User-independent accelerometer-based gesture recognition for mobile devices

Xian Wang, Paula Tarrío, Ana M. Bernardos, Eduardo Metola, José R. Casar

Data Processing and Simulation Group, Universidad Politécnica de Madrid

KEYWORD

Gesture recognition
Accelerometers
Mobile devices
Human-robot interaction

ABSTRACT

Many mobile devices embed nowadays inertial sensors. This enables new forms of human-computer interaction through the use of gestures (movements performed with the mobile device) as a way of communication. This paper presents an accelerometer-based gesture recognition system for mobile devices which is able to recognize a collection of 10 different hand gestures. The system was conceived to be light and to operate in a user-independent manner in real time. The recognition system was implemented in a smart phone and evaluated through a collection of user tests, which showed a recognition accuracy similar to other state-of-the art techniques and a lower computational complexity. The system was also used to build a human-robot interface that enables controlling a wheeled robot with the gestures made with the mobile phone.

1 Introduction

In the past few years, human-computer interaction has been enriched by the availability of new sensors embedded in consumer electronics devices, which offer new and more natural possibilities of interaction to users. Gestural interfaces, for example, enable a natural interaction between users and computers through the motion of the body, face or hands. These movements or "gestures" can be captured using either a device equipped with inertial sensors carried by the user, such as in the Nintendo's Wii [NINTENDO, 2012], or from a remote device provided with image or depth sensors, such as in the Kinect [KINECT, 2012].

Camera-based gestural interfaces use computer vision algorithms to identify the gestures and trigger the corresponding actions. These techniques have been widely studied and used in many applications, but they also have some limitations. They usually require a high computational power, and they are restricted to the limited areas in which the camera sensors are deployed. In addition, they rely on good lighting conditions, so they cannot be used in all types of environments (for example, it would be very difficult to control the TV volume using a camera-based interface when watching a movie with the lights off) [WU, J. *et al*. 2009]. In contrast, inertial-based gestural interfaces may run over mobile devices, and thus, are also appropriate for applications that require mobility, such as outdoor games or sports. Using embedded sensors also has the advantage that the cost and power consumption are lower [NIEZEN, G. & HANCKE, G.P. 2009] and that they are not dependent on lighting conditions.

The recent and increasing trend of integrating inertial sensors in mobile devices enables the development of new applications based on inertial-based gesture recognition. However, gesture recognition in mobile devices also presents some specific challenges, such as the need of real time operation and low energy consumption, to be able to run efficiently in resource constrained systems.

Several gesture recognition systems based on inertial sensors have been developed to date. The two most popular approaches, taken from the field of speech recognition, are to use hidden Markov models (HMMs) [PYLVÄNÄINEN, T. 2005], [NIEZEN, G. & HANCKE, G.P. 2008], [JOSELLI, M. & CLUA, E. 2009], [WESTEYN, T. *et al*. 2003], [KAUPPILA, M. *et al*. 2007], [KAUPPILA, M. *et al*. 2008],

[HOFMANN, F.G. *et al*. 1998], [SCHLÖMER, T. *et al*. 2008], [MÄNTYJÄRVI, J. *et al*. 2004] or Dynamic Time Wrapping (DTW) [NIEZEN, G. & HANCKE, G.P. 2008] , [LIU, J. *et al*. 2009], although other methods have also been utilized, such as conditional Gaussian models, support vector machines [WU, J. *et al*. 2009] and Bayesian networks [CHO, S.J. *et al*. 2004]. However, only a few [NIEZEN, G. & HANCKE, G.P. 2008], [JOSELLI, M. & CLUA, E. 2009], [NIEZEN, G. & HANCKE, G.P. 2009], [LIU, J. *et al*. 2009] of the techniques mentioned above are implemented on a resource-constrained platform such as a mobile phone. What is more, most of these proposals target at user-dependent gesture recognition, in which an initial training is required for each user in order to calibrate the performance of the system.

In this work, however, our goal is to develop a user-independent gesture recognition system which is light, real-time, and has low energy consumption. Our gesture recognition system was conceived as an intuitive user input interface for a variety of applications in which the user wants to control external devices or equipment using his mobile phone. The requirements of the system are the following:

- User-independent operation: the gesture recognition system should be able to recognize gestures performed by different users without the need of training.
- Lightness: as the system is devised to run on resource-constrained devices, it should have low computational complexity and low energy consumption.
- Real-time operation: the system should be able to operate in real-time without introducing appreciable delays.

To this end, we define a dictionary of simple gestures, and develop time-domain algorithms with low computational complexity to recognize the individual gestures. The proposed gesture recognition algorithms were implemented in a Google Nexus S smart phone and evaluated with real user tests, which showed a good performance in terms of both recognition accuracy and processing time. In addition, the classical DTW algorithm was implemented and the proposed algorithm was compared with it.

As a concrete application of our gesture recognition system, we developed a simple gesture-based human-robot interaction interface, which allows the user to control a wheeled robot by performing gestures with his mobile phone. This kind of interfaces is increasingly becoming more and more important for non-expert users given the rising popularity of social robots.

To sum up, the main contributions of the work are: 1) a light, user-independent and real-time accelerometer-based gesture recognition algorithm and 2) the implementation of a simple human-robot interaction interface based on the previous algorithm.

The paper is organized as follows. Section 2 reviews related work on gesture recognition for mobile devices. Section 3 describes the gesture set selected for our system. Section 4 describes in detail the system architecture and the developed gesture recognition algorithms. The evaluation results are reported in Section 5 and the application to robot control is described in section 6. Finally, Section 7 concludes the paper and proposes some future research directions.

2 Related work

The design of a gesture recognition system can follow a user-independent or a user-dependent approach. The difference lies in whether the user has to train the system before she/he actually utilizes it. User-independent systems are oriented to general users and do not need a training phase before being usable; conversely, user-dependent systems require the user to repeat the gesture movements several times to train the system.

As pointed out in [LIU, J. *et al*. 2009] a great deal of work targets user-dependent gesture recognition only, due to the difficulties in user-independent gesture recognition produced by the great variation between different users, even for the same predefined gesture. The authors of [LIU, J. *et al*. 2009] report that if the collected data are evaluated in a user-independent way instead of user-dependent, the recognition accuracy decreases significantly, from 98.4% to 75.4%. Moreover, the absence of a standard or commonly accepted gesture set – in contrast with the case of speaker-independent speech recognition – has not facilitated the implementation of practical user-independent gesture recognition systems so far. Nevertheless, we aim to develop a user-independent recognition method motivated by the convenience this could bring to the users. It makes the method feasible to carefully select gestures that are intuitive and have common performances among different users.

The selection of the gesture set has a great influence on the recognition accuracy. More complicated gestures may lead to higher accuracy because they have more features to distinguish. Nevertheless, complicated gestures pose a burden on the users: they have to remember how to perform the gestures and associate them with unrelated functions. Therefore, there should not be many complicated gestures to build a user-friendly system [LIU, J. *et al*. 2009].

Some work makes comparisons between several popular algorithms. In [NIEZEN, G. & HANCKE, G.P. 2008], gesture recognition techniques including Hidden Markov Models (HMMs), artificial neural networks and Dynamic Time Warping (DTW) are evaluated based on computational efficiency, recognition accuracy and storage efficiency. In [NIEZEN, G. & HANCKE, G.P. 2009], the same three algorithms are compared and the authors argue that the DTW algorithm runs faster than the other two and that there is no need for preprocessing in the DTW algorithm, which is necessary for the other two. However, since the DTW system has to maintain the template to match, as the amount of gestures to be recognized increases, the storage space required by the DTW algorithm will increase linearly. On the other hand, the storage requirement of the other two algorithms will remain invariable. In [LIU, J. *et al*. 2009], the authors compared HMMs and DTW algorithms. They believe that HMM-based methods require extensive training data to be effective. Furthermore, to configure the models properly, the knowledge of the gesture vocabulary is necessary, which may restrict the variation of gesture vocabulary. According to [LIU, J. *et al*. 2009], the evaluation datasets and testing procedures of the literature using HMMs did not consider gesture variation over the time.

As a summary, Table 1 compares some of the reviewed literature about accelerometer-based gesture recognition systems. In this table, the term "user dependent" means the training samples are from the same subject as the testing sample, while "user independent" means the training samples are not or not all from the same subject as the testing sample. As it can be seen, the recognition accuracy obtained with HMMs and DTW algorithms is quite good, but there is still work to be done for solving the user-independent case. Our previous work [WANG, X. *et al*. 2012] explores this question and proposes a time-domain algorithm for recognizing translation/turn gestures based on the accelerometer/gyroscope signals. A good accuracy and time-efficiency were obtained, but the system was nonflexible, as the translation gestures had to be performed while holding the mobile device in a fixed position. Here we extend and improve our previous algorithm by allowing holding the device in different positions. We also compare its performance with a DTW technique, showing similar accuracy results and lower computation times.

Table 1: Summary of accelerometer-based gesture recognition systems

Paper	Working mode	Recognition method	Platform where implemented	Dataset size (no. gestures / total no. samples)	Recognition accuracy
[CHAMBERS, G.S. et al. 2004]	User dependent	HMM	PC	10 / 548	99.2% the best
[NIEZEN, G. & HANCKE, G.P. 2008]	User dependent	DTW	Mobile phone	8 / 80	96.25%
[LIU, J. et al. 2009]	User dependent	DTW	Mobile phone	8 / 4480	98.6%
[JOSELLI, M. & CLUA, E. 2009]	User dependent	HMM	Mobile phone	10 /400	89%
[KAUPPILA, M. et al. 2007]	User dependent	HMM	PC	9 / unavailable	98.76%
[WESTEYN, T. et al. 2003]	User dependent	HMM	Not on mobile phone	Varies among projects	Varies among projects
[PYLVÄNÄINEN, T. 2005]	User dependent and user independent	HMM	Mobile phone	10 / 1400	99.76%

Table 2: Summary of human-robot interaction systems based on gesture recognition

Paper	Sensing method	Vocabulary	Recognition Method	Application
Sketch interface for HRI [SHAH, D. et al. 2012]	Sketch interface, pen tablet, mouse, etc.	9 drawings: box, arrow, circle, etc.	VDHMM	Search-and-identify missions
Gesture recognition based robot control with LAN [YAN, R. et al. 2012]	Upper body motion capture system	6 gestures: left, right, forward, backward, stop, slow, fast	LAN (localist actuator network)	The recognition system is applied to robot control
Recognition of whole body key gestures for HRI [YANG, H.D. et al. 2007]	Stereo camera	10 whole body gestures: walking, running, bending, jumping, etc.	HMM, with cluster index as observation	The recognition approach has been integrated into a mobile robot
Pointing behavior recognition for HRI [SATO, E. et al. 2007]	Cameras	Pointing	Fuzzy associative memory (FAM)	Manipulating the robot, path set, parking instruction
Body gesture recognition with depth camera [GONZALEZ-SANCHEZ, T. & PUIG, D. 2011]	Depth camera	13 body gestures	Gaussian HMM	Tele-operate a robot
Proximate HRI using mobile phone accelerometer [SERAFIMOV, K. et al. 2012]	Accelerometer	9 gestures: left, right, up, etc.	HMM	Soccer robot control

Gesture recognition techniques have been recently used to develop several human-robot interaction systems with a natural and intelligent interface. Some of these systems, [SHAH, D. et al. 2012], [YANG, H.D. et al. 2007], [GONZALEZ-SANCHEZ, T. & PUIG, D. 2011], [SERAFIMOV, K. et al. 2012] use HMM or its variations, although some other recognition techniques have also been used, such as localist actuator network (LAN) [YAN, R. et al. 2012] and fuzzy associative memory (FAM) [SATO, E. et al. 2007]. Table 2 summarizes some of the characteristics of several human-robot interaction systems based on gesture recognition. All of these systems run gesture recognition algorithm on a computer except for [SERAFIMOV, K. et al. 2012], which, similar to ours, uses a mobile phone to interact.

3 Gesture set selection

The first aspect that has to be decided when designing a gesture recognition system is the vocabulary of gestures that the system is going to use. This decision will influence both the performance of the system (recognition accuracy, processing time, computational complexity, etc.) and its usability. On one hand, in order to maximize the accuracy of gesture recognition, it is convenient to design a vocabulary of gestures which produce inertial sensor responses as different as possible. On the other hand, it is appropriate to use simple gestures, which are easier to perform and remember by users, and also yield lower computational complexity. Therefore, we can select a collection of simple gestures which are supposed to produce distinct inertial stimuli, such as accelerations in different axes. But, how many gestures should be included in the dictionary? According to [PYLVÄNÄINEN, T. 2005], a large dictionary is not practical, as users need to remember how to perform the different gestures and their meaning. Previous gesture recognition systems typically consider from 8 to 10 gestures, to reach a compromise between the possible set of actions that can be triggered by the system and the amount of gestures to remember. In this work, we have considered a collection of 10 simple gestures, which are described in Figure **1**. This selection of gestures is inspired by the user studies reported in [KELA, J. *et al*. 2006]. The meaning of each gesture will depend on the application. We define the *user coordinate system* as the one with the origin of coordinates in the user's hand, the Z axis perpendicular to the ground and pointing up, the X

axis parallel to the ground and pointing to the user, and the Y axis parallel to the ground and pointing to the right of the user. In this system, all the gestures are performed in the Y-Z plane, except Forward and Backward, which are performed along the X axis.

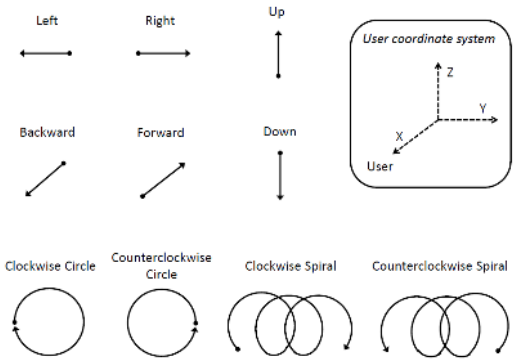

Figure 1: Gesture vocabulary

The defined gestures should be performed maintaining the device in the initial orientation (approximately). However, the user can choose between six possible initial orientations, which are pictured in Figure **2**. These orientations are defined according to the axis of the device pointing to the ground:

- Vertical Up/Down: -Y/Y axis pointing to the ground
- Horizontal Up/Down: -Z/Z axis pointing to the ground
- Lateral Left/Right: -X/X axis pointing to the ground

Figure 2: Possible initial orientations of the device.

The user may perform the gestures defined in Figure **1** holding the device in any of these initial orientations. Note that there is still a degree of liberty defined by a rotation around the axis pointing to the ground. For example, when the device is held in a Vertical Up orientation, the user may be looking at the screen, or at the back of the device (or at any intermediate point). A forward movement in the first case is indeed indistinguishable from a backward movement in the second case, as the accelerations suffered by the device are the same in the two cases. To avoid this degree of freedom, the user is supposed to hold the device in the most usual manner, that is, looking at the screen, for vertical and lateral orientations, and looking at the device in a reading position for horizontal orientations.

4 Gesture recognition system architecture

The current gesture recognition system architecture consists of three different layers- a sensor layer, a processing layer and a communications layer. The sensor layer acquires the information provided by the inertial sensors embedded in the device. The processing layer includes the recognition algorithms to recognize the gestures and the transformation of the recognition result into a specific action. Finally, the communications layer transmits this information through a wireless connection to the external devices that are being controlled. The mobile phone can also receive feedback from the device through this connection.

In the following we describe in detail the proposed gesture recognition algorithm, which is the core of our system. Our method based on the analysis of the acceleration signal and the extraction of its features. The input of the algorithm is a time series provided by a three-axis accelerometer embedded in the device, and the output is the estimated gesture.

4.1 Analysis of the accelerometer signals

In practice, the signal returned by the accelerometer is quite noisy. The velocity and displacement information integrated from this acceleration signal deviates a lot from the real values. For this reason, in our system, the acceleration signal is processed directly.

Let us analyze first the expected shape of the acceleration system for the considered gestures. In the following, we use the *device coordinate system* shown in Figure 2 and suppose the orientation of the device is Vertical Up. We will not take into account

the acceleration of gravity at this point. On the one hand, straight movements can be modeled as two parts with constant accelerations a_1 and a_2, where a_1 and a_2 have opposite sign. Figure 3 shows an example of this straight movement model.

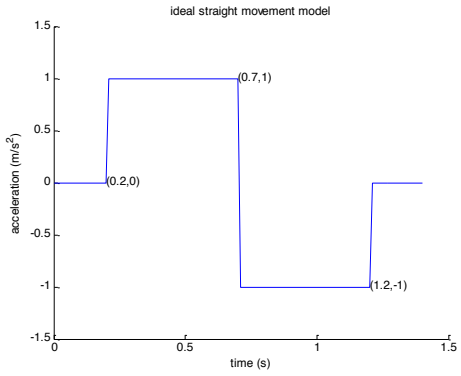

Figure 3: Ideal straight movement model, with $\boldsymbol{a_1 = 1}$, $\boldsymbol{a_2 = -1}$.

Note that this straight movement model applies to all X, Y or Z dimension. Without considering the gravity, the initial and final accelerations of the movement are zero. On the other hand, a circle gesture can be ideally modeled as a uniform circular motion in the X-Y plane. Then the acceleration along X and Y axis can be expressed as:

$$a_x(t) = -m\cos(\theta_t) = -m\cos(\omega t + \theta_0) \quad (1)$$

$$a_y(t) = -m\sin(\theta_t) = -m\sin(\omega t + \theta_0) \quad (2)$$

where m is the module of the acceleration vector, θ_t is the phase at time t, ω is the angular velocity, and θ_0 the initial angle (with respect to the positive X-axis). Figure 4 shows an example of the acceleration signal for this kind of gesture.

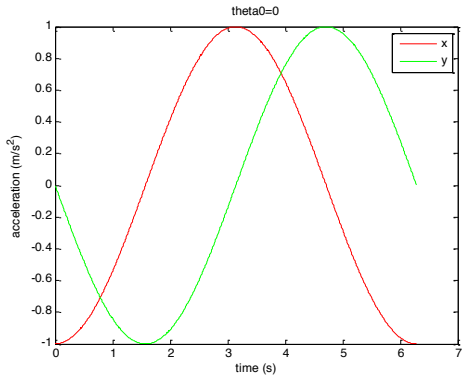

Figure 4: Acceleration signal along x and y axis for the ideal circle movement, with initial phase $\boldsymbol{\theta_0 = 0}$. Here $\boldsymbol{m = 1 m/s^2}$, $\boldsymbol{\omega = 1\ rand/s}$.

4.2 Preprocessing

In practice, accelerations signals are not as simple as the ones shown above, being more complicated and distorted by noise. It is not possible that the users move the mobile phone exactly along one axis and that they complete the 2π circle trajectory exactly. The ideal movement models, however, still provide an abstraction of the real movements and help explain the experimental signals.

The measurements collected from the accelerometer include gravity acceleration no matter what the actual acceleration of the mobile phone is. If we the user holds the mobile phone in the Vertical Up orientation of Figure 2 when he/she moves it, the component due to the gravity acceleration will be on the Y axis. However, in practice, when the user performs the gestures, there is an inevitable tilt of the mobile phone, which will introduce acceleration on Z or X axis, and affect the accuracy of gesture recognition. On the other hand, the signals sensed by the accelerometer are distorted by noise. So, before running the gesture recognition algorithm, we should first rectify the tilt and reduce the effect of noise.

A. Tilt Compensation

In this work, the tilt is compensated with the normalizing method mentioned in [PYLVÄNÄINEN, T. 2005] by estimating the direction of the actual gravitational pull and calculating the rotation matrix. In order to apply this method, the tilt should be small. Therefore, prior to applying the tilt compensation, we first estimate the axis which is more affected by the gravity and change the coordinate system so that the final signal has the strongest gravity component in the Y axis (corresponding to a Vertical Up orientation with some small tilt).

After tilt compensation, the gravity acceleration is subtracted from the Y axis.

B. Noise removing and signal smoothing

In order to reduce the effect of noise, we include a Butterworth low pass filter. The cutoff frequency and the order of the filter were set to 0.3 (normalized, 1 corresponds to the Nyquist frequency) and 2 respectively. These values were determined empirically (too little smoothing will not eliminate enough noise, too much smoothing will blur the features we are interested in).

After passing through the low pass filter, the signal is further smoothed with a moving average method with un-weighted mean. A sliding window with length w is used to calculate the moving average. In our case, w was set to 11. The element in the center of the window is updated with the average of the samples inside the sliding window. Suppose $\boldsymbol{a}(n)$ with $1 \le n \le N$ is a vector with length N, then the smoothed signal $\boldsymbol{a}'$ is calculated by:

$$a'(i) = \frac{a(i-r)+a(i-r+1)+\cdots+a(i)+\cdots+a(i+r)}{w} \quad (3)$$

where $r < i < N - r + 1$, $r = (w-1)/2$, and w is supposed to be odd.

The first and last r elements of vector a' are arbitrarily set to 0.

C. Preprocessed signal

Figure 5 shows the raw original data and the preprocessed signal after tilt compensation, gravity subtraction, Butterworth filtering and smoothing for a "left" gesture.

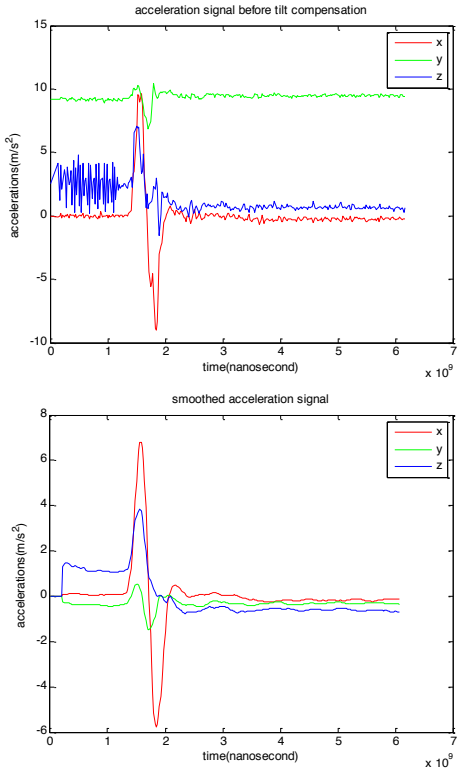

Figure 5: Original signal of a "left" gesture (upper) and corresponding preprocessed signal (bottom).

It can be observed that straight movements happen in practice not only along one dimension (as in the ideal case represented in Figure 3). Furthermore, the module of the acceleration is far from constant.

For the case of the "circle" gestures, it was observed that the movement was not restraint to the X-Y plane, as in the ideal case. From the collected dataset we also observed that the users tend to move the mobile phone more than 2π radians in the circle movements.

The spiral movements could be treated as a circle movement plus a straight movement, but the intensity of the straight movement component is not strong enough compared with the intensity of the acceleration in the circle movement. Therefore, in practice, the spiral movement seems like two or more repetitions of circle movements.

4.3 Classification

Intuitively, comparing the plots of straight, circle and spiral movements performed by several users, we found one main difference between them: the number of "peaks" and "valleys", which are the extrema of the curves, in the X and Y dimensions. The proposed method tries to count the number of peaks and valleys, according to which it classifies the input signal into one of three groups: straight movements, circle movements and spiral movements. Then, other features of the classified signal are extracted and the final recognition results are further determined. The data flow is sketched in Figure 6.

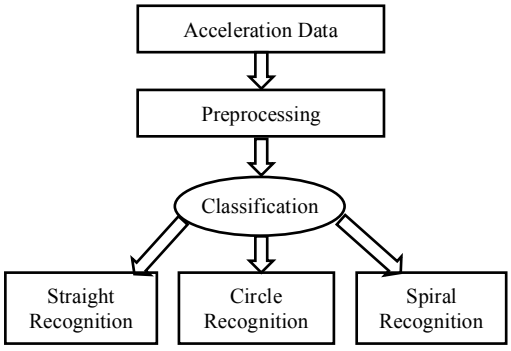

Figure 6: Data flow of the gesture recognition system

We need to detect extrema points to count the peaks and valleys. Consider a one dimensional signal $s(t)$, where $0 \le t \le N$. Then a sample point $s(i)$ marks a peak if $(i-1) < s(i) \,\&\, s(i+1) < s(i)$, and a valley if $s(i-1) > s(i) \,\&\, s(i+1) > s(i)$,

where the operator & means logical "and". According to this definition, many extrema points are usually detected in the preprocessed signal due to the residual noise. Figure 7 shows an example.

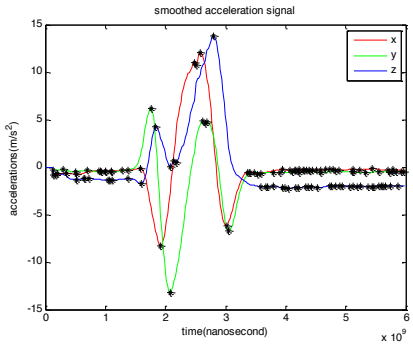

Figure 7: Preprocessed signal corresponding to a "counter-clockwise circle" gesture and extracted extrema points (marked with black stars).

We need to refine the extracted extrema points so that the ones due to small fluctuations are removed. The algorithm to refine extrema points is detailed in the following, assuming that the original signal is $s^{(0)}$, which is along X or Y axis:

1. Calculate the extrema: $s^{(1)} = ext(s^{(0)})$. The first and last extrema will not be removed in the whole algorithm.

2. Remove from $s^{(1)}$ peak extrema that have negative values and valley extrema that have positive values. The remaining sampling points constitute $s^{(2)}$.

3. $s^{(3)} = ext(s^{(2)})$

4. If the length of $s^{(3)}$ equals to the length of $s^{(1)}$, stop. Otherwise, $s^{(1)} = s^{(3)}$, return to step 2.

The function $ext(s)$ extracts extrema points from signal s and also ensures that the absolute difference between any two adjacent extrema points is above a threshold, which is set as 2 in our algorithm.

The refined extraction of extrema for Figure 7 is shown in Figure 8.

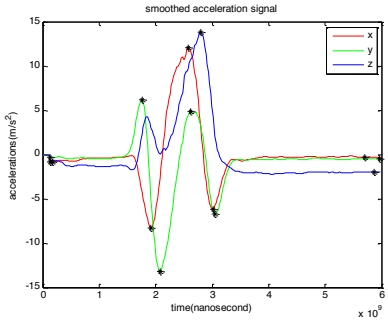

Figure 8: Performance of the extrema refinement algorithm.

Finally, we count the extracted refined extrema in the X and Y dimensions to categorize the input signal into one of the three groups. The conditions were set according to the observations from our dataset:

- If $sum \leq 3 || (nx = 2 \,\&\, ny = 2)$, the gesture is straight movement
- If $sum > 8$, the gesture is spiral movement
- The rest are circle movements

Here, nx and ny are the number of refined extrema points (excluding the first and the last one), along X and Y axis respectively.

Depending on the result of this classification, a different recognition algorithm is further applied, as explained in the following.

4.4 Recognition of straight movements

In the straight movements, the curve in the axis along the movement direction looks like a sinusoid of one period, as the one shown in Figure 5. We treat the sinusoid-like curve as a pattern defined by four key points: the starting point, the ending point, the peak and the valley. At first, the peak and valley are selected as the maximum and minimum point; the starting and ending points are the two extrema adjacent to the peak and valley points. Note that the extrema used here are the original ones instead of the refined ones in the classification stage. Then, given the noisy acceleration measurement and that this method is sensitive to fluctuations of the signal, an adjustment of the key points is employed: we look forward or backward several extrema points from the initial four pattern points and check whether there are higher extrema (for the peak) or lower extrema (for the valley). In this way, the method is robust to small fluctuations.

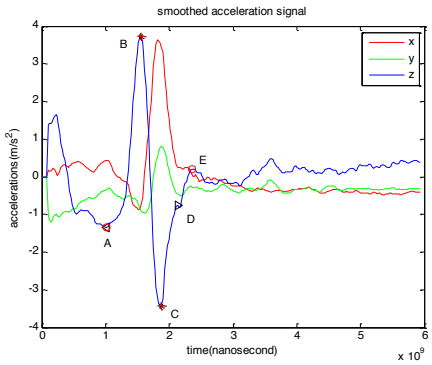

Figure 9: Illustration of adjustment of key points of the pattern

In Figure 9, we try to extract pattern key points along Z-axis. The end point is set as D, which is the right adjacent extremum of the valley point C. Then, because D is also a peak (but small), we look forward within certain range to see whether there is other extrema which have larger values than the ending point. Finally we find point E as the adjustment of ending point.

With this pattern extraction and adjustment, we get the four key points for X, Y and Z dimensions. A symmetry condition is set to avoid unwanted noise. If we denote the acceleration values of the four key points - the starting, the peak (or valley), the valley (or peak) and the ending point - as a_s, a_{pv1}, a_{pv2} and a_e respectively, we then define two ratios:

$$r_1 = min\left\{\frac{|a_s - a_{pv1}|}{\frac{1}{2}|a_{pv1} - a_{pv2}|}, \frac{\frac{1}{2}|a_{pv1} - a_{pv2}|}{|a_s - a_{pv1}|}\right\} \quad (4)$$

$$r_2 = min\left\{\frac{|a_e - a_{pv2}|}{\frac{1}{2}|a_{pv1} - a_{pv2}|}, \frac{\frac{1}{2}|a_{pv1} - a_{pv2}|}{|a_e - a_{pv2}|}\right\} \quad (5)$$

We refine the extracted pattern by omitting those which have r_1 and r_2 values below a predefined threshold, which is tuned as 0.4. Then, we select the axis that has the greatest difference between peak and valley and that agrees with our symmetry requirement, as the dimension where the movement occurred. The appearance order of peak and valley further determines the direction of the movement along this dimension. For example, the plot in Figure 9 represents "Backward" movement.

4.5 Recognition of circle and spiral movements

Suppose that we already know the input signal is a circle or spiral movement, we need to further determine the direction of the movement. To this end, we use the phase information of the signal, both for circle and spiral movements.

It is found that in practice, the phase relationship between X and Y signal of circle movements is similar to the ideal case. As explained before, the spiral movement can be treated as several circle movements because the straight movement along X dimension is very weak. What is more, the direction of the spiral is determined by how the user moves the mobile phone, clock wisely or counter clock wisely. Using the refined extrema extraction obtained in the classification step, we detect the monotonicity of a_y at the point of the first refined extremum of a_x; if it is increasing, then the extremum in a_y following the first refined extremum in a_x is a peak; otherwise it is a valley. Then, according to the following table the direction of circle or spiral movements are determined. The peak and valley are represented by the signs of their acceleration value.

Table 3: Judging table for the direction of circles and spirals. CCW refers to counterclockwise circle and CW to clockwise circle.

First refined extremum in x	+	+	-	-
The extremum followed in y	+	-	+	-
Type of circles	CCW	CW	CW	CCW
Type of spirals	Left	Right	Right	Left

5 Evaluation

In this section we describe the results of a collection of experiments that were defined to evaluate the system with respect to its recognition accuracy and its processing time characteristics. The performance of the proposed system is compared against the classical DTW algorithm, which works in a user-dependent manner. Since this algorithm is mature and reliable, it can provide a state-of-the-art recognition performance, to which the proposed kinematic feature based method can be compared and evaluated.

In order to carry out these experiments, the system described above was implemented in a Google Nexus S smart phone, running Android operating system.

5.1 Description of the dataset and running environment

To test the gesture recognition methods, a dataset was collected with 14 subjects (10 males and 4 females), who repeated 10 times each gesture using a Google Nexus S smartphone. So we gathered 14x10x10 testing samples. Each user completed the data collection in the same day he/she started. The starting and ending timestamps for each gesture were marked by pressing a virtual button of a mobile application prepared to gather gesture logs in the smartphone. The data communication was done by event notifications and the approximate sampling rate of the accelerometer was 0.02 seconds per measurement. The snapshot of the application is shown in Figure 10. Before collecting the gesture data, the users were given a brief introduction about how to perform each gesture. They were told to hold the mobile phone vertically facing the screen.

Both algorithms (proposed and DTW) were tested on Matlab over the collected dataset, on a desktop with Intel 2 Quad CPU (2.5 GHz, 2.5GHz) and 4GB memory. The proposed method was also implemented and tested on the Google Nexus S smartphone.

Figure 10: Snapshot of the mobile application for gathering gesture logs

5.2 Performance of the proposed kinematic feature based method

As this method does not require training, each single gesture sample is processed directly by this recognition method. Then, the recognition accuracy for a given gesture is calculated as the number of correctly detected samples of this gesture divided by the total number of samples of this gesture. The average recognition accuracy is then the mean of the recognition accuracies for all gestures.

Table 4 shows the final recognition results for the proposed method expressed by a confusion matrix. Each row shows the recognition results for a certain gesture. For example, the second row lists the result of the recognition of the “Backward” gesture. There are 93.57% “Backward” movement samples recognized as “Backward” correctly. No samples are recognized as “Clockwise circle” movement. And 2.14% samples are detected as “Counter clockwise circle”, and so on. An average recognition accuracy of 94.14% is achieved.

As mentioned before, the proposed kinematic feature based algorithm has also been implemented on a smartphone running an Android platform. This implementation runs in real-time and was tested with 17 subjects, each of which repeated ten times each of the ten gestures, holding the device with different orientations. In this real-time experiment we obtained an average recognition accuracy of about 89.07%.

5.3 Performance of the DTW method

The DTW algorithm is a dynamic programming based time-normalization algorithm, popular in spoken word recognition field. It was introduced in [SAKOE, H. & CHIBA, S. 1978] to deal with the highly complicated nonlinear fluctuations in the speech pattern time axis. The recognition problem is treated as an optimization problem which tries to minimize the time-normalized distance between two pattern series. So given a limited set of known templates of patterns, an unknown testing pattern can be recognized by calculating the time-normalized distances between it and each of the known templates and selecting the template which is “nearest” to the testing pattern as the recognition result.

In the problem of gesture recognition, there also exist nonlinear fluctuations in the time axis. For example, the speed of the performance of the same gesture varies from one user to another. Therefore, it makes sense to transplant the DTW algorithm to solve gesture recognition problem.

Table 4: Confusion matrix for the ten gestures with kinematic feature based method (%), run with Matlab.

	Backw.	**Cw c.**	**Ccw c.**	**Down**	**Forw.**	**Left**	**Right**	**S. left**	**S.right**	**Up**
Backw.	93.57	0	2.14	1.43	0.71	2.14	0	0	0	0
Cw c.	0	94.29	0.71	0	0	0	0	0	4.29	0.71
Ccw c.	0	2.14	92.14	0	0.71	2.86	0	2.14	0	0
Down	0	2.86	1.43	92.86	2.14	0.71	0	0	0	0
Forw.	2.14	1.43	1.43	0	94.29	0	0.71	0	0	0
Left	0.71	0.71	0	0	0	98.57	0	0	0	0
Right	0	1.43	0.71	0	0	0	97.86	0	0	0
S. left	0	0.71	2.86	0	0	0	0	90.71	5.71	0
S.right	0	2.86	0	0	0	0	0	0	97.14	0
Up	3.57	3.57	2.14	0	0.71	0	0	0	0	90.00

The DTW method is classical and reliable, and provides a performance benchmark for other newly developed methods, as ours. In this work, we implemented a DTW method based on the methods introduced in [SAKOE, H. & CHIBA, S. 1978] and [MYERS, C. *et al*. 1980].

Before running the DTW algorithm, the input signal is down sampled with a Butterworth filter to enhance the computational efficiency. After down sampling, the length of the signal is normalized to enhance the performance. The method is the same as mentioned in [MYERS, C. *et al*. 1980].

We tested this DTW method over the collected dataset described in section 5.1. The DTW method recognizes gestures by comparing the testing sample with known templates and selecting the template which has the smallest mapping cost as the detected result. The template should be "standard" enough, as the template selection has great influence on the recognition accuracy. The evaluation method used here is the leave-one-out method. The idea is to use a single observation from the original samples as the testing data and the remaining observations as the training set. This is repeated so that every original sample in the dataset is utilized as testing data. In our test, we tested the dataset user by user. For a certain user and a certain gesture type, we selected one template from the 9 training samples (there are 10 data samples per gesture per user) with the *minimum selection* method mentioned in [KO, M.H. *et al*. 2008].

In the evaluation, we took into consideration the constraints mentioned in [SAKOE, H. & CHIBA, S. 1978] and [MYERS, C. *et al*. 1980], and we especially examined the effect of the local continuity constraints [MYERS, C. *et al*. 1980], the window length in adjustment window [SAKOE, H. & CHIBA, S. 1978], the form of weighting coefficient [SAKOE, H. & CHIBA, S. 1978] and the different types of distance definitions (including cosine correlation coefficient, Euclidean distance and Chebyshev distance). We found that small values of the window length lead to higher recognition accuracy and less computing time. The two different forms (symmetric and asymmetric) of the weighting coefficient have similar performance in terms of recognition accuracy and computational efficiency. The Type I local continuity constraint in [MYERS, C. *et al*. 1980] results in higher recognition accuracy than the Type II constraint. On the other hand, the Euclidean distance measure has lower computing time than the distance measure with cosine correlation coefficient. The combination of Type I local continuity constraint and cosine correlation coefficient distance measurement attains the highest recognition accuracy.

If we select the Type I local continuity constraint and the cosine correlation coefficient distance measurement, the recognition accuracy is 94.04% and the average computing time for each DTW matching of two data samples is 5.8ms. The down sampling step takes 0.157ms for each data sample on average.

5.4 Comparison of the two recognition methods

¡Error! La autoreferencia al marcador no es válida. shows a comparison of the two recognition methods in terms of different parameters.

Table 5: Comparison of the two recognition methods

	Kinematic feature based	**DTW**
Accuracy	94.14%	As high as 94.04%
Time efficiency (ms)	4.8	(0.157+5.8)*10
Efficiency (memory)	O(n)	$O(\tilde{N}^2)$
Working mode	User independent	User dependent
Ease of development	A little difficult	Easy
Knowledge about vocabulary	Requires a lot	Not much
Flexibility	Small	Large

For this dataset, the kinematic feature based method has a little higher recognition accuracy. However, the recognition result of this method is based on the fact that the parameters are tuned according to the dataset, which means that the 94.14% accuracy is almost the best one for this method. On the other hand, no sample-specific parameters are needed in the DTW method, except for the down sampling rate, so its recognition result has less dependency on dataset.

The kinematic feature based method needs 4.8ms to process one data sample, including preprocessing and recognition. For the DTW, the down sampling spends 0.157ms and each mapping cost calculation needs 5.8ms. The DTW method determines the final recognition result after comparing the matching costs with the 10 templates. Then, to recognize one input data sample, the DTW method needs $(0.157 + 5.8) * 10ms = 59.57ms$, more than 12 times the one of the proposed method.

With respect to the memory requirements, we denote the length of acceleration signal as n and the normalized length as $\tilde{N}$. For this test, n is 304 and $\tilde{N}$ is 20, so the DTW method requires a little more memory. For longer signals, the proposed method would be more efficient.

One important advantage of the kinematic feature based method is that it does not need a training step, while the DTW method has to select template in the training phase first to then do the test. The user-independent method is preferred because the user can enjoy the human-device interaction technique directly without the boring training step.

However, in general, it is more difficult to develop a user-independent gesture recognition method than a user-dependent one, due to the great variation of gesture performance between different users for the same gesture. A uniform standard rule may not always be found to distinguish gestures.

The proposed method also requires an exhaustive knowledge of the gesture vocabulary to select proper features, thresholds and rules to distinguish the different gestures, while DTW provides a general procedure to differentiate gestures without a detailed knowledge of gesture vocabulary.

Flexibility here refers to how the methods can be applied on different users and on a different gesture vocabulary. The proposed method relies heavily on the threshold tuning and on the concrete form of the movements. So it can only be applied on a certain group of users with the specified gesture vocabulary. However, the DTW method, due to its user-dependent manner, applies to everybody and, since it works by matching testing sample with templates which are not restricted within a fixed gesture vocabulary, it can be applied to a different gesture vocabulary.

6 Robot control application

In order to define meaningful gestures for the experimental tests, we decided to configure the system for controlling the actions of a wheeled robot in an indoor space. The mobile robot used in the work is P3-DX by MOBILEROBOTS Inc. There is an onboard computer which bears a 1.79 GHz Intel Pentium M processor, 476M RAM, and the operating system is Debian lenny with Linux kernel version 2.6.26-2-686. The robot has an antenna which works as an access point. When the robot is moving, it can be connected by a laptop or mobile phone with wireless connection through the SSH protocol. The wire-

less network driver is MadWifi. This robot is also provided with sonar and laser sensors, which are used to build 2D maps of areas of interest. We also attached to the robot a camera sensor (Logitech Webcam C600, supported by a Linux UVC driver) that allows taking photos of the area surrounding the robot.

We used the implementation of the kinematic feature based method on the Android mobile phone to construct a human-robot interaction system via setting up a wireless connection between the mobile phone and the robot. The objective was to control the robot with gestures made with the mobile phone.

When the user performs one of the predefined gestures, the recognition system running on the mobile phone recognizes the gesture type and sends it to the robot through the wireless connection. The robot then performs an action corresponding to that gesture type. The gestures and the corresponding actions implemented in this work are listed in Table 6.

Table 6: Gestures and corresponding commands to control the robot.

Gesture	**Command**
Right	Take a photo
Spiral to the right	360-degree sonar measurement
Forward	Move to a given position
Down	Stop
Left	Pause
Up	Continue

Three actions were implemented in this work, including simple navigation (moving to a given position), taking a photo with the web camera mounted on the robot, and taking 360-degree sonar measurement with the sonar disk array in the front of the robot. After completing the action, the robot sends feedback to the mobile phone through the wireless connection. This feedback includes the task status, and in the two last cases, the collected information (photo or sonar measurement). The other three commands- stop, pause and continue- are employed to control the progress of the moving action.

The complete system was tested with several experiments, which showed good user friendliness and satisfactory interaction, with fast recognition times and good correspondence between the expected and the final robot action.

7 Conclusions

In this paper, a kinematic feature based gesture recognition method has been proposed and compared with the classical DTW algorithm in various aspects. For the offline evaluation, its recognition accuracy was similar to the one achieved by the DTW algorithm, and for the online test the accuracy was also appropriate, with about 89.07%. The advantage of our method is that it is light-weighted, has higher computational and storage efficiency than the DTW algorithm, and works in user-independent mode, so that it is suitable for resource-restricted platforms and real-time applications, and helps to enhance user experience. However, the cost to obtain these gains is that it is vocabulary-specific, resulting in limited flexibility because it cannot recognize gestures outside the vocabulary.

We also developed a human-robot interaction system based on the proposed method that enables controlling a wheeled robot through gestures performed with a mobile phone. This experiment shows the feasibility of using the proposed technique as an intuitive user input interface for applications in which the user wants to control external devices or equipment using his mobile phone.

We are planning to examine in a systematic way the performance of the human-robot interaction system in terms of interaction intuition, response delay and error tolerance. With respect to the gesture recognition system, a natural extension would be recognizing sequences of gestures, which would allow users to send series of commands and may be also used to generate new commands with the combination of the predefined individual gestures.

8 Acknowledgment

This work has been supported by the Region of Madrid Government under grant S2009TIC-1485 (CONTEXTS) and by the THOFU project (CENIT CDTI).

9 References

[CHAMBERS, G.S. *et al*. 2004] CHAMBERS, G.S.; VENKATESH, S.; WEST, G.A.W. & BUI, H.H. Segmentation of Intentional Human Gestures for Sports Video Annotation, in: MMM '04 Proceedings of the 10th International Multimedia Modeling Conference, Brisbane, Australia, 2004, pp. 124-129.

[CHO, S.J. *et al*. 2004] CHO, S.J.; OH, J.K.; BANG, W.C.; CHANG, W.; CHOI, E.; JING, Y.; CHO, J. & KIM, D.Y. Magic wand: a hand-drawn gesture input device in 3-D space with inertial sensors, in: Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on, Kokubunji, Tokyo, Japan, 2004, pp. 106-111.

[GONZALEZ-SANCHEZ, T. & PUIG, D. 2011] GONZALEZ-SANCHEZ, T. & PUIG, D. Real-time body gesture recognition using depth camera, Electronics Letters 47 (12) (2011) 697-698.

[HOFMANN, F.G. *et al*. 1998] HOFMANN, F.G.; HEYER, P. & HOMMEL, G. Velocity profile based recognition of dynamic gestures with discrete Hidden Markov Models, in: International Gesture Workshop, Bielefeld, Germany, 1998, pp. 81-95.

[JOSELLI, M. & CLUA, E. 2009] JOSELLI, M. & CLUA, E. gRmobile: A Framework for Touch and Accelerometer Gesture Recognition for Mobile Games, in: Games and Digital Entertainment (SBGAMES), 2009 VIII Brazilian Symposium on, Rio de Janeiro, Brazil, 2009, pp. 141-150.

[KAUPPILA, M. *et al*. 2007] KAUPPILA, M.; PIRTTIKANGAS, S.; SU, X. & RIEKKI, J. Accelerometer Based Gestural Control of Browser Application, in: International Workshop on Real Field Identification (RFId2007), Tokyo, Japan, 2007, pp. 25-28.

[KAUPPILA, M. *et al*. 2008] KAUPPILA, M.; INKEROINEN, T.; PIRTTIKANGAS, S.; & RIEKKI, J. Mobile phone controller based on accelerative gesturing, in: Pervasive 2008, the Sixth International Conference on Pervasive Computing, Sydney, Australia, 2008, pp. 130-133.

[KELA, J. *et al*. 2006] KELA, J.; KORPIPÄÄ, P.; MÄNTYJÄRVI, J.; KALLIO, S.; SAVINO, G.; JOZZO, L. & MARCA, S.D. Accelerometer-based gesture control for a design environment, Personal and Ubiquitous Computing 10 (5) (2006) 285-299.

[KINECT, 2012] Kinect website, http://www.xbox.com/kinect, 2012

[KO, M.H. *et al*. 2008] KO, M.H.; WEST, G.; VENKATESH, S. & KUMAR, M. Using dynamic time warping for online temporal fusion in multisensor systems, Information Fusion 9 (3) (2008) 370-388.

[LIU, J. *et al*. 2009] LIU, J.; WANG, Z.; ZHONG, L.; WICKRAMASURIYA, J. & VASUDEVAN, V. uWave: Accelerometer-based Personalized Gesture Recognition and its Applications, in: Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on, Galveston, TX, USA, 2009, pp. 1-9.

[MÄNTYJÄRVI, J. *et al*. 2004] MÄNTYJÄRVI, J.; KELA, J.; KORPIPÄÄ, P. & KALLIO, S. Enabling fast and effortless customisation in accelerometer based gesture interaction, in: MUM '04: Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia, College Park, MD, USA, 2004, pp. 25-31.

[MYERS, C. *et al*. 1980] MYERS, C.; RABINER, L.R. & ROSENBERG, A.E. Performance tradeoffs in dynamic time warping algorithms for isolated word recognition, IEEE Transactions on Acoustics, Speech, and Signal Processing 28 (6) (1980) 623-635.

[NIEZEN, G. & HANCKE, G.P. 2008] Niezen, G. & Hancke, G.P. Gesture Recognition as Ubiquitous Input for Mobile Phones, in: UbiComp '08 Workshop W1 – Devices that Alter Perception (DAP

2008), Seoul, South Korea, 2008.

[NIEZEN, G. & HANCKE, G.P. 2009] NIEZEN, G. & HANCKE, G.P. Evaluating and Optimising Accelerometer-based Gesture Recognition Techniques for Mobile Devices, in: AFRICON 2009, Nairobi, South Africa, 2009, pp. 1-6.

[NINTENDO, 2012] Nintendo's Wii website, http://www.nintendo.com/wii, 2012

[PYLVÄNÄINEN, T. 2005] PYLVÄNÄINEN, T. Accelerometer Based Gesture Recognition Using Continuous HMMs, in: Second Iberian Conference, IbPRIA 2005, Estoril, Portugal, 2005, pp. 639-646.

[SAKOE, H. & CHIBA, S. 1978] SAKOE, H. & CHIBA, S. Dynamic programming algorithm optimization for spoken word recognition, IEEE Transactions on Acoustics, Speech and Signal Processing 26 (1) (1978) 43-49.

[SATO, E. *et al*. 2007] SATO, E.; YAMAGUCHI, T. & HARASHIMA, F. Natural Interface Using Pointing Behavior for Human–Robot Gestural Interaction, Industrial Electronics, IEEE Transactions on, 54 (2) (2007) 1105-1112.

[SCHLÖMER, T. *et al*. 2008] SCHLÖMER, T.; POPPINGA, B.; HENZE, N. & BOLL, S. Gesture recognition with a Wii controller, in: TEI '08: Proceedings of the 2nd international conference on Tangible and embedded interaction, Bonn, Germany, 2008, pp. 11-14.

[SERAFIMOV, K. *et al*. 2012] SERAFIMOV, K.; ANGELKOV, D.; KOCESKA, N. & KOCESKI, S. Using Mobile-phone Accelerometer for Gestural Control of Soccer Robots, in: Embedded Computing (MECO), 2012 Mediterranean Conference on, Bar, Montenegro, 2012, pp. 140-143.

[SHAH, D. *et al*. 2012] SHAH, D.; SCHNEIDER, J. & CAMPBELL, M. A Sketch Interface for Robust and Natural Robot Control, Proceedings of the IEEE 100 (3) (2012) 604-622.

[WANG, X. *et al*. 2012] WANG, X.; TARRÍO, P.; METOLA, E,; BERNARDOS, A.M. & CASAR, J.R. Gesture recognition using mobile phone's inertial sensors, in: 9th International Conference on Distributed Computing and Artificial Intelligence, Salamanca, Spain, 2012, Advances in Intelligent and Soft Computing, Volume 151, pp. 173-184.

[WESTEYN, T. *et al*. 2003] WESTEYN, T.; BRASHEAR, H.; ATRASH, A. & STARNER, T. Georgia Tech Gesture Toolkit: Supporting Experiments in Gesture Recognition, in: ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces, Vancouver, British Columbia, Canada, 2003, pp. 85-92.

[WU, J. *et al*. 2009] WU, J.; PAN, G.; ZHANG, D.; QI, G.& LI, S. Gesture recognition with a 3-d accelerometer, in: 6th International Conference, Ubiquitous Intelligence and Computing (UIC) 2009, Brisbane, Australia, 2009, pp. 25-38.

[YAN, R. *et al*. 2012] YAN, R.; TEE, K.P.; CHUA, Y.; LI, H. & TANG, H. Gesture Recognition Based on Localist Attractor Networks with Application to Robot Control, Computational Intelligence Magazine, IEEE 7 (1) (2012) 64-74.

[YANG, H.D. *et al*. 2007] YANG, H.D.; PARK, A.Y. & LEE, S.W. Gesture Spotting and Recognition for Human–Robot Interaction, Robotics, IEEE Transactions on 23 (2) (2007) 256-270.