision and Image Understanding*, 104(2-3):210–220, 2006. **2**
- [8] S. Wang, A. Quattoni, L. Morency, D. Demirdjian, and T. Darrell. Hidden Conditional Random Fields for Gesture Recognition. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 2, 2006. **2**