Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 82638, 11 pages
doi:10.1155/2007/82638

Research Article

MOCA: A Low-Power, Low-Cost Motion Capture System Based on Integrated Accelerometers

Elisabetta Farella,[1] Luca Benini,[1] Bruno Riccò,[1] and Andrea Acquaviva[2]

[1] *Dipartimento di Elettronica, Informatica e Sistemistica (DEIS), Universitá di Bologna, Viale Risorgimento 2, 40136 Bologna, Italy*
[2] *Information Science and Technology Institute (ISTI), University of Urbino, Piazza Repubblica 13, 61029 Urbino, Italy*

Received 5 September 2006; Revised 18 January 2007; Accepted 8 March 2007

Recommended by Yong Pei

Human-computer interaction (HCI) and virtual reality applications pose the challenge of enabling real-time interfaces for natural interaction. Gesture recognition based on body-mounted accelerometers has been proposed as a viable solution to translate patterns of movements that are associated with user commands, thus substituting point-and-click methods or other cumbersome input devices. On the other hand, cost and power constraints make the implementation of a natural and efficient interface suitable for consumer applications a critical task. Even though several gesture recognition solutions exist, their use in HCI context has been poorly characterized. For this reason, in this paper, we consider a low-cost/low-power wearable motion tracking system based on integrated accelerometers called motion capture with accelerometers (MOCA) that we evaluated for navigation in virtual spaces. Recognition is based on a geometric algorithm that enables efficient and robust detection of rotational movements. Our objective is to demonstrate that such a low-cost and a low-power implementation is suitable for HCI applications. To this purpose, we characterized the system from both a quantitative point of view and a qualitative point of view. First, we performed static and dynamic assessment of movement recognition accuracy. Second, we evaluated the effectiveness of user experience using a 3D game application as a test bed.

1. INTRODUCTION

Motion tracking is a key enabling technology that is nowadays widely applied into research and commercial fields ranging from human-computer interaction to robot navigation, virtual and augmented reality. Tracking systems are now a mature technology [1] for being used in ambient intelligence applications to provide users with personalized services depending on their location or position. They are also used in augmented reality applications to generate virtual scenes depending on the user point of view.

Recently, tracking of human gestures has been extensively studied to promote human-like input interfaces to machines. In particular, virtual reality (VR) applications and immersive games critically require natural human-computer interfaces (HCIs), which reduce the gap between virtual-world and real-world interactions. For instance, in learning or gaming applications where navigation in virtual environments is needed, the effectiveness of the user experience depends not only on the rendering quality but also on the interaction with the system. A successful HCI system must guarantee real-time feedback to the user and be neither intrusive nor cumbersome [2]. Furthermore, to allow deployment on commercial-off-the-shelf computer systems, it must be designed with commodity components.

Most of the research on motion tracking is concerned with the design of highly accurate and expensive systems for absolute positioning in space. Some specific tracking tasks, such as gesture recognition, do not require high precision in absolute positioning, and currently available commercial systems based on electromagnetic and optical sensors are too expensive for consumer products.

Accelerometer-based sensing methods are a promising technology for low-cost motion capture systems [2, 3], since they can be implemented with low-power integrated components. Furthermore, unlike more accurate tracking techniques based on optical or movement sensors [4–6], they do not require video cameras, making them more suitable for mobile interaction. In fact, in a virtual or augmented reality environment, the user may want to move around without bounds to interact with virtual objects.

In this work, we focus on a low-cost motion capture interface system targeted to the navigation in virtual interactive environment called motion capture with accelerometers (MOCA) [7]. This recognition technology has been used to implement a body sensor network targeted to human posture recognition [1]. In this paper, the objective is to demonstrate, through extensive user experience characterization, that our low-cost, low-power while robust motion recognition system is valuable to implement human-like interaction. We selected a subset of the possible recognizable postures using a single node to test the effectiveness of the recognition algorithm. This choice has been made to characterize the recognition performance independently from network and synchronization issues.

The system is based on commodity components such as integrated accelerometers and completed by many software parts. The main features of the system are (i) a low-cost and low-power implementation providing easy deployment and wearability; (ii) an optimized data acquisition process; (iii) a lightweight and efficient position and motion recognition software. It is driven by a geometrical tracking algorithm, that guarantees robustness with respect to drift problems typical of solutions based on integration of acceleration data [2, 8, 9]. All these characteristics make the MOCA system suitable to interactive environments and allow its implementation on resource-constrained devices such as PDAs.

The recognition process works as follows. Data transmitted by sensing units are sampled and collected by an acquisition/control board that communicates to the machine running the virtual world by means of a serial link. The limited size and weight of the acquisition board make it suitable to be installed on a palmtop computer or PDA, providing unconstrained mobility. Communication between sensing units and the acquisition board takes place by means of a dedicated wired link. However, a fully wireless solution where both sensing unit and acquisition board are provided with RF interface has also been developed [10]. In both cases, the cost of the interface hardware is around $100 (even in small volume prototyping). In this scenario, the palmtop computer may represent a preprocessing stage for final interaction to a remote sensor fusion system for ambient intelligence or location-aware applications. On the other hand, it may represent the target system if low-end interactive applications are running on it (e.g., games).

We performed a number of tests to verify accuracy, power consumption, and performance of our system. Even if acceleration sensors can be placed anywhere on the body, we performed tests using representative rotational movements of the hand. We obtained a static accuracy of 3 degrees for different contiguous orientations. With regard to dynamic characteristics, our system is suitable to detect human body movements for HCI applications, that are typically characterized by an angular velocity of around 3 radians/s. Power consumption measurements show that the system can operate continuously at least for 12 hours with 100 g, 750 mAh batteries. For the wireless prototype we followed the same low-cost/low-power design guide, additional power consumption being only 15% [1]. Our system has been also extensively tested in the field. First, the performance of the interface in terms of response time and movement loss has been analyzed. Then, a 3D game was used as a test bed to evaluate usability and user experiences.

The contribution of this work is thus twofold. First, we demonstrate that a low-cost, low-power motion tracking solution based on integrated accelerometers and on lightweight, efficient, and robust motion tracking algorithm provides an accuracy suitable for HCI applications. Second, we present the results of extensive characterization on the field through the evaluation of human experience with real applications.

2. RELATED WORK

Motion tracking has been studied in many fields and a variety of solutions have been proposed depending on the targeted application requirements [10].

2.1. Motion tracking solutions

For human-computer interaction (HCI), which aims at building natural perceptual user interfaces, human motion tracking is used to ease the interactions with machines and to enhance the mobility of portable devices. The design of interfaces for natural human-computer interaction is a critical research problem that has been addressed in different ways depending on the targeted systems and applications [11]. Several studies have been performed to design unobtrusive input interfaces for wearable computers [2, 3, 12, 13]. In fact, these devices cannot be equipped with traditional input devices like keyboards and mice for mobility reasons. In this field various modalities are investigated: gesture, gaze, facial expressions as they occur in human-human communication. Gesture recognition systems have lower accuracy and jitter requirements compared to other applications where motion tracking is applied such as robotic and augmented reality [14–25]. Instead power consumption and cost are the most critical factors. For these reasons, previous solutions are not well suited to these kinds of applications. The purpose of tracking in this context is to recognize a set of movements indicating commands. Thus, the accuracy affects just the number of different commands that can be distinguished.

Both vision-based and accelerometer-based techniques have been proposed for gesture recognition [14, 17, 26]. One general weakness of optical tracking systems is that there must be line of sight between the camera and the tracked object. Another drawback of the optical method is its strong dependency on environmental luminance. In addition, it is difficult to use this tracking system for real-time applications since the system needs postprocessing of the data to extract motion information. Moreover, commercially available products adopting these kinds of techniques are currently too expensive for consumer applications [27]. The motes presented in [28] are a very interesting solution in terms of wearability, since they are tiny and modular. However, at present they are not optimized for a particular application scenario, such as movement tracking and gesture recognition.

Accelerometer-based solutions, even though less accurate than optical tracking, are a promising alternative for gesture recognition in mobile environments. Commercial systems based on inertial sensors [29, 30] are in many cases accurate enough for HCI domain, but still expensive (also because being sold in pair with dedicated software), even if the cost is generally lower compared with multicamera-based motion trackers. For these commercial solutions, the bridge between sensors on body and the gateway is wireless, but cables are still present among the units worn by the user, limiting usability.

Several academic projects propose various types of input hardware [3, 9, 12]. The work of Bang et al. [27] is a dedicated solution consisting of a pen system based on accelerometers and gyroscopes. The system is used for recognizing handwriting motion in 2D and 3D spaces. One major problem faced in [9] is the compensation for inaccuracies in position and orientation detection due to the integration process. This problem is common to most of the accelerometer-based solutions we analyzed. Different approaches have been evaluated to face it, ranging from Kalman filtering [2], usage of secondary sensors [8] to compensation techniques exploiting external measurements or introducing reinitialization, as it happens in [9] where the zero-velocity compensation (ZVC) technique is exploited. This technique is not tailored to the interaction with virtual environments.

A microaccelerometer-based interface for interaction with mixed reality in wearable computers has been presented by Cheok et al. [2]. Accelerometers are mounted on various kinds of input devices: a pen and a tilt and gesture pad. They provide 3D information about motion of the body part on which they are worn. Authors present a detailed testing of the filtering approach they exploited to reduce position and velocity measurement errors caused by the integration step. Results obtained show that filtering helps reducing even though not eliminating the measurement error. Moreover, the application of a complex filtering step may consistently degrade the dynamic performance of the recognition. For this reason, the applicability of this approach to interactive input systems is still not proven.

Motion tracking with accelerometers can be used also for interaction with intelligent environments, as shown in a recent work by Wilson and Shafer [8]. This approach envisions the collaboration between a hand-held sensor (a wand called *XWand*) system with the external environment (e.g., a living room) to let the user control intelligent devices. The presented system exploits an infrared link to point the device to be controlled, then uses gesture recognition to control it. Position information is computed using integration of sensor data and gesture mapping is performed exploiting the knowledge of interaction context. A set of gestures may be significant only for that particular device pointed by the wand.

Gesture processing systems have been designed for interaction with VR systems using immersive 3D displays (such as for navigation in virtual theaters) [5, 33]. They are mainly targeted to particular movement, such as the sentient floor [31] and the SenToy doll [6].

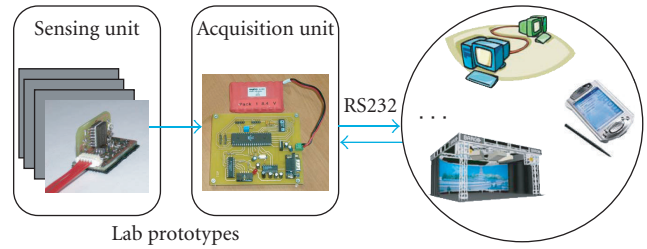

Figure 1: The gesture recognition interface can interact with various hardware targets.

2.2. The MOCA solution

The motion tracking system described in this paper targets gesture recognition for interaction with a virtual environment. The main characteristics of the motion capture system are the following: consumer, low-power and wearable hardware; low-complexity position recognition algorithm (tilt-based) with high dynamic performance suitable for interactive systems. In our system, accelerometers are used to recognize inclinations of human body. In our example application we used two accelerometers to recognize the arm position and movements.

Main contribution of this work with respect to the state of the art is the followings. First, we do not perform integration on the accelerometer data, instead we rely on a geometrical approach based on tilt. We use this approach to compute inclination with respect to gravity since we do not need information on absolute position, which can be obtained through double integration. Our approach eliminates the need for compensating for the error introduced by the integration step common to the techniques described before [2, 8, 9]. This is a critical advantage since noise cancellation from double integration is an extremely challenging task, for which no general solution is known. In addition, the reduced computational complexity makes our system more suitable for mobile devices with limited computational resources such as palmtop computers.

The second distinction concerns the qualitative assessment. In our work, we targeted the evaluation of gesture recognition effectiveness for interaction with VR applications like 3D games. Instead of just focusing on accelerometer accuracy, we provide a full evaluation of user experience by measuring the number of recognized movements for different user behaviors.

3. GENERAL SYSTEM ARCHITECTURE

The general system architecture is composed of a fixed system back end and a flexible and reconfigurable front end (Figure 1).

The system back end consists of (i) the MOCA prototype (one acquisition board and up to four sensing units) (ii) the MOCA interface, and (iii) the implementation of the tracking algorithm. It is responsible for collecting and interpreting data (from basic driver operation, to position recognition or

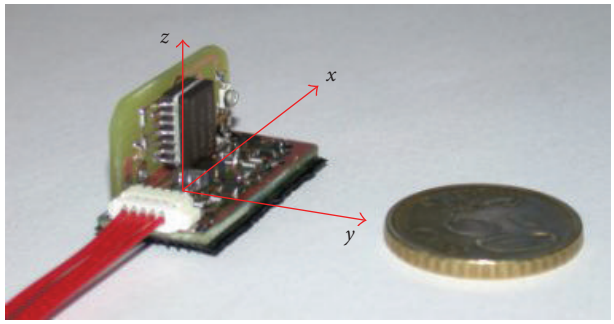

Figure 2: A sensing unit.

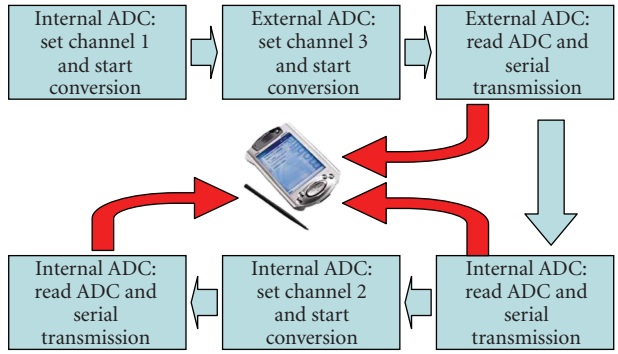

Figure 3: Acquisition framework.

complex processing such as gesture recognition). The front end is the target application, which can take many forms from simple visualization of the data collected to sophisticated interaction with a virtual environment. In fact, two configurations are possible. In the first one the mobile terminal (a palmtop computer, a notebook, or a more complex processing unit in a network with wireless capabilities) is the target for data acquired by sensors. Alternatively, the wireless mobile terminal is a bridge to interact with a remote system. Thanks to general purpose features and wireless capabilities, the entire system becomes a flexible and reconfigurable architecture. Thus the system can benefit from the ability to execute the processing algorithm either locally on the mobile unit (as one of our example application) or remotely depending on its complexity.

3.1. Motion capture back end

Moca sensing units and acquisition board

MOCA is built entirely using low-cost commodity components and it is based on two boards: (i) sensing units (shown in Figure 2) and (ii) acquisition board. The user will place a number of sensors corresponding to the body part to track (e.g., in our first implementation we tested arm tracking) and the number of degrees of freedom considered.

The acquisition board is equipped with a RISC microcontroller with integrated analog-to-digital converter (ADC) and UART controller, a serial port, and an external ADC. Each sensing unit contains two different MEMS (micro electronic mechanical system) accelerometers. In order to collect 3D acceleration vector, a monoaxial sensor is mounted orthogonally with respect to biaxial accelerometer.

The acquisition board is responsible for data sampling, conversion from a set of sensing units, processing, and transmission to the gesture recognition software (on a wearable computer or a PC) through a serial link. Acquisition takes place exploiting both internal and external ADCs in an optimized way, as described later. The processing step involves the arrangement of data samples into packets with marking stamps indicating the beginning and end of one acquisition step. For each step, data from all accelerometers are collected.

Acquisition time optimization

The acquisition process accounts for a time interval short enough (about 350 microseconds from the start of sampling to the end of the serial transmission) with respect to data sampling rate (10 milliseconds for 100 Hz accelerometer bandwidth), to save the rest of the time for local processing.

Acquisition time has been optimized by using both internal (on the microcontroller) and external (on board) ADCs. The overall processing time is reduced by exploiting the parallelism of these components. Figure 3 shows the acquisition and transmission sequence for a single sensing unit. The sequence is composed of four steps: (i) channel selection, (ii) start acquisition, (iii) read ADC, (iv) serial transmission. Internal ADC is associated with channels 1 and 2, while external ADC is associated with channel 3. In order to exploit hardware parallelism, acquisition from channels 1 and 3 starts consecutively. Two reads and transmissions are subsequently performed from these channels (2 bytes and 1 byte, resp.) before starting the acquisition from channel 2, read, and transmit (1 byte).

MOCA interface

We implemented two alternative drivers as support for interfacing with the prototype: a Linux driver and a set of Java classes. Both of them are responsible for controlling the acquisition and process data packets received from the sensing/acquisition board and extract the payload. A specific task implements the handshaking protocol required to communicate with the MOCA board. It consists of sending a start character to the microcontroller to trigger the beginning of the acquisition. A stop character causes the board to go in waiting state or completely stop the acquisition. After payload extraction, the driver performs some minimal processing to convert input data, expressed as voltage values, into acceleration values. Finally, the driver configures some parameters such as gain and offset using default values. These parameters are needed to tune the above-mentioned conversion process. The Java solution is motivated by the need to guarantee platform independence. This can be used to easily port the code to different operating systems (OSs) and hardware platforms. The SOFTWARE CLASS responsible for

reading data from the serial port is implemented exploiting the Java COMM extension package. This library contains the necessary classes to handle the communication ports, enabling access to them at many abstraction levels, depending on programmer needs. The Linux driver implemented offers a standard application level interface to communicate with MOCA board. Developing a specific Linux driver guarantees better efficiency than the JAVA-based solution, as the processor does not experience the workload generated by the JAVA virtual machine.

Tracking algorithm

The module in charge of processing the data acquired from the acquisition part of the system performs two tasks. Firstly, it translates bytes coming from the serial port into acceleration values. Then, it derives the user's arm position from these values. The position is calculated geometrically, without the use of integration, which introduces measurement errors. In fact, integration actually increases the amount of noise, causing a shift of the measures. In order to overcome this problem, the tracking system must be periodically reset [3] or complex filtering algorithm must be applied [2]. A geometric approach based on gyroscopes and accelerometers has been shown to give best results in terms of both accuracy and reliability [26, 32, 33]. In this situation, redundant information coming from the use of more sensing units can be exploited. Accelerometers can be used to compute the inclination of steady objects. As shown in Figure 4, from the knowledge of an acceleration value, a (where $\overline{a} = a \cdot \hat{a}$),[1] collected using a monoaxial accelerometer, the angle θ between the gravity vector $\overline{g}$ and the direction of the accelerometer's axis $\hat{a}$ can be determined by applying the following trigonometric relationship:

$$\cos\theta = \frac{a}{g}, \tag{1}$$

where both g (9.80665 ms^{-2}) and a are known.

Thus, the data collected from one sensor represent the projection of the vector $\overline{g}$ along the direction sensed by the accelerometer. Through similar considerations, we can compute the two angles Ω and θ that determinate the inclination of $\overline{g}$ with respect to the plane π defined by the axes $\hat{x}_1$ and $\hat{y}_1$ of two accelerometers.

Let us call a_x and a_y the accelerometers outputs, and $g_\Pi = \sqrt{a_x^2 + a_y^2}$. As shown in Figure 5, the geometric interpretation of g_Π is the modulus of the projection $\overline{g}_\Pi$ of $\overline{g}$ onto π. Thus, we obtain Ω and θ as follows:

$$\cos\Omega = \frac{a_x}{g_\Pi} \quad \left(\text{or in alternative } sen\Omega = \frac{a_y}{g_\Pi}\right)$$
$$\cos\theta = \frac{g_\Pi}{g}. \tag{2}$$

[1] In general $\overline{v}$ is a vector, v is its modulus with sign, and $\hat{v}$ is its unit vector, so that $\overline{v} = v \cdot \hat{v}$.

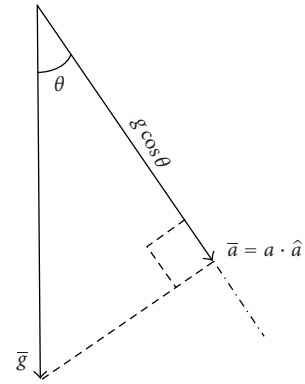

Figure 4: Measuring tilt from one acceleration value.

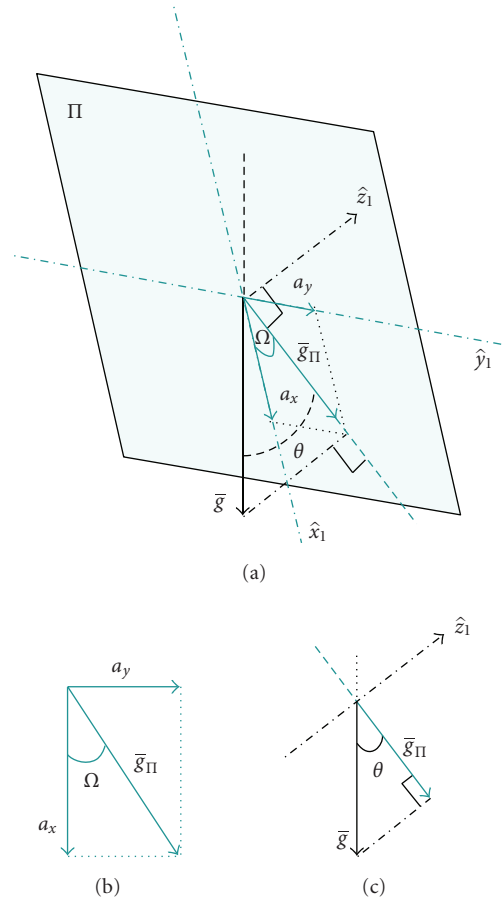

Figure 5: Measuring tilt from two acceleration values.

In real life, redundant information, such as a third acceleration component, can be used to optimize the computation as explained later.

Note that there is a blind rotation, marked with x in Figure 6, which is the one around the vertical axis. In this case, in fact, the rotation is fully contained within the plane orthogonal to gravity and the accelerometer cannot measure any variation in the projection of $\overline{g}$, being always null. All

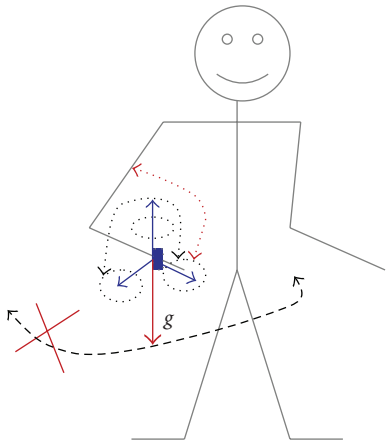

Figure 6: Example of two possible measurable rotation angles.

these algorithms are based on the assumption that the system is in steady state. In both cases, the particular angle that identifies a position is derived as a composition of trigonometric functions of the accelerations in one or more points. For this approach to be applicable, the only hypothesis is that the gravitational acceleration is the main acceleration component.

Accuracy optimization

In practical cases, sensor characteristics affect the accuracy of the measurement. As such, optimization is required. In our case, when the angle between the direction of the gravitational acceleration and the axis of measurement of the accelerometer is small, the accuracy of the sensing unit decreases. To overcome this problem, it is possible to exploit information coming from additional accelerometers to maximize robustness. The sensors should be placed in such a way that the measurement axes are orthogonal to each other. With this setup we can select the most appropriate accelerometer to measure any given orientation and ensure high sensor accuracy for the entire range. If we use orthogonally mounted accelerometers, it is always possible to calculate one of the acceleration components from the other two. In fact the following equation holds:

$$a_x^2 + a_y^2 + a_z^2 = g^2, \tag{3}$$

where a_x, a_y, and a_z are the accelerations measured by orthogonal accelerometers (with respect to z axis). Threshold values have been determined experimentally. They discriminate between the use of the value measured by a certain accelerometer and the use of (3) to compute the same quantity through the other two components. The thresholds chosen guarantee good overall accuracy results, as shown in the last section.

3.2. Motion capture front end

The front end of the MOCA system exploits data acquired and processed by the back end to recognize a set of user movements and to enable interaction with the application. The tracking algorithm described in the previous section provides inclination values. Once a gesture is detected and identified, an application interface layer translates them into application specific commands. The application we target in this work is the navigation in a virtual environment. Our aim was to test the interaction system, thus we tried a subset of the many possible interaction metaphors with the application. For the same reason, more attention was given on the easy and fast deployment of a test bed virtual environment than on the performance of the graphical libraries.

Table 1: Mapping between gestures and encoding.

Gesture	X	Y	Z
Stay	0	0	−1
Turn right	0	−1	0
Turn left	0	1	0
Turn up	1	0	0
Turn down	−1	0	0

Gesture recognition framework

The gesture recognition framework relies on the Java MOCA driver to obtain acceleration values. Information on the users' movements is derived exploiting the recognition algorithm described later. The system is flexible, in that it allows the specification of many different patterns that will be recognized as specific gestures. For the sake of clarity we describe a practical example, namely, a simple gesture recognizer for single arm gestures. In this setup, there is a single sensing unit, tied to the user's wrist. Arm gestures are defined to control navigation in the virtual world described in the following subsection. We defined 5 gestures, namely, "rest" position, "turn right," "turn up," "turn left," "turn down." This is only a small subset of possible gestures. In fact, the system can distinguish between at least 20 different inclinations (and therefore gestures) as demonstrated in Section 4. Relying on tilt provides independence from sensor location and to recover the gesture performed we do not need complex human body models. As explained in the previous section, the system is based on a geometric approach and on the already mentioned condition (steady-state conditions and exploitation of the tridimensional acceleration vector to avoid nonlinearity errors).

After normalization, orientation values are first averaged and then translated into gesture based on the mapping table shown in Table 1. A "1" in the table means that the accelerometer is positioned in the same direction as the reference axis. A "−1" means that the direction is opposite to the reference axis, while a "0" value identifies an orthogonal condition. To decrease sensitivity due to movement inaccuracies, a range of valid acceleration values has been defined for each gesture, instead of identifying with a single value. A *recognition window* corresponds to the extension of this range. The interface system defines also another type of window that will be referred to as *time window*. The time window

size is identified by the sampling period times the number of samples, m, (equal for all acceleration components) used to average sampled data. Since the accelerometer bandwidth is 100 Hz, 10 milliseconds elapse between two consecutive samples, so that time window is proportional to 10 milliseconds through m. The application interface layer derives control events from the gesture recognition process and it maps them in understandable commands for the end-user application.

Navigation environment

The navigation application chosen as a case study is a Java3D virtual world. On the application side, commands corresponding to recognized gestures are processed by an input interface layer responsible for the final conversion into actions understandable by the rendering engine (Figure 7). Java 3D provides a complete environment to define, render, and interact with a navigable virtual world. We exploited the platform independence of JAVA to rapidly build an application prototype which can be easily executed on a number of different hardware targets, ranging from standard PCs with low-cost graphics cards to high-end rendering engines in virtual theaters [34]. The application creates a virtual world based on a 2-dimensional map input with additional annotations to describe textures and various objects placed in the virtual world. Several users can be present in the virtual environment at the same time, navigating the space and interacting with one another. They are represented as simple avatars (3D geometric shapes). The application supports multiple viewpoints for each user and it is capable of being rendered on multiple, distributed visualization systems.

This flexibility allows us to readily deploy the application in a number of different scenarios.

4. EXPERIMENTS

We performed two sets of experiments with different setup. The first set was performed on the basic application using a palmtop computer as the target platform. The aim of the experiments in this context was the characterization of MOCA's static and dynamic properties. The power consumption of the system was also measured. The second block of tests was aimed at evaluating the gesture recognition interface in terms of usability and performance.

4.1. Setup

The test setup for our system is composed of one sensing unit, using a combination of a MEMS biaxial (ADLX202) and a MEMS monoaxial (ADXL105) accelerometers produced by analog devices. The range for the ADLX202 is ± 2 g and for ADXL105 is ± 5 g. The acquisition board is based on an ATMEL AVR AT90S8535 microcontroller, working at 8 MHz, equipped with an internal 10-bit A/D converter and a UART module; on the board we also placed a 16-bit, 200 kSPS A/D converter (AD974 from analog devices). The dimensions of miniaturized components are $25 \times 30 \times 20$ mm for the sensing units and $75 \times 60 \times 15$ mm for the acquisition board.

Figure 7: A set of screen captures of the virtual world, as seen from an LCD while a user is interfacing with it through MOCA.

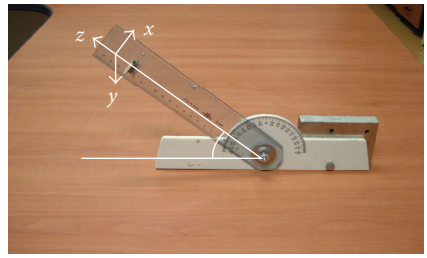

Figure 8: The equipment used for measurements.

The first block of tests was performed using a personal digital assistant (Compaq IPAQ) featuring a RISC processor (Intel XSCALE 400 MHz). The operating system running on the iPAQ is PocketPC 2002 combined with Jeode by Insignia, a Java virtual machine for WinCE, and the COMM API. As an alternative, in order to test the Linux driver, the iPAQ also runs Pocket Linux, familiar distribution, kernel version 2.4.19.

The second set of tests used a laptop PC equipped with a Pentium 4, Windows XP(c) operating system, and a stepper to perform test at different controlled speeds. The system was equipped with a Java virtual machine, the Java3D API, and the binary code to run the virtual environment to test user satisfaction while performing navigation.

4.2. Characterization results

The first experiment shows the accuracy of the system while measuring static orientations. The measurement is taken when the accelerometer is still. To perform accuracy tests, we contrasted the angle computed by our tracking system against the real angle. The latter was measured by mounting the sensing unit on a test fixture (see Figure 8) mainly composed by a goniometer. The darker color represents the uncertainty for each orientation. Here we show a range of 90 degrees with a step of 5 degrees (Figure 9). Similar results were found testing a 360-degree spectrum and repeating measurements placing the device in all possible orientations. The lowest accuracy obtained was 3 degrees. Thus it is possible

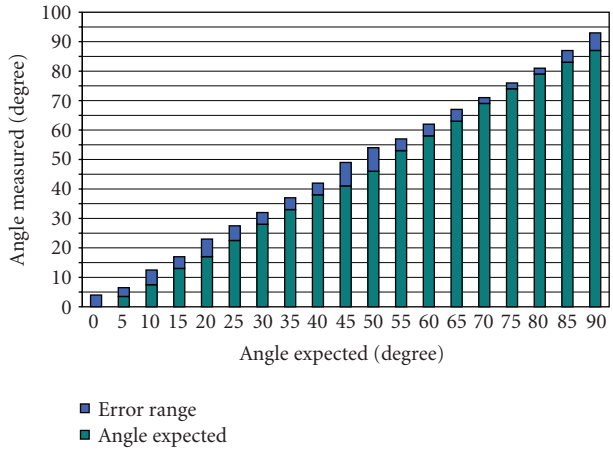

Figure 9: Accuracy as a function of the angle.

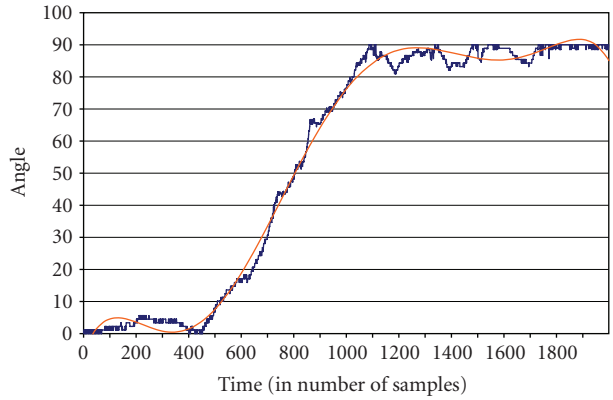

Figure 10: Dynamic performance. (Sampling time 0.5 millisecond)

to distinguish among at least 60 different orientations along a 180-degrees spectrum. The tested accuracy reveals that our system is suitable for a human-computer interface where interaction is performed through discrete events. It is worth noting that these results do not suffer from drift problems, since we do not use integration techniques to compute the angle. No update or compensation is needed as in [2, 8, 9].

Since we exploit a geometric approach, the requirement is that the main contribution to the accelerations must be due to gravity. Therefore this assumption may lead to problems while tracking fast movements, since other acceleration components may pollute the measurement. As such, there is a minimum time that has to be waited so that the system is stable before another inclination can be detected. The shorter this time, the higher the ability of the system to follow fast user movements. To characterize our system from this dynamic perspective, we performed another experimental test, whose results are shown in Figure 10. Here we show the actual measured orientation (y-axis) as a function of the time (x-axis) and we compared it with the expected curve (the broken line). The time is expressed in number of samples, considering a sample rate of 2000 sample/s (0.5 millisecond between samples). This experiment shows that the system is able to follow correctly movements of a stepper motor with rotational speed up to 3 radians/s. The equipment is the same used for the static characterization (Figure 8).

Table 2: Power contributors.

System component	Power consumption (mW) @ Vcc = 5 V
Sensing unit	60
Acquisition board	230
Microcontroller	50
ADC	75
MAX232	75

Finally, we measured the power consumption of the sensing/acquisition unit. We evaluated that the current drawn is 60 mA when the voltage is 6 V. Powering the system with ordinary 750 mAh batteries, the present implementation is continuously operative for at least 12 hours, guaranteeing a sufficient lifetime and freedom of usage. Battery weight is less than 100 g, light enough to allow the system to be worn. Table 2 shows how the power consumption is divided among the different components on the board. It must be stressed that these power consumption measures are related to prototype components. Using miniaturized components will lead to lower power consumption. The MOCA system costs less than $100, making it an extremely cheap and viable system for many end-user applications. The total component and assembly cost for the acquisition board and one sensing unit is around $75, where each sensing unit accounts for around $25 and the acquisition board accounts for $50 (with component bought in small volumes).

Our system is suitable for consumer market and widespread diffusion, while other motion tracking systems are targeted for a niche market and for professional use, often with a cost in excess of $50 000, and requiring greater learning times and calibration procedures.

4.3. Interaction results

Two groups of tests were performed. The first evaluated the user-perceived quality of interaction. The second quantitatively measured the recognition loss with the help of a stepper motor. The recognition window was of the same size for each gesture and was kept constant during the tests. The performance of the system was evaluated as a function of various parameters, namely, speed of movement, settling time, and time window. Settling time refers to the time that a user remains in a certain position. For both groups of tests, a two-step calibration process was performed before starting the measurements. The calibration process is fast and it is needed only once after placing the sensing unit along the body in the first time. During the first step, the sensing unit was positioned so that the biaxial accelerometer was orthogonal to gravity. In the second step the same procedure was repeated for the monoaxial accelerometer. This was used to center the voltage range with respect to the horizontal position.

Figure 11: Wearable setup used in human-based tests.

Table 3: Human-based test results.

$m = 10$ ms	*Slow*	*Spontaneous*	*Fast*
50 ms	100%	100%	95%
100 ms	100%	95%	85%
150 ms	95%	95%	80%
200 ms	95%	100%	75%
250 ms	100%	85%	60%
500 ms	85%	35%	0%
750 ms	100%	20%	0%
1 s	95%	0%	0%
1,5 s	100%	0%	0%

Here m is the length of the time window.

4.4. User experience

A group of ten users (aged between 23 and 30) was chosen. Quality of user interaction was evaluated using two metrics, namely, gesture loss and general user satisfaction. Users were asked to wear one of the sensing units on their wrist or on the back of their hand, depending on their personal preference (see Figure 11). They were then asked to perform a sequence of gestures such as moving their hand up, down, right, and left. To increase the range of movements tested, users were requested to perform the same sequence once at a natural pace (defined as "spontaneous" in the tables and plots), once fast and once slowly. Visual feedback was provided to indicate the result of the recognition process. The test was repeated for each user with different sizes of time window.

Results of the first test are shown in Table 3 and plotted in Figure 12. For each sequence under test the percentage of successful recognitions was determined. Both not recognized and not correctly recognized gestures were considered as errors. As expected, the worst case performance corresponds to the user moving as fast as possible (rightmost column of Table 3). Clearly, if the time window (rows in Table 3) is fixed, the number of samples falling in the recognition window increases as the movement speed decreases, thus helping the recognition. Looking at each column in Table 3, we observe that lower values of time windows provide a better performance, since the system is less sensitive to previous values caused by movements.

The minimum time window represented in Table 3 is 50 milliseconds. Smaller values are not represented since almost all recognitions failed. This is due to the noise introduced by the user stopping in incorrect positions at the end

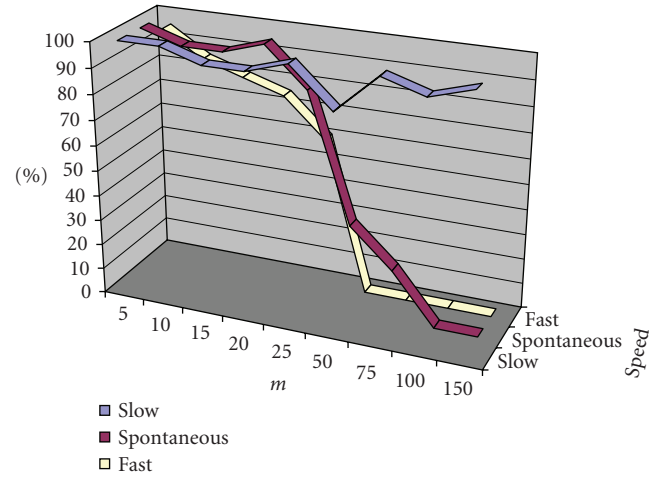

Figure 12: Successful recognition percentage as function of the recognition window and of speed of movement.

of the movement. This noise is not averaged out for very small time windows. It must be observed that the percentage of successful recognitions decreases rapidly as a function of the movement speed. This is due to the contribution of the time users which remained in a given orientation t_{settle}. In fact, this settling time was not forced (the user had no indication on how long she/he should stay in a certain position), but almost all users waited in the final position until feedback was provided by the system, showing some kind of adaptation to t_{settle} to achieve a successful recognition.

A final test was performed to assess general user satisfaction by letting users to freely navigate the virtual environment. The time window was set to a value corresponding to the one that produced the lowest errors for each user in the previous study. This value corresponds to 50 milliseconds in the majority of cases. All the users expressed satisfaction and two of them were positively impressed by the usability of the system and the fast learning curve. In some cases, a little latency was experienced, due to the Java3D-based rendering. One of the users lamented the muscular effort required to use the system.

These experiments clearly suggest that improvements to the system can be obtained by a configurable recognition window. In fact, the recognition window must be established depending on both the characteristics of user gestures and the position of the sensing unit on his or her body. Some of the errors were clearly caused by the fact that some gestures were less natural than others for a particular user. For example, placing the sensing unit on the right hand causes the gesture of turning left to be more difficult than turning right, indicating the need of extending the recognition window for the left position.

4.5. Synthetic experience

The second group of tests was performed by securing the sensing unit to a bar on a revolving support. The bar was

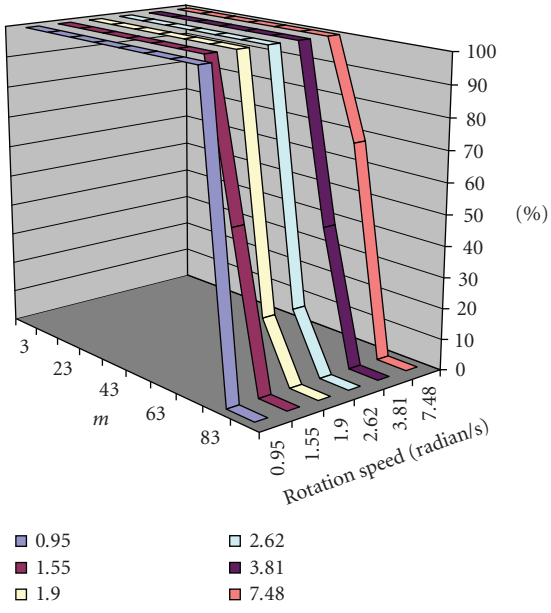

Figure 13: Successful recognition percentage as function of time window (m) and speed of movement for t_{settle} = 900 milliseconds.

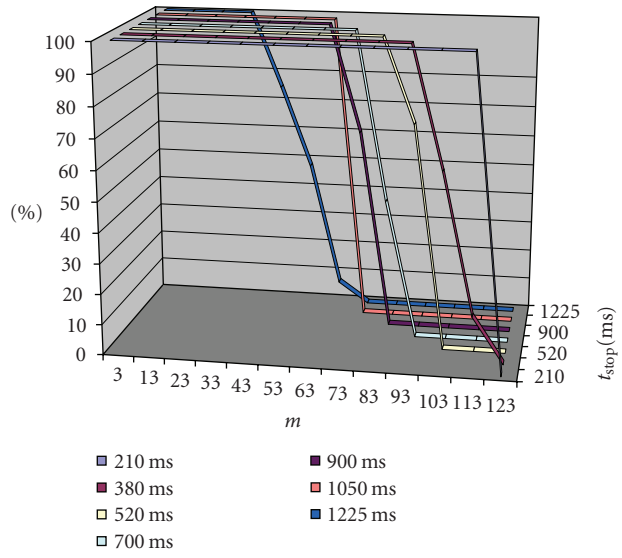

Figure 14: Successful recognition percentage as function of t_{settle} and time window (m) for speed of movement 7,48 radians/s.

moved by a stepper motor whose rotation speed[2] and settling time (t_{settle})[3] could be predeterminated. These experiments eliminated the noise introduced by the users and allowed us to assess the system's basic recognition capabilities.

We performed two sets of experiments. The first one was performed for a constant t_{settle} value. The second set was performed with a constant speed value. Results of the first test, shown in Figure 13 for t_{settle} of 900 milliseconds, confirmed those obtained in the user experience evaluation. It can be observed that the number of successful recognition was in general higher than human-based tests because of the improved synthetic movement precision.

The second set of experiments, shown in Figure 14, outlines the effect of t_{settle} on the recognition process for various time window sizes. Results suggest that small time windows improve the system robustness, especially for short t_{settle}. It must be noted that, compared to the human-based experiments, smaller t_{settle} values are permitted, due to the higher movement precision of the stepper motor.

To summarize, by combining human-based and mechanical tests, we can conclude that low inertial recognition (i.e., small time windows) provides the best recognition performance. The lower bound for the time window should be determined by the level of noise introduced by lack of precision in the human movement. In any case, both user and synthetic experimental results demonstrate the effectiveness of the proposed gesture interface system. Moreover, our solution is robust with respect to drift problems, being based on accelerometers used as inclinometers.

5. CONCLUSION

In this work we presented a low-cost, low-power gesture recognition system based on integrated accelerometers. Acquisition and processing steps have been optimized to be efficient for wearable mobile systems with limited resources. Experimental results show that the accuracy of our solution is suitable for gesture recognition and guarantee its low-power consumption.

Future work will improve MOCA in many directions. First and foremost, the sensing units will be miniaturized and wireless capability will be embedded. Miniaturization and the wireless enhancement will improve the wearability of the system. Another improvement will be on the power dissipation for both boards.

REFERENCES

[1] G. Welch and E. Foxlin, "Motion tracking: no silver bullet, but a respectable arsenal," *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 24–38, 2002.

[2] A. D. Cheok, K. Ganesh Kumar, and S. Prince, "Micro-accelerometer based hardware interfaces for wearable computer mixed reality applications," in *Proceedings of the 6th IEEE International Symposium on Wearable Computers (ISWC '02)*, pp. 223–230, Seattle, Wash, USA, October 2002.

[3] I. Oakley and S. O'Modhrain, "Tilt to scroll: evaluating a motion based vibrotactile mobile interface," in *Proceedings of the 1st Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC '05)*, pp. 40–49, Pisa, Italy, March 2005.

[4] G. B. I. P. Regenbrecht and M. Billinghurst, "A cable-less interaction device for AR and VR environments," in *Proceedings of the 2nd International Symposium on Mixed Reality (ISMR '01)*, pp. 151–152, Yokohama, Japan, March 2001.

[5] V. Tanriverdi and R. J. K. Jacob, "Interacting with eye movements in virtual environments," in *Proceedings of the*

[2] Values tested as rotation speed are (radian/s) 0,95; 1,10; 1,27; 1,55; 1,90; 2,62; 3,81; 7,48.

[3] Values tested as t_{settle} are 210, 380, 520, 700, 900, 1050, 1225.

Conference on Human Factors in Computing Systems 'The Future is Here' (CHI '00), pp. 265–272, The Hague, The Netherlands, April 2000.

[6] R. Headon and R. Curwen, "Movement awareness for ubiquitous game control," *Personal and Ubiquitous Computing*, vol. 6, no. 5-6, pp. 407–415, 2002.

[7] R. Barbieri, E. Farella, L. Benini, B. Riccò, and A. Acquaviva, "A low-power motion capture system with integrated accelerometers," in *Proceedings of the 1st IEEE Consumer Communications and Networking Conference (CCNC '04)*, pp. 418–423, Las Vegas, Nev, USA, January 2004.

[8] A. Wilson and S. Shafer, "XWand: UI for intelligent spaces," in *Proceedings of the New Horizons Conference on Human Factors in Computing Systems (CHI '03)*, pp. 545–552, ACM Press, Ft. Lauderdale, Fla, USA, April 2003.

[9] J. Skaloud and K.-P. Schwarz, "Application of inertial technology to underground positioning: the Soudan mine shaft survey," *Zeitschrift für Vermessungswesen*, vol. 125, no. 8, pp. 292–299, 2000.

[10] E. Farella, A. Pieracci, and A. Acquaviva, "Design and implementation of WiMoCA node for a body area wireless sensor network," in *Proceedings of Systems Communications*, pp. 342–347, Montreal, Quebec, Canada, August 2005.

[11] M. Pantic and L. J. M. Rothkrantz, "Toward an affect-sensitive multimodal human-computer interaction," *Proceedings of the IEEE*, vol. 91, no. 9, pp. 1370–1390, 2003.

[12] J. K. Perng, B. Fisher, S. Hollar, and K. S. J. Pister, "Acceleration sensing glove (ASG)," in *Proceedings of the 3rd International Symposium on Wearable Computers (ISWC '99)*, pp. 178–180, San Francisco, Calif, USA, October 1999.

[13] R. Headon and G. Coulouris, "Supporting gestural input for users on the move," in *Proceedings of the IEE Eurowearable*, pp. 107–112, Birmingham, UK, September 2003.

[14] R. W. Hogg, A. L. Rankin, S. I. Roumeliotis, et al., "Algorithms and sensors for small robot path following," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '02)*, vol. 4, pp. 3850–3857, Washington, DC, USA, May 2002.

[15] W. Ponweiser, M. Ayromlou, M. Vincze, C. Beltran, O. Madsen, and A. Gasteratos, "RobVision: vision based navigation for mobile robots," in *Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI '01)*, pp. 109–114, Baden-Baden, Germany, August 2001.

[16] C. Tarin, H. Brugger, R. Moscardo, B. Tibken, and E. P. Hofer, "Low level sensor fusion for autonomous mobile robot navigation," in *Proceedings of the 16th IEEE Instrumentation and Measurement Technology Conference (IMTC '99)*, vol. 3, pp. 1377–1382, Venice, Italy, May 1999.

[17] P. Lang, A. Kusej, A. Pinz, and G. Brasseur, "Inertial tracking for mobile augmented reality," in *Proceedings of the 19th IEEE Instrumentation and Measurement Technology Conference (IMTC '02)*, vol. 2, pp. 1583–1587, Anchorage, Alaska, USA, May 2002.

[18] M. Ribo, P. Lang, H. Ganster, M. Brandner, C. Stock, and A. Pinz, "Hybrid tracking for outdoor augmented reality applications," *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 54–63, 2002.

[19] Y. Yokokohji, Y. Sugawara, and T. Yoshikawa, "Accurate image overlay on head-mounted displays using vision and accelerometers," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '99)*, vol. 4, pp. 3243–3248, Detroit, Mich, USA, May 1999.

[20] R. Zhu and Z. Zhou, "A real-time articulated human motion tracking using tri-axis inertial/magnetic sensors package," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 12, no. 2, pp. 295–302, 2004.

[21] A. R. Golding and N. Lesh, "Indoor navigation using a diverse set of cheap, wearable sensors," in *Proceedings of the 3rd International Symposium on Wearable Computers (ISWC '99)*, pp. 29–36, San Francisco, Calif, USA, October 1999.

[22] H. Junker, P. Lukowicz, and G. Troster, "PadNET: wearable physical activity detection network," in *Proceedings of the 7th IEEE International Symposium on Wearable Computers(ISWC '03)*, pp. 244–245, White Plains, NY, USA, October 2005.

[23] T. Harada, H. Uchino, T. Mori, and T. Sato, "Portable orientation estimation device based on accelerometers, magnetometers and gyroscope sensors for sensor network," in *Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI '03)*, pp. 191–196, Tokyo, Japan, July-August 2003.

[24] G. M. Lyons, K. M. Culhane, D. Hilton, P. A. Grace, and D. Lyons, "A description of an accelerometer-based mobility monitoring technique," *Medical Engineering and Physics*, vol. 27, no. 6, pp. 497–504, 2005.

[25] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *Proceedings of the 2nd International Conference on Pervasive Computing (PERVASIVE '04)*, pp. 1–17, Vienna, Austria, April 2004.

[26] S. Kurata, M. Makikawa, H. Kobayashi, A. Takahashi, and R. Tokue, "Joint motion monitoring by accelerometers set at both near sides around the joint," in *Proceedings of the 20th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEMBS '98)*, vol. 4, pp. 1936–1939, Hong Kong, October-November 1998.

[27] W.-C. Bang, W. Chang, K.-H. Kang, E.-S. Choi, A. Potanin, and D.-Y. Kim, "Self-contained spatial input device for wearable computers," in *Proceedings of the 7th IEEE International Symposium on Wearable Computers (ISWC '03)*, pp. 26–34, White Plains, NY, USA, October 2003, http://www.measurand.com/.

[28] C. Park, J. Liu, and P. H. Chou, "Eco: an ultra-compact low-power wireless sensor node for real-time motion monitoring," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, pp. 398–403, Los Angeles, Calif, USA, April 2005.

[29] http://www.intersense.com/.

[30] http://www.xsens.com/.

[31] A. Paiva, R. Prada, R. Chaves, et al., "Towards tangibility in gameplay: building a tangible affective interface for a computer game," in *Proceedings of the 5th International Conference on Multimodal Interfaces (ICMI '03)*, pp. 60–67, ACM Press, Vancouver, BC, Canada, November 2003.

[32] J. Lee and I. Ha, "Sensor fusion and calibration for motion captures using accelerometers," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '99)*, vol. 3, pp. 1954–1959, Detroit, Mich, USA, May 1999.

[33] http://www.sgi.com/.

[34] T. Sakaguchi, T. Kanamori, H. Katayose, K. Sato, and S. Inokuchi, "Human motion capture by integrating gyroscopes and accelerometers," in *Proceedings of the IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI '96)*, pp. 470–475, Washington, DC, USA, December 1996.