Continuous Realtime Gesture Following and Recognition

Frédéric Bevilacqua, Bruno Zamborlin, Anthony Sypniewski, Norbert Schnell, Fabrice Guédy, and Nicolas Rasamimanana

Real Time Musical Interactions Team,
IRCAM, CNRS - STMS,
1, Place Igor Stravinsky, 75004 Paris, France
Frederic.Bevilacqua@ircam.fr

Abstract. We present a HMM based system for real-time gesture analysis. The system outputs continuously parameters relative to the gesture *time progression* and its *likelihood*. These parameters are computed by comparing the performed gesture with stored reference gestures. The method relies on a detailed modeling of multidimensional temporal curves. Compared to standard HMM systems, the learning procedure is simplified using prior knowledge allowing the system to use a single example for each class. Several applications have been developed using this system in the context of music education, music and dance performances and interactive installation. Typically, the estimation of the *time progression* allows for the synchronization of physical gestures to sound files by time stretching/compressing audio buffers or videos.

Keywords: gesture recognition, gesture following, Hidden Markov Model, music, interactive systems.

1 Introduction

Gesture recognition systems have been successively developed based on methods such as Hidden Markov Models (HMM), finite state machines, template matching or neural networks [1]. In most cases, gestures are considered as "unbreakable units" that must be recognized once completed. Typically, on-line systems output the recognition result at the end of each gesture. Motivated by the development of interactive systems in performing arts, we present here a different approach for online gesture analysis : the system updates "continuously" (i.e. on a fine temporal grain) parameters characterizing the performance of a gesture. Precisely, these parameters are made available during the temporal unfolding of the performed gesture.

We are first particularly interested in computing the *time progression* of the performance, or in other words answering the question "where are we within the gesture ?". We refer this as *following* the gesture. Second, we are interested in computing *likelihood* values between a performed gesture and pre-recorded gestures stored in a database. This can be used to perform a *recognition* task, but

S. Kopp and I. Wachsmuth (Eds.): GW 2009, LNAI 5934, pp. 73–84, 2010.

also to characterize gestures. As this will be illustrated by application examples, these parameters are particularly useful to build systems enabling expressive gestural control of sonic and/or visual media. Moreover, the estimation of both the *time progression* and *likelihood* values enable another important feature of such a system: the possibility to predict the evolution of the current gesture.

We assume here that *gestures* can be represented as multidimensional temporal curves. Importantly, our approach focuses on a detailed modeling of temporal profiles. High resolution temporal modeling is indeed essential for the estimation of the *time progression* of a gesture. This approach is thus especially suited for cases where the gesture temporal evolution are intrinsically relevant, and performed in a consistent manner. This is typically found in performing arts: measurements of dancers or musicians gestures reveal very consistent temporal profiles[2,3].

Our system is essentially based on Hidden Markov Models, with a particular implementation guaranteeing precise temporal modeling of gesture profiles and allowing for a simplified learning procedure. This latter point is essential for making such a system workable in the context of performing arts. As a matter of fact, building general gesture databases can reveal to be unpractical, as also discussed in Ref.[4], since gesture data are typically highly dependent on the artistic contexts and idiosyncrasies of performers. For these reasons, we developed a particular learning scheme based on a single recorded example only, using a priori knowledge. In this case, the learning phase is simple to operate and can be easily achieved during the normal flow of rehearsals. This approach has been iteratively designed through several specific cases[5,6,7] and implemented in a software called the *Gesture Follower*.

This paper is structured as follows. After a short summary of related works, we describe the algorithm, along with numerical simulations using synthetic data assessing quantitatively the algorithm. Second, we present typical applications of this system related to music and dance practices.

2 Related Works

The use of machine learning techniques for gesture recognition has been widely covered. Hidden Markov Models represents one of the mostly used methods [1,8,9]. Taking notice that training Hidden Markov Models might represent a cumbersome task, several authors proposed various approaches to facilitate the training process. Bobick and Wilson have proposed a state-based approach using a single prototype [10]. Using HMM they also later proposed an online adaptive algorithm for learning gesture models [11]. Rajko et al. also proposed a HMM based system, with the aim of reducing training requirements and allowing precise temporal gesture modeling [4,12,13]. Artieres et al. [14] proposed a recognition scheme based on segmental HMM that can be trained with very few examples.

Concerning realtime recognition Bloit and Rodet [15] developed a modified Viterbi decoding, called short term Viterbi, that allows for low latency recognition.

Mori et al. [16] proposed a system for early recognition and anticipation, based on continuous dynamic programming.

These works are generally principally oriented towards recognition tasks. The case of *following* a gesture in realtime, i.e. determining the time progression of a gesture during a performance, is generally not explicitly covered with the exception of score following systems for musical applications. Several authors have proposed systems based on HMM (or Semi-Markov Models) [17,18]. Nevertheless, in such cases, the Markov structure is essentially built from a symbolic representation given by the musical score, and not from continuous gesture data.

3 Gesture Modeling and Analysis Algorithm

As generally found in machine learning techniques, the system operation is separated into two procedures, learning and decoding. Our approach is based on Hidden Markov Models but with a modified learning schema. The algorithm fundamentally works with any type of regularly sampled multidimensional data flow.

3.1 Learning

Our system has been developed with the constraint that only few examples will be available. This constraint is incompatible with the use of statistical training (for example using the Baum-Welch algorithm), as found in standard implementations of Hidden Markov Models (HMM) [8].

The learning procedure is illustrated in Figure 1. The recorded temporal profile is used to create a left-to-right Hidden Markov Model. We build a model that fits the recorded reference by directly associating each sampled points to a state in a left-to-right Markov chain.

Each state i emits an observable O with a probability b_i, following a normal distribution (the vectorial case can be generalized in a straightforward way):

$$b_i(O) = \frac{1}{\sigma_i\sqrt{2\pi}} exp[-(\frac{O-\mu_i}{2\sigma_i})^2] \tag{1}$$

μ_i is the *ith* sampled value of the recorded reference. The σ_i parameter can be interpreted as the standard deviation of differences in x occurring between performances. Obviously, σ_i cannot be estimated from a single example. Therefore, it is either estimated for a given context based on prior experiments and/or knowledge. The influence of this parameter will be further discussed in section 3.4.

This HMM structure statistically models the recorded data sequence, considering additional assumptions on the transition probabilities. We define a limited number of permitted transitions by setting the transition probabilities a_0, a_1, a_2 (self, next, skip transitions, see Figure1). These probabilities a_0, a_1, a_2 must satisfy the Equation 2.

$$a_0 + a_1 + a_2 = 1 \tag{2}$$

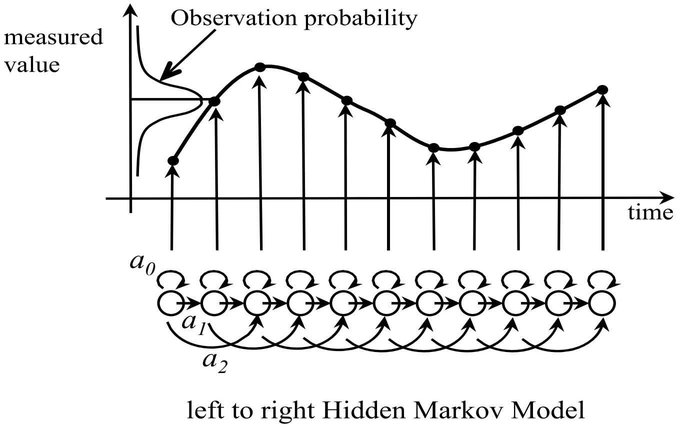

Fig. 1. Learning procedure: a left-to-right HMM is used to model the recorded reference

As for σ_i, these parameters cannot be precisely estimated from a single example. Nevertheless, their values can be set based on prior knowledge or measurements in specific applications. The following discussion clarifies the role of these parameters.

1. $a_0 = a_1 = a_2 = 1/3$: this case corresponds to have equal probabilities slower or faster performance of the gesture.
2. $a_0 < a_1$ and $a_2 < a_1$ this case corresponds to have lower probability for speeding up or slowing down.
3. $a_0 < a_2$ this case corresponds to have lower probability for slowing down than speeding up
4. $a_0 > a_2$ this case corresponds to have higher probability for slowing down than speeding up.

Note that the relative maximum speed (between the performed and reference gesture) is 2 in the example shown in Figure1. This value can higher by setting additional transitions (for example $a_3 > 0$).

Based on experiments (see section 4), we found that empirical values such as $a_0 = a_1 = a_2 = 1/3$ or $a_0 = a_2 = 0.25$ and $a_1 = 0.5$ work for a large set of applications. A similar discussion can be found in [4].

We also implemented a further simplified HMM structure. As described in [6], a HMM structure with only two possible transitions, self and next transitions can be used considering $a_0 = 1/n$ and $a_1 = 1 - 1/n$, and downsampling the recorded reference by a factor n. This configuration has been used for the assessments described in section 3.4.

3.2 Decoding

As explained in the introduction, we are interested in two types of quantities: *time progression index* and *likelihood* values, computed from the online comparison between the gesture being performed and recorded references. Similarly to score following algorithms [17], we base our decoding scheme on the standard forward procedure in HMM [8].

Consider the partial observation sequence $O_1, O_2, ...O_t$. The forward procedure requires the computation of the $\alpha_i(t)$ variable which corresponds to the probability distribution of the partial observation sequence until time t, and state i . It is computed inductively as follows:

Initialisation

$$\alpha_1(i) = \pi_i b_i(O_1) \qquad 1 \leq i \leq N \tag{3}$$

where π is the initial state distribution, and b is the observation probability distribution.

Induction

$$\alpha_{t+1}(i) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_i(O_t) \qquad 1 \leq t \leq T-1, 1 \leq j \leq N \tag{4}$$

where a_{ij} is the state transition probability distribution.

From the $\alpha_i(t)$ variable we can compute two important quantities:

1. Time progression of the sequence, related to the recorded example

$$time\,progression\;index(t) = argmax[\alpha_t(i)] \tag{5}$$

   Note that this *index* can be alternatively estimated by the mean (expected value) of the distribution $\alpha_i(t)$.
2. Likelihood of the sequence.

$$likelihood(t) = \sum_{i=1}^{N} \alpha_t(i) \tag{6}$$

   This quantity can been used directly as a *similarity measure* between the gesture being performed and the recorded reference. Other similarity measures could also be derived by combining the *likelihood* and the smoothness of the *time progression* index.

3.3 Windowing Technique

A major limitation of the algorithm described above is the large number of states of the HMM when dealing with long phrases, which can be an issue for real-time computation. For example, with a data sampling rate of 100 Hz, the number of states is typically 600 for an one minute phrase. Typically, a number of states larger than 1000 might be too CPU intensive for our applications. To avoid this problem, we developed a sliding window technique that uses a fixed number of

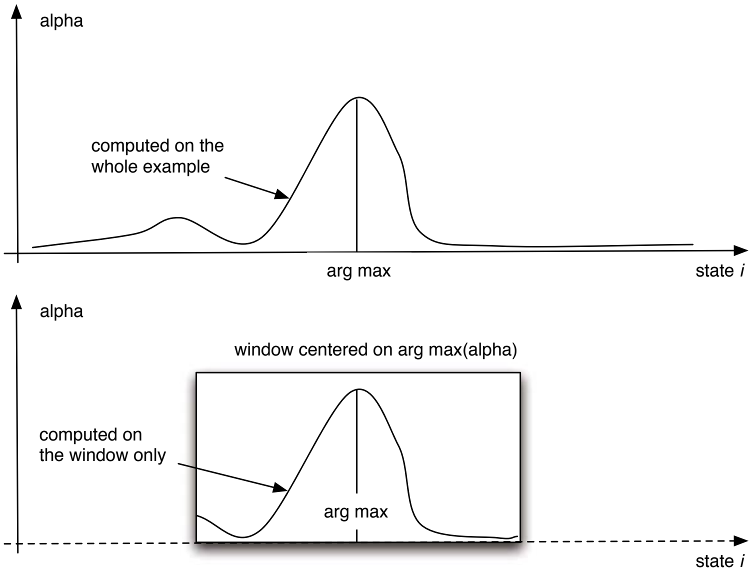

Fig. 2. Windowing technique used in the decoding scheme to reduce the CPU load

states, thus limiting the computation load, similarly to beam search techniques. The method is illustrated in Figure 2.

As explained earlier, the decoding procedure requires the computation of the probability $\alpha_i(t)$. In the basic algorithm, this probability distribution is computed on the entire state structure. In the windowed version, this computation is limited to a section of the state structure. Precisely, it is evaluated on a window centered around the arg max of the $\alpha_i(t)$ distribution, i.e. around the *time progression index*. At each new step of the evaluation, the window location is moved. α_i values that were not considered in the previous window are initialized to zero in the new window.

This technique allows for the computation with a fixed number of states, that is adjustable by the user. Thus the CPU load remains a constant value independent of the length of the gesture data. Tests showed that this method was effective.

Importantly, this technique can also be seen as adding constraints to the estimation of the *time progression index*, since the range of possible values is reduced at a given time. This procedure can make the estimation of the *time progression index* more robust to outlier data that could otherwise provoke unrealistic jumps.

3.4 Assessments on Synthetic Data

Simulations were performed with Matlab using synthetic signals to evaluate quantitatively the accuracy of the *time progression index*. As shown in Figure 3,

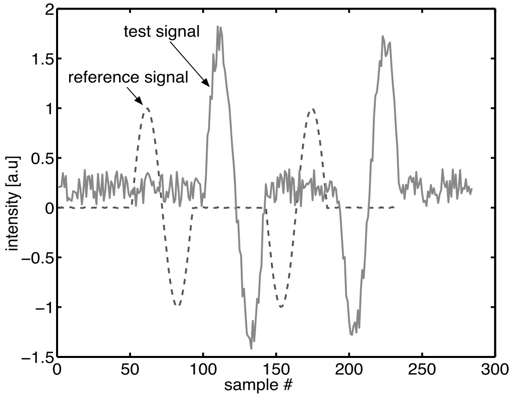

Fig. 3. Synthetic signals used for the algorithm assessment

reference and test signals were created by concatenating truncated sine functions and constant signals.

Precisely, the reference signal was obtained by concatenating the following parts: constant zero signal (50 samples), one period of a sine signal (40 samples), constant zero signal (50 samples), one period of a sine signal (40 samples), constant zero signal (50 samples) (total length = 230 samples). The tests signals were obtained from the reference signal by applying various transformations in the amplitude, offset and noise level.

The algorithm was applied to these altered test signals, and average errors obtained in the *time progression* index were computed. These error values can be associated to a time jitter. For example, with a sampling rate of 200 Hz, an error of one sample would correspond to a jitter of 5 ms.

The assessments reported here were performed in the case of the simplified HMM state structure, where we retain only two types of possible transitions, *self* and *next* transitions ($a_0 = a_1 = 0.5$). As noted in section 3.1, a downsampling of a factor 2 was thus applied to the reference signals. In this simplified case, the standard deviation σ_i is the only parameter to be adjusted. We performed a complete set of assessments varying the σ_i values.

Interestingly, we found that, as long as σ_i values lies in an given interval, the results for the *time progression* index are weakly affected by the σ_i absolute value. For example, considering reference signals normalized between -1 and 1, we found that σ_i should lie approximately in the interval [0.1 0.5]. This result confirmed us that our algorithm can operate in cases where the σ_i values are known only approximately.

Figure 4 (a), (b) and (c) show the results for three different signal alterations: scaling the amplitude, adding a constant offset and adding gaussian noise, respectively. The value σ_i is $= 0.2$ in all cases. These results show that, as expected, the accuracy in the estimation of the *time progression index* decreases while increasing the alteration level. Nevertheless, it is important to note that the errors

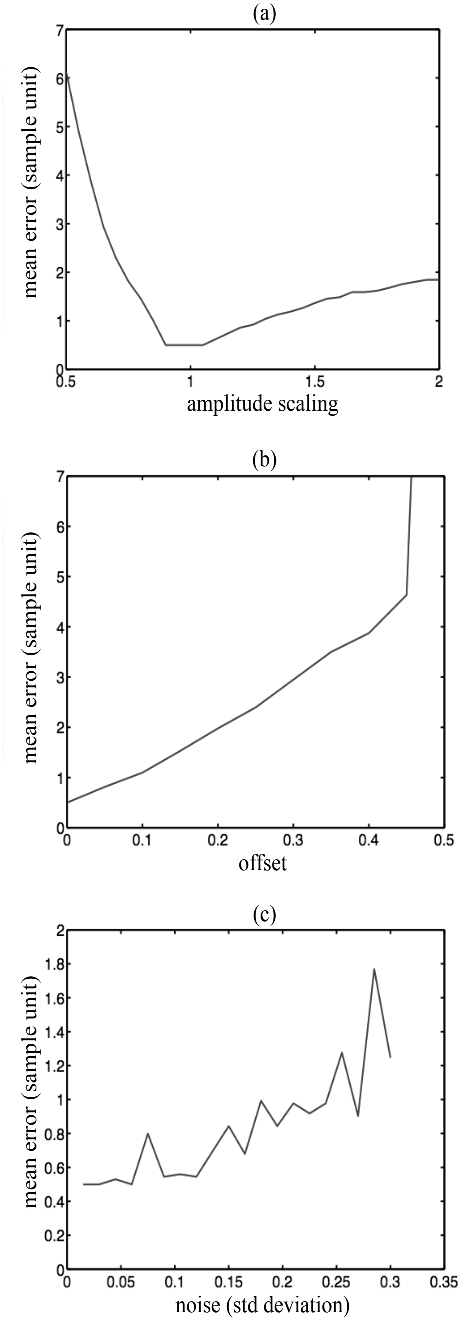

Fig. 4. (a) Error in the time progression index for various amplitude scaling of the test signals. (b) Error in the time progression index for various offset of the test signals. (c) Error in the time progression index for various noise levels of the test signals. In all cases, value $\sigma_i = 0.2$, and the reference signal were normalized between -1 and 1. In the cases of (a) and (b), a fixed gaussian noise of 1% was also added.

remain at an acceptable level considering relatively large alterations. For example, a test signal of amplitude twice the reference signal induces an average error less than 2 samples, for a total signal length of 250 samples (see Figure 3). Interestingly, the algorithm is more sensitive to a decrease than to an increase of the test signal amplitude. These results indicate that theses limits can be large enough to work with real data, which has been confirmed later during our applications as described in section 4.

4 Implementation and Applications

The system described in this paper is implemented as a collection of modules in the Max environnement[1] called the *Gesture Follower*, taking advantage of the data structures of the FTM library[2] such as matrices and dictionaries[19,20]. Recently, the core algorithm was developed as an independent C++ library and can therefore be implemented in other environnements.

We applied the *Gesture Follower* to build interactive systems in music and dance performances. The system was also applied to design experiments aimed at studying gesture and movement sonification [21].

Most of the applications make use of the *time progression index* given by the *Gesture Follower*. Particularly, this parameter allows for the design of applications based on a time synchronization paradigm: digital media and effects can be synchronized to any particular moments of a given reference gesture. This reference gesture is set by the user by a simple recording. Thanks to the windowing technique (section 3.3), there is no limitation other than the computer memory for the gesture length. This opens interesting perspectives, such as following an entire music performance. A particular interesting case is illustrated in Figure 5, were the *time progression index* is used to synchronize the speed of the playback of an audio recording. For example, audio time stretching/compressing can be performed using phase vocoder techniques.

Specific cases of this paradigm were experimented in the context of music pedagogy [22]. In particular, "virtual" conducting was achieved using a wireless sensor module transmitting hand gesture accelerations to the *Gesture Follower* (Figure 6). With this system, students were able to control precisely the playing speed of an orchestra recording.

A particular interesting feature of our system resides in the possibility to continuously compare the live performance with different interpretations previously recorded. For example, the *likelihood* value can be therefore used to control the sound intensity (see Figure 5) and further sound transformation.

Other applications were achieved in the fields of dance performance and interactive installations. Experiments showed that the system was able to distinguish easily between fifteen short dance phrases, based on 3D accelerometers wore on a dancer wrists. Since the *likelihood* parameters were continuously updated, it was

[1] http://www.cycling74.com

[2] http://ftm.ircam.fr

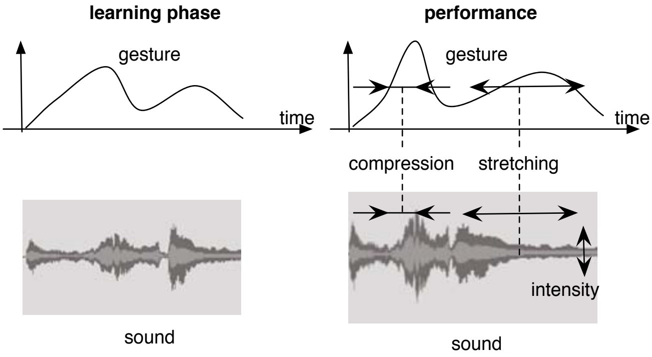

Fig. 5. Gesture synchronization to audio time stretching/compression

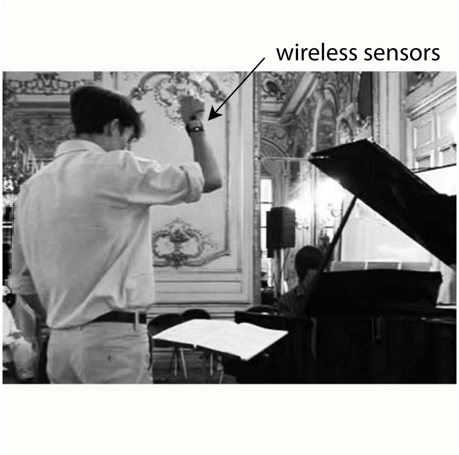

Fig. 6. "Virtual" conducting using wireless motion sensors (accelerometers and gyrsocopes and the *gesture follower*. The system synchronizes the gesture with the playback of an orchestra recording.

possible to recognize a gesture early, without waiting for its completion to operate a choice in the interaction process. Parallel to this recognition procedure, it was possible to effectively synchronize video materials to the dancer movements using the *time progression index*.

5 Conclusion and Future Work

We presented a HMM based system for real-time gesture analysis. The system relies on a detailed temporal modeling and outputs continuously two main parameters: the *time progression index* and *likelihood* values which can be used to estimate *similarities* between a gesture being performed and recorded references. One advantage of this system resides in the simplified learning process.

Various applications, mainly based on a *following* paradigm, were built with this system, and proved the validity of our approach. Refinements of this system are currently implemented and further applications are foreseen, especially taking advantage of the prediction capabilities of this system.

Acknowledgements

We acknowledge partial support of the following projects: The EU-ICT i-Maestro project, the EU-ICT Project SAME and the ANR project EarToy (French National Research Agency). We would like to thank Remy Muller for his important contribution in the early development of this work, Richard Siegal, Jean-Philippe Lambert, Florence Baschet, Serge Lemouton and the students of "Atelier des Feuillantines" for contributing to experiments.

References

1. Mitra, S., Acharya, T., Member, S., Member, S.: Gesture recognition: A survey. IEEE Transactions on Systems, Man and Cybernetics - Part C 37, 311–324 (2007)
2. Rasamimanana, N.H., Bevilacqua, F.: Effort-based analysis of bowing movements: evidence of anticipation effects. The Journal of New Music Research 37(4), 339–351 (2009)
3. Rasamimanana, N.H., Kaiser, F., Bevilacqua, F.: Perspectives on gesture-sound relationships informed from acoustic instrument studies. Organised Sound 14(2), 208–216 (2009)
4. Rajko, S., Qian, G., Ingalls, T., James, J.: Real-time gesture recognition with minimal training requirements and on-line learning. In: CVPR 2007: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2007)
5. Muller, R.: Human Motion Following using Hidden Markov Models. Master thesis, INSA Lyon, Laboratoire CREATIS (2004)
6. Bevilacqua, F., Guédy, F., Schnell, N., Fléty, E., Leroy, N.: Wireless sensor interface and gesture-follower for music pedagogy. In: NIME 2007: Proceedings of the 7th international conference on New interfaces for musical expression, pp. 124–129 (2007)
7. Bevilacqua, F.: Momentary notes on capturing gestures. In: (capturing intentions). Emio Greco/PC and the Amsterdam School for the Arts (2007)
8. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE 77(2), 257–286 (1989)
9. Fine, S., Singer, Y.: The hierarchical hidden markov model: Analysis and applications. In: Machine Learning, pp. 41–62 (1998)
10. Bobick, A.F., Wilson, A.D.: A state-based approach to the representation and recognition of gesture. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(12), 1325–1337 (1997)
11. Wilson, A.D., Bobick, A.F.: Realtime online adaptive gesture recognition. In: Proceedings of the International Conference on Pattern Recognition (1999)
12. Rajko, S., Qian, G.: A hybrid hmm/dpa adaptive gesture recognition method. In: International Symposium on Visual Computing (ISVC), pp. 227–234 (2005)

13. Rajko, S., Qian, G.: Hmm parameter reduction for practical gesture recognition. In: 8th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2008), pp. 1–6 (2008)
14. Artieres, T., Marukatat, S., Gallinari, P.: Online handwritten shape recognition using segmental hidden markov models. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(2), 205–217 (2007)
15. Bloit, J., Rodet, X.: Short-time viterbi for online HMM decoding: evaluation on a real-time phone recognition task. In: International Conference on Acoustics, Speech, and Signal Processing, ICASSP (2008)
16. Mori, A., Uchida, S., Kurazume, R., Ichiro Taniguchi, R., Hasegawa, T., Sakoe, H.: Early recognition and prediction of gestures. In: Proceedings of the International Conference on Pattern Recognition, vol. 3, pp. 560–563 (2006)
17. Schwarz, D., Orio, N., Schnell, N.: Robust polyphonic midi score following with hidden markov models. In: Proceedings of the International Computer Music Conference, ICMC (2004)
18. Cont, A.: Antescofo: Anticipatory synchronization and control of interactive parameters in computer music. In: Proceedings of the International Computer Music Conference, ICMC (2008)
19. Schnell, N., Borghesi, R., Schwarz, D., Bevilacqua, F., Müller, R.: Ftm - complex data structures for max. In: International Computer Music Conference, ICMC (2005)
20. Bevilacqua, F., Muller, R., Schnell, N.: Mnm: a max/msp mapping toolbox. In: NIME 2005: Proceedings of the 5th international conference on New interfaces for musical expression, pp. 85–88 (2005)
21. Viaud-Delmon, I., Bresson, J., Pachet, F., Bevilacqua, F., Roy, P., Warusfel, O.: Eartoy: interactions ludiques par l'audition. In: Journées d'Informatique Musicale - JIM 2007, Lyon, France (2007)
22. Rasamimanana, N., Guedy, F., Schnell, N., Lambert, J.P., Bevilacqua, F.: Three pedagogical scenarios using the sound and gesture lab. In: Proceedings of the 4th i-Maestro Workshop on Technology Enhanced Music Education (2008)