Adaptive Gesture Recognition with Variation Estimation for Interactive Systems

BAPTISTE CARAMIAUX, Goldsmiths, University of London, UK / IRCAM, Paris, France
NICOLA MONTECCHIO, University of Padova, Italy
ATAU TANAKA, Goldsmiths, University of London
FREDERIC BEVILACQUA, STMS Lab IRCAM-CNRS-UPMC, Paris, France

This paper presents a gesture recognition/adaptation system for Human Computer Interaction applications that goes beyond activity classification and that, complementary to gesture labeling, characterizes the movement execution. We describe a template-based recognition method that simultaneously aligns the input gesture to the templates using a Sequential Montecarlo inference technique. Contrary to standard template-based methods based on dynamic programming, such as Dynamic Time Warping, the algorithm has an adaptation process that tracks gesture variation in real-time. The method continuously updates, during execution of the gesture, the estimated parameters and recognition results which offers key advantages for continuous human-machine interaction. The technique is evaluated in several different ways: recognition and early recognition are evaluated on a 2D onscreen pen gestures; adaptation is assessed on synthetic data; and both early recognition and adaptation is evaluation in a user study involving 3D free space gestures. The method is not only robust to noise and successfully adapts to parameter variation but also performs recognition as well or better than non-adapting offline template-based methods.

Categories and Subject Descriptors: H.5.2 [**User Interfaces**]: Interaction styles; I.5.5 [**Pattern Recognition**]: Implementation

General Terms: Design, Algorithms, Performance, Deployment of Gesture Interaction Systems

Additional Key Words and Phrases: Gesture Recognition, Particle Filtering, Continuous Gesture Modeling, Adaptive Decoding, Gesture Analysis, Realtime

ACM Reference Format:
Baptiste Caramiaux, Nicola Montecchio, Atau Tanaka, and Frédéric Bevilacqua. Adaptive Gesture Recognition with Variations Estimation for Interactive Systems. *ACM Trans. Interact. Intell. Syst.* V, N, Article (YYYY), 34 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Gesture is increasingly used in Human Computer Interaction (HCI) involving several forms of activity recognition. This is a need to elaborate interaction paradigms based on body movements (e.g. hand, whole body) or tangible interfaces [Dourish 2004; Jordà 2008] that could enable natural and fluid interaction. Methods for gesture recognition [Mitra and Acharya 2007] and continuous gesture recognition [Weinland et al. 2011] have been proposed and successfully implemented. These methods have been developed for the most part to label gesture. We propose a technique that goes beyond clas-

---

The first author was affiliated to IRCAM Centre Pompidou when this work started. He is now at Goldsmiths College, University of London. The second author was a visiting researcher at IRCAM Centre Pompidou, affiliated to the University of Padova, Italy. He is now at The Echo Nest, Boston.

DOI: http://dx.doi.org/10.1145/0000000.0000000

sification by, complementary to gesture labeling, characterizes movement execution, providing the possibility of innovative interaction scenarios.

Movement-based interactive systems generally assume a closed action-perception loop, especially in cases of continuous control. This means that the users continuously *adapt* their movements, relying for example on visual or sound feedback. In other words, the gesture being recognised might appear as "distorted" compared to gesture references. A robust recognition system should be able to adapt to such changes. In addition, it would be useful to estimate parameters of the gesture execution incrementally *during* the performance, for two reasons. First, it allows the system to take into account variations occurring during the motion and update the motion model accordingly. Second, such parametrization could be directly used in the design of interaction: for example gesture size variation might allow continuous control over an *expressive* component of the interaction.

We propose a gesture recognition system that is designed to take into account movement variation incrementally during the performance and to provide users real-time parameter feedback. It can accommodate a broad set of gesture variations within each class, for example in the speed, the amplitude or the orientation. Importantly, these variations can be estimated continuously during gesture execution.

This system is suited for trajectory-based gestures such as finger trajectories on a tablet, hand motions manipulating an interface (e.g. game interface or mobile phones) or free space body movements. It can be seen as an extension of a previous system we developed called *Gesture Follower* [Bevilacqua et al. 2010; Bevilacqua et al. 2011b], that was found to be effective in recognizing and synchronizing continuous execution of gesture to media such as sounds and visuals [Bevilacqua et al. 2012; Caramiaux et al. 2010]. It was shown to be successful for musical control using tangible interfaces [Rasamimanana et al. 2011; Zamborlin et al. 2014], music and dance pedagogy [Bevilacqua et al. 2011a; Bevilacqua et al. 2007] and gesture-based gaming systems [Rasamimanana and Bevilacqua 2012]. This body of work allowed us to establish a series of requirements that guided the development of the method we report in this paper:

(1) The training procedure must be based on a single template, to allow users to define their gesture vocabulary with simple and direct procedures.
(2) Results should be updated continuously during the gesture. In order to allow them to be used in continuous interaction paradigms or for anticipation (as proposed for example by Bau et al. [Bau and Mackay 2008]). This generally requires taking into account the gesture's fine temporal structure.
(3) Gesture variations that occur during execution should be taken into account and estimated as a way to encode expressive aspects of the performance.

The *Gesture Follower* (GF) handles Points 1 and 2 above, but does not handle Point 3. The method presented here is a state-space model where states are variations to be estimated online. To do so, the method makes use of Particle Filtering that tackles the challenge of continuously adapting to the gesture variation. The goal of this paper is to formally introduce the method and to show its accuracy with both real world databases and interactive applications.

The paper is structured as follows. First, we review the state of the art on gesture recognition systems for interaction (Section 2). This provides the technical motivation for the design of our method. In addition, in Section 3 we present the interaction model that informs the design from an applicative point of view. The computational model is presented in Section 4. In Section 5 we present an evaluation on recognition with adaptation performed on real data in the case of 2D pen-gestures from the state of the art. Then, in Section 6, we evaluate the process of adaptation to movement vari-

ations on synthetic data. This is followed by a user study assessing the ability of the method to adapt in realtime in an interactive context (Section 7). The results obtained in the previous sections are discussed in Section 8 together with technical and human constraints of using gesture variations for HCI. Finally we conclude in Section 9.

2. RELATED WORK

In this section we review the methods most often used to recognize gestures represented as multidimensional times series for human-computer interaction. Note that the multidimensional time series represent the trajectory of one point on a surface or in the 3-dimensional space.

For 2D drawing gestures, several basic methods take advantage of simple distance functions between gestures. Rubine [Rubine 1991] proposes a geometric distance measure based on examples of single-stroke gestures. Wobbrock et al. propose a simple template-based method that makes use of Euclidean distance [Wobbrock et al. 2007], after a pre-processing stage in order to take into account geometric variations (such as scaling and rotation) and speed variations (by uniformly resampling the data).

Several methods are based on Dynamic Programming (DP) to handle local time variations. The most widely used technique is Dynamic Time Warping (DTW), that requires the storage of the whole gesture temporal structure [Gavrila and Davis 1995; Liu et al. 2009]. A similarity matrix is computed between the test gesture and a reference template and the optimal path is computed, representing the best alignment between the two time series. There are various applications such as gesture control [Merrill and Paradiso 2005], communicative gesture sequences [Heloir et al. 2006], querying based on human motion [Forbes and Fiume 2005]. An extension to DTW has been proposed by Bobick et al. [Bobick and Wilson 1997], to take into account several examples, using principal curve in DP computation. A similar approach by Yacoob et al. [Yacoob and Black 1998] considers an "EigenCurve" representation of several examples and carries out recognition based on this representation. One of the main drawbacks of methods based on DP is that they do not provide an explicit noise model, and do not prevent errors due to unexpected or lost observations in the incoming sequence.

Statistical methods, such as the widely used Hidden Markov Model (HMM) [Rabiner 1989] prevent such shortcomings. HMMs are based on a probabilistic interpretation of observations (gesture samples) and can model the gesture's temporal trajectory through a compact representation. HMMs have been successfully applied in human motion recognition from vision-based data [Mitra and Acharya 2007]. HMM-based methods are generally robust since they rely on learning procedures that use large databases, allowing a model of variations occurring within a gesture class to be created [Bilmes 2002].

Variation in gesture are thus handled for the most part by methods like HMM that use comprehensive databases taking into account all possible variations. They typically require cumbersome training procedures. Consequently, several authors propose so-called adaptive systems, where the system adapts to variability of input, user [Licsár and Szirányi 2005; Wilson 2000; Caridakis et al. 2009], or sensor location [Chavarriaga et al. 2013]. These systems are fundamentally not designed to take into account variations that occur *during* the movement performance.

In this paper, we refer to variations occurring *within* gesture classes. Wilson and Bobick propose a model that takes into account parametric changes in execution [Wilson and Bobick 1999]. They describe an application where bi-handed gesture semantics are related to global trajectories (for example actions on an object) while variations provide additional meaning (for example the size of the object). In this case, the amplitude is defined globally on the whole gesture (see also [Brand and Hertzmann 2000]).

In [Wilson and Bobick 2000], Wilson and Bobick describe an online learning method that can be applied to each different user. A case study is described, where simple gestures such as ”rest”, ”down”, and ”up” are recognized.

The Gesture Follower described above makes use of the HMM statistical framework, but with an approach that differs from standard implementations. Initially, the aim of the GF method was to estimate the time progression of a gesture in real-time, using a template reference [Bevilacqua et al. 2012]. The time progression information can then be used in the interaction. Similarly to DTW, this method uses the whole time series and assigns a state to each sample. This allows for the modeling of fine-temporal gestural structure (similarly to the approach of Bobick and Wilson in [Bobick and Wilson 1997]). The system makes use of a forward procedure simultaneously on several template gestures, which allows for the estimation, during the gesture performance, of its time progression and likelihood related to each template. However, GF cannot adapt to variations occurring during the gesture.

We will show in this paper that an adaptive approach using an extended state model and a different decoding scheme is possible. We do this by considering the recognition problem as a tracking problem, where Particle Filtering (PF) techniques have been widely used, and have been proved effective adapting continuously features of the tracked objects. An exhaustive review of particle filtering literature is beyond the scope of this paper, and we refer the reader to [Arulampalam et al. 2002] and [Doucet et al. 2001] for more specific theoretical works on PF. For example, methods based on PF for tracking were used on hand gestures and faces [Bretzner et al. 2002; Zhou et al. 2004; Mitra and Acharya 2007; Shan et al. 2007]. In these previous works, PF is used to estimate the position of the area of importance in image sequences. PF has also been proved efficient when the training and testing data may have significant differences (see for instance [Wei et al. 2013] for such application) which will be the case in our context of application.

The method we propose is inspired by the work of Black et al. [Black and Jepson 1998a], based on the *condensation* algorithm [Isard and Blake 1998], for the recognition of spatio-temporal gesture templates. The model was applied to data recorded using a 2-dimensional augmented whiteboard. The implementation allowed for the tracking of speed and scaling variation. It will be denoted PF–condensation. A similar inference model has been used by Visell et al. [Visell and Cooperstock 2007] to estimate parameters of a non-linear dynamical system for the analysis and rehabilitation of gait using non-visual feedback. However, no experimental results are reported.

In this paper, we generalize the approach by [Black and Jepson 1998a] by estimating not only scaling but also other parameters such as rotation, and propose a different observation function that facilitates parameter estimation. We also propose and evaluate the explicit use of the estimated variations in interaction.

Since the method we propose can be seen as an extension of our *Gesture Follower* system in order to allow for *following* variation in the gesture, we will refer it as *Gesture Variation Follower* (GVF).

3. INTERACTION PRINCIPLES

We present interaction principles that we consider important for target applications in sound manipulation or visual processing in contexts such as gaming, interactive art, or rehabilitation. These interaction principles are grounded in prior works in interaction design.

According to Verplank [Verplank et al. 2001], both discrete and continuous commands are critical to the design of interaction based on motion. Our interaction model incorporates these two types of control, to return not only *which* gesture is performed but also *how* it is performed. This paradigm can be explained with an example taken

from musical performance. A musician performs actions that articulate discrete notes but also modify continuous parameters such as amplitude and timbre. Continuous variation takes place *while* playing the note. Hence, in our interaction model, a combination of *which* and *how* is seen as critical for expressive interaction [Caramiaux 2012].

The schematic view of the interaction model is seen in Figure 1. User body movement (fingers, hands, whole body) are captured by a sensor system. The data can be represented as a multidimensional time series. From this representation, the proposed algorithm is able to recognize the gesture performed, adapt to its variation and return variation parameters based on a previously learned template. Consequently, the model output has two components: an index for the recognized gesture, and a vector of continuous values estimating the variation. Importantly, recognition and adaptation are performed in real-time which implies that the algorithm continuously updates and outputs the recognized gesture and variation values during gesture execution.

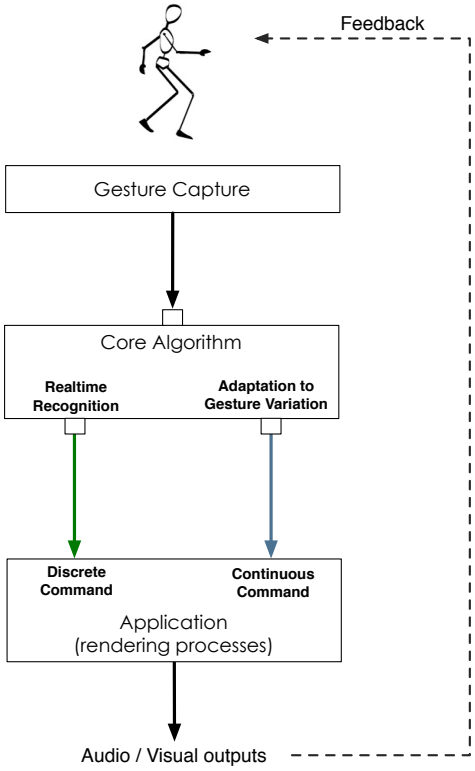

Fig. 1. Interaction Model. User body motion are sensed by a motion capture device. The proposed algorithm is then able to recognize the gesture performed, adapt to its variation and return variation parameters considering previously learned templates. Recognition and adaptation are performed in real-time which implies that the algorithm outputs continuously updated recognized gesture and variation values during the performance. An application is then plugged to the system and returned processed/synthesized audio/graphics.

It is also important that the user be able to easily establish the reference gestures/actions in order to facilitate fast testing sessions. The learning procedure should

therefore remain as simple as possible. Based on this, we explicitly designed our learning system to require only a single template for defining a given gesture class.

The interaction model can be applied in a range of different application scenarios. In the user study described in Section 7, we focus on realtime audio manipulation that, due to the temporal nature of sound, is suitable for continuous interaction. Motion-based sonic interaction has been used in various applicative contexts such as for interactive motion *sonification* to help people with visual impairments in various activities such as sport [Höner and Hermann 2005]; audio-motor loop for stroke rehabilitation [Boyer et al. 2013]; everyday activity sonification [Rocchesso et al. 2009]; or musical applications and gaming [Rasamimanana and Bevilacqua 2012]. In these examples, interaction depends on continuous mapping between movement and audio feedback. Sound brings an additional information channel to vision that has been shown relevant for temporal data and temporal data with recurrent patterns [Barrass and Kramer 1999].

To that extent, we designed a computational model motivated by the presented interaction principles and types of application. The next section presents the technical description of the method whose main features, gesture recognition and adaptation to movement variation, will be evaluated in the subsequent sections.

4. COMPUTATIONAL MODEL

Our working definition of gesture is body limb movement represented by a temporal series of a fixed number of parameters. For a given *input* gesture, the recognition task selects the best match among a set of pre-recorded *template* gestures. The input gesture is denoted $\mathbf{z} = \mathbf{z}_1 \dots \mathbf{z}_N$ (or $\mathbf{z}_{1:N}$) and the template gesture is denoted $\mathrm{g} = \mathrm{g}_1 \dots \mathrm{g}_T$ (or $\mathbf{g}_{1:T}$). $\mathbf{z}$ can be of different length than g. As described in the next section, we use a Bayesian approach with a continuous state representation. The model is inspired by the work of Black et al. [Black and Jepson 1998b]. In our system, we explicitly changed the latent state space in order to fit the constraints of the interaction scenarios described above. In addition, we propose the use of an alternative observation distribution allowing for less dependency on the model parameters, which has critical consequences for its practical use.

4.1. Continuous state model

The model can be formulated with the following dynamical system:

$$\begin{cases} \mathbf{x}_k &= f_{\mathrm{TR}}(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}) \\ \mathbf{z}_k &= f_{\mathrm{OB}}(\mathbf{x}_k, \mathbf{w}_k; \mathbf{g}) \end{cases} \tag{1}$$

where, at discrete time k,

— $\mathbf{x}_k$ is a vector representing the *system state*, state elements are the varying gesture characteristics;
— f_{TR} is a (possibly non linear) function that governs the evolution of the system state, depending on $\mathbf{x}_{k-1}$ and an independent and identically distributed (i.i.d.) process noise sequence $\mathbf{v}_k$;
— f_{OB} is a (possibly non-linear) function that generates the *observations* $\mathbf{z}_k$, depending on the system state $\mathbf{x}_k$, an i.i.d. measurement noise sequence $\mathbf{w}_k$ and a template gesture $\mathbf{g}$.

The problem is formulated as a tracking problem, i.e. tracking and adapting to the values of $\mathbf{x}_k$. Precisely, state variables $\mathbf{x}_k$ are the varying gesture characteristics (speed, size, etc.) that are chosen depending on the input signal and the context, as detailed in Section 4.2. Estimation of varying characteristics is governed by the par-

ticular form of transition between states, f_{TR} (described in Section 4.3) and an observation function f_{OB} (Section 4.4) used to compute the likelihood of the estimation according to the incoming gesture $\mathbf{z}$ and the template gesture $\mathbf{g}$. The extension of the tracking algorithm for the recognition task is detailed in Section 4.6. The inference is based on particle filtering with a resampling process. The inference used in the model is reported in Appendix A.1.

4.2. State space model

A critical aspect of the model design is defining the state space. The state of the system is composed of the varying gesture characteristics that have to be estimated over time. In other words, the state space comprises the features we are able to assess online and, consequently, used as continuous outputs during the interaction (see Figure 1).

The process is adaptive since the features are updated at each time step. Formally, the system state at instant k is denoted as:

$$\mathbf{x}_k = \begin{pmatrix} \mathbf{x}_k(1) \\ \vdots \\ \mathbf{x}_k(D) \end{pmatrix} \in \mathbb{R}^D$$

where D is the dimensionality of the state space.

In our model, the first dimension $\mathbf{x}_k(1)$ is set to be the *phase* p_k at discrete time k, which represents the alignment between the template gesture and the incoming gesture at time k, as illustrated in Figure 2 (or in other words, p_k can be seen as the time progression of the gesture). The phase is normalized in the [0,1] range (0 being the beginning and 1 the end of the gesture time).

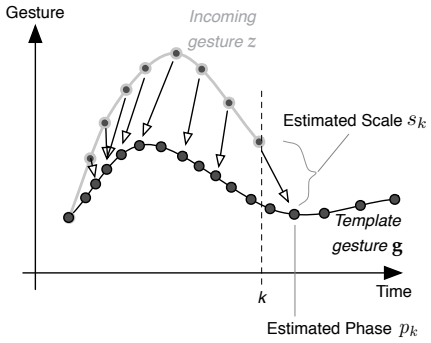

Fig. 2. Illustration of the alignment and adaptation. An incoming gesture $\mathbf{z}$ is aligned onto a template gesture $\mathbf{g}$ based on the continuous adaptation of gesture features $\mathbf{x}_k$ illustrated as p_k and s_k in the figure.

The second dimension $\mathbf{x}_k(2)$ is set to be the *speed* v_k at k. The speed v_k is actually a speed ratio between the speed of the incoming gesture (first derivative of the phase) to the speed the template gesture.

The state space can contain additional dimensions. In particular, we will extend it to include to other features such as the *scaling* (i.e. amplitude ratio, see Figure 2), and the *rotation* angles in 3-dimensions.

The configuration of the state space depends on two independent criteria. First the input data drives the type of feature we can track. As an example, the notion of rotation does not represent the same feature for 2-dimensional shapes performed on a tactile surface and for 3-dimensional accelerometer data. Second, the applicative context also

drives the state space in order to make the estimated features suitable and usable in a given scenario, something we will discuss below.

Note that defining the state space does not depend on the gesture template $\mathbf{g}$.

4.3. State transition

In the proposed model, the state transition function f_{TR} (see Equation (1)) is linear, given by the matrix A, and modeled probabilistically as a Gaussian distribution:

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{x}_{k-1}) &= \mathcal{N}(\mathbf{x}_k|A\mathbf{x}_{k-1}, \Sigma) \\ \Sigma &= \text{diag}\,(\sigma_1 \dots \sigma_D) \end{aligned} \tag{2}$$

We choose to add a constraint, by setting the relationship between the phase and the velocity, corresponding to a first-order motion equation (other choices can easily be made):

$$p_k = p_{k-1} + \frac{v_k}{T} + \mathcal{N}(0, \sigma_1) \tag{3}$$

where T is the template's length and σ_1 is the first element in the diagonal of Σ. This constraint can simply be taken into account by setting the first row of the matrix A to $(1\ \frac{1}{T}\ 0 \cdots 0)$. The other terms, set to zero in the first row of the matrix A, implies that the estimation of the phase is independent of the other features ($\mathbf{x}_k(j), j > 2$).

The transition parameters play an important role on adaptation as we will show in Sections 5 and 7. They govern the dynamic of the variation estimations, in other words: the speed of convergence to the accurate estimation and the precision of the estimation.

4.4. Observation function

The observation function evaluates the accuracy of the state estimation according to the input observation and the template. Parameters of the observation function govern how discriminant the method is.

In our model, the observation function f_{OB} (see Equation (1)) is chosen to be a Student's t-distribution that depends on three parameters: the mean $\boldsymbol{\mu}$, the covariance matrix Σ and the degree of freedom ν. For a K-dimensional input vector $\mathbf{z}_k$ at time k, the Student's t-distribution is as follows:

$$St(\mathbf{z}_k|f(\mathbf{x}_k, \mathbf{g}(p_k)), \Sigma, \nu) = C(\Sigma, \nu)\left(1 + \frac{d^2(\mathbf{z}_k, f(\mathbf{x}_k, \mathbf{g}))}{\nu}\right)^{-\frac{\nu+K}{2}} \tag{4}$$

where

$$C(\Sigma, \nu) = \frac{\Gamma(\nu/2 + K/2)}{\Gamma(\nu/2)} \frac{|\Sigma|^{-1/2}}{(\nu\pi)^{K/2}}$$

where $f(\mathbf{x}_k, \mathbf{g})$ is a function of the template $\mathbf{g}$ and the state value at k. Precisely, $f(\mathbf{x}_k, \mathbf{g})$ adapts the expected template sample $\mathbf{g}(p_k)$, given the phase p_k at k. The distance d between the adapted template sample and the incoming observation is given by:

$$d(\mathbf{z}_k, f(\mathbf{x}_k, \mathbf{g})) = \sqrt{[\mathbf{z}_k - f(\mathbf{x}_k, \mathbf{g})]^T \Sigma^{-1} [\mathbf{z}_k - f(\mathbf{x}_k, \mathbf{g})]} \tag{5}$$

The choice of Student's t-distribution is motivated by its heavier tails compared to Gaussian distribution (i.e. the distribution is wider around the mean). In the limit $\nu \to \infty$, the t-distribution reduces to a Gaussian with mean $\boldsymbol{\mu}$ and covariance Σ. We will see in Section 5 that the choice of the Student's t-distribution has interesting properties that can reduce the sensitivity of the system to the covariance matrix.

4.5. Inference and implementation

Realtime estimation of the state values (inference) is performed using Particle Filtering, a special case of sequential Montecarlo method. Sequential Montecarlo methods work by recursively approximating the current distribution of the system state using the technique of Sequential Importance Sampling: state samples are drawn from a simpler distribution and then weighted according to their importance in estimating the "true" distribution. Importance is driven by incoming samples. At each step k, a particle $\mathbf{x}_k^i$ represents a possible value of the state space which is weighted by its probability w_k^i. The expected value of the features, at time k, is:

$$\hat{\mathbf{x}}_k = \sum_{i=1}^{N_s} w_k^i \mathbf{x}_k^i$$

where N_s denotes the number of particles. Inferred feature values $\hat{\mathbf{x}}_k$ constitute the adaptation process since at each time k, we assess the variation values defined as state variables. The particle filtering algorithm is reported in Algorithm 2 in Appendix A.1 together with the GVF algorithm pseudo code (Algorithm 1)[1].

4.6. Handling recognition

Finally, we extend the model to handle recognition by taking into account several templates. To do so, we must change the state space in order to estimate the likeliest template in addition to the varying gesture characteristics.

Consider M templates of respective length $L_1 \dots L_M$ denoted $\mathbf{g}^1 \dots \mathbf{g}^M$. At initialization, we assign to each state particle $\mathbf{x}_k^i$ a *gesture index* between $1 \dots M$ (denoted m_k), based on a initial distribution. Generally, a uniform distribution is chosen, that is, by distributing particles evenly across the gesture templates. This extends the state configuration applied to each particle as follows:

$$\mathbf{x}_k^i = \begin{pmatrix} \mathbf{x}_k^i(1) \\ \vdots \\ \mathbf{x}_k^i(D) \\ m_k \end{pmatrix} \in \mathbb{R}^D \times \mathbb{N} \tag{6}$$

The transition probability is then adapted as follows:

$$\begin{aligned} p(\mathbf{x}_k^i | \mathbf{x}_{k-1}^i) &= \mathcal{N}(\mathbf{x}_k^i | A\mathbf{x}_{k-1}^i, \Sigma) \\ \Sigma &= \mathbf{diag}\,(\sigma_1 \dots \sigma_D\, 0) \end{aligned} \tag{7}$$

By summing the weights w_k^i corresponding to the particles' gesture indexes, it is straightforward to compute the probability of each gesture:

$$p(\mathbf{g}_k^l | \mathbf{g}_k^m) = \sum_{j \in \mathcal{J}} w_k^j \quad , \forall l \in [1, M], \forall m \in [1, M], m \neq l$$

$$\text{where} \quad \mathcal{J} = \left\{ j \in [1, N_s] / \mathbf{x}_k^j(D+1) = l \right\} \tag{8}$$

5. RECOGNITION TASKS ON REAL-WORLD 2-DIMENSIONAL GESTURE DATA

The goal of the experiment is to assess the recognition accuracy of the proposed method on a database from the state of the art and comparing it with established techniques.

[1]Note that the algorithm has been implemented in C++ and it is also available online and starts to be used in other projects: `http://www.github.com/bcaramiaux/gvf`.

We use Wobbrock's [Wobbrock et al. 2007] database of 2-dimensional pen gestures[2]. The database contains 16 gestures that are meant to be commands for selection, execution and entering symbols in HCI applications (Figure 3). It has been created as follows. Ten participants were recruited to perform 16 gestures. For each gesture in the vocabulary, "subjects entered one practice gesture before beginning three sets of 10 entries at slow, medium, and fast speeds" [Wobbrock et al. 2007]. Hence, the whole database contains: 10 participants $\times$ 16 gestures $\times$ 3 speeds $\times$ 10 trials $= 4800$ gesture examples.

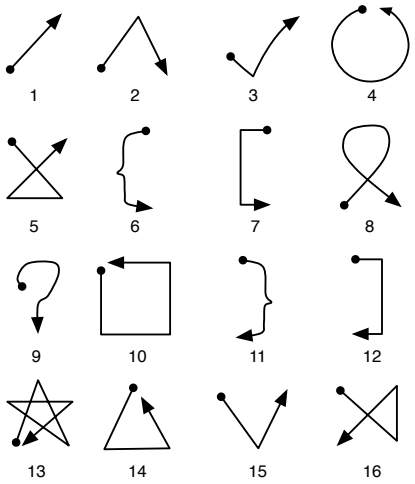

Fig. 3. Gesture vocabulary from [Wobbrock et al. 2007].

We used this database to test the recognition accuracy of GVF. We used the same evaluation procedure than in [Wobbrock et al. 2007], based on a statistical "leave-one-out" approach. One template per gesture is randomly chosen from the 10 trials, and one test example is chosen randomly from the remaining trials. This process is repeated 100 times. The procedure is applied in four distinct tests: 1) Replicated Wobbrock's evaluation procedure; 2) Looked at the effect of changing distribution parameters; 3) Took training and testing examples randomly from different speeds; and 4) Assessed the recognition accuracy on partial gestures as the testing gesture is being performed. In tests 1, 2, and 4, the training examples and the testing examples are taken from the same speed (either *slow*, *medium*, or *fast*). In test 3 the training and testing examples are taken from two different speeds.

Wobbrock uses the database to propose a simple gesture recognition method that performs pre-processing step to rotate, scale and translate data before applying different recognition algorithms. Importantly, the rotation angle and the scaling coefficient are considered to be invariant in the recognition process.

Our proposed method, GVF, on the other hand, adapts to, and is able to report on variations in these characteristics. Consequently, the state space $\mathbf{x}_k$ is comprised of four elements: the phase p_k, the speed v_k, a scaling coefficient s_k and the angle of rotation α_k. The number of particles is set to $N_s = 2000$. The complete model used for 2-dimensional input is detailed in Appendix A.2.1.

[2]Database available at: http://depts.washington.edu/aimgroup/proj/dollar/

5.1. Recognition results for same-speed examples

GVF was compared to three other methods using mean and standard deviations for recognition rate (Table I). Two of the methods ($1 recognizer and DTW) are offline methods. They both ran following a pre-processing step to correct for variations in scaling and rotation. $1 recognizer is based on the Euclidean distance between the uniformly resampled template and test shape. DTW is based on the Euclidean distance between the temporally aligned template and test shape. The other two methods, GVF and GF are online methods, reporting results as the gestures are performed. Contrary to GF, GVF can incrementally adapt to dynamic scaling and rotation variation.

Table I. Results obtained on a unistroke gesture database presented in [Wobbrock et al. 2007]. Our model has the following parameterization: $\sigma = 130$, $\nu = 0.1$.

	$1 **recognizer**	**DTW**	**GF**	**GVF**
	offline operated after scaling and rotation estimation		**online** no adaptation of scaling neither rotation	**online** incremental adaptation of scaling and rotation
Mean	97.27 %	97.86 %	95.78 %	**98.11** %
Std	2.38 %	1.76 %	2.06 %	2.35 %

First, a comparison of recognition rates of the on-line methods show that GVF gives better results than GF ($98,11\%$ vs $95,78\%$). This is due to the fact that GVF adapts to scaling and rotation. Next, comparing our method with the off-line methods, GVF gives slightly better results to the $1 recognizer and to DTW. These results are consistent with what was expected, confirming that the incremental adaptation of GVF is effective, and that the recognition accuracy of this on-line method can be at least as equivalent to standard off-line methods that correct for invariance.

5.2. Influence of the observation distribution parameters

We describe here how the parameters, standard deviation σ and Student's ν (used in the observation function, section 4.4), influence recognition accuracy. Results were obtained for a fixed set of these two parameters, and we report on the recognition rate for a large set of σ values (from 10 to 150 with step$= 10$) and ν values (0.5, 1.0, 1.5 and ∞ = Gaussian distribution). Note that these values are related to the range of the input data. In the case of the database considered here, the range is $[5, 181]$.

The variability of the recognition rate is plotted in Figure 4, superimposed on the results obtained with the $1 recognizer and DTW (methods that do not depend on distribution parameters)

Two important points must be noted. First, the best recognition rate is obtained, as expected, for a restricted range of σ and ν values. Nonetheless, the recognition varies smoothly, with a single maximum (no other local maxima). Second, the Student's t-distribution is advantageous to the Gaussian distribution ($\nu = \infty$), since it significantly reduces the sensitivity of the recognition rate to σ values. This demonstrates that data specific training procedures may not be required, since the recognition remains optimal over a large range of the Student's t-distribution parameters.

5.3. Recognition results for cross-speed examples

Complementary to the previous tests, we conducted an evaluation of recognition accuracy taking a training example from a given speed and testing with an example from the database at another speed. Table II reports the mean recognition accuracy in percent.

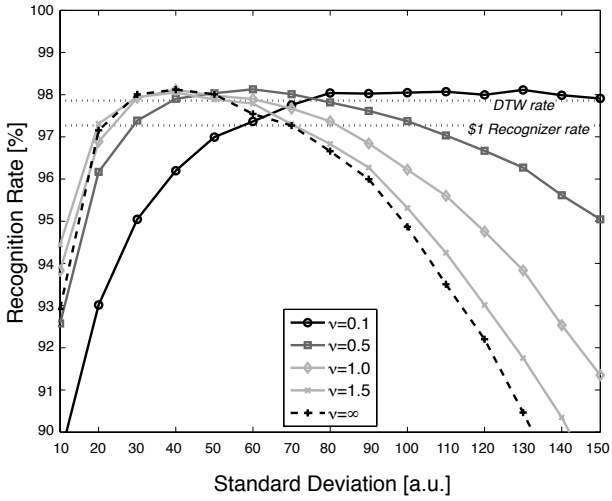

Fig. 4. Recognition rates obtained from the 2-dimensional pen gesture database by evolving observation distribution (defined as a Student's t-distribution) parameters σ and ν. While σ evolves from 10 to 150 with a step$= 10$, ν takes 4 values: $0.5, 1.0, 1.5$ and ∞.

Table II. Cross-speed results obtained on a unistroke gesture database presented in [Wobbrock et al. 2007] considering training and testing examples from different speeds $\sigma = 130$, $\nu = 0.1$.

		Training Examples					
		Slow		Medium		Fast	
		$1	GVF	$1	GVF	$1	GVF
Testing Examples	Slow	96.2%	97.3 %	94.9%	93.2 %	91.6%	85.9 %
	Medium	96.1%	95.4 %	97.1%	98.6 %	94.1%	96.6 %
	Fast	92.9%	88.3 %	94.6 %	97.5 %	95.5%	98.2 %

The results show that the global recognition rate obtained with GVF remains high at 94.6% (std=4.6%) and is equivalent with the $1 method that obtains a recognition rate of 94.8% (std=1.7%). The lowest rates are obtained when both the testing and training examples are taken from contrasted speeds: either slow–fast (recognition rate at 88.3%) or fast–slow (recognition rate at 85.9%).

5.4. Early Recognition Results

Finally we assess the evolving recognition rate while the testing gesture is performed, also called early recognition. Figure 5 illustrates the results that are compared to the GF early recognition rates. The recognition rate obtained with GVF attains 67% when just 10% of the gesture has been performed, and goes up as more of the entire gesture is available for continual testing. 90% recognition is reached at 40% of the gesture on average. On the contrary, GF globally attains lower recognition accuracy.

5.5. Observations from the evaluation on 2-dimensional gesture database

The experiment on 2D drawing gestures presented in this section confirms that the GVF method works equally or better than state-of-the-art recognition methods. This demonstrates that the on-line scaling, rotation and speed estimation of the GVF method performs at least as efficiently as standards off-line methods ($1 recognizer

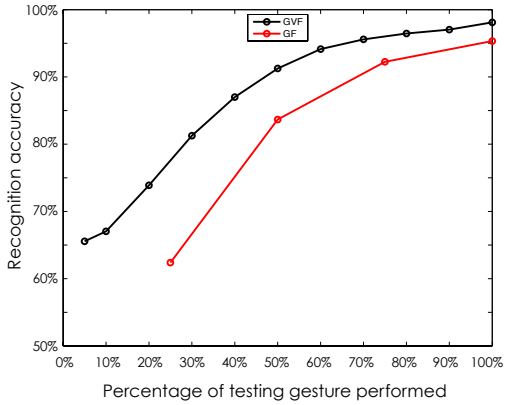

Fig. 5. Early recognition rates obtained with GVF and GF (for a more complete study of the GF evaluation on the same database, please refer to [Zamborlin et al. 2014])

and DTW implementation of Wobbrock et al. [Wobbrock et al. 2007]). The GF, which does not taking into account such invariance, performs worse.

Compared to the GF method that uses a Gaussian distribution for observation likelihood function, the GVF method uses a Student's t-distribution. This choice significantly reduces the sensitivity of the standard deviation parameter σ. Since this parameter which might *a priori* be difficult to estimate with limited training data, the use of Student's t-distribution can broaden the applicability of the method.

In addition, the results obtained with GVF are remarkable considering the fact that it operates in a causal manner: the recognition results and the parameters adaptation are updated each time a new sample is received. On the contrary, standard recognition schemes compute the results only once the gesture is finished (as DTW, Rubine or $\$1$ recognizer), which generally allows for a more comprehensive decoding algorithm. Thus, it demonstrates that the causal inference is robust, thanks to the coupling imposed between the phase and velocity estimation. Precisely, the phase and velocity are coupled through a kinematic model (similarly to a Kalman filter, see 4.3). This forces the tracking to be continuous along the state sequence.

Finally, the early recognition has been shown to be accurate. Indeed, we reached a recognition rate of 65% by considering only the first 5% of the gesture performed. In comparison, GF needs almost 35% of the gesture to be completed before reaching the same level of recognition accuracy. This is a clear advantage for the aimed interactive applications.

6. ASSESSING ADAPTATION ON SYNTHETIC DATA

In this section we present an evaluation of GVF for the adaptation task, independent of recognition. Since it would not be possible for a human user to provide ground truth on which we could evaluate the ability of the algorithm to adapt to varying features, we use synthetic data as a means to provide controlled, quantitative variations. We consider two different cases. In the first, only the phase and scaling are adapted in the inference. In the second case, the system also adapts to rotation angle.

In both cases, we consider synthetic data obtained by using Viviani's curve:

$$\mathbf{C}(t) = \begin{cases} x(t) &= a(1+\cos(t)) \\ y(t) &= a\sin(t) \\ z(t) &= 2a\sin(t/2) \end{cases} \tag{9}$$

6.1. Temporal Alignment Assessment

We define two distinct curves for the test and template data. The template gesture is obtained by a regular sampling of the curve described by Equation (9), and the input gesture is obtained by a non-linear sampling ($t \mapsto t^3$) of the same function, and by adding a uniformly distributed noise. We denote with $\mathbf{C}$ and $\hat{\mathbf{C}}$, the original and the resampled curves, respectively.

$$\hat{\mathbf{C}}(t) = \mathbf{C}(t^3) + \mathcal{N}(\mathbf{0}, \sigma_{\mathbf{C}}) \tag{10}$$

For this first case, we used a state space defined as a three-dimensional vector, consisting of the phase p_k, the speed v_k and the scale s_k (this model is similar to the one presented in [Black and Jepson 1998a]):

$$\mathbf{x}_k = (p_k, v_k, s_k)^T \in [0, 1] \times \mathbb{R}^2$$

The phase feature p_k lies in the interval $[0, 1]$. The velocity v_k and scale s_k are normalized, a value of 1 corresponding to the speed (resp. scale) of the template. The f function involved in the distance function for the observation likelihood (cf. Equation (5)) is

$$f(\mathbf{x}_k, \mathbf{g}) = \text{diag}(s_k)\mathbf{g}(p_k)$$

where $\text{diag}(s_k)$ is the diagonal matrix, of size 3×3, which elements are equal to the scaling s_k. The scaling coefficient is identical for all three input observations $x(t)$, $y(t)$, $z(t)$ (homothetic transformation).

In this first experiment, we compare GVF with the GF model. To do so, we set $\nu \to \infty$, converging to the gaussian distribution with standard deviation σ used in the GF. The influence of the ν value will be discussed in Section 5.

We present here results to do with the estimation of the phase p_k, that describes the alignment between the test and template data. For each test, our model returns the estimated phase p_k which, in this evaluation, should ideally follow the cubic function that was used to synthesize the curve $\hat{\mathbf{C}}$. From this, we calculated the mean square error between p_k and the ground-truth cubic function k^3. The number of particles was set to $N_s = 1800$.

In addition, we compare the estimated phase p_k obtained by GVF against the same estimated alignment feature obtained by GF. The results are seen in Figure 6. The estimated phase p_k is plotted along the cubic function (at the top of Fig. 6). For both models, the estimated p_k is close to the expected curve (Figure 6, middle and bottom plots). The average error for GVF (resp. GF) is 1.3 sample with std=0.7 (resp. 2.3 samples with std=1.4).

We further examined the influence of the parameter σ, used in the probability distribution, as well as the noise level in the input data (measured as the signal-to-noise ratio [SNR] in dB). Figure 7 shows those results. On the leftmost plot, we varied σ between 0.05 and 1.0 (with a step of 0.05), for both GVF and GF. On the rightmost plot, we varied the noise level in the input data (SNR was varied between 12dB and 45dB). In both plots, the gray curve is the error curve obtained from the GF model and the black curve is from our model.

We found, first that, for all σ values between 0.05 and 1.0, the errors for our model are lower than for the HMM-based model (Figure 7, left). Interestingly our model, based on approximative inference (particle filter), can obtain better results than the exact inference of GF (forward procedure). This is due to the continuous latent model that better represents the data as well as the fact that p_k and v_k values are linked through a first order motion equation, while such a constraint is not taken into account in the

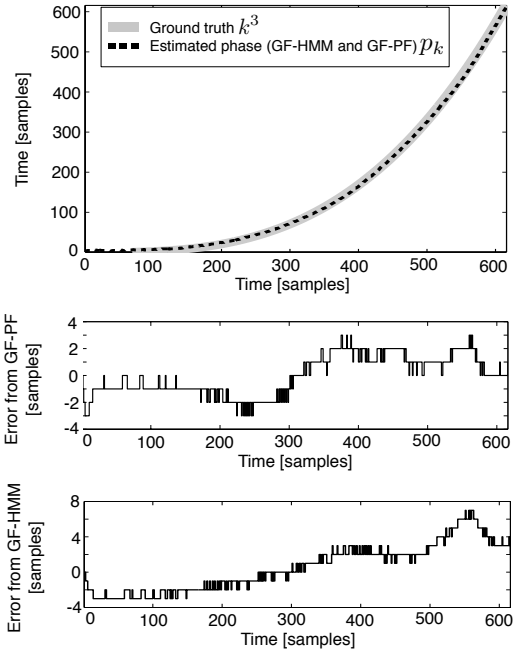

Fig. 6. Example of the estimated phase p_k (top plot, dashed black line) compared to the ground truth defined as a cubic function (top plot, gray solid line). The example is obtained with a observation likelihood with standard deviation $\sigma = 0.1$ that equals to the additive gaussian noise used for the test. Middle, resp. bottom, plot reports the error between the estimated alignment and the ground truth for GVF, resp. GF.

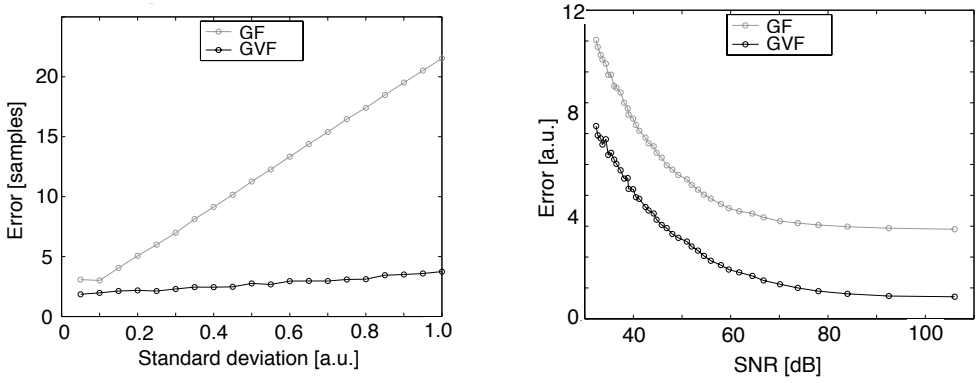

Fig. 7. Error curves obtained from both our model (black line) and GF (gray line). The error is computed from the mean square error between the ground truth cubic function and the estimated phase. On the left, the tested parameter is the parameter σ of the Gaussian distribution of the observations. On the right, the tested parameter is the noise level in the input data, given by the standard deviation of the additive Gaussian noise.

GF model. In other words, the phase p_k estimation is made more robust by the joint estimate of the speed v_k. Second, we found that our model better handles the level of noise in the input data compared to GF (Figure 7, right).

6.2. Rotation matrix adaptation

In this section, we examine the case where gesture rotation angle varies dynamically over time. We consider the three angles ϕ, θ, ψ around x, y, z, respectively, in a Cartesian coordinate system. Their time series are defined as follows:

$$\begin{cases} \phi(t) = t^2 \\ \theta(t) = t \\ \psi(t) = -t^{1/3} \end{cases} \tag{11}$$

The 3-dimensional template curve $\mathbf{C}$ (equation (9)) is rotated according to this matrix in the Cartesian frame (x, y, z). The input curve is the rotated version of $\mathbf{C}$ with added gaussian noise:

$$\hat{\mathbf{C}}(t) = R(\phi(t), \theta(t), \psi(t))\mathbf{C}(t) + \mathcal{N}(\mathbf{0}, \sigma_{\mathbf{C}})$$

The rotation matrix $R(\phi(t), \theta(t), \psi(t))$ is computed each time step. Details on our use of conventions for angles and rotation in the 3-dimension Cartesian frame are given in Appendix A.2.

The state space is defined as a 5-dimensional vector that consists of the phase p_k, velocity v_k, and the angles ϕ_k, θ_k, ψ_k. The state variable at time k is:

$$\mathbf{x}_k = (p_k, v_k, \phi_k, \theta_k, \psi_k)^T \in [0, 1] \times \mathbb{R}^4$$

The observation likelihood is entirely defined by the following f function:

$$f(\mathbf{x}_k, \mathbf{g}) = R(\phi_k, \theta_k, \psi_k)\mathbf{g}(p_k)$$

Figure 8 shows an example where the estimated angles ϕ_k, θ_k, ψ_k are plotted against the ground truth (defined by Equation (11)). In this example, the standard deviation σ is set to the standard deviation of the input data ($\sigma = \sigma_{\mathbf{C}} = 0.1$). The number of particles was set to $N_s = 1800$.

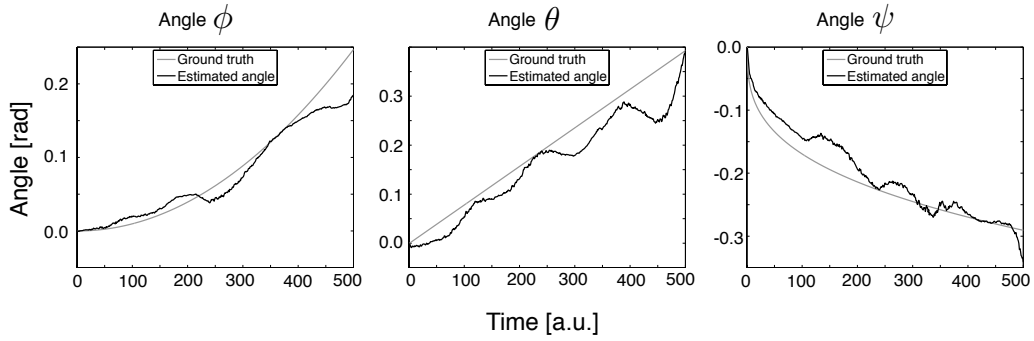

Fig. 8. Dynamic rotation estimation. The input curve is the one given by Equation 11 with $\sigma_{\mathbf{C}} = 0.1$. The model is configured to estimate the phase p_k, the speed v_k and the three angles ϕ_k, θ_k, ψ_k. The standard deviation used in the model is 0.1.

We performed the same evaluation as the one assessing temporal alignment. We tested the effect of the standard deviation σ of the Gaussian observation distribution by varying its value between 0.05 to 1.4 (step$= 0.05$) as well as the effect of noise in the input data, varying the SNR between 3dB to 30dB. Figure 9 reports the results. On the left, we report the error curve function of σ. On the right, we report the error

curve function of data noise given by the SNR. In both cases, the error is computed as the mean square error between estimation of the angle ϕ, θ, ψ and the ground truth angles.

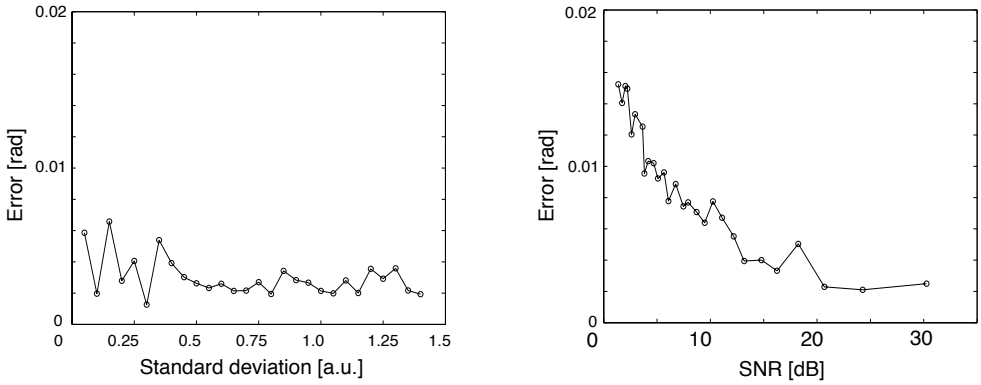

Fig. 9. Error curves obtained from our model. The error is computed from the mean square error between the ground truth angles and the estimated ones. On the left, the tested parameter is the parameter σ of the Gaussian distribution of the observations. On the right, the tested parameter is the noise level in the input data, given by the standard deviation of the additive Gaussian noise.

First, the results illustrate that the standard deviation σ of the observation distribution has a weak influence on the angle accuracy (on the left of Figure 9). For small values of σ, the error curve shows more variations than for higher values of σ where the adaptation is more stable.

Second, the noise in the data has a significant influence on the angle accuracy, as shown on the right of Figure 9. The error curve increases as noise level increases (i.e. decreasing SNR).

6.3. Observations from the evaluation on synthetic data

The results of this evaluation show that for a fixed parameter σ and a fixed level of noise, the temporal alignment is reliable (the average absolute error is 2.3 samples with a standard deviation of 1.4). For an incoming data stream at a sampling period of 20 milliseconds, this means that the results would be estimated with a latency of just 46 ms.

Both evaluations on the temporal alignment and the rotation angles estimation showed that the errors obtained while varying the parameter σ remains low and barely varies. This is important considering that only one or few examples are available for training the model: the σ value can be easily initiated and easily optimized.

The evaluation also shows that the level of noise in the data has an expected effect on the accuracy. The error can be diminished by increasing the number of particles (which increases the computational cost) and by optimally adjusting the variance in the observation distribution. Nevertheless, the algorithm still operates even in the presence of significant noise.

7. USER STUDY: ADAPTATION AND RECOGNITION IN AN INTERACTIVE SCENARIO

In this section, we present a user study evaluation of the GVF method in an interactive context using 3-dimensional gestures. We assess the ability of the method to efficiently recognize real world gestures and adapt to their variation in a context where users' arm movements manipulate audio playback and processing.

7.1. Study

7.1.1. Presentation. We built an application that uses of sound feedback to respond to salient gesture variation. Performing a given gesture initiates a specific sound, which are then continuously manipulated depending how the gesture is performed. Interaction with the application involves the two fundamental aspects of the interaction model illustrated in Figure 1: selection (discrete command) and manipulation (continuous command).

— **Selection**. As our method allows for early recognition, recognition output will be used to trigger a sound associated to the gesture performed while the gesture is still being executed. We built a vocabulary of 3 gestures taken from the previous experiments presented earlier. Each gesture has a associated sound. Figure 10 illustrates each gesture and associated sound. Each sound is of short length with an average length of 4.9 seconds ($\sigma = 0.7$ sec).

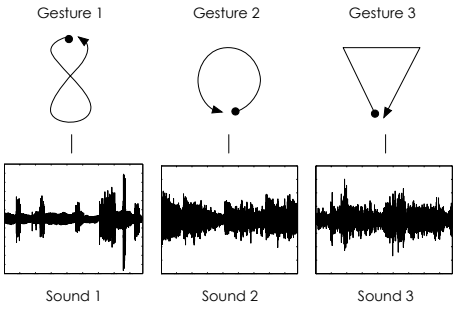

Fig. 10. Gesture vocabulary and sounds associated.

Given the ambiguity that continuous recognition has in recognizing at the beginning of gesture performance, in the application we will trigger the sound as soon as the recognized gesture has a probability greater than 0.5.

— **Manipulation**. Variations in gesture characteristics of the recognized gesture are estimated and used to continuously manipulate characteristics in the sound. In the application, we allow the following gesture variations: *slower / faster, smaller / bigger, tilt*. Variation in gesture speed, relative to the template gesture, is mapped to time stretching of the sound playback. Variation in size is mapped to the volume of the sound: a smaller gesture will play a sound more quietly while bigger gestures play the sound louder. Finally variation in tilt controls the cut-off frequency of a high-pass filter creating a stifling, distancing effect. Figure 11 illustrates the variations allowed in the application and their link to the sound manipulation.

7.1.2. Hypothesis and experiment design. In this study we seek to validate the hypothesis that our algorithm is able to dynamically and accurately adapt to gesture variation in order to be used in a closed-loop gesture-to-sound interaction.

In order to validate this hypothesis we propose to define a set of tasks asking the participant to play a given sound and to modify it (through one, or a combination of multiple, variations). To that extent, the participant must use the sound feedback, entirely controlled by the estimated variations, in order to achieve the task. A set of 7 sound modifications corresponding to gesture variations (Figure 11) are summarized in Table III. Note that Task 1 does not involve any variation, Tasks 2 to 4 involve a global modification of one aspect of the sound, Task 5 involves a dynamic change of one

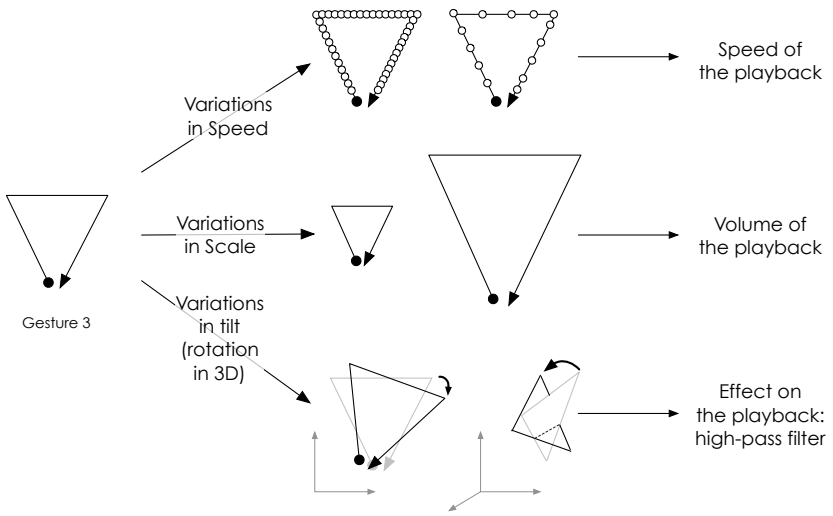

Fig. 11. Gesture variations allowed in the application: variations in size, speed and tilt. Each variation is used to control the sound feedback parameter: loudness, playback speed and filtering.

characteristic and Tasks 6 and 7 involve the global change of two characteristics of the sound. The study follows a within-subject design with one factor tested: the TASK.

Table III. Set of 7 tasks used in the user study

Modifications	
Id.	Description
T1	Original
T2	Louder
T3	Faster
T4	High-pass filtered
T5	Louder then quieter
T6	Slower and quieter
T7	Louder and high-pass filtered

7.1.3. Apparatus. Participants perform free space gestures with their hand. The hand motion is capture through an infra-red based device tracking finger and hand motion[3]. The device returns the position in the 3-dimensional space of the palm's centroid sampled at 80 frames per second. The data are then slightly downsampled in the application at a rate of 50Hz. The application is implemented in the realtime audiovisual programming environment Max/MSP[4]. The raw 3-dimensional gesture data are used as input data for the GVF implemented as a plug-in in Max/MSP (note that the code source is open and available online[5]). The output of the GVF is mapped to parameters of a phase vocoder synthesizer, SuperVP[6]. Participants listened to stereo audio feedback through a pair of high quality speakers set up in an isolated control booth.

[3]Leap Motion, `http://www.leapmotion.com`
[4]`http://www.cycling74.com`
[5]https://github.com/bcaramiaux/gvf
[6]`http://forumnet.ircam.fr/product/supervp-max/`

7.1.4. Procedure. The procedure is comprised of three steps. In Step 1, the participants are asked to perform each gesture a few times in order to get accustomed to the apparatus. The gestures themselves were chosen to be fairly simple and do not require a time to be mastered by the participants. After the participant has practiced several times, we record one example of the gestures the GVF template that will subsequently be used for recognition and adaptation.

In the second step, we introduce the sound playback and processing part of the application to the participants. We explain that each sound is associated to one gesture and we play each of the sounds by way of example. We then explain the mapping, telling them that performing a gesture will trigger the associated sound and varying some specific aspect of the sound. In order for them to understand the mapping, we give them a few minutes to freely explore the interaction.

Step 3 is the main part of the experiment: the controlled session. During this part, for each of the 3 base sounds, the participants are asked to play the sound by performing the corresponding gesture and modify it by applying specific variations reported in Table III. The order of the tasks presented is randomized as well as the order of the gestures. Participants have an unlimited number of trials to play a sound in a given task. Once they feel comfortable, they record the gesture 3 times.

7.1.5. Data collection and analysis. We invited 10 participants to be part of the user study. All the participants have a background in sound or music, meaning that they understood the audio processing nature of the task. We collected a total of 10 (participants) $\times$ 3 (gestures) $\times$ 3 (trials) $\times$ 7 (tasks) $=$ 630 gesture examples.

For each gesture collected, we analyzed the data by computing the relative size and speed. The relative size is given by the square root of the quotient of the area of a given gesture, performed under task i, divided by the area of the same gesture recorded as the reference in the first step of the experiment (see Section 7.1.4 for the whole procedure). Similarly, the relative speed is computed by taking the length of the given gesture (under task i) divided by the length of the reference gesture. We call such characteristics we computed offline, *post-processing analysis*, to distinguish them from the online estimation returned by GVF.

Statistics on GVF variations estimation is done by taking the estimated value at 75% of the gestures. The rationale behind the choice is that taking the mean over the whole estimation, from the starting point of the gesture to the ending point, will underestimate the actual estimation since the starting values (initial conditions) are always 1 (the original size and speed deviation) or 0 (original angle deviation).

Finally, we compute the characteristics of each sound produced in order to assess if the participants actually achieved the task. By analyzing the audio signal, we compute three characteristics: relative amplitude, relative duration and the relative spectral centroid. The three characteristics are relative to the original sounds in the database.

7.2. Assessing recognition accuracy through selection rate

An initial quantitative evaluation of the recognition was carried out on the GVF estimated gesture probabilities for the three trials recorded by the participants for each gesture and each task. Here we report the statistics on recognized gestures across all the participants. Overall we found a recognition rate of 97.3%. In other words, over the 610 gesture examples, the algorithm misclassified only 17 of them.

We then performed an analysis to assess the ability of GVF to provide realtime recognition of the input gesture. A gesture is considered recognized and triggers the corresponding sound when its probability with respect to the others is greater than 0.5. We computed the point within the incomplete gesture on average recognition is achieved. The results (Figure 12) show that recognition is achieved on average at 12.6%

(std=2.4%) into the gesture (where 0% is the beginning of gesture and 100% is gesture end). Converted to time, the average latency created by this selection criterion is 490.6ms (std=70.2ms).

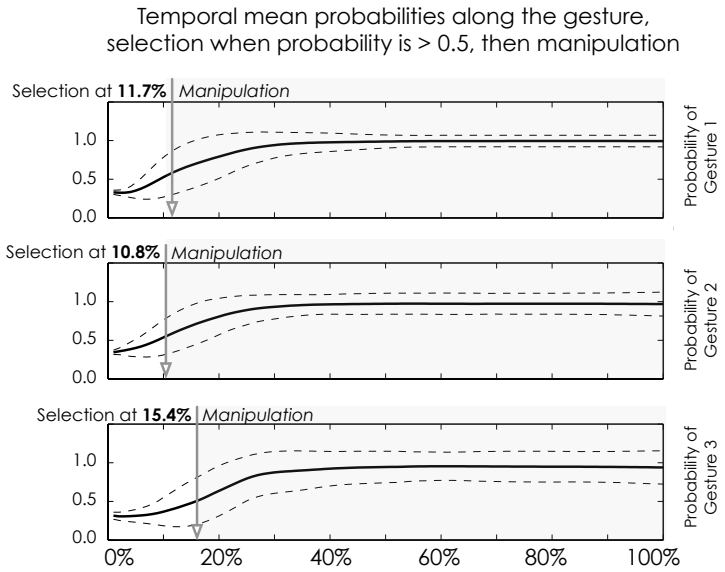

Fig. 12. Sound selection: proportion of gesture at which the gesture probability attains 0.5 and consequently the associated sound is selected.

The sound manipulation phase based on variation adaptation starts once the selection has been made. In the following we turn our attention to this part of the task.

7.3. Adaptation of one characteristic

Participants are asked to change one characteristic of the sound: *volume* (Task 2), *speed* (Task 3), or *filter* (Task 4). We report on the results from these three tasks below. For each Task we compared with the statistics obtained from Task 1: playing the original sound. We used a Student's T-Test with $\alpha = 0.01$ to assess differences between mean values obtained between these two Task.

7.3.1. Task 2: Playing the sound Louder. Figure 13 shows results obtained from Task 2 compared to Task 1. On the left is the average estimation of relative size according to the task for both the online results reported by GVF (green) plotted next to post-processing analysis (yellow) as reference. The right side of the figure shows the actual relative audio amplitude computed by analyzing the sound output.

First, the right side of the figure shows that the participants successfully accomplished the task by producing a sound that was louder than the original sound (mean values for the audio amplitude are significantly different between Task 1 and Task 2, $p < 0.01$).

Let us now inspect the gesture sizes on the left side of the figure. The mean relative size values obtained by post-processing analysis shows a significant increase between Task 1 and Task 2 ($p < 0.01$) meaning that participants did actually perform a bigger gesture. Regarding the estimated relative size by GVF, it also shows a significant increase between these two tasks ($p < 0.01$). In addition, the estimations values obtained

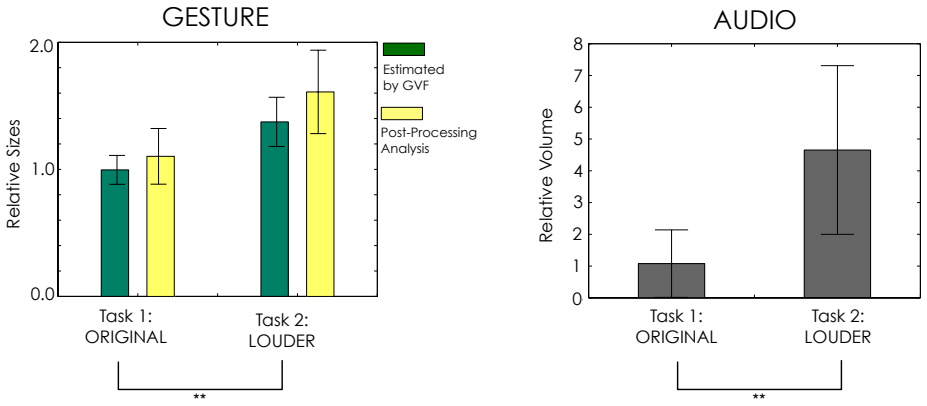

Fig. 13. Task 2: Playing the sound louder. Significance code: $**, p < 0.01$, $*, p < 0.05$.

from the post-processing analysis and by GVF do not differ at a significant level, which means that GVF converged to the actual global size of the gesture.

7.3.2. Task 3: Playing the sound Faster. We perform a similar analysis when participants were asked to play the sounds faster (Task 3) compared to the original (Task 1). The results obtained are illustrated in Figure 14. The left shows the average estimation of the relative speed reported online by GVF (green) and by post-processing analysis (yellow). On the right, the actual resulting average relative sound durations are calculated from the recorded sound output.

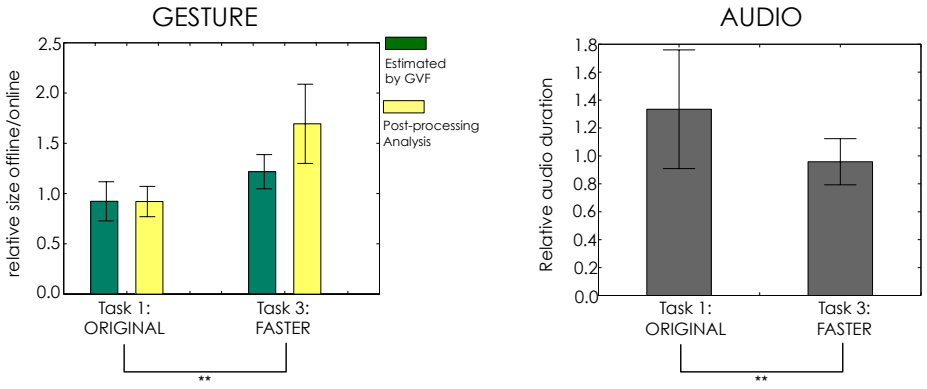

Fig. 14. Task 3: Playing the sound Faster. Significance code: $**, p < 0.01$, $*, p < 0.05$.

The right side of the figure shows that the mean sound duration is significantly lower for Task 3 than Task 1 ($p < 0.01$) which means that the participants achieved the task by producing faster (i.e. shorter) sounds. Note that participants seemed to play the sound slightly slower than the real sound when asked to play the original sound (the mean relative duration is 1.3, std= 0.6).

If we examine the relative speeds of the gestures performed to produce the sounds (Figure 14, left), the post-processing analysis shows a significant increase of the rel-

ative speed between Task 1 and Task 3 ($p < 0.01$) meaning that the participants performed their gesture faster in order to play the sounds faster. Regarding the online estimation given by GVF, the relative speed also significantly increases between Task 1 and Task 3 ($p < 0.01$).

7.3.3. Task 4: Playing the sound High-pass filtered. Finally, we examine the angles of rotation estimated in the experiment. The left side of Figure 15 shows the norm of the vector of the three relative angles (ϕ, θ, ψ) estimated by GVF. Note that the post-processing analysis is not reported here. Post-processing analysis using Horn's quaternion-based method has been tested on the data and returned inconsistency. This was due mainly to the fact that the tilt was not constant over time due to the lack of physical reference in free space to maintain constant tilt. On the right, we report the average relative spectral centroid values computed from the sounds produced for Tasks 1 and 4 compared to the original sound (1.0).

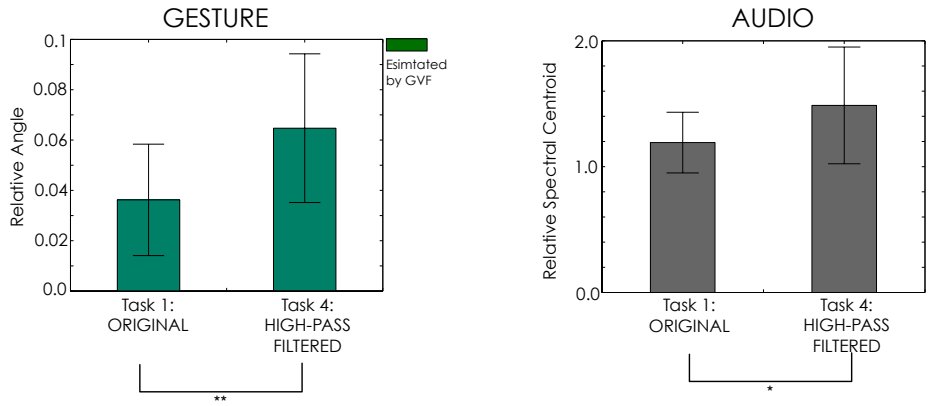

Fig. 15. Task 4: Playing the sound high-pass filtered. Significance code: $**, p < 0.01$, $*, p < 0.05$.

The relative spectral centroid values in the audio output (right) show a significant increase between both tasks ($p < 0.05$). In other words, the mean frequency in the sound produced increases, revealing the application of a high-pass filter. Regarding the estimation of the angles (left), the norm of the relative angle also increases between Task 1 and Task 4 ($p < 0.01$). Note that the angle estimation is not zero for gestures performed where the task was to play the original sound, leading to a relative spectral centroid slightly greater than 1.

7.4. Dynamic adaptation of one characteristic: Size

Task 5 involved a dynamic, time varying modification of the volume of the sound. We asked the participants to play the sound louder at the beginning, gradually becoming quieter at the end. Figure 16 reports the results by plotting the mean curve of estimated size over the course of gesture execution (in percentage). The mean curve is reported as the solid black line while dashed lines represent the dynamic standard deviations.

Considering the values at 0%, 50% and 100%, the statistical test shows that the size significantly increases between the beginning and the middle of the gesture ($p < 0.01$) and significantly decreases between the middle and the end of the gesture ($p < 0.01$),

showing that the participants successfully changed the size dynamically according to the task, relative to the initial amplitude (1.0).

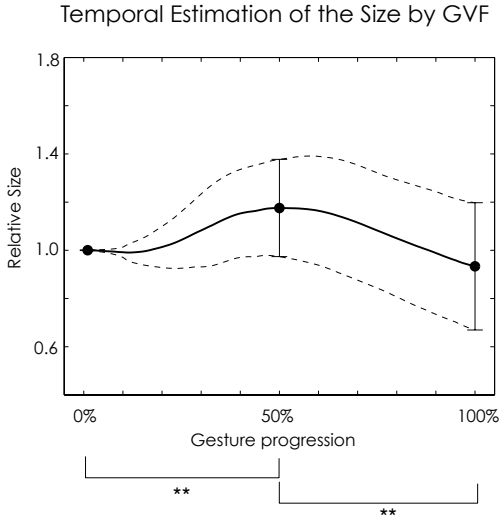

Fig. 16. Dynamic variations of size in gestures during Task 5. On the left we report the correlation between both offline and online correlations. On the right we report the size estimations averaged over the first, second and last third of the gesture.

7.5. Adaptation of two characteristics

The two last tasks involved the joint simultaneous modification of two sound characteristics: Task 6 was to play Slower and Quieter while Task 7 Louder and High-Pass Filtered.

7.5.1. Task 6: Slower and Quieter. Figure 17 reports the results in a similar way to Figure 15 but with two graphs, one for each sound feature being modified. The right side of the figure shows the relative duration (above) and the relative volume (below) of the sound output . On the left side, the top plot illustrates the estimated speed and below the relative size of the gesture performed by the user in the task.

The figure shows that the actual duration of sound output significantly increases (right, top) while the volume significantly decreases (right, bottom) ($p < 0.01$). This is linked to the gesture characteristics - faster gesture for shorter sound, and larger gesture for louder sound. Indeed, the post-processing analysis values (left, yellow) significantly decrease when performed Task 6 while the size also decreases ($p < 0.01$). The online estimation by GVF gives a similar result (with $p < 0.01$).

7.5.2. Task 7: Louder and High-pass Filtered. Figure 18 reports on the results from Task 7. On the right top we have the Relative Volume and below the Relative Spectral Centroid of the sound output. The left top plot illustrates the estimated Size and below the norm of the Angles of rotation.

For the audio (right), the analysis shows that the volume as well as the spectral centroid significantly increase ($p < 0.01$). In other words, the sounds produced by the users' gestures are globally louder and high pass filtered, showing that the users accomplished the given task. The relative gesture size (left) given by post-processing

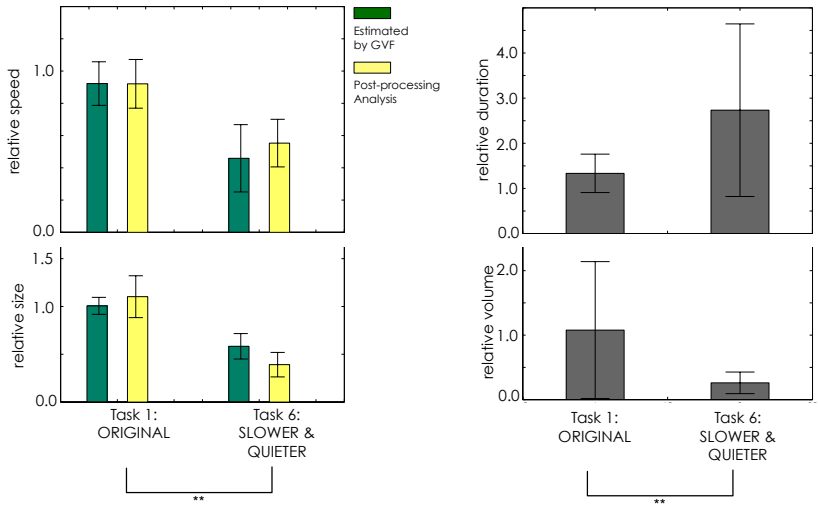

Fig. 17. Task 6: Slower and Quieter. Significance code: $**, p < 0.01$, $*, p < 0.05$.

calculation significantly increases, $p < 0.01$, when trying to play the sound louder and filtered. The online estimation by GVF, parallels the post-processing estimated values and the size significantly increases ($p < 0.01$), with the norm of the angles also increases ($p < 0.01$).

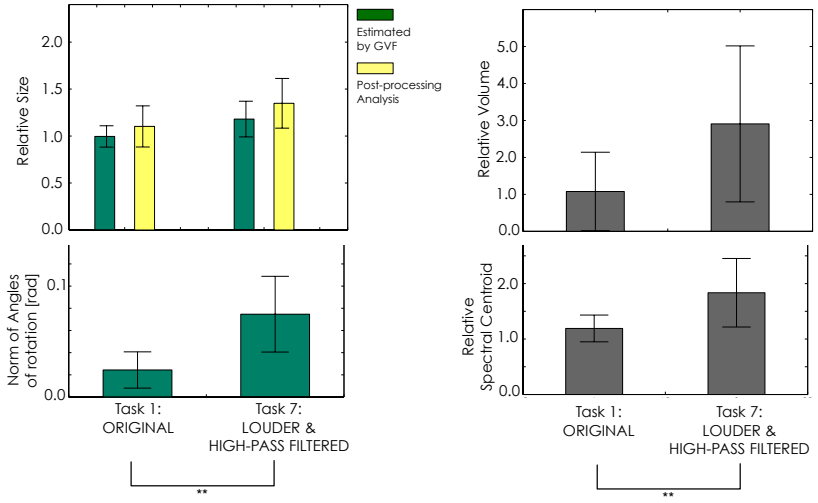

Fig. 18. Task 7: Louder and High-pass Filtered. Significance code: $**, p < 0.01$, $*, p < 0.05$.

7.6. Observations from the user study

Participants in this study were on the whole successful in playing sounds using variation in their gesture to articulate changes in speed, volume or filtering, and combi-

nations of these manipulations. The task was understood and successfully fulfilled by the participants. Audio analysis of the sound output showed that the users succeeded in modifying the sounds' characteristics according to the tasks considered. In addition, the gesture variations used to achieve the tasks were coherent with the sound produced and the gesture-to-sound mapping implemented. Consequently, the online estimation by GVF embody the variations asked on the sound either in speed, size or orientation and converge with the reference post-processing calculation of gesture differences. This has been shown to also be accurate when users were asked to vary two sound characteristics simultaneously.

A very important feature resides in that the adaptation is dynamic, along the gesture progression, starting at the selection. As it was configured, the algorithm imposes certain initial conditions that are as follows: phase set to 0, scales and speed to 1 and angles of orientation to 0. Then the algorithm dynamically adapts to the variations in gesture performance. We saw that the participants were successful in controlling dynamically the size: bigger at the beginning and smaller at the end.

8. DISCUSSION

We discuss in this section the main features of the algorithm based on the results obtained from the algorithm evaluation and user study. We look at capability of the method to carry out early recognition and adaptation, and we outline limitations of the algorithm and of its use.

8.1. Early recognition and adaptation

Early recognition is a process which performs realtime classification. In other words, once testing data is received, the process continuously assigns probabilities to each class of the base vocabulary and returns the class with highest probability. We showed that the GVF method needs fewer frames than the HMM-based GF method to perform accurate classification on a database of 2-dimensional pen gestures.

This ability to perform recognition mid-gesture is interesting in the interactive context illustrated in Figure 1. Early recognition allows for the selection of a media asset (such as sound) at the beginning of a gesture and provides scope for continuous interaction throughout the rest of the gesture. This creates an important gestural interaction dynamic in which continuous control is coincident with continuous input. On the other hand, recognition techniques such as the $\$1$ recognizer [Wobbrock et al. 2007] perform selection after completion of the gesture, not allowing for continuous interaction. The GVF method has been shown to perform with a recognition accuracy as good as those methods while allowing low latency selection in a continuous interaction context (Section 7).

Continuous interaction leverages the algorithm's adaptation feature. GVF has been shown to be able to dynamically adapt to gesture variation. Results obtained in Section 7.4 of the user study illustrate the dynamic process of adaptation to gesture size. It shows that the estimated value starts at 1 (the initial condition) and then converges towards the correct size values as the users modify the gesture size dynamically: bigger gesture at the middle and smaller at the end (see Figure 16). On the other hand, strategies based on pre-processing used in offline methods such as the ones presented in [Wobbrock et al. 2007] (based on Euclidean distance or Dynamic Time Warping) would not be able to handle these dynamically changing variations.

Dynamic adaptation is an advantageous feature for the estimation of variations that are hard to maintain constant throughout the gesture performance, such as rotation angles. Given that tilting a gesture in 3 dimensions is a difficult task, users tend to rotate gestures at angles which are not constant throughout the performance. This leads to non-affine transformations of the pattern which require us to take into ac-

count time-varying parameters. GVF is designed to handle such transformations. We observed that offline methods returned non-consistent estimations of the rotation angles, whereas GVF was able to adapt and report a consistent value.

Note that probabilistic machine learning methods such as HMMs could adapt to dynamic variations provided prior knowledge on the type and range of variations, but would require actual examples of these variations. On the contrary our method only requires one template reference per gesture class and is able to adapt to a wide range of variations of each gesture without explicit examples. This represents an advantage in applications for which an exhaustive database containing all gesture variations is not readily available, or potential gesture variations cannot be known beforehand.

8.2. Limitations

Dynamic adaptation is not instantaneous and implies a latency that must be taken into account in the application design. In the user study, we showed that the method significantly converges towards increasing or decreasing speeds but, in the task, *Faster*, the algorithm does not attain the actual change in speed: speed is underestimated. This is a constraint of the algorithm due to the convergence time required by the particle filtering implementation. The time needed by the particle filter to converge to the correct estimation is longer than the actual duration of the gesture, in the case of a quickly executed variant. The speed of convergence is determined by the noise parameters of the Gaussian transition distribution, which govern the speed and precision of adaptation. Hence a trade-off has to be found in order to balance convergence time and estimation precision. Consequently, this also reveals that the algorithm parameters need to be fine tuned to the interaction context. These parameters have a direct impact on the performance of the algorithm but also allows it to be more flexible to different interaction scenarios: wide or narrow variations, fast or slow convergence, precise or loose. Future work would investigate the impact of these parameters on the usability of the interaction potential provided by GVF.

Note that a possible improvement of the speed of convergence would then be in adding constraints in the transition model which would require prior knowledge on variations possible dynamics.

8.3. The user factor

Usability of the proposed method depends on the algorithm itself but also the user's ability in controlling variations in their gestures. Indeed, to some extent, gesture characteristics are coupled due to constraints from the human motor system.

The ability of users to reliably control combinations of variations of gesture has been previously illustrated in [Caramiaux et al. 2013]. While performing a gesture at normal or fast speeds, the "2/3 Power Law" of motion applies. This law states that there is a strong correlation between instantaneous speed and curvature [Viviani and Flash 1995]. This means that the drawing speed cannot be constant over the pattern (even if the user perceives it as constant), with each drawing pattern having a specific time/speed profile. Therefore, in order to compare gesture speed across a vocabulary of different stroke gestures, we need to consider average speed as calculated over the whole pattern. The law called *isochrony* further establishes that average execution speed tends, for a given person, to be constant independent of size. These two laws clearly establish that speed is not a parameter we are accustomed to controlling or varying in everyday drawing tasks. These facts only hold for "ballistic gestures", performed sufficiently quickly without feedback. When performed sufficiently slowly, gestures (non-ballistic in nature) are controlled through a sensorimotor loop using feedback such as vision. In such cases, the human motion law we mentioned does not hold

and an independent control on the variation of gesture characteristics (e.g. speed, size) is possible.

Based on these results, in the user study presented in Section 7, we placed the participants in a closed sensorimotor loop: gesture performed continuously affects a sound that in turn feedback to influence the gesture being performed. Indeed, the purpose was to reduce the impact of coupling between gesture characteristics resulting from motor control. Although, an important future work is to investigate the precise impact of these laws of motion on 3-dimensional gestures in interactive contexts.

9. CONCLUSION

In this paper, we described a method, called GVF, for gesture recognition that can adapt and estimate in real-time variations occurring during gesture execution. We demonstrated in this paper the ability of our method to track changes in phase, scaling and rotation. Extension of the method to adapt to other types of variation would be straightforward using the same formalism, and can be defined in a flexible manner. Feature initial values can be chosen arbitrary in one or several distinct intervals.

GVF belongs by design to the family of template-based methods. Therefore, we compared it with similar approaches such as DTW or more recent methods such as the $\$1$ recognizer [Wobbrock et al. 2007] or GF [Bevilacqua et al. 2010]. It globally obtained better results in accuracy, either in adaptation only and recognition with adaptation. We did not to perform comparisons with other methods that require training on a large number of examples because such approaches do not fit our application constraint of making the training phase as easy as possible for the end user.

The first important feature of the algorithm is that GVF adapts dynamically to large differences between the gesture performance and the templates without need for explicit examples of the variations themselves. This dynamic adaptation is particularly important since it is aimed to be used where the gesture classes are defined using single templates. In practice, this allows user for authoring small gesture datasets while ensuring good recognition accuracy. In our case, adaptation is used not only to improve the recognition, as generally found in the state of the art, but also allows us to characterize gesture variation. Such variation estimations are are useful in interaction design contexts requiring continuous interaction. This feature has been illustrated in an application involving continuous manipulation of sound playback.

Another important feature of the algorithm resides in causal inference. This represents a clear advantage for interactive applications, since partial results are available during the gestures (and not only after gesture completion). This allows the use of the running estimation of the scaling as a control parameter during the gesture, or to anticipate which gesture is currently performed by *early* recognition.

In summary, the GVF method we propose represents an improvement over the previous GF algorithm. Its contribution lies in the use of a general formalism based on a particle filtering inference, allowing for the online adaptation of gesture features. We proved the validity of such an approach on 2-dimensional motion data as well as on 3-dimensional gestures. In particular we demonstrated recognition accuracy, early recognition, and adaptation in an end user application.

This research is motivated by the design of expressive interaction models that allow the use of a mixed strategy between discrete commands and continuous control. It is influenced by application contexts for the expressive control of digital media (sound and visuals), with a particular emphasis on low latency, early recognition capabilities. The method proposed can be seen as part of broader research on adaptive gesture feature estimation used to describe *how* a gesture is performed for use in expressive interaction.

A. APPENDIX

A.1. Inference and algorithm for the alignment and adaptation

Here we denote by N_s the number of particles used to approximate the distribution. We denote $\mathbf{x}_k^i$ the i^{th} state sample drawn and w_k^i its respective weight at time k. The weights are normalized such as $\sum_{i=1}^{N_s} w_k^i = 1$. The set of support points and their associated weights $\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s}$ is a random measure used to characterize the posterior *pdf* $p(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})$. The continuous "true" state distribution can be approximated with a series of weighted Dirac's Delta functions:

$$p(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i)$$

The term $\mathbf{x}_0$ represents the prior distribution (i.e., the initial state), and the posterior distribution is updated at each time step. Finally, the expected value of the resulting random measure is computed as $\hat{\mathbf{x}}_k$.

An optional resampling step is used to address the *degeneracy* problem, common to particle filtering approaches, as discussed in details in [Arulampalam et al. 2002; Douc and Cappé 2005]. Resampling is introduced because after a few iterations of the inference algorithm, only a few particles have non-negligible weights (it can be shown that the variance of the importance weights can only increase over time). The resampling step corresponds to draw the particles according to the current distribution $\left\{w_k^i\right\}_i^{N_s}$. Intuitively, resampling replaces a random measure of the true distribution with an equivalent one (in the limit of $N_s \to \infty$).

In [Black and Jepson 1998b] Black et al. choose to randomly select 5 to 10% of particles to be replaced by randomly taken initial values. This process is performed during transition and may introduce discontinuities. In our approach, the degeneracy problem is handled by defining a criterion based on effective sample size N_{eff}, as specified by Arulampalam [Arulampalam et al. 2002]: $N_{eff} = 1/\sum_{i=1}^{N_s}(w_k^i)^2$ where N_{eff} is an estimate of the effective sample size, i.e. an approximation of the number of particles that are contributing significant information to the estimation of the posterior *pdf*. The N_{eff} value is used as a criterion to operate the resampling step as shown in the Algorithm 2.

ALGORITHM 1: GVF: realtime recognition and adaptation to gesture variations

 # T: number of templates

for $i = 1 \dots T$ **do**
  | ADDTEMPLATE (TemplateGesture$_i$)
end

 # L: length of incoming gesture TestingGesture

SPREADPARTICLES()

for $k = 1 \dots L$ **do**
  | PARTICLEFILTER (TestingGesture[k])
  | P = PROBABILITIES() *# P contains gesture probabilities*
  | S = STATUS() *# S contains variation estimations (phase, speed, size, angle, etc.)*
end

ALGORITHM 2: PARTICLEFILTER($\mathbf{z}_k$): Realtime temporal alignment (step at time k with observation $\mathbf{z}_k$).

for $i = 1 \dots N_s$ **do**
  | $\mathbf{x}_k^i \sim \mathcal{N}(\mathbf{x}_k | A\mathbf{x}_{k-1}, \Sigma)$
  | $p_k^i := \mathbf{x}_k^i(1)$
  | $p(\mathbf{z}_k | \mathbf{x}_k^i) = St(\mathbf{z}_k | f(\mathbf{x}_k^i, \mathbf{g}(p_k^i)), \Sigma, \nu)$
  | $\hat{w}_k^i \leftarrow w_{k-1}^i p(\mathbf{z}_k | \mathbf{x}_k^i)$
end

$$w_k^i \leftarrow \frac{\hat{w}_k^i}{\sum_j \hat{w}_k^j}, \qquad \forall i = 1 \dots N_s$$

$$N_{eff} \leftarrow (\sum_{i=1}^{N_s} (w_k^i)^2)^{-1}$$

if $N_{eff} <$ *resampling threshold* **then**
  | resample $\mathbf{x}_k^1 \dots \mathbf{x}_k^{N_s}$ according to ddf $w_k^1 \dots w_k^{N_s}$ $w_k^i \leftarrow N_s^{-1} \qquad \forall i = 1 \dots N_s$
end

return $\hat{\mathbf{x}}_k = \sum_{i=1}^{N_s} w_k^i \mathbf{x}_k^i$

ALGORITHM 3: SPREADPARTICLES(): Initial Conditions by spreading particles over the varying gesture characteristics

 # D is the state space dimension
for $i = 1 \dots N_s$ **do**
  | **for** $l = 1 \dots D$ **do**
  | | $\mathbf{x}_0^i(l)$ = UNIRANDOM($range_l$) *# Particle sets to a random value uniformly drawn from a given range*
  | **end**
  | $w^i = 1/N_s$ *# Uniform distribution over the particle weights*
end

A.2. Rotation matrix convention

Let us consider the Cartesian frame (x, y, z), the three Euler angles ϕ, θ, ψ rotating vectors about respectively x, y and z induce the three following rotation matrices:

$$R_\phi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{pmatrix}$$

$$R_\theta = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix}$$

$$R_\psi = \begin{pmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The rotation matrix in 3-dimension considered in the paper the clockwise rotation defined as: $R = R_\phi R_\theta R_\psi$

A.2.1. Model configuration for 2-dimensional gesture data. The GVF method allows for taking into account these invariants by defining them as state variables, s_k and r_k respectively. The gesture features estimated are the following: phase p_k, velocity v_k, scaling coefficient s_k, rotation angle r_k, and the gesture index $m_k \in [1 \dots 16]$:

$$\mathbf{x}_k = (p_k, v_k, s_k, r_k, m_k)^T \in [0, 1] \times \mathbb{R}^3 \times \mathbb{N}$$

The invariance by rotation and scaling leads to the following non linear function of state variables:

$$f(\mathbf{x}_k, \mathbf{g}(p_k)) = \mathbf{diag}(s_k) \begin{pmatrix} \cos(r_k) & -\sin(r_k) \\ \sin(r_k) & \cos(r_k) \end{pmatrix} \mathbf{g}(p_k)$$

The state transition matrix A_l, for the template gesture index $l \in [1, M]$ is given by:

$$A_l = \begin{pmatrix} 1 & 1/T_l & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & & & \\ 0 & 0 & & I_3 & \\ \vdots & \vdots & & & \end{pmatrix}$$

where T_l is the length of the l-th gesture template.

A.3. Model configuration for 3-dimensional gesture data

The model is configured in order to be able to track variations in speed, scale and orientation for 3-dimensional gesture inputs. We denote these variations as follows: phase p_k, velocity v_k, scaling coefficients $\mathbf{s}_k$, rotation angles $\mathbf{r}_k$, and the gesture index $m_k \in [1 \dots 3]$. Note that we have to consider scaling along the three dimensions and rotation in a 3-dimensional space: $\mathbf{s}_k = (s_k^x, s_k^y, s_k^z)$, $\mathbf{r}_k = (\phi_k, \theta_k, \psi_k)$.

$$\mathbf{x}_k = (p_k, v_k, s_k^x, s_k^y, s_k^z, \phi_k, \theta_k, \psi_k, m_k)^T \in [0, 1] \times \mathbb{R}^7 \times \mathbb{N}$$

The invariance by rotation and scaling leads to the following non linear function of state variables:

$$f(\mathbf{x}_k, \mathbf{g}(p_k)) = \begin{pmatrix} s_k^x & 0 & 0 \\ 0 & s_k^y & 0 \\ 0 & 0 & s_k^z \end{pmatrix} R(\phi_k, \theta_k, \psi_k) \mathbf{g}(p_k)$$

where $R(\phi_k, \theta_k, \psi_k)$ is the rotation matrix in three dimensions given by the Euler angles. As previously, we refer the reader to the Appendix A.2 for the rotation conventions. The state transition matrix A_l depends on the gesture template l and is written:

$$A_l = \begin{pmatrix} 1 & 1/T_l & 0 & 0 & \dots \\ 0 & 1 & 0 & 0 & \dots \\ 0 & 0 & & & \\ 0 & 0 & & I_7 & \\ \vdots & \vdots & & & \end{pmatrix}$$

where I_7 is the identity matrix of size 7×7 and T_l is the length of the l-th template gesture.

A.4. Complementary Study: stimuli

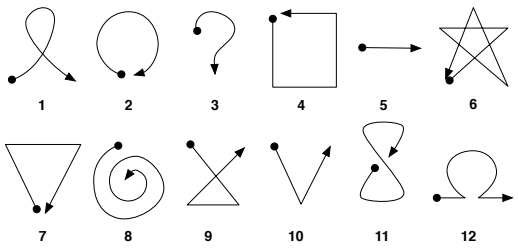

Fig. 19. Gesture vocabulary used in the first experiment

Table IV. Set of 11 variations combinations used in Step 2 of Study 1.

Variations	
Id.	Description
V1	slower
V2	faster
V3	change size
V4	change orientation
V5	slower and change size
V6	faster and change size
V7	slower and change orientation
V8	faster and change orientation
V9	change size and orientation
V10	slower and change size and orientation
V11	faster and change size and orientation

REFERENCES

M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. 2002. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on signal processing* 50, 2 (2002), 174–188.

Stephen Barrass and Gregory Kramer. 1999. Using sonification. *Multimedia systems* 7, 1 (1999), 23–31.

O. Bau and W.E. Mackay. 2008. OctoPocus: a dynamic guide for learning gesture-based command sets. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*. ACM, 37–46.

F. Bevilacqua, F. Baschet, and S. Lemouton. 2012. The augmented string quartet: experiments and gesture following. *Journal of New Music Research (Accepted)* (2012).

F. Bevilacqua, F. Guédy, N. Schnell, E. Fléty, and N. Leroy. 2007. Wireless sensor interface and gesture-follower for music pedagogy. In *Proceedings of the 7th international conference on New interfaces for musical expression*. ACM, 124–129.

F. Bevilacqua, N. Schnell, and S. Fdili Alaoui. 2011a. Gesture Capture: Paradigms in Interactive Music/Dance Systems. *Emerging Bodies: The Performance of Worldmaking in Dance and Choreography* (2011), 183.

F. Bevilacqua, N. Schnell, N. Rasamimanana, B. Zamborlin, and F. Guédy. 2011b. Online Gesture Analysis and Control of Audio Processing. *Musical Robots and Interactive Multimodal Systems* (2011), 127–142.

F. Bevilacqua, B. Zamborlin, A. Sypniewski, N. Schnell, F. Guédy, and N. Rasamimanana. 2010. Continuous realtime gesture following and recognition. In *In Embodied Communication and Human-Computer Interaction, volume 5934 of Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 73—84.

J. Bilmes. 2002. *What HMMs Can Do*. Technical Report. University of Washington, Department of EE, Seattle WA, 98195-2500.

M. Black and A. Jepson. 1998a. A probabilistic framework for matching temporal trajectories: Condensation-based recognition of gestures and expressions. *Computer Vision?ECCV'98* (1998), 909–924.

M.J. Black and A.D. Jepson. 1998b. Recognizing temporal trajectories using the condensation algorithm. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*. IEEE, 16–21.

A.F. Bobick and A.D. Wilson. 1997. A State-Based Approach to the Representation and Recognition of Gesture. *IEEE Trans. Pattern Anal. Mach. Intell.* 19, 12 (1997), 1325–1337.

Eric O Boyer, Bénédicte M Babayan, Frédéric Bevilacqua, Markus Noisternig, Olivier Warusfel, Agnes Roby-Brami, Sylvain Hanneton, and Isabelle Viaud-Delmon. 2013. From ear to hand: the role of the auditory-motor loop in pointing to an auditory source. *Frontiers in computational neuroscience* 7 (2013).

M. Brand and A. Hertzmann. 2000. Style machines. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 183–192.

L. Bretzner, I. Laptev, and T. Lindeberg. 2002. Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering. In *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*. IEEE, 423–428.

B. Caramiaux. 2012. *Studies on the Relationship between Gesture and Sound in Musical Performance*. Ph.D. Dissertation. University Paris VI, IRCAM Centre Pompidou.

B. Caramiaux, F. Bevilacqua, and N. Schnell. 2010. Analysing Gesture and Sound Similarities with a HMM-based Divergence Measure. In *Proceedings of the 6th Sound and Music Conference*. Barcelona, Spain.

B. Caramiaux, F. Bevilacqua, and A. Tanaka. 2013. Beyond recognition: using gesture variation for continuous interaction. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*. ACM, 2109–2118.

G. Caridakis, K. Karpouzis, N. Drosopoulos, and S. Kollias. 2009. Adaptive gesture recognition in Human Computer Interaction. In *Image Analysis for Multimedia Interactive Services, 2009. WIAMIS'09. 10th Workshop on*. IEEE, 270–274.

R. Chavarriaga, H. Bayati, and J.D. Millán. 2013. Unsupervised adaptation for acceleration-based activity recognition: robustness to sensor displacement and rotation. *Personal and Ubiquitous Computing* 17, 3 (2013), 479–490.

R. Douc and O. Cappé. 2005. Comparison of resampling schemes for particle filtering. In *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*. IEEE, 64–69.

A. Doucet, N. De Freitas, and N. Gordon. 2001. *Sequential Monte Carlo methods in practice*. Springer Verlag.

Paul Dourish. 2004. *Where the action is: the foundations of embodied interaction*. MIT press.

K. Forbes and E. Fiume. 2005. An efficient search algorithm for motion data using weighted PCA. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*. ACM, 67–76.

D.M. Gavrila and L.S. Davis. 1995. Towards 3-d model-based tracking and recognition of human movement: a multi-view approach. In *International workshop on automatic face-and gesture-recognition*. Citeseer, 272–277.

A. Heloir, N. Courty, S. Gibet, and F. Multon. 2006. Temporal alignment of communicative gesture sequences. *Computer animation and virtual worlds* 17, 3-4 (2006), 347–357.

Oliver Höner and Thomas Hermann. 2005. Listen to the ball!'-sonification-based sport games for people with visual impairment. In *APA: a discipline, a profession, an attitude (Proceedings of the 15th International Symposium Adapted Physical Activity)*.

M. Isard and A. Blake. 1998. Condensation conditional density propagation for visual tracking. *International journal of computer vision* 29, 1 (1998), 5–28.

S. Jordà. 2008. On stage: the reactable and other musical tangibles go real. *International Journal of Arts and Technology* 1, 3 (2008), 268–287.

A. Licsár and T. Szirányi. 2005. User-adaptive hand gesture recognition system with interactive training. *Image and Vision Computing* 23, 12 (2005), 1102–1114.

J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan. 2009. uWave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing* 5, 6 (2009), 657–675.

D. Merrill and J.A. Paradiso. 2005. Personalization, expressivity, and learnability of an implicit mapping strategy for physical interfaces. In *Proceedings of the CHI Conference on Human Factors in Computing Systems, Extended Abstracts*. 2152–2161.

S. Mitra and T. Acharya. 2007. Gesture recognition: A survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 37, 3 (2007), 311–324.

L.R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* (1989), 257–286.

N. Rasamimanana and F. Bevilacqua. 2012. Urban Musical Game. In *Proceedings of the 2012 annual conference on Human factors in computing systems (CHI2012)*.

N. Rasamimanana, F. Bevilacqua, N. Schnell, F. Guedy, E. Flety, C. Maestracci, B. Zamborlin, J.L. Frechin, and U. Petrevski. 2011. Modular musical objects towards embodied control of digital music. In *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*. ACM, 9–12.

Davide Rocchesso, Pietro Polotti, and Stefano Delle Monache. 2009. Designing Continuous Sonic Interaction. *International Journal of Design* 3, 3 (2009).

D. Rubine. 1991. Specifying gestures by example. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*. ACM, 329–337.

Caifeng Shan, Tieniu Tan, and Yucheng Wei. 2007. Real-time hand tracking using a mean shift embedded particle filter. *Pattern Recognition* 40, 7 (2007), 1958–1970.

B. Verplank, C. Sapp, and M. Mathews. 2001. A Course on Controllers. In *NIME Workshop at ACM Conference on Computer-Human Interaction (CHI'2001)*.

Y. Visell and J. Cooperstock. 2007. Enabling gestural interaction by means of tracking dynamical systems models and assistive feedback. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*. IEEE, 3373–3378.

P. Viviani and T. Flash. 1995. Minimum-jerk, two-thirds power law, and isochrony: converging approaches to movement planning. *Journal of Experimental Psychology: Human Perception and Performance* 21, 1 (1995), 32.

Zhao Wei, Tao Tao, Ding ZhuoShu, and Enrico Zio. 2013. A dynamic particle filter-support vector regression method for reliability prediction. *Reliability Engineering & System Safety* 119 (2013), 109–116.

D. Weinland, R. Ronfard, and E. Boyer. 2011. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding* 115, 2 (2011), 224–241.

A.D. Wilson. 2000. *Adaptive Models for Gesture Recognition*. Ph.D. Dissertation. Massachusetts Institute of Technology.

A.D. Wilson and A.F. Bobick. 1999. Parametric hidden markov models for gesture recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 21, 9 (1999), 884–900.

A.D. Wilson and A.F. Bobick. 2000. Realtime online adaptive gesture recognition. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, Vol. 1. IEEE, 270–275.

J.O. Wobbrock, A.D. Wilson, and Y. Li. 2007. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*. ACM, 159–168.

Y. Yacoob and M.J. Black. 1998. Parameterized modeling and recognition of activities. In *Computer Vision, 1998. Sixth International Conference on*. IEEE, 120–127.

B. Zamborlin, F. Bevilacqua, M. Gillies, and M. D'inverno. 2014. Fluid gesture interaction design: Applications of continuous recognition for the design of modern gestural interfaces. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 3, 4 (2014), 22.

S.K. Zhou, R. Chellappa, and B. Moghaddam. 2004. Visual tracking and recognition using appearance-adaptive models in particle filters. *Image Processing, IEEE Transactions on* 13, 11 (2004), 1491–1506.

Received February 20XX; revised March 20XX; accepted June 20XX