# Probabilistic Models for Designing Motion and Sound Relationships

Jules Françoise, Norbert Schnell, Riccardo Borghesi, Frédéric Bevilacqua
STMS Lab
IRCAM—CNRS—UPMC
1, Place Igor Stravinsky
75004 Paris, FRANCE
{jules.francoise, norbert.schnell, riccardo.borghesi, frederic.bevilacqua}@ircam.fr

## ABSTRACT

We present a set of probabilistic models that support the design of movement and sound relationships in interactive sonic systems. We focus on a mapping–by–demonstration approach in which the relationships between motion and sound are defined by a machine learning model that learns from a set of user examples. We describe four probabilistic models with complementary characteristics in terms of multimodality and temporality. We illustrate the practical use of each of the four models with a prototype application for sound control built using our Max implementation.

## Keywords

Movement, Sound, Motion, Machine Learning, Max, Mapping, GMM, HMM, Mapping-by-Demonstration.

## 1. INTRODUCTION

Designing the relationships between movement and sound is one of the crucial issues in interactive systems, especially to develop musical expressivity. As reviewed by Caramiaux et al. the methods for creating movement–sound mappings have evolved with the advent of practical and interactive tools based on machine learning techniques [5]. Indeed, design practices are shifting from an analytical view of the links between parameters to interaction–driven approaches.

In this paper we detail a set of methods that follow the general approach called *Mapping–by–Demonstration* [9], that aims to learn the relationships between movement and sound from examples created by the user. In these methods, a probabilistic model is trained with examples of movements and sounds during a phase called *training*. In *performance* phase, the trained model generates a mapping between movement and sound processing parameters in real-time.

The aim of this paper is to present four probabilistic models that represent complementary aspects of the motion-sounds relationships. Each model is supported by a specific Max external – available to the community though the MuBu library, – that enables the creation of applications in motion-based sonic interactions for music and sound design. We also present four different use cases that further illustrate possible implementations of these probabilistic models.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
*NIME'14,* June 30 – July 03, 2014, Goldsmiths, University of London, UK.
Copyright remains with the author(s).

The paper is structured as follows. First, we review the similar works in term of methodology or model implementation (Section 2). Second, we describe the general workflow of our approach (Section 3) and the four different models (Section 4). Third, we describe the implementation of these models (Section 5) and prototype applications (Section 6).

## 2. RELATED WORK

Many approaches take advantage of machine learning as a design support tool in music interaction. In this section we review this body of work, with a distinction between computational models that focus on movement only and methods that directly learn the mapping by jointly representing movement and sound.[1]

Several methods have implemented movement models with a focus on gesture recognition, using Dynamic Time Warping [1], Hidden Markov Models (HMMs) [14] or Support Vector machines [13, 8]. Often, These methods target discrete interaction paradigms where recognition is used for triggering or selecting sounds. Recently, interests in computational models of movement shifted towards a better integration of expressivity. Bevilacqua *et al.* proposed a particular implementation of HMMs for continuous gesture recognition able to 'follow' the performance of a gesture in real-time [3]. Two extensions of this method were respectively proposed by Caramiaux *et al.* [4] who integrated the estimation of other attributes of movement (scaling, rotation, etc.), and by Françoise *et al.* whose hierarchical model integrates a representation of higher level temporal sequences [10].

Multimodal approaches propose a joint representation of movement and sound. Fels [6] use Neural Networks to learn a regression between movement and sound parameters. This approach was recently extended using deep learning models that integrate multilevel representations and intrinsic feature extraction [12]. Françoise proposed a multimodal temporal model able to learn dynamic movement–sound mappings using a multimodal extension of Hidden Markov Models [11].

Several works formalize the process of designing interactions with the support of machine learning. Drawing from knowledge in Interactive Machine Learning, Fiebrink emphasizes the role of the user in machine learning-based systems [8]: for example the 'Wekinator' allows a *playalong* definition of the training examples, in which movements parameters are recorded while listening to a predefined sequence of sound parameters [7]. In this paper we adopt the broader framework of *mapping–by–demonstration* proposed by Françoise [9], that does not necessarily assume the

[1] A broad overview of machine learning goes beyond the scope of this paper, and we refer the reader to [5] for a tutorial on machine learning for musical gestures.

synchronous performance. We extend this framework by formalizing different strategies based on either movement models or multimodal mapping models.

## 3. WORKFLOW AND DEFINITIONS

This paper focuses on supervised machine learning techniques implemented in an interactive workflow: the user defines movement–sound mappings through examples. In this section, we present the general workflow from the user's point of view. This workflow is supported by models and tools we describe in Sections 4 and 5, respectively.

As depicted in Figure 1, the *mapping–by–demonstration* approach involves an interaction loop that consists of two phases: a *training* phase that allows the user to define mappings and a *performance* phase in which the user controls sound synthesis through the previously defined mappings [9]. These two phases can be repeated and seamlessly integrated. Our library is indeed designed to make this interaction loop transparent to the user without requiring expert knowledge of machine learning algorithms and methods.

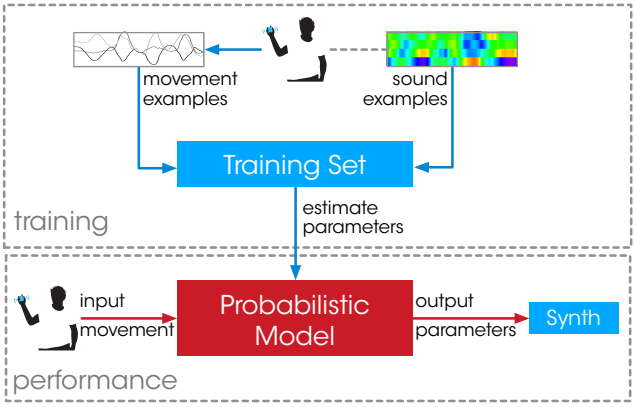



**Figure 1: Workflow of a Mapping–by–Demonstration System**

During *training*, the user demonstrates examples of movements. The sound may be taken from pre-recorded material, or captured synchronously to the movement (e.g. vocalizations). Each recorded *phrase* is represented by features describing the movement and – optionally – the associated sound.[2] A *training set* is constituted by a set of phrases labeled by the user to define *classes*.[3] A training algorithm is used to estimate the optimal parameters of the model for each class. This way, the user can integrate multiple movement–sound relationships within a model.

In the *performance* phase, the trained system is used to control sound processing in real-time driven by movement features.[4] In our implementation, each model outputs a set of parameters for every frame of the incoming stream of motion features. This guarantees continuous sound control with a fine time grain. Depending on the used model, the system simultaneously reports recognition scores, i.e. *likelihoods*, and estimates *control* parameters, which can be instantaneous sound synthesis parameters or temporal profiles.

## 4. MODELS

We present four supervised[5] probabilistic models, corresponding to four movement–sound relationship paradigms: Gaussian Mixture Models (GMM), Gaussian Mixture Regression (GMR), Hierarchical Hidden Markov Models (HHMM), and Multimodal Hierarchical Hidden Markov Models (MHMM).

In this section, we briefly introduce these models with a practical perspective, aiming at giving insights into for their specific advantages for interaction design.[6] The models can be organized regarding two criteria, *multimodality* and *temporality*. The former criterion is concerned with whether the model is limited to movement features, or whether it provides a joint (i.e. multimodal) representation of movement and sound. The latter considers whether and how the models take into account the temporal evolution of movement and sound.

### 4.1 Two Criteria

**Multimodality**

We make a distinction between movement models and multimodal models. Many approaches to sound control involving gesture recognition are based on movement models that are not intrinsically related to sound modeling (Figure 2**(a)**). In this case, the user defines the mapping between the recognized gesture classes and the sound classes after the learning the phase. Such mappings could consist, for example, in triggering a particular sound segment each time a particular gesture is recognized. More advanced mappings may allow for aligning the playback of a sound segment to the performance of a gesture [2].

Alternatively, *multimodal* models are trained with sequences of joint movement–sound representations and therefore enables to learn movement–sound relationships (Figure 2**(b)**). Consequently, these probabilistic models allow for generating sound features – or synthesis parameters – from motion features input into a trained system.

**Temporality**

We differentiate *instantaneous* models from *temporal* models. Instantaneous models learn and perform static instantaneous movement–sound relationships without taking into account any temporal modeling. Practically, this means that the recognition or generation performed by the model at any given instant is independent of previous input. On the contrary, *temporal* models take into account time series. In this case, the recognition or generation performed by the model depends on the history of the input.

### 4.2 Four Models

The implemented models are summarized in Table 1. Each of the four model addresses a different combination of the multimodal and temporal aspects. We implemented two instantaneous models based on Gaussian Mixture Models and two temporal models with a hierarchical structure, based

[2]While movement are usually represented by a time series of movement features extracted from motion capture data, sound sequences can be either represented by a stream of audio features or sound synthesis parameters.

[3]In the case that each class is defined by a single example, the labeling might be implicit.

[4]The same movement features are used in training and performance phase.

[5]We argue that supervised learning strongly support a controlled design of the mapping in most cases, by letting the user associate movements and sounds. However, some of the implemented models can be used in an unsupervised manner (for example: GMMs).

[6]The technical description of the models goes beyond the scope of this paper. We refer the reader to specialized publications for more detailed descriptions of the implemented machine learning techniques.

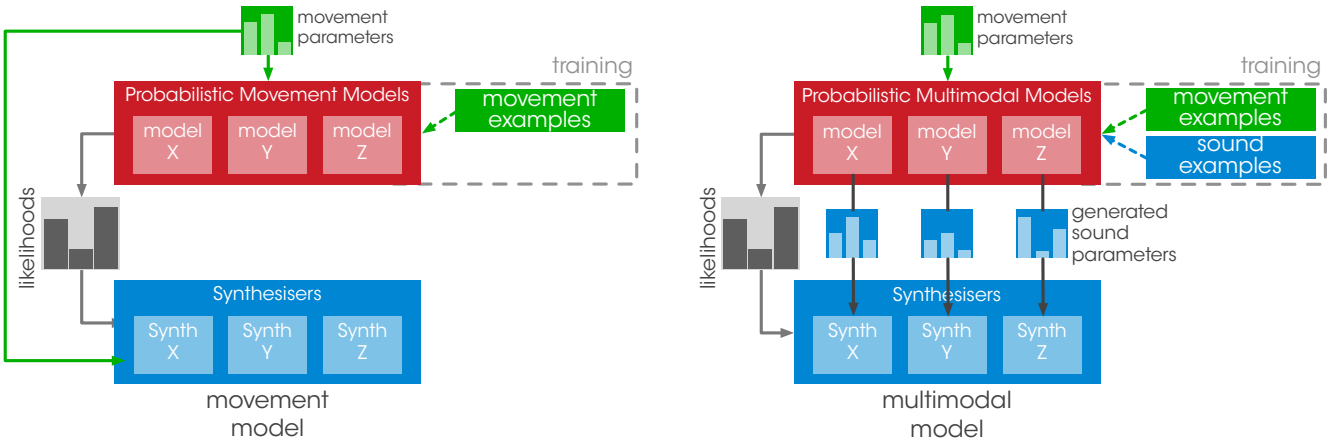



**Figure 2: Probabilistic sound control strategies based on movement models and multimodal models.**

on an extension of the basic Hidden Markov Model (HMM) formalism.

| | Movement | Multimodal |
|---|---|---|
| Instantaneous | Gaussian Mixture Model (GMM) | Gaussian Mixture Regression (GMR) |
| Temporal | Hierarchical Hidden Markov Model (HHMM) | Multimodal Hierarchical Hidden Markov Model (MHMM) |

**Table 1: Summary of the probabilistic models.**

**Gaussian Mixture Models (GMMs)** are instantaneous movement models. The input data associated to a class defined by the training sets is abstracted by a mixture (i.e. a weighted sum) of Gaussian distributions. This representation allows recognition in the *performance* phase: for each input frame the model calculates the likelihood of each class (Figure 3**(a)**).

**Gaussian Mixture Regression (GMR)** [18] are a straightforward extension of Gaussian Mixture Models used for regression. Trained with multimodal data, GMR allows to predict the features of one modality (e.g. sound) from the features of another (e.g. movement) through non-linear regression between both feature sets (Figure 3**(b)**).

**Hierarchical HMM (HHMM)** [10] integrates a high-level structure that governs the transitions between classical HMM structures representing the temporal evolution of – low-level – movement segments. In the *performance* phase of the system, the hierarchical model estimates the likeliest gesture according to the transitions defined by the user. The system continuously estimates the likelihood for each model, as well as the time progression within the original training phrases (Figure 3**(c)**).

**Multimodal Hierarchical HMM (MHMM)** [11] allows for predicting a stream of sound parameters from a stream of movement features. It simultaneously takes into account the temporal evolution of movement and sound as well as their dynamic relationship according to the given example phrases. In this way, it guarantees the temporal consistency of the generated sound, while realizing the trained temporal movement-sound mappings (Figure 3**(d)**).

## 5. IMPLEMENTATIONS

We developed a software library implementing the general framework presented in Section 3 as well as of the four models introduced in Section 4. The library is built upon a set of C++ classes representing phrases, training sets, and models.

### 5.1 Architecture

Our implementation follows the workflow presented in Section 3 with a particular attention to the interactive training procedure, and to the respect of the real-time constraints of the *performance* mode. The library is built upon four components representing phrases, training sets, models and model groups, as represented on Figure 4. A phrase is a multimodal data container used to store training examples. A training set is used to aggregate phrases associated with labels. It provides a set of function for interactive recording, editing and annotation of the phrases. Each instance of a model is connected to a training set that provides access to the training phrases. Performance functions are designed for real-time usage, updating the internal state of the model and the results for each new observation of a new movement. The library is portable and cross-platform. It defines a specific format for exchanging trained models, and provides Python bindings for scripting purpose or offline processing.

### 5.2 Max Implementation

Max is a visual programming environment dedicated to music and interactive media. We provide an implementation of our library as a set of Max externals and abstractions articulated around the *MuBu* collection of objects developed at Ircam [16][7].

Training sets are built using *MuBu*, a generic container designed to store and process multimodal data such as audio, motion tracking data, sound descriptors, markers, etc. Each training phrase is stored in a buffer of the container, and movement and sound parameters are recorded into separate tracks of each buffer. Markers can be used to specify regions of interest within the recorded examples. Phrases are labeled using the markers or as an attribute of the buffer. This structures allows to quickly record, modify, and an-

[7] `http://forumnet.ircam.fr/product/mubu/`

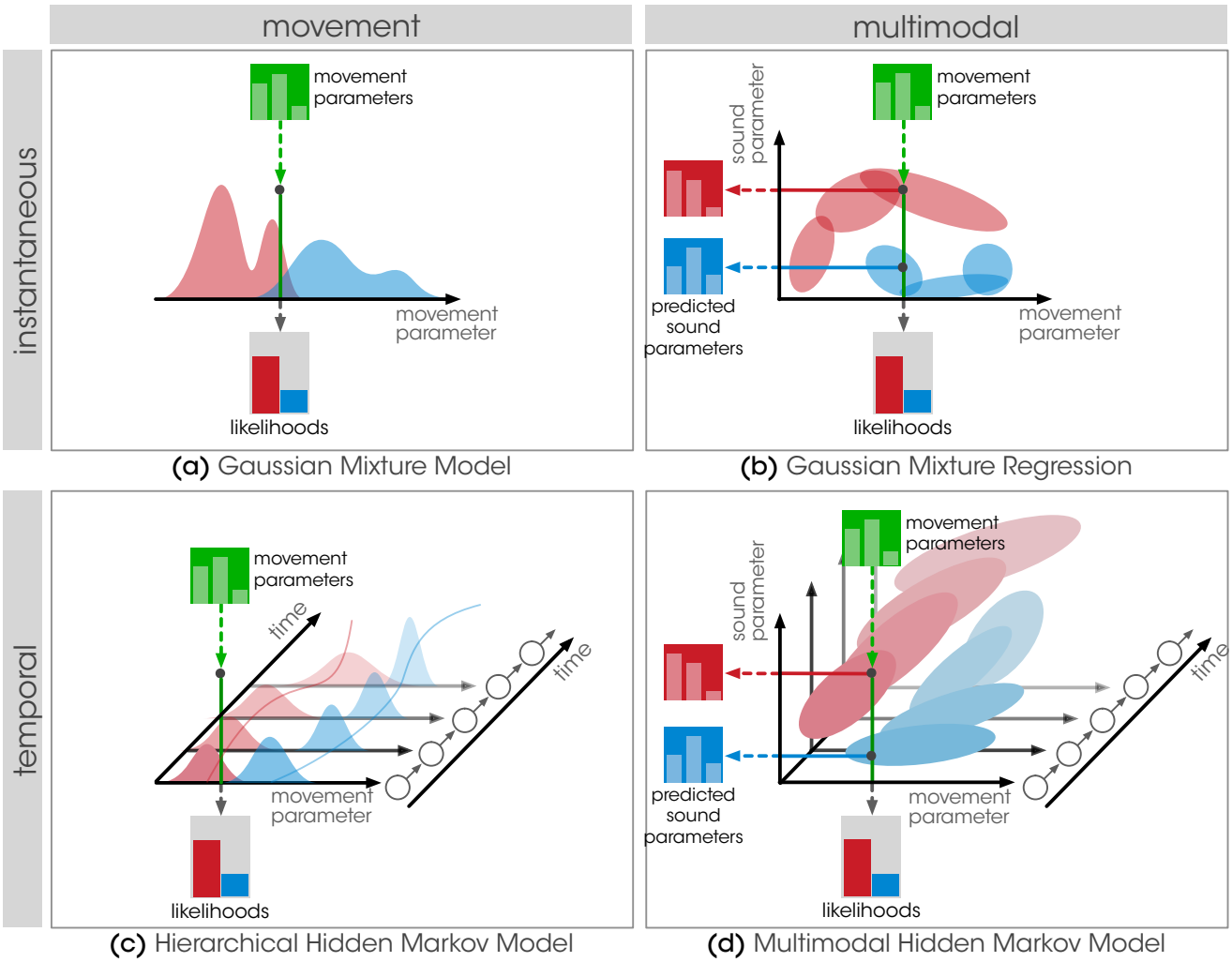



**Figure 3: Schematic representation of the characteristics of the 4 models.**

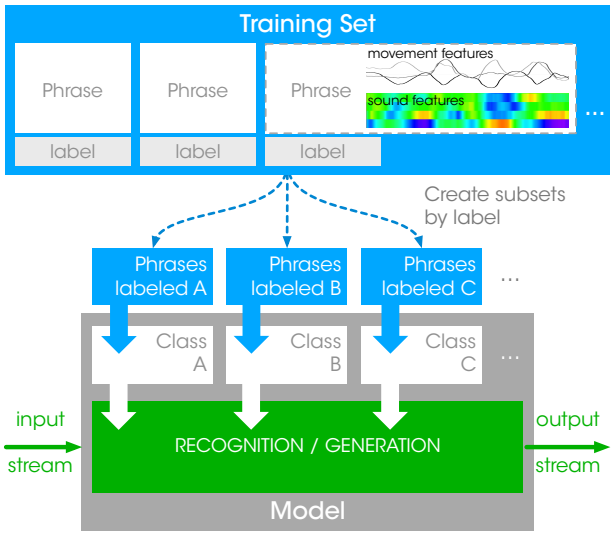



**Figure 4: Architecture of the implementation.**

notate the training examples. Training sets are thus autonomous and can be used to train several models.

Each model can be instantiated as a max object referring to a MuBu container that defines its training set. For training, the model connects to the container and transfers the training examples to its internal representation of phrases. The parameters of the model can be set manually as attributes of the object, such as the number of Gaussian components in the case of a GMM, or the number of states in the case of a HMM. The training is performed in background.

For performance, each object processes an input stream of movement features and updates the results with the same rate. For movement models, the object output the list of likelihoods, complemented with the parameters estimated for each class, such as the time progression in the case of a temporal model, or the weight of each Gaussian component in the case of a GMM. For multimodal models, the object also outputs the most probable sound parameters estimated by the model, that can be directly used to drive the sound synthesis.

## 6. PROTOTYPE APPLICATIONS

In this section, we illustrate the usage of our library through a set of prototype applications that emphasize the different characteristics of the four models. We illustrate the generality of our implementation with different types of sound synthesis and movement tracking systems. The four applications are illustrated in Figure 5, and a demonstration

video as well as further examples can be found on the webpage: `http://ismm.ircam.fr/nime2014_mbd/`.

## 6.1 Resonant Scratching

This application aims at sonifying touching movements using a set of resonant models.[8] The application is depicted in Figure 5**(a)**, and a screenshot of the Max patch is reported in Figure 6.

Motion capture is performed using a contact microphone placed on the control surface. Our goal is to classify different touching modes from the audio signal in order to select the separate resonant model. This classification only requires the instantaneous description of the timbre of the scratching sound. Therefore, we do not consider the temporal dynamics in this case, which justifies the use of an instantaneous movement model. We use a GMM to classify touch using Mel-Frequency Cepstral Coefficients (MFCCs), that we consider here as movement features since they directly relate to touch qualities.

During *Training*, we demonstrate several examples of 3 classes of touch: for instance *rub*, *scratch* and *tap*, by recording and analyzing the sound of each touching mode. Each class is represented by a GMM with 3 Gaussian components, and is associated with a resonant model. During *Performance*, the sound from the contact microphone is then directly filtered using the resonant model. The amount of each filter is determined by the likelihood of each class.

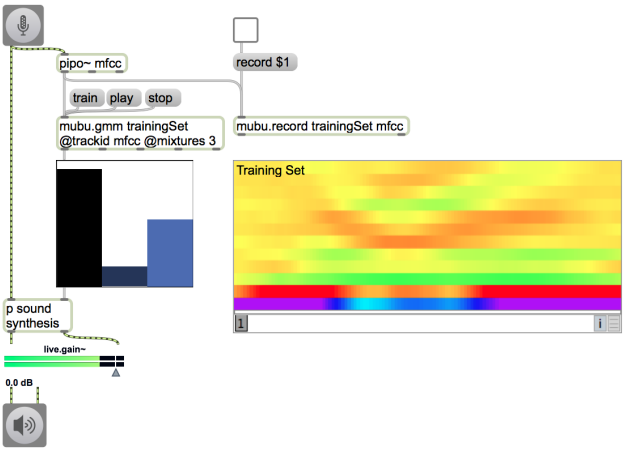



**Figure 6: Screenshot of the Max patch of the *Scratching* application.**

## 6.2 Physical Sound Design

In this application, we map in-air movement to physical modeling sound synthesis, as shown in Figure 5**(b)**). Using a Leapmotion™ hand tracking system, hand speed and orientation are directly available as movement features. The goal here is to learn the mapping between these movement features and the control parameters of physical models. Therefore, this application requires an instantaneous multimodal model, namely GMR.

For *Training*, we start by designing sounds using a graphical editor that allows to draw time profiles of the physical models' input parameters. After recording several examples of movements with each preset, one model is trained for each physical model using movement and sound parameters sequences. During Performance, the GMR generates the control parameters of each physical models, and estimates the likelihoods, that are used to mix the sound output of each synthesizer.

[8]This application draws from previous research from the Interlude project (see: `http://interlude.ircam.fr/`) [15]

## 6.3 Gesture-based Sound Mixing

This use case illustrates the use of the continuous estimation of the likelihoods in gesture recognition (Figure 5**(c)**). The goal is to continuously control the mixing of a set of recorded sounds, from a set of dynamic gestures captured using the Leapmotion. As dynamic gesture recognition is required here, we use a temporal movement model, namely a HHMM.

After defining the gesture vocabulary, we record several examples of each gesture to recognize, taking care of varying particular aspects such as the speed and breadth of each movement to ensure generalization and robustness of the recognition method. The movement models are learned using a HHMM in which each sub-model represents a particular class of gesture. As shown in Figure 5**(c)**, during performance the HHMM is used to evaluate the likelihood of each gesture, that is used to drive the playback level of the associated recorded sound.

## 6.4 Interactive Vocalization

This prototype focuses on sonic interaction design based on movements and non-verbal vocal sketches (Figure 5**(d)**). The application allows for performing interactive vocalizations where the relationships between motion and sounds are learned from direct demonstration of movements and vocalizations performed synchronously during the *training* phase. Movements are captured using MO interfaces [15], that integrate 3D accelerometers and gyroscopes. In order to guarantee a consistent reconstruction of the vocal sketches, this application requires the use of a temporal model. Therefore, we use the MHMM model to learn this multimodal and temporal mapping.

Each training phrase associates a sequence of motion features with a sequence of MFCCs computed from the audio. From this multimodal data, a hierarchical model (MHMM) is learned, in which each sub-model represents a multimodal primitive linking movement and voice. During *performance*, the model recognizes the movement and estimates the MFCCs accordingly. We use a corpus-based granular synthesis engine. The estimated stream of MFCCs is used to re-synthesize the vocalizations by concatenating the grains that match the sound description using a KNN search [17]. As before, the likelihoods are used to control the level of each class of vocalization.

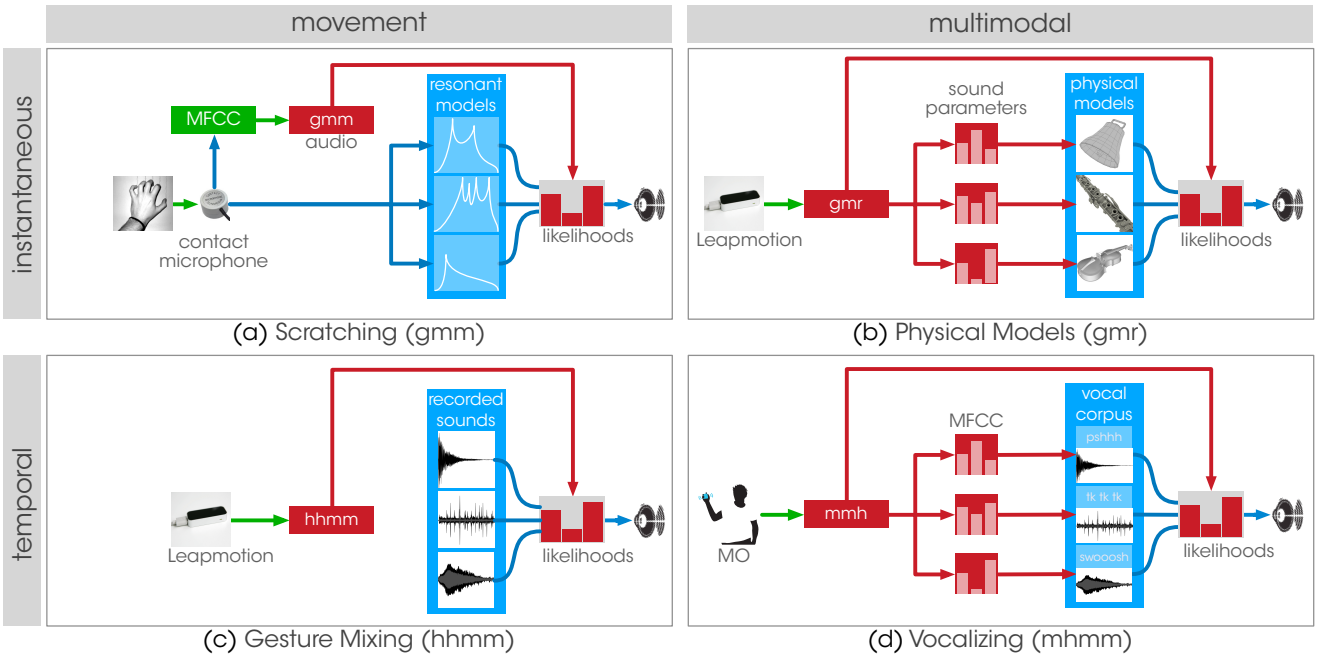
# 7. CONCLUSION AND FUTURE DEVELOPMENTS

We presented four different probabilistic models and their implementation for the design of movement-sound interaction by demonstration. These four models offers a consistent approach to use interactive machine learning techniques, considering different cases: movement models and multimodal models, instantaneous and temporal models.

These models have been implemented in the Max environment and are distributed in the MuBu library. We believe that the NIME community will benefit of these tools. Further examples will also be available that should foster discussion in the emerging use of machine learning in interactive systems.

# 8. ACKNOWLEDGMENTS

We acknowledge support from the ANR project Legos 11 BS02 012.




(a) Scratching (gmm) (b) Physical Models (gmr)

(c) Gesture Mixing (hhmm) (d) Vocalizing (mhmm)

**Figure 5: Workflow of the *performance* phase of each application.**

## 9. REFERENCES


[1] F. Bettens and T. Todoroff. Real-time dtw-based gesture recognition external object for max/msp and puredata. *Proceedings of the SMC 2009 Conference*, 9(July):30–35, 2009.

[2] F. Bevilacqua, N. Schnell, N. Rasamimanana, B. Zamborlin, and F. Guédy. Online Gesture Analysis and Control of Audio Processing. In *Musical Robots and Interactive Multimodal Systems*, pages 127–142. Springer, 2011.

[3] F. Bevilacqua, B. Zamborlin, A. Sypniewski, N. Schnell, F. Guédy, and N. Rasamimanana. Continuous realtime gesture following and recognition. *Gesture in Embodied Communication and Human-Computer Interaction*, pages 73–84, 2010.

[4] B. Caramiaux, N. Montecchio, and F. Bevilacqua. Adaptive Gesture Recognition with Variations Estimation for Interactive Systems. *(Under review)*.

[5] B. Caramiaux and A. Tanaka. Machine Learning of Musical Gestures. In *proceedings of the International Conference on New Interfaces for Musical Expression (NIME 2013)*, Seoul, South Korea, 2013.

[6] S. Fels and G. Hinton. Glove-talkII: A neural network interface between a data-glove and a speech synthesizer. *Neural Networks, IEEE Transactions on*, 4(1):2–8, 1993.

[7] R. Fiebrink, P. R. Cook, and D. Trueman. Play-along mapping of musical controllers. In *In Proceedings of the International Computer Music Conference*, 2009.

[8] R. Fiebrink, P. R. Cook, and D. Trueman. Human model evaluation in interactive supervised learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'11)*, page 147, Vancouver, BC, Canada, 2011. ACM.

[9] J. Françoise. Gesture–Sound Mapping by Demonstration in Interactive Music Systems. In *Proceedings of ACM Multimedia (MM'13)*, pages 1051—-1054, Barcelona, Spain, 2013.

[10] J. Françoise, B. Caramiaux, and F. Bevilacqua. A Hierarchical Approach for the Design of Gesture–to–Sound Mappings. In *Proceedings of the 9th Sound and Music Computing Conference*, pages 233–240, Copenhagen, Denmark, 2012.

[11] J. Françoise, N. Schnell, and F. Bevilacqua. A Multimodal Probabilistic Model for Gesture–based Control of Sound Synthesis. In *Proceedings of ACM Multimedia (MM'13)*, pages 705–708, Barcelona, Spain, 2013.

[12] O. Fried and R. Fiebrink. Cross-modal Sound Mapping Using Deep Learning. In *New Interfaces for Musical Expression (NIME 2013)*, Seoul, Corea, 2013.

[13] N. Gillian and R. Knapp. A Machine Learning Toolbox For Musician Computer Interaction. In *proceedings of the 2011 International Conference on New Interfaces for Musical Expression (NIME 2011)*, Oslo, Norway, 2011.

[14] P. Kolesnik and M. M. Wanderley. Implementation of the Discrete Hidden Markov Model in Max/MSP Environment. In *FLAIRS Conference*, pages 68–73, 2005.

[15] N. Rasamimanana, F. Bevilacqua, N. Schnell, E. Fléty, and B. Zamborlin. Modular Musical Objects Towards Embodied Control Of Digital Music Real Time Musical Interactions. *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*, pages 9–12, 2011.

[16] N. Schnell, A. Röbel, D. Schwarz, G. Peeters, and R. Borghesi. Mubu & friends - assembling tools for content based real-time interactive audio processing in max/msp. In *Proceedings of International Computer Music Conference*, Montreal, 2009.

[17] D. Schwarz. Corpus-based concatenative synthesis. *Signal Processing Magazine, IEEE*, 24(2):92–104, 2007.

[18] H. G. Sung. *Gaussian Mixture Regression and Classification*. Phd dissertation, Rice University, Houston, TX, 2004.