Inertial Gesture Recognition with BLSTM-RNN

Grégoire Lefebvre, Samuel Berlemont, Franck Mamalet, Christophe Garcia

Abstract This chapter presents a new robust method for inertial MEM (MicroElectroMechanical systems) based 3D gesture recognition. The linear acceleration and the angular velocity, respectively provided by the accelerometer and the gyrometer, are sampled in time resulting in 6D values at each timestep, which are used as inputs for our gesture recognition system. We propose to build a system based on Bidirectional Long Short-Term Memory Recurrent Neural Networks (BLSTM-RNN) for gesture classification from raw MEM data. We compare this system to a statistical method based on HMM (Hidden Markov Model), to a geometric approach using DTW (Dynamic Time Warping), and to a specific classifier FDSVM (Frame-based Descriptor and multi-class Support Vector Machine) using filtered and denoised MEM data. Experimental results, on a dataset composed of 22 individuals producing 14 gestures, show that the proposed approach outperforms classical methods with an average classification rate of 95.57% and a standard deviation of 0.50 for 616 test gestures. Furthermore, these experiments underline that combining accelerometer and gyrometer data gives better results than using a single inertial description.

Grégoire Lefebvre
Orange Labs, R&D, Meylan, France, e-mail: gregoire.lefebvre@orange.com

Samuel Berlemont
Orange Labs, R&D, Meylan, France, e-mail: samuel.berlemont@orange.com

Franck Mamalet
Orange Labs, R&D, Rennes, France, e-mail: franck.mamalet@orange.com

Christophe Garcia
LIRIS, UMR 5205 CNRS, INSA-Lyon, F-69621, France, e-mail: christophe.garcia@liris.cnrs.fr

1 Introduction

Accelerometers and gyrometers are nowadays present in our Smartphones. These sensors capture hand movements when users grasp their devices. We can consider two main issues: posture recognition and symbolic gesture recognition. In the first case, the user maintains a posture during a certain period of time, keeping for instance the device upside down. In the second situation, the user may produce a gesture to execute a system command, like drawing a *heart* symbol in 3D space in order to call its favorite phone number. Dynamic gesture recognition based on inertial sensors is a very challenging task. Algorithms are confronted to numerous factors causing errors in the recognition process: dynamical differences (intense versus phlegmatic gestures), temporal variations (slow versus fast movements), physical constraints (device weight, human body elasticity, left or right-handed users, seated or standing up users, users on the move, etc.), classification constraints (mono versus multi users, open or closed world paradigm, etc.). Moreover, inertial MEM introduce some noise in the data. For example, the accelerometer, as a 3-axis capacitive mass spring system, interpret gravity as an acceleration in the opposite direction. This constant bias is difficult to extract due to the reference frame of the sensor, which is fixed in relation to the device (see [23] for more details).

Classically, several steps operate from signal data processing to gesture classification with some intermediate steps like data clustering and gesture model learning. The processing steps aim at building a more compact representation of the input signals that characterize the corresponding gestures. Different methods can then be applied: calibration, filtering, normalization or thresholding. Data clustering is often applied to reduce the input space dimension and find class referent gesture vectors. A learning phase of a gesture model follows this clustering step and finally a decision rule or a specific classifier is built in order to label the input data as a recognized gesture or an unknown gesture.

In this chapter, we propose to learn an efficient gesture classifier without any preprocessing method (*i.e.* from raw MEM data) using a BLSTM-RNN model. This chapter is organized as follows. Section 2 presents a survey of sensor-based gesture recognition. In section 3, we describe in detail the proposed gesture recognition method. We present in depth experimental results in section 4. Finally, conclusions and perspectives are drawn in section 5.

2 Accelerometer based 3D Gesture Recognition

3D gesture recognition using accelerometers has been studied in recent years, and for gesture classification three main strategies stand out which are based on statistics, on geometry or on boosting classifiers.

The first strategy has been deeply studied in the last decade, the methods being based on Hidden Markov Models (HMM) [12, 13, 14, 21] and Bayesian networks

[4]. Hofmann *et al.* [12] propose to use discrete HMM (dHMM) for recognizing dynamic gestures thanks to their velocity profile. This approach consists of two levels and stages of recognition: a low-level stage essentially dividing the input data space into different regions and assigning each of them to a representative codebook, and a high-level stage taking the sequences of vector indexes from the first stage and classifying them with discrete HMM. The experiments are performed using a training set of 500 gestures with 10 samples per gesture realized by two persons. Each sample represents hand orientations, acceleration data and finger joint angles. A vector codebook is obtained by an input space clustering method (*i.e.* the K-means algorithm). The clustering essentially serves as an unsupervised learning procedure to model the shape of the feature vector distribution in the input data space. Here, the observation alphabet size equals 120. The comparison between ergodic HMM and left-to-right HMM shows similar results with 95.6% of correct recognition rate for 100 gestures from two users describing German Sign Language.

Hand gesture recognition based on HMM is also proposed by Mantyla *et al.* [18]. Their approach differs by a uniform vector quantization strategy based on 512 3D codewords. Here, minimum and maximum values of each acceleration component were evaluated using the training data in order to define the bounding box of the activity in the feature space. The experimental results show an average correct recognition rate of 96% on a small dataset of six gesture categories. The input acceleration data is low-pass Butterworth filtered after $100Hz$ sampling.

Similar results are presented in [13, 14]. Kallio *et al.* [13] use five HMM states and a codebook size of eight for 16 gestures. The authors highlight that the performances decrease when using four sequences for training the system instead of 20 sequences. The recognition rate falls from 95% to 75% even for this mono-user case study. In [14], a 37 multi-user case is studied for 8 gestures, evaluating the effect of vector quantization and sampling. A rate of 96.1% of correct classification is obtained with five HMM states and a codebook size of height. However, this study can be considered as biased since the K-means clustering is performed from all the available data set and not only the training database.

In opposition to the previous studies, and taking into consideration that gesture data are correlated in time, Pylvänäinen proposes in [21] to build a system based on continuous HMM (cHMM). Again, the results are convincing, with 96.76% on a dataset providing 20 samples for 10 gestures performed by 7 persons. An evaluation of the quantization and sampling effects is demonstrated, and the best performances are obtained by using $7-12$ bits per sample and a frequency from 20 to $30Hz$.

In [4], Cho *et al.* build Bayesian networks models based on the extraction, within the acceleration signals, of primitives which correspond to portions of the signals whose values increase or decrease monotonously. These primitives are recursively decomposed in mid-points models, represented by the Gaussian distributions of the samples whose time indexes are the mean of the extracted primitive end-points. Gestures are classified based on a maximum likelihood matching to the models created. Cho *et al.* show a mean correct recognition rate of 96.3% with a 4-fold test, with 11 gestures carried out 3 times by 100 users.

The second strategy for recognizing 3D gestures is based on distances between geometric models. The goal is to provide a gallery of some gesture references to model each gesture into class with time warping strategies and design a decision rule for a test gesture regarding the respective distance to these referent instances. On the contrary to the HMM strategy, no learning phase is needed but the computational time is high as a test gesture has to be compared to all referent instances. Consequently, the main drawback of this approach is the necessity to find the most relevant samples to represent a gesture class while keeping the number of these referents low in order to minimize the final evaluation processing time. Wilson *et al.* in [22] compare Linear Time Warping (LTW) and Dynamic Time Warping (DTW) to the HMM based strategy. Their experiments with 7 types of gesture from 6 users shows an advantage for HMM with 90% in opposition to the score of LTW and DTW of respectively 40% and 71% of correct recognition rate.

Liu *et al.* experiment with more success the DTW strategy in [17]. Gesture recognition and user identification are performed with good recognition rates of respectively 93.5% and 88%. The gesture data, performed over multiple days, consists of 30 samples of 8 gestures for 8 individuals and the user recognition results are obtained from 25 participants. The authors introduce an averaging window of 50 ms for reducing noise and erratic moves. It appears that preprocessing methods are here crucial to increase the classification results. Likewise, in [1], Akl *et al.* use DTW and affinity propagation for dimension reduction for recognizing 3D gestures. 7 subjects participated producing 3700 gesture traces for a good classification rate of 90%.

The third strategy for recognizing 3D gestures is to learn a specific classifier. Hoffman *et al.* in [11] improve 3D gesture recognition with a linear classifier and the Adaboost method, inspired by the method proposed in [15] for 2D symbol writer recognition. The experiments show an accuracy of 98% for 13 gestures made by 17 participants. Other studies focus on Support Vector Machine (SVM) like in [24]. This study uses Frame-based Descriptor and multi-class SVM (FDSVM). Each gesture is divided into segments, from which are extracted statistical descriptors such as: mean, energy, entropy, standard deviation and correlation. These descriptors form the feature vectors to be classified by a multi-class SVM. The obtained results achieve 95.21% of good recognition for 12 gestures made by 10 individuals.

In [4], Cho *et al.* also use a confusion pair discrimination strategy, enhancing their 96.3% average recognition rate to 96.9% with a bi-class SVM (*i.e.* letter *O* versus number 6), in order to better separate similar gestures, which generate most of the recognition errors.

Consequently, many strategies are explored with different paradigms and specific data processing methods. However, theses approaches depend heavily on the choice of descriptors which may not be optimal in general. Moreover, it is challenging to compare these methods on different databases. To cope with these issues, we develop hereafter our 3D gesture recognition method based on BLSTM-RNN that learns to automatically extract relevant features from raw input data and compare it with the main state-of-the-art methods on a common database.

3 The Proposed Gesture Recognition Method

3.1 Bidirectional Long Short-Term Memory Recurrent Neural Networks

Classical Recurrent Neural Networks (RNNs) are a common learning technique for temporal sequence analysis. It can be seen as a particular neural network, which can remember previous inputs and use them to influence the network output, using recurrent connections in the hidden layers to store some information about the context. Nevertheless, even if they are able to learn tasks which involve short time lags between inputs and corresponding teacher signals, Hochreiter and Schmidhuber in [10] have shown that this short-term memory becomes insufficient when dealing with "real world" sequence processing, such as inertial gesture sequences.

In order to alleviate this problem, they introduced the Long Short Term Memory RNNs (LSTM-RNN), which neurons contain a constant "memory cell" – namely constant error carousel (CEC) –. This allows for constant error signal propagation through time, and thus, provides remedies for the RNN's problem of *exponential error decay* [10]. Figure 1 presents in detail a LSTM neuron.

Improved versions of the LSTM-RNN were proposed by Graves *et al.* [9], adding some multiplicative gates to control the access to the CEC. These gates are neurons that can set (input gate), reset (forget gate) or hide (output gate) the internal value of the CEC according to neuron input values and context. Additional direct connections – namely peephole – have also been introduced to achieve finer gate control using the CEC value as input.

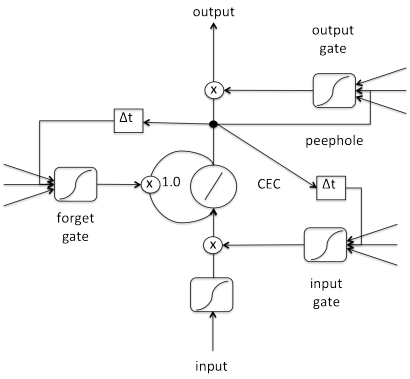


Fig. 1 A LSTM neuron: the Constant Error Carousel (CEC) is controlled by three multiplicative gates (input, forget and output). Peepholes enable to transmit directly the CEC value to the gate, at a given timestep.

Furthermore, since in sequence classification, at a given timestep, past and future context may have similar importance, Graves and Schmidhuber [9] also introduced

a so-called bidirectional LSTM-RNN model (BLSTM-RNN), that consists in two separate hidden layers, the forward and backward layers able to deal respectively with past and future context. As shown in Figure 2, the output layer is connected to both hidden layers in order to fuse past and future contexts.

Let's give the following notations:

- $G = \{G_0, ..., G_{T-1}\}$ is a gesture with T denoting the size of the sequence;
- $G_t = (x_0(t), ..., x_{N-1}(t))$ is a vector at timestep t and N being the sensor number;
- $(o_0(t), ..., o_{n-1}(t))$ is the BLSTM-RNN output set at timestep t with n is the number of gestures to be classified.

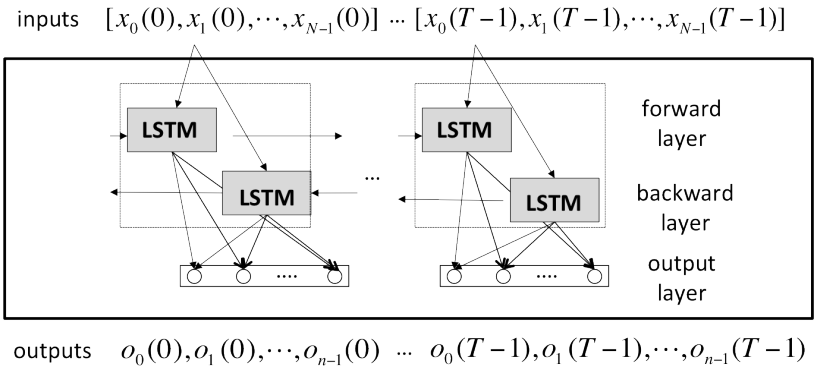


Fig. 2 A BLSTM-RNN architecture: the forward (resp. backward) layer processes the input sequence in the (resp. reverse) order. Output layer concatenates hidden layers values at each timestep to make a decision by considering both past and future contexts.

(B)LSTM-RNNs have proven their great ability to deal with temporal data in many applications such as: phoneme classification [9], action classification [2], facial expression recognition [3], rhythm or timed event recognition [6], robot trajectory learning [19], handwriting recognition [8, 7], or text recognition [5].

In this chapter, we consider gesture data using 6D input vectors through sampling timestep. These data are correlated during the user gestural production, and time lags between the beginning and the end of gesture can be long. Furthermore, past and future context are both essential for gesture recognition. For these reasons, BLSTM-RNN is chosen to classify the input MEM data sequence.

3.2 BLSTM-RNN Architecture, Training and Decision Rule

The proposed gesture classification scheme based on BLSTM-RNN is described in Figure 3. First, the input layer consists in the concatenation of 3-axis accelerometer and 3-axis gyrometer information synchronized in time (*i.e.* $N = 6$). Notice that our system relies only on the raw MEMs data, without any preprocessing in opposition to most of state-of-the-art methods. These data are only rescaled between -1 and $+1$ according to the maximum value that sensors can provide.

The forward and backward LSTM hidden layers are fully connected to the input layer and consist in multiple LSTM neurons each with full recurrent connections. Several experiments have been conducted with different hidden layer sizes and 100 neurons lead to the best results (cf. section 4.2).

The output layer has a size equals the number of gesture to classify (*i.e.* $n = 14$ in our experiments) . The SoftMax activation function is used for this layer to give network responses $o_i(t)$ between 0 and 1, and a sum $\sum_{i\in[0,n[} o_i(t)$ equals one, at every timestep t. Classically, these outputs can be considered as posterior probabilities of the input sequence to belong to a specific gesture class.

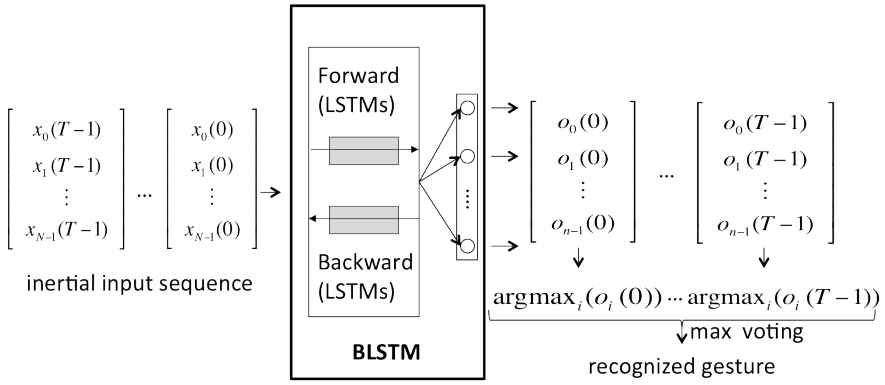


Fig. 3 BLSTM-RNN gesture classification architecture: input layer is a 6D inertial sequence. The largest BLSTM-RNN output is computed at each timestep and a Max voting scheme enables to determine the recognized gesture class.

This network is learned using classical on-line backpropagation through time with momentum (*i.e.* learning rate $5e{-}4$, momentum 0.2, experimented in section 4.2). As described in [9], on a training set, we target the same corresponding gesture class at each timestep for each input example. For classifying a new gesture sequence, we use a majority voting rule over the outputs along the sequence (*i.e.* keeping only the most *probable* class $argmax_{i\in[0,n[} o_i(t)$ at each timestep) to determine the final gesture class.

4 Experimental Results

4.1 Inertial Gesture Dataset

To the best of our knowledge, the community does not provide yet any public dataset for benchmarking inertial symbolic gesture recognition methods. Therefore, we collected our own 3D gesture dataset to compare our classification method with classical approaches in this research field. Our dataset has been captured on an Android Nexus S Samsung device. The sampling time is 40*ms* during the accelerometer and gyrometer capture (*i.e.* a frequency of 25*Hz*). 22 novice participants, from 20 to 55 years old, all right-handed, performed five times each of the 14 symbolic gestures that we designed. This corresponds to 1540 gestures. As shown in Figure 4, the 14 symbolic gestures are divided into two families: linear gestures (*e.g. north*, *east*, *west* and *south flicks*, and *down*, *up*, *throw* and *pick* gestures) and curvilinear gestures (*e.g. clockwise*, *counter-clockwise*, *letter Z*, *letter N*, *alpha* and *heart*). These choices make the dataset particularly difficult. For instance, there classically are confusions between *flick* gestures and *letter N* and *Z*. Likewise, the *clockwise* movement is often confused with *alpha* or *heart* symbols.

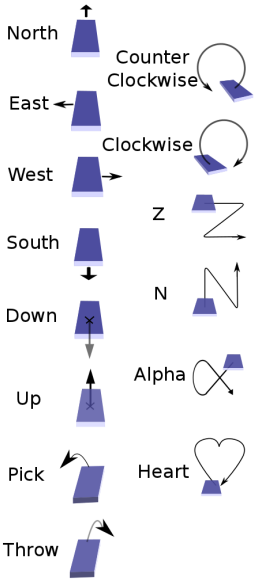


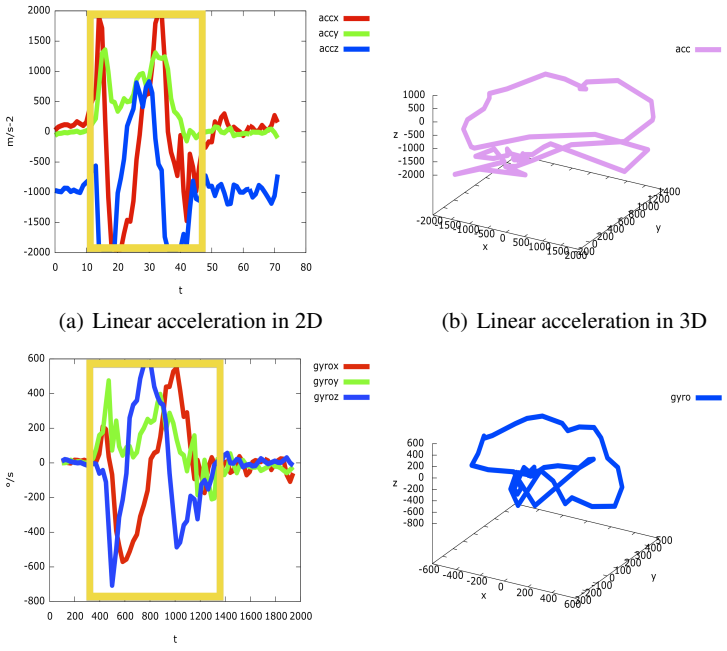
Fig. 4 Gesture dataset : the 14 symbolic gestures to be classified.

In order to locate and use only the informative part of the input sequence for performing gesture classification, an automatic temporal segmentation is performed. As described in [16], this temporal segmentation is based on a signal energy indicator

obtained by the difference of two instant power estimators. A gesture indicator $I(t)$ can be defined from two estimators $E_0(t)$ and $E_1(t)$ of the instant signal power $P(t)$ (see Equation 1). Some heuristics on this gesture indicator identify the beginning and the end of a symbolic gesture before processing the recognition phase.

$$\begin{cases} P(t) &= \frac{1}{2}\sqrt{\sum_{i=0}^{N-1} x_i(t)} \\ E_0(t) &= \delta_0 E_0(t-1) + (1-\delta_0)P(t), \delta_0 \in [0;1] \\ E_1(t) &= \delta_1 E_1(t-1) + (1-\delta_1)P(t), \delta_1 \in [0;1], \delta_0 > \delta_1 \\ I(t) &= |E_0(t) - E_1(t))| \end{cases} \tag{1}$$

For instance, Figure 5 shows raw input signals to be classified as a clockwise gesture and the yellow parts correspond to segmented gestures. Six degrees of freedom are then provided by the accelerometer and the gyrometer with respectively linear accelerations and angular velocities. In Figure 6, we show the 3D gesture trajectory reconstruction using Simpson's 3/8 rule for numerical integration.



(a) Linear acceleration in 2D (b) Linear acceleration in 3D

(c) Angular velocity in 2D (d) Angular velocity in 3D

Fig. 5 Clockwise gesture sample : accelerometer and gyrometer features to be classified. The yellow parts correspond to segmented gestures.

We then use three different configurations of our dataset to compare several state-of-the-art methods with our solution. The first configuration (DB1) corresponds to the personalization paradigm, where only one user is considered with few learning

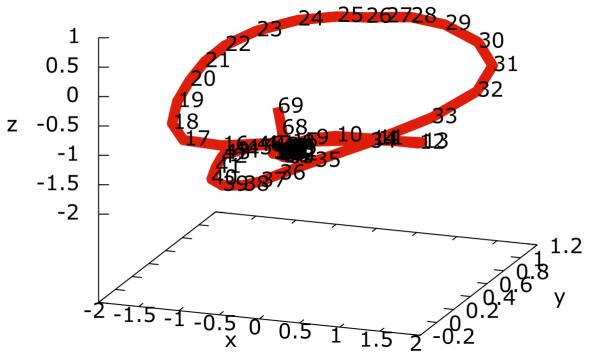


Fig. 6 Clockwise trajectory reconstruction using Simpson's 3/8 rule for numerical integration.

examples. For this configuration we have used the 70 gestures of a participant in the learning phase, and ask him to process 16 more instances of each gesture for test (*i.e.* 224 gestures). The second configuration (DB2) uses three instances of each gesture per user for the learning phase: 924 gestures (*i.e.* 60% of all data) are used for the learning phase and 616 gestures (*i.e.* 40%) for the test phase. This case corresponds to a multi-user system and a closed world paradigm. The third configuration (DB3) is composed of all samples from 17 users (*i.e.* 1190 gestures) and the test data uses the other available gestures (*i.e.* 350 gestures from five unknown users). This case is close to a real system trained with a several users and having to generalize to new users who want to use it without any training phase. Here, the third configuration represents the open world paradigm.

Our experimental results aim at assessing the method based on BLSTM-RNN and compare it to three state-of-the-art solutions: cHMM[21], DTW[20] and FDSVM [24] methods. These three state-of-the-art solutions represent three main strategies which are based on statistics, on geometry or on boosting classifier approaches. We also compare our results obtained with BLSTM-RNN and LSTM-RNN methods.

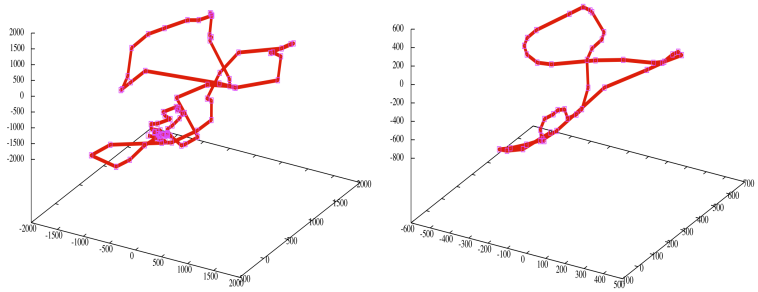
For the three state-of-the-art solutions and in all experiments, we use normalized, filtered and thresholded gesture signals. In opposition, LSTM-RNN and BLSTM-RNN based solutions use raw MEM data. As shown in Figure 7, the main objective for classical gesture recognition methods is to operate on preprocessed input signal for facilitating gesture recognition. From the raw data, some signal processings are operated to filter, normalize and threshold the gesture data.

We define (*cf.* Section 3 for notations) the normalized signal $x_i^{norm}(t)$ by Equation 2 being a weighting by the inverse of the maximal Euclidean norm for every gesture sequence element $x_i(t)$. Equation 3 gives the filtered signal $x_i^{filter}(t)$ with a low-pass filter of smoothing parameter β. Equation 4 is the condition to keep an original data in a thresholding process (*i.e.* two consecutive data differ from a measure Δ).

$$\forall i \in \{0,...,n-1\}, x_i^{norm}(t) = \frac{x_i(t)}{max_{t=0}^{T-1} \|G_t\|}, \tag{2}$$

$$\forall i \in \{0,...,n-1\}, x_i^{filter}(t) = \beta x_i^{filter}(t-1) + (1-\beta)x_i^{norm}(t), \beta \in [0,1], \quad (3)$$

$$\forall t \in \{0,...,T-2\}, \|G_t - G_{t+1}\| > \Delta, \Delta \in \Re. \quad (4)$$



(a) Raw acceleration input signal (b) Normalized, filtered and thresholded acceleration signal

Fig. 7 An *alpha* gesture with raw MEM data (a) and after preprocessing methods (b).

4.2 Preliminary Classification Results

We have conducted some experiments to explore the BLSTM architecture and find the best configuration and parameters (*i.e.* learning rate and momentum). Figure 8 gives some primarily results with different BLSTM-RNN architectures and parameters on DB3. Here, the inertial gesture recognition task is performed with the better results obtained for a configuration of 100 LSTM neurons with a learning rate of $5e{-}4$ and a momentum of 0.2.

In order to obtain the best results for the state-of-the-art methods, we have also made some primarily experimentations to find the best parameters:

- The filtering process use a low-pass filter with a smoothing parameter β of 0.8 ;
- The thresholding process approximates a minimal local error Δ of 0.1 ;
- The cHMM based method is left-to-right and uses the maximum of likelihood as a decision rule and is composed of nine states using Gaussian distribution ;
- The DTW based method uses a 5-nearest-neighbor classification ;
- The FDSVM based method applies polynomial kernel SVMs on 19-dimensional feature descriptors (mean, energy, entropy, standard deviation and correlation) for nine time segments with six signal dimensions, *i.e.* feature vectors of 1026 dimensions.

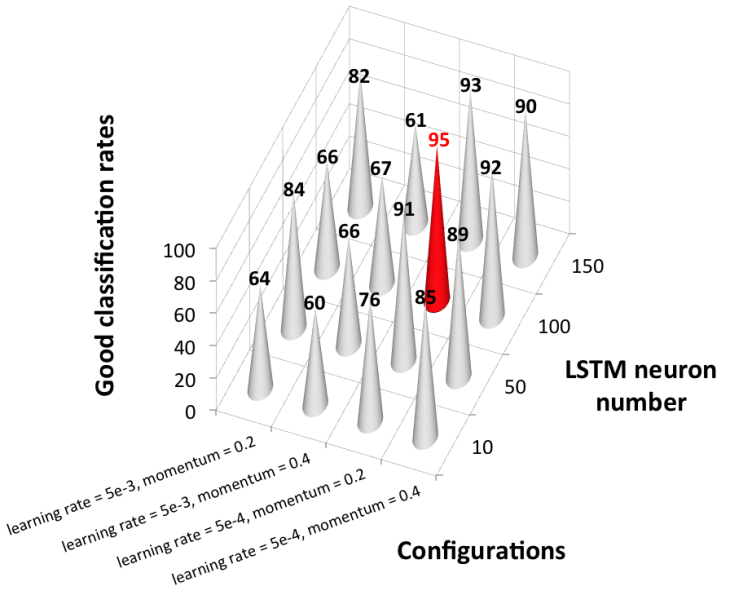


Fig. 8 Recognition rates for different BLSTM-RNN architectures on DB3 with multiple learning configurations.

4.3 Classification Results

In the following, we use a three fold cross validation to obtain mean and standard deviation, as shown in Table 1, which outlines the global performances of each classifier for configurations DB1, DB2 and DB3 using either accelerometer or gyrometer data or both.

Firstly, when comparing these methods using a single input MEM sensor (accelerometer or gyrometer), we can see that using only the gyrometer data is less efficient than using the single accelerometer data. Moreover, when these two information are combined, the global performances increase.

Secondly, considering coupled input data, this table shows that our BLSTM-RNN based classifier gives the best results on DB2 and DB3, with respectively $95.57\% \pm 0.50$ and $92.57\% \pm 2.85$.

On DB1 configuration, the DTW and cHMM based methods achieve the equivalent best performances (*i.e.* the mean recognition rates are upper than 99%, due to the strong similarity between test and training gestures provided by the same user) while our BLSTM-RNN approach is less efficient with $86.75\% \pm 0.75$. This is mainly due to the reduced number of training data which leads to the classical neural network over-fitting issue. The attempts made with smaller BLSTM-RNN did not allow any improvement on generalization.

Table 1 Classification rates on DB1, DB2 and DB3 using accelerometer (acc), gyrometer (gyr), and both sensors (acc+gyro)

Databases	DB1	DB2	DB3
Methods	Mean & Standard Deviation		
cHMM acc	99.02% ± 0.81%	83.99% ± 1.09%	80.09% ± 2.82%
cHMM gyro	95.05% ± 2.62%	70.92% ± 0.74%	70.76% ± 0.58%
cHMM acc+gyro	**99.86% ± 0.20%**	85.79% ± 0.67%	82.76% ± 1.41%
DTW acc	99.40% ± 0.21%	92.59% ± 0.20%	90.29% ± 2.07%
DTW gyro	95.39% ± 0.56%	80.63% ± 2.39%	79.81% ± 1.72%
DTW acc+gyro	99.70% ± 0.42%	94.04% ± 0.15%	91.71% ± 1.46%
FDSVM acc	94.94% ± 2.42%	93.07% ± 0.88%	91.56% ± 0.28%
FDSVM gyro	92.26% ± 1.38%	85.66% ± 0.47%	84.52% ± 0.48%
FDSVM acc+gyro	96.38% ± 1.89%	95.39% ± 0.57%	92.38% ± 1.27%
LSTM acc	86.46% ± 5.83%	94.29% ± 1.07%	89.14% ± 2.03%
LSTM gyro	77.53% ± 5.26%	81.54% ± 1.79%	72.29% ± 1.30%
LSTM acc+gyro	88.10% ± 3.72%	95.18% ± 0.88%	92.47% ± 1.34%
BLSTM-RNN acc	84.15% ± 0.67%	94.86% ± 1.23%	89.42% ± 2.45%
BLSTM-RNN gyro	68.90% ± 4.85%	83.39% ± 0.65%	74.19% ± 1.55%
BLSTM-RNN acc+gyro	86.75% ± 0.75%	**95.57% ± 0.50%**	**92.57% ± 2.85%**

In comparison, the FDSVM solution provides good and stable performances with an average recognition rate greater than 92% on the three datasets, but does not achieve the best results.

The main conclusions of a deep analysis of confusion matrices on DB3 (*i.e.* open world paradigm) are the following. The main drawback for the cHMM based method in this context is the incorrect classification of some linear gestures as *down*, *throw* and *up* gestures. 48% of the *down* and *throw* gestures are confused with *N* gestures and 52% of the *up* gestures are indeed confused with *pick* gestures. These gestures share indeed some features that proved to be difficult to distinguish for this method.

On the contrary, the DTW based method provides a good solution to classify linear gestures except for *throw* gestures which are often recognized as *east*, *north flick* and *pick* gestures, which can be explained by the similar nature of production of these three gestures.

The FDSVM method achieves good performances in general. Nevertheless, some misclassifications appear for instance between opposite gestures as *pick* and *throw* or *down* and *up* gestures. Opposite gestures may be misclassified for instance when some users anticipate a *throw* gesture by slightly moving back the device in the beginning of the production as in a *pick* gesture.

Our BLSTM-RNN approach (see Table 2) have some issues distinguishing respectively the *letter N* and the *up* gesture from the *up* and the *pick* gesture. This may

be due to the uniform learning target chosen (*i.e.* same class at each timestep), or the majority voting scheme in the recognition phase.

Table 2 Best Confusion Matrix on DB3 for the BLSTM based method.

BLSTM	alpha	ccw	cw	flickE	flickN	flickS	flickW	heart	N	pick	down	throw	up	Z	recognition rate
alpha	**25**														1
ccw		**25**													1
cw			**24**											1	0.96
flickE				**24**			1								0.96
flickN					**23**	1						1			0.92
flickS						**25**									1
flickW				1			**24**								0.96
heart			1					**24**							0.96
N									**21**	1			3		0.84
pick										**25**					1
down											**25**				1
throw										1	1	**23**			0.92
up										2	1		**21**		0.88
Z														**25**	1
error rate	0	0	0.04	0.04	0	0.04	0.042	0	0	0.16	0.08	0.04	0.12	0.04	**0.96**

4.4 Computing Times

Table 3 presents the computing times for all methods for the three configurations in recognition phase executed on an Intel Core i5 CPU at 2.67GHz with 3.42Go of RAM. These experimental results show that the computing time for the HMM, FDSVM, LSTM-RNN and BLSTM-RNN based solutions is quite constant on the different datasets (*i.e.* for instance around 30 ms for BLSTM-RNN and 43ms for cHMM to classify one input gesture on DB1). The learning process is indeed built off-line and, consequently, the recognition process from a learnt model is quite fast. On the contrary, the DTW based method requires to compare the test gesture with all training reference samples. That is why the computing time increases in average from 11.93ms for 70 learning samples to 44.58ms for 1190 learning samples. The DTW based method requires then a small number of reference gestures which makes it hard to cover all user gesture variations. This trade-off may reduce the performances.

Consequently, our proposed system based on BLSTM-RNN, achieving the best performances in multi-user configuration with a recognition computing time independent of training dataset size, is a very challenging solution.

Table 3 Computing times (in *ms*) to classify one unknown gesture.

Databases	DB1	DB2	DB3
Training examples	70	924	1190
Test examples	224	616	350
cHMM accgyro	42.53±1.97	23.89±2.74	30.19±1.65
DTW accgyro	11.93±0.02	34.57±0.47	44.58±0.38
FDSVM accgyro	23.81±2.10	28.14±0.77	30.82 ±0.47
LSTM accgyro	27.98±0.90	26.52±0.77	25.42±0.84
BLSTM-RNN accgyro	30.47±0.23	31.12±0.57	29.56±0.48

4.5 Final Results

In order to assess more precisely our BLSTM-RNN based method, we investigate seven performance criteria which we quantified from to 0 to 5 in Figure 9 (*e.g.* the number 5 denotes a good performance regarding the specific criterion):

1. Classification: the overall performance regarding the classification rates over the three datasets (cf. Table 1);
2. Training time: the average time requested to learn all reference gestures (*i.e.* lower the training time, higher the grade);
3. Test time: the average time to test all unknown gestures (cf. Table 3);
4. Parametrization: the number of parameters to be specified to get a valuable gesture model (*i.e.* fewer parameters equals higher grade);
5. Fast recognition: the ability to classify an unknown gesture as soon as possible (*i.e.* before the gesture ending);
6. Personalization: the ability to add new user gestures to rebuild gesture models;
7. Universality: the ability to generalize gesture models without adding new user samples.

The BLSTM-RNN based method gives the best overall performance with the biggest covered area in the radar chart (cf. Figure 9). Therefore, with the best classification rates on two datasets and a competitive computing time for testing unknown gestures, this method offers good perspectives for universality issues, extracting gesture features from the raw signal data and dealing with strong gesture variations. The parametrization is also simplified by the choice of three criteria: the number of LSTM neurons, the learning rate and the momentum. Nevertheless, some difficulties remain when gestures should be classified as soon as possible (*i.e.* fast recognition). The temporal gesture segmentation must be indeed properly provided to maximize the classification results. The main drawbacks are when new gestures are added to the model for personalization purposes, given that the neural network must be learnt once again with a substantial computing time.

In opposition to the BLSTM based method, the three other methods perform better in term of training speed, but with less success on classification results. As previously exposed in section 4.4, the DTW based method requires also more time

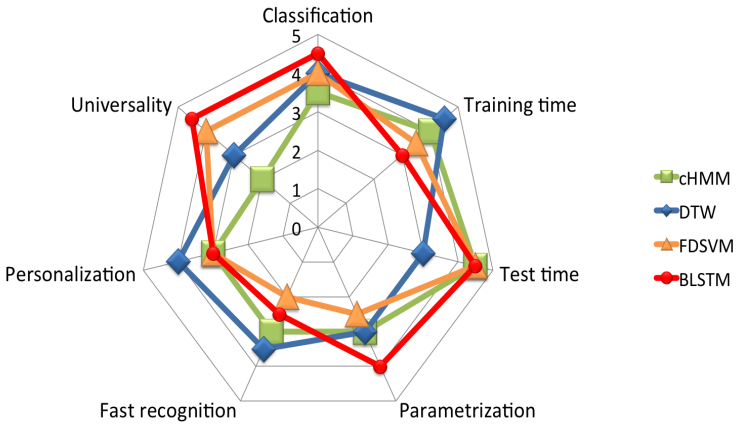


Fig. 9 Benchmark of the four main methods in regard to the seven criteria.

during test. Concerning the parametrization criterion, these three methods are more complex to tune with a need of a preprocessing step with specific parameters for filtering, normalizing and vectorizing input signals. Additional parameters are also inherent to each method. A fast recognition may be better realized with the DTW based method, computing a similarity between training samples and an unknown gesture even if the last one is incomplete. A fast recognition for the FDSVM based method is difficult because the whole input gesture must be segmented into frames in order to compute the final feature vector to be classified. For personalizing the system with new user gestures, the DTW based method seems preferable because only a selection of these instances is needed to be designed as new gesture references. Finally, Table 1 presents the universality criterion on DB3 with an open world paradigm. We show then that the best performance is obtained respectively in order by the BLSTM, FDSVM, DTW, and cHMM based method.

5 Conclusion and Perspectives

In this chapter, we have presented a contribution based on BLSTM-RNN for inertial MEM based gesture recognition. This study about symbolic gesture recognition compares our contribution to three classical pattern recognition methods: the geometric approach using DTW and the statistical method based on cHMM and a specific FDSVM classifier. We have shown that for a multi-user configuration our approach achieves the best average classification rates, up to 95.57%, in a closed world configuration, and up to 92.57%, in a open world configuration. Computing times are also very challenging for an industrial solution, being independent to the

learning sample number. The main remaining confusions with the proposed solution are when two 3D trajectories are similar or share some initial movements. A new approach, ,height=5cmusing a modified objective function, such as a Connectionist Temporal Classification [9], that permits to jointly learn to localize and classify events in input sequences, might be used to overcome this issue or to classify non segmented gestures. Another interesting extension of this work will be to study how Deep Learning models like Convolutional Neural Networks can be used in our scheme to enhance the feature extraction stage by automatically learning sparse discriminant features from the raw data.

References

1. Akl, A., Valaee, S.: Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, & compressive sensing. In: ICASSP (2010)
2. Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., Baskurt, A.: Sequential Deep Learning for Human Action Recognition. In: HBU, Lecture Notes in Computer Science, pp. 29–39 (2011)
3. Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., Baskurt, A.: Spatio-Temporal Convolutional Sparse Auto-Encoder for Sequence Classification. In: BMVC (2012)
4. Cho, S.J., Choi, E., Bang, W.C., Yang, J., Sohn, J., Kim, D.Y., Lee, Y.B., Kim, S.: Two-stage Recognition of Raw Acceleration Signals for 3-D Gesture-Understanding Cell Phones. In: G. Lorette (ed.) Tenth International Workshop on Frontiers in Handwriting Recognition. Université de Rennes 1, Suvisoft, La Baule (France) (2006). URL http://hal.inria.fr/inria-00103854. Http://www.suvisoft.com Université de Rennes 1
5. Elagouni, K., Garcia, C., Mamalet, F., Sébillot, P.: Text Recognition in Videos using a Recurrent Connectionist Approach. In: International Conference on Artificial Neural Networks (ICANN 2012), pp. 172–179 (2012)
6. Gers, F.A., Schraudolph, N.N., Schmidhuber, J.: Learning precise timing with lstm recurrent networks. J. Mach. Learn. Res. **3**, 115–143 (2003)
7. Graves, A.: Offline Arabic Handwriting Recognition with Multidimensional Neural Networks. Springer (2012)
8. Graves, A., Liwicki, M., Fernandez, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE TPAMI **31**(5), 855–868 (2009)
9. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural Networks (18), 5–6 (2005)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation (9), 1735–1780 (1997)
11. Hoffman, M., Varcholik, P., LaViola, J.: Breaking the status quo: Improving 3d gesture recognition with spatially convenient input devices. In: Virtual Reality Conference (VR), pp. 59 –66 (2010). DOI 10.1109/VR.2010.5444813
12. Hofmann, F., Heyer, P., Hommel, G.: Velocity profile based recognition of dynamic gestures with discrete hidden markov models. In: Gesture and Sign Language in Human-Computer Interaction, *Lecture Notes in Computer Science*, vol. 1371, pp. 81–95 (1998). URL http://dx.doi.org/10.1007/BFb0052991
13. Kallio, S., Kela, J., Mantyjarvi, J.: Online gesture recognition system for mobile interaction. In: Systems, Man and Cybernetics, vol. 3, pp. 2070 – 2076 vol.3 (2003)
14. Kela, J., Korpipää, P., Mäntyjärvi, J., Kallio, S., Savino, G., Jozzo, L., Marca, D.: Accelerometer-based gesture control for a design environment. Personal Ubiquitous Comput. **10**(5), 285–299 (2006). DOI 10.1007/s00779-005-0033-8. URL http://dx.doi.org/10.1007/s00779-005-0033-8

15. LaViola Jr., J.J., Zeleznik, R.C.: A practical approach for writer-dependent symbol recognition using a writer-independent symbol recognizer. IEEE Trans. PAMI **29**(11), 1917–1926 (2007). URL http://dx.doi.org/10.1109/TPAMI.2007.1109
16. Lefebvre, G., Roux, S., Petit, E.: Automatic temporal segmentation method for instrumented gesture, devices and associated terminal. Patent FR2013/51320 (2013)
17. Liu, J., Wang, Z., Zhong, L., Wickramasuriya, J., Vasudevan, V.: uwave: Accelerometer-based personalized gesture recognition and its applications. In: IEEE PerCom, pp. 1 –9 (2009)
18. Mantyla, V.M., Mantyjarvi, J., Seppanen, T., Tuulari, E.: Hand gesture recognition of a mobile device user. In: IEEE ICME, vol. 1, pp. 281–284 (2000)
19. Mayer, H., Gomez, F., Wierstra, D., Nagy, I., Knoll, A., Schmidhuber, J.: A system for robotic heart surgery that learns to tie knots using recurrent neural networks. In: Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on (2006)
20. Petit, E.: Grasp : Moteur de reconnaissance de gestes. Inter Desposit Digital Number IDDNFR.001.030023.000.S.P.2010.000.31500 (2010)
21. Pylvänäinen, T.: Accelerometer Based Gesture Recognition Using Continuous HMMs Pattern Recognition and Image Analysis. chap. 77, pp. 413–430. Berlin, Heidelberg (2005)
22. Wilson, D.H., Wilson, A.: Gesture recognition using the xwand. Tech. Rep. CMU-RI-TR-04-57, Robotics Institute (2004)
23. Woodman, O.J.: An introduction to inertial navigation. Tech. rep., University of Cambridge (2007). URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.63.7402
24. Wu, J., Pan, G., Zhang, D., Qi, G., Li, S.: Gesture recognition with a 3-d accelerometer. UIC '09, pp. 25–38 (2009)