A smartphone-based fall detection system

Stefano Abbate[a], Marco Avvenuti[a], Francesco Bonatesta, Guglielmo Cola[a], Paolo Corsini[a], Alessio Vecchio[a,*]

[a]*Dip. di Ingegneria dell'Informazione, University of Pisa*
Largo L. Lazzarino 1
56122-Pisa, Italy

Abstract

Falls are a major cause of injuries and hospital admissions among elderly people. Thus, the caregiving process and the quality of life of older adults can be improved by adopting systems for the automatic detection of falls. This paper presents a smartphone-based fall detection system that monitors the movements of patients, recognizes a fall, and automatically sends a request for help to the caregivers. To reduce the problem of false alarms, the system includes novel techniques for the recognition of those activities of daily living that could be erroneously misdetected as falls (such as sitting on a sofa or lying on a bed). To limit the intrusiveness of the system, a small external sensing unit can also be used for the acquisition of movement data.

Keywords:
Pervasive healthcare, Fall detection, Wearable sensors

1. Introduction

A fall event is one of the main factors that influence the physical and psychological health of an elderly person. Injuries related to falls include physical damages like skin abrasions, bone fractures and general connective tissue lesions [1, 2]. A fall has also dramatic psychological consequences, since it drastically reduces the self-confidence and independence of affected people. This may contribute to future falls with more serious outcomes or it may lead to a decline in health. Other frequent consequences include early nursing home admission and continuous fear of falling, lowering the quality of everyday life [3]. The consequences of a fall event depend also on the *long-lie* period, i.e. the time interval during which the person remains involuntarily on the ground after the fall [4]. Therefore, it is of fundamental importance to provide quick support to the injured people as soon as a fall happened.

The simplest solution to the fall detection problem consists of providing people with a Personal Emergency Response System (PERS), a small, light-weight and battery-powered device with a "help" button that can be carried on a belt, in a pocket, on a necklace or on a wrist band.

*Corresponding author.
Email addresses: stefano.abbate@imtlucca.it (Stefano Abbate), m.avvenuti@iet.unipi.it (Marco Avvenuti), guglielmo.cola@iet.unipi.it (Guglielmo Cola), p.corsini@iet.unipi.it (Paolo Corsini), a.vecchio@ing.unipi.it (Alessio Vecchio)

This kind of device also embeds a radio transmitter which is able to connect to the user's home telephone and to dial preselected numbers in case of emergency. Many of these systems have been successfully deployed in several countries and require almost no configuration [5]. However, they suffer from a major issue: the need for the user to press a button. Unfortunately, it is common that after a fall a person is unable to perform even this simple action, for example because of a loss of consciousness or because the final lying position could prevent the victim from being able to reach the button.

Thus, in the last few years, research about systems for the *automatic* detection of falls gained momentum, pushed by the growing number of elderly citizens in a large fraction of the world [6, 7, 8, 9, 10, 11]. The techniques for the automatic detection of falls can be substantially divided into two categories [12]. The first category includes the approaches based on instrumenting the environment; examples include equipping rooms with cameras able to track the movements of people or placing pressure sensors in specific areas (e.g., in the vicinity of beds). The second category includes techniques based on wearable sensors: accelerometers and/or gyroscopes are used to collect kinematic information about the monitored person and then to detect falls. The advantage of using wearable sensors is that almost no installation or set-up is required and the system is immediately available for deployment. Moreover, the area of operation is not limited to the instrumented spaces as the system can be carried by users wherever they go. This enlarges the number of users and situations, ideally including all the activities that expose people to long periods of being alone with a high risk of falling. Nevertheless, the user is required to wear at least a device and this can pose some intrusiveness and usability concerns. In other situations, automatic fall detection can be integrated within the monitoring system dedicated to specific pathologies [13].

Many automatic fall detection systems suffer from the problem of false alarms, caused by some fall-like activities of daily living (ADLs), such as sitting on a sofa or lying on a bed. For this reason, in our approach to fall detection we devoted a special attention to the study of the acceleration signal produced by fall-like ADLs and to the design of novel filtering techniques [14]. In this paper, we describe the design rationale and the implementation of a fall detection system based on wearable sensors. The system relies on commercially available smartphones and is capable of automatically sending an alarm message to the caregivers in case of fall. The acquisition of kinematic data can be carried out either using the accelerometer available on many smartphones or using an external sensing unit. The usability of the system has been confirmed by a set of interviews with some aged people, while its performance, in terms of precision and recall, has been evaluated both in lab sessions and through continuous monitoring of three subjects (including indoor and outdoor activities). A comparison with similar existing fall detection techniques is also reported.

2. System design

2.1. Requirements and architectural guidelines

A fall detection system can be useful for people working or doing recreational activities in isolated places with high risk of falls, such as country workers, mushroom hunters, or skiers. However, the category of people that can benefit more significantly from a fall detection system is the one of elderly. In fact, in the last years life expectancy has increased making a larger fraction of the population more prone to falls. Unfortunately, the injuries due to falls are a major cause of hospitalization, disabilities and even deaths for elderly people.

Thus, the target user of our fall detection system is an elderly person who frequently turns out to be alone for an extended period of time. An effective fall detection system for such users must address the following requirements: *automation*, to be able to send an alarm without the need for the user to press a button; *promptness*, to provide quick help and avoid worsening of health condition; *reliability*, concerning the capability of detecting fall events while filtering fall-like ADLs; *communication*, in order to be always connected and able to alert the care givers, relatives or friends; *usability*, for facilitating users' acceptance.

We argue that, in order to fulfill the mentioned requirements, a fall detection system should be designed based on the following guidelines:

1. Detection of falls should be carried out using only acceleration-based information. Previous work demonstrated that acceleration is the most reliable information that can be used in detecting a fall, while other kinematic data, such as angular velocity, is less relevant [15].
2. Usability is strongly influenced by the number of wearable sensors and by their placement on the user's body. For this reason, detection should be performed using a single sensor. Waist and head provide the most relevant acceleration data for the purpose of fall detection [10]. However, waist represents the most comfortable position for the user. Also, placing the accelerometer at the waist, close to the center of gravity of the user's body, makes the sensor less sensible to spurious movements.
3. Some fall detection systems use posture information to improve their precision [11, 16, 17]. Posture can be calculated either assuming a known orientation of the axes of the accelerometer with respect to the user's body, or using two or more sensors. As both approaches reduce the usability of the system, solutions that do not need posture information should be preferred.
4. Still for usability reasons, the fall detection algorithm should work only with the magnitude of acceleration and not with the values along each of the three accelerometer's axes, as this, again, would require a known and fixed orientation of the device with respect to the user's body.
5. The fall detection algorithm has to be self-learning; in this way, it can automatically adjust the parameters of operation according to the specific characteristics of the user (height, weight, movement speed, etc.).
6. A human-machine interface, even if rudimentary, is mandatory for two reasons: *i*) a stop button is necessary to avoid sending a request for help in case of false alarms; *ii*) to provide a feedback to the self-learning detection algorithm.

A wearable device with a small form factor seems to adhere to such guidelines. To allow the system to operate outdoors, even far from any building, communication should rely on the existing cellular network. Clearly, smartphones are good candidates: they are now an effective computing platform and include a variety of sensing subsystems, such as accelerometers and gyroscopes (normally used to manage the orientation of the screen and to use the position of the device as a form of input). At the same time, providing a new functionality, such as fall detection, through a familiar device would greatly facilitate an extensive deployment of the application.

We designed and implemented a fall detection system that satisfies the given requirements by using a smartphone attached to the user's belt. An application, with self-learning properties, continuously collects acceleration data through the built-in accelerometer, applies the fall detection algorithm, and sends an alarm to a set of predefined contacts when a fall is detected. Interaction between the user and the application takes place using the standard phone interface.

Questions
Gender and age of the interviewee.
How do you judge the project's usefulness and benefits?
Do you already own a smartphone? How do you use it?
How do you rank the usability of the smartphone plus external unit solution?
Would you prefer a solution where only the external sensing unit is used or a solution based only on a smartphone?
In the latter case, would you be annoyed if you were obliged to hang the smartphone on your belt?
Do you feel intrusive and/or difficult to use the request for suggestions in case of false alarm?
Would you prefer a completely automatic solution?

Table 1: A small survey

A smartphone-based solution still has some drawbacks: not all the smartphones embed an accelerometer and, even if it is available, it may be not adequate for the fall detection purposes (3 axes, acceleration range $\geq \pm 4g$, sampling rate $\geq 50Hz$). Moreover, a usability obstacle must be considered: not all the people carry their mobile phone in a fixed position or wish to wear a device of the size of a smartphone; they would rather carry a small and non invasive sensor, more versatile and easy to wear, being free to put their smartphones wherever they want.

To address the above issues, we added the possibility of collecting acceleration data also through an external accelerometer. The external sensor must be placed at the user's waist and communicates wirelessly with the smartphone (via Bluetooth), where data is elaborated. This way, the user can carry the smartphone wherever he/she wants, with the only constraint of keeping it within the radio communication range.

2.2. End users interview

As discussed in the previous section, usability is a critical design issue of a wearable fall detection system. Even the most reliable algorithm can be useless if the overall system does not address the users' acceptance of a number of issues like ease of use, perceived effectiveness, privacy, cost.

A common and effective methodology to evaluate a system's usability is by interviewing end-users. To evaluate our approach, we showed a prototype of the system to a small group of people, 6 men and 4 women between 60 and 82, and we asked them to judge its usability. The survey focused on the users' reaction to a wearable system and their willing to use it in daily life. Also, the users' confidence with technology and their preferences about the proposed solutions were investigated, together with the problem of sensor positioning. Lastly, the users' need for interactivity and feedback was analyzed.

After a briefing about the project and a simple demonstration using the smartphone and the external sensing unit, the questions listed in Table 1 were posed to the interviewees. Despite the small size of the sample group, the following considerations emerged quite clearly:

- All the interviewees agreed on the benefits and importance of an automatic fall detection system. Some of them also reported about injuring falls occurred to close relatives, which seriously affected their quality of life, because of late arrival of medical teams. All interviewees owned a mobile phone, most of them using it for both voice and text messages.

- Men consider comfortable wearing a device on their belt. They like having the possibility of switching between the external and the built-in sensor depending on the situation. For example, while at home they would prefer to wear a small sensing unit rather than a fully fledged mobile phone. But sometimes, especially outdoors, carrying just a single device may be easier, even if it is needed to place it on one's belt.

- On the contrary, women do not consider comfortable the idea of wearing a device on their belts, for the evident reason that they seldom use belts. However, they found reasonable the idea of attaching a small sensing unit to an elastic band and place it at the waist under their dresses. Also, the sensor would be invisible, thus improving the users' privacy.

- As far as the user interface is concerned, interviewees unanimously agreed on the need of having a button to prevent false alarms. Also, they expressed the opinion that interacting with the system, with the purpose of giving feedback, is a small burden if compared with the advantage of increasing the user's confidence and trustworthiness on a continuously updated system.

As a final remark, even though the small size of the sample group did not allow us to draw general conclusions with high confidence, the replies received from the interviewees were definitely of great help in identifying users' requirements and wishes. Some related studies are reported in Section 7.2.

2.3. Basic fall detection

In our system, a *fall-like event* is defined as an acceleration peak of magnitude greater than $3g$ followed by a period of 2500 ms without further peaks exceeding the threshold. The accelerometer sampling rate has been set at 50 Hz, a tradeoff between resolution and power consumption [18]. Threshold values around $3g$ (in general, ranging from $2.5g$ to $3.5g$) have been widely used in other fall detection systems [8, 11]. The $3g$ value is small enough to avoid false negatives, since real falls are likely to present an acceleration pattern containing a peak that exceeds such a value.

After a fall, the unfortunate usually remains lying on the ground. According to our experiments, in case of fall the body becomes inactive by 1000 ms after the peak that exceeds the $3g$ threshold. Such an interval is due to the fact that the first peak is usually followed by other smaller peaks generated by different parts of the body impacting the ground at subsequent times. During inactivity, the acceleration magnitude is stable around $1g$.

Based on this consideration, a simple *Activity Test* can be performed to avoid some false alarms: a fall-like event is discarded if the acceleration magnitude reveals a significant body activity starting from 1000 ms after the first peak. The Activity Test can be safely performed in the $[1000, 2500]$ ms interval after the first peak. A possible implementation of the Activity Test is discussed in Section 4.3.

This basic fall detection technique can achieve up to 100% sensitivity. Unfortunately, some ADLs can produce peaks exceeding the $3g$ threshold, followed by a period where the person is almost inactive. Examples of such activities include: sitting on a sofa, lying on a bed, walking/running followed by a stop, or simply unintentionally hitting the sensor with a hand or a bag. In all these cases, basic fall detection is likely to confuse the event with a fall, leading to a great number of false positives.

In order to further reduce the number of false alarms, we built a classification engine that considers a fall-like event as a real fall only if its pattern does not correspond to the typical patterns of the most common ADLs. This approach was suggested by experimentally observing that the acceleration traces of several ADLs have more regular patterns than falls. Intuitively, this is due to the fact that such events originate from controlled movements, thus different individuals produce similar patterns and different iterations show little variations. On the contrary, trying to

recognize falls is more difficult, as it is an ill-defined process and a thorough description is not easy to achieve [14].

In the end, we identified a set of fall-like activities and we noticed that they are characterized by consistent acceleration-based properties. The classification engine receives as input the data corresponding to a fall-like event and produces as output a membership value for each of the different ADL classes and one membership value for the class of falls. This is achieved using soft computing techniques to make the system resilient to the variations introduced by the different executions and the individual body characteristics. If one of the memberships associated to the ADL classes is higher than the membership associated to the Falls class, the fall-like event is considered as a normal ADL and the alarm is not raised.

3. Implementation

This section describes the hardware platform used to build a prototype of the system and the software running on the smartphone and on the external sensing unit when provisioned.

3.1. Hardware

We implemented the system using a HTC Google Nexus One smartphone, which features hardware characteristics suitable to our purpose. This smartphone embeds an accelerometer (BMA150 by Bosch Sensortec) which provides 3-axial measurements in the ranges $\pm 2g/\pm 4g/\pm 8g$.

A Shimmer2 wireless sensor, produced by Shimmer Research [19], has been selected as the external sensing unit. This device incorporates a 3-axis accelerometer (MMA7260Q by Freescale Semiconductor) which provides measurements in the ranges $\pm 1.5g/\pm 2g/\pm 4g/\pm 6g$ per axis. The Shimmer Platform also features a 3-axis gyroscope but, given its high current drain and the small benefits that can be achieved by measuring the angular velocity, it has not been used for our purposes [20]. The device includes 802.15.4 and Bluetooth radios. The latter has been used for communication with the smartphone.

3.2. Software

The software on the smartphone side runs on top of the Android operating system, while the software running on the external sensing unit has been written using the TinyOS/nesC platform [21].

3.2.1. Sensing process

The sensing process can be described as a finite state machine (shown in Figure 1). Initially, the machine stays in the *Sampling* state until a *threshold peak* is detected (acceleration magnitude $\geq 3g$). After that, the machine moves to the *Post-peak* state and starts waiting for a *bouncing timer* of 1000 ms, during which no further threshold peaks must be seen.

When the bouncing interval is elapsed, the machine moves to the the *Post-fall* state and starts a *post-fall timer* of 1500 ms. During this interval, the detection of a new threshold peak makes the machine return to the Post-peak state. When the post-fall timer fires, the *Activity test* introduced in Section 2.3 is performed. If the event does not pass the test, i.e. there is a high level of body activity immediately after the supposed fall, the event is considered a false alarm and the system returns to the Sampling state. Otherwise, the detection of a *fall-like event* is signaled to the application.

The sensing process is executed either on the smartphone or on the external sensing unit, depending on the adopted configuration.

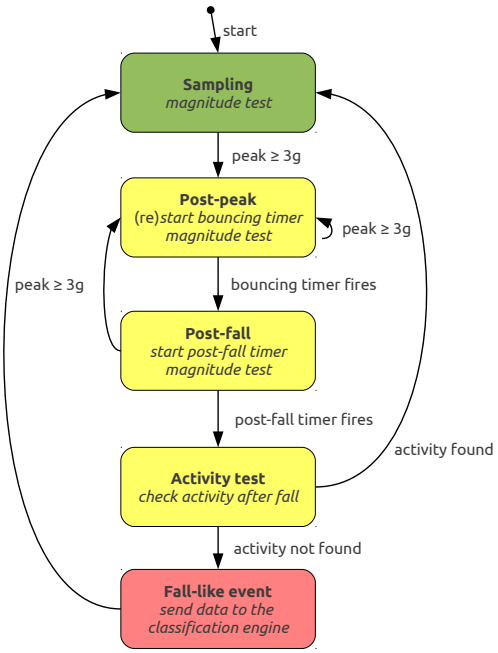


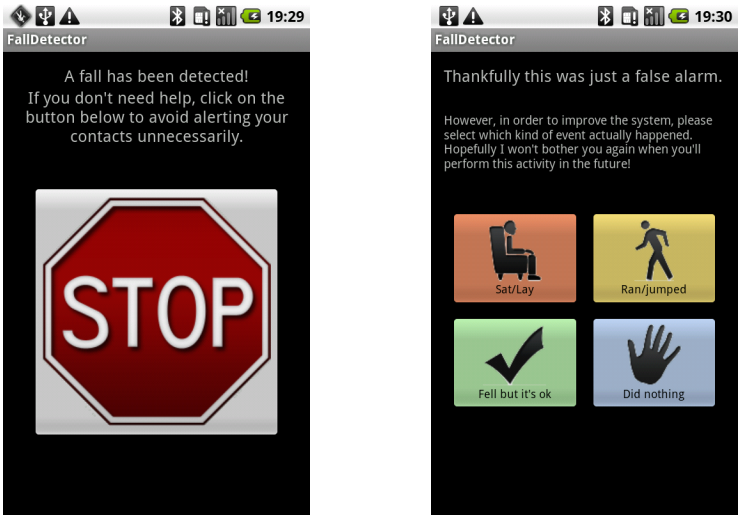
Figure 1: Sensing process as a finite state machine

3.2.2. Smartphone

On the smartphone side, the application is made of four major components: *Background Service*, *Classification Engine*, *Notification System* and *Graphical User Interface*.

Background Service. The monitoring service runs in the background. Data coming from the sensor, either internal or external, is forwarded to the Classification Engine, which in turn classifies the event as a fall or as an ADL. If the external sensing unit is used, the service takes care of starting the communication and of handling connection losses. If the event has been classified as a fall, the service invokes the Notification System, which informs the user about the detected fall. The user can stop the notification if a false alarm occurred. In addition, the Background Service periodically gets a location fix from the phone's GPS antenna or cellular network and passes it to the Notification System when a fall occurs.

Classification Engine. Whenever a fall-like event is detected, the Background Service activates the Classification Engine providing it with the array of acceleration data related to the fall-like event. For the event, the Classification Engine calculates a membership value to the ADL classes and to the Falls class. The classification method is described in the next section. If the membership value corresponding to the Falls class is greater than all the membership values related to the ADL classes, then the fall-like event is classified as a fall and the Notification System is started. Otherwise, the fall-like event is considered as a false alarm and discarded.



(a) Stop button

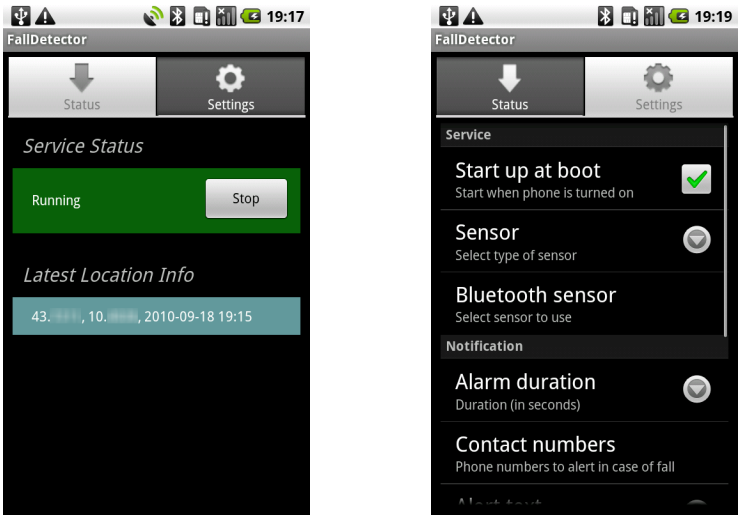
(b) Suggestions screen

Figure 2: Notification screenshots

The Classification Engine is also in charge of receiving feedback from the Notification System to improve the classification accuracy.

Notification System. The Notification System is responsible for alerting the caregivers whenever a fall is detected. Each time a fall-like event is classified as a real fall by the Classification Engine, an acoustic alarm is emitted continuously for 30 seconds. At the same time, a notification appears on the Android's notification bar and a foreground screen is shown. The screen simply contains some informative text and a flashing "stop" button (Figure 2(a)), that can be used to deactivate the alarm. If the user does not deactivate the alarm by a given time, the stop screen is dismissed and a text message is sent to one or more specified contacts. The message contains basic details about the event and information about the last known location. If the "stop" button is pressed, the user can provide some feedback to the Classification Engine. Through another screen, shown in Figure 2(b), the user specifies the type of activity that raised the false alarm. When any of these buttons is selected, the acceleration pattern is archived together with information related to that event.

Graphical User Interface. As shown in Figure 3, the user interface is made of two tabs. The first tab shows status information about the Background Service and provides a way to start/stop it. It also shows information about the last location, time and date, if available. The second tab provides a way to set up all necessary system settings. The most important options are: *Start up at boot*, to autostart the system when the phone is turned on; *Sensor*, lets the user switch between the internal and the external sensor; *Bluetooth sensor*, provides a screen for selecting an external sensing unit; *Alarm duration*, set the time available for stopping an alarm before sending the request for help (default 30 seconds); *Contact numbers*, to enter the phone numbers to alert in case of fall (contacts can be added, deleted, modified or selected from the main list of the phone); *Alert text*, content of the message sent in case of fall; *Send location*, location information can be



(a) Status tab (b) Settings screen

Figure 3: The Graphical User Interface

appended to the text message; *Improve system*, if the user wishes to give suggestions after a false alarm.

3.2.3. External sensing unit

The software running on the external sensing unit manages the communication with the smartphone and takes care of the sensing process. When a fall-like event is detected, the software running on the sensing unit checks the connectivity status and, if a connection is already established, it sends the relevant data to the smartphone. In case of communication errors, the anomalous status is signaled by using a led. When the communication is successfully completed, the application re-enters the *Sampling* state.

It should be noted that the Activity Test can be executed on the external sensing unit rather than on the smartphone. This way, a large number of events can be discarded by the sensing device itself, thus reducing the amount of communication with the smartphone and improving the battery lifetime. Other power saving techniques can be put into operation in case of excessive consumption due to communication [22, 23]. Customization of the sensing process could also be achieved through specific techniques [24].

4. Detection of real falls and filtering of fall-like ADLs

The building blocks of the classification engine are shown in Figure 4. As said in the previous section, fall-like events that are not discarded through the Activity Test are forwarded to the Classification Engine. The input to the engine is a vector of acceleration magnitude values collected in the interval $[-2200, 1000]$ *ms*, centered at the instant of the peak exceeding the $3g$ threshold. These values are fed into a *Feature Extractor* whose task is to reduce the number

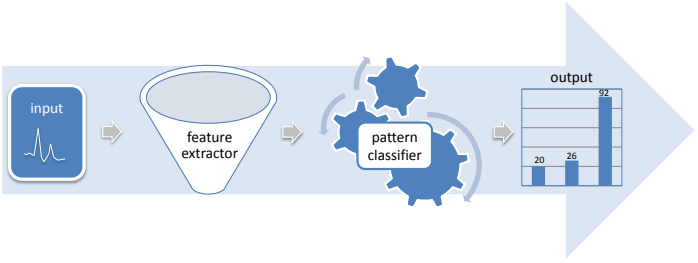


Figure 4: The Classification Engine

of the input values. This is carried out by taking advantage of domain knowledge in order to cut away unnecessary information. In our case, we used eight features that characterize fall-like events originated from ADLs. The *Pattern Classifier* takes such values and returns a membership value for each ADL class and the Falls class. Finally, the event is associated to the class with the highest membership.

The conceptual borderline between feature extraction and pattern classification is somewhat arbitrary. In fact, a perfect feature extractor would produce features that are directly membership values, or values from which the desired output can be trivially derived. This kind of extractor would require no classifier. Conversely, a perfect classifier would work without a sophisticated feature extractor [25]. From a practical point of view, the task of feature extraction depends on the problem and on the domain to a much greater extent than classification. In fact, a good feature extractor for a given problem can be of little use in another domain. The structure of a classifier, instead, can be easily adapted to new problems and reused by changing few configuration parameters.

4.1. Activities of daily living

We identified three ADL classes that may be confused with real falls: Sitting/Lying, Jumping/Running/Walking and Hitting the sensor. Then, we collected and studied the acceleration patterns of such ADLs. Here we briefly report the main characteristics of events belonging to each class.

4.1.1. Sitting/Lying

Sitting or lying on soft or hard surfaces can generate a fall-like event when a person, due to a reduced muscular force or tiredness, moves in a less controlled way and approaches the surface quite quickly. Different patterns are produced by elastic, soft and hard surfaces.

When sitting or lying on an elastic surface like a bed, a fall-like event is unlikely to be produced. However, in certain circumstances an acceleration peak could be present, due to the fact that the body has a relatively high kinetic energy. After the first contact, energy is dissipated very slowly so that the acceleration trace shows several smooth peaks. As shown in Figure 5, there is a single peak higher than $3g$ followed by some oscillations. A similar behavior can be seen in the case of soft surfaces, like when sitting on a sofa. In this case, the first peak is generally very smooth and it is followed by one or two smaller peaks.

Sitting on a hard surface like a chair or an armchair frequently produces a fall-like event, as a sharp acceleration peak is generated when the body impacts the surface. The kinetic energy is quickly absorbed and a quick stabilization of the acceleration magnitude can be seen.

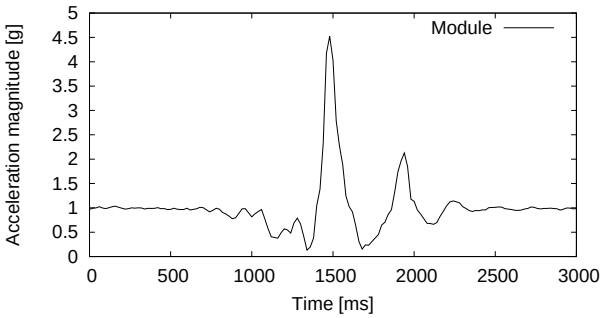


Figure 5: Sitting on elastic surface

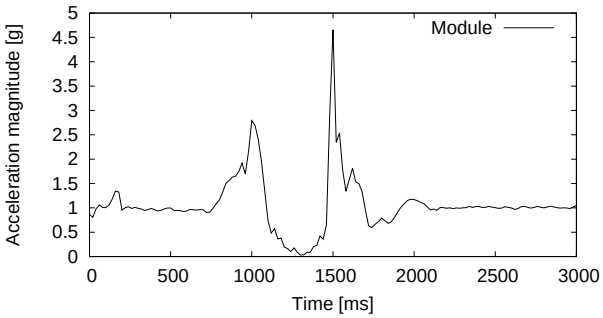


Figure 6: Jumping

To summarize, the distinguishing feature of sitting or lying on elastic/soft surfaces is that *acceleration traces show relatively slow oscillations*, while with hard surfaces *there are few oscillations that stabilize quite quickly*.

4.1.2. Jumping/Running/Walking

The Activity Test can strongly reduce the number of false alarms produced during actions that have a significant duration, such as walking or running. However, when the user stops moving, a fall-like event can still be triggered.

Running and jumping activities easily lead to relatively high acceleration magnitude peaks, which exceed the $3g$ threshold. Jumping has a very peculiar acceleration pattern that may ease its identification. As shown in Figure 6, a first smooth peak, usually lower than $3g$ and corresponding to the time the person starts to leap, is followed by a relatively long free fall phase during which the acceleration magnitude is close to $0g$. Finally, landing produces a sharp and higher

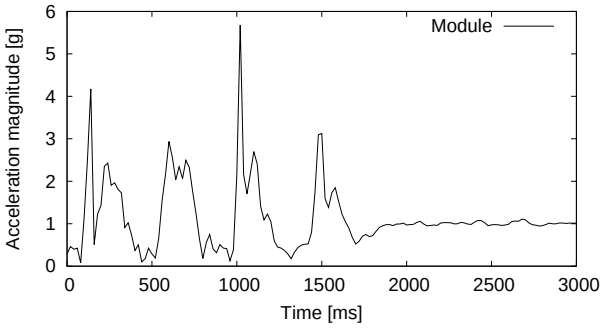


Figure 7: Running (followed by a stop)

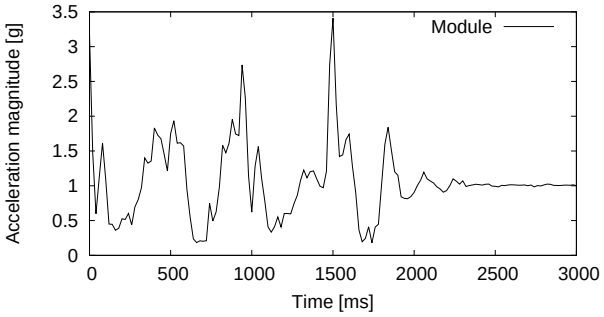


Figure 8: Walking (followed by a stop)

peak (or two rapidly subsequent and partially overlapped sharp peaks if the feet do not land at the same time) followed by a quick stabilization to $1g$.

Running (Figure 7) can be modeled as a sequence of small jumps, where free fall phases between steps are shorter, and landing peaks are often made of two or more overlapped sub-peaks, due to the landing foot starting a new leap.

Walking (Figure 8) is made of a sequence of steps, and produces an acceleration pattern that has some similarities with running, since it presents a regular sequence of peaks. As expected, the peaks are lower and the free-fall phase between steps is less evident. Nevertheless, a walk characterized by a relatively high pace may easily lead to fall-like events when the user stops moving.

4.1.3. Hitting the sensor

Hitting the sensor unintentionally with a hand or with an object, such as a bag, can produce one or more high and sharp acceleration peaks, as it is shown in Figure 9. These events proved

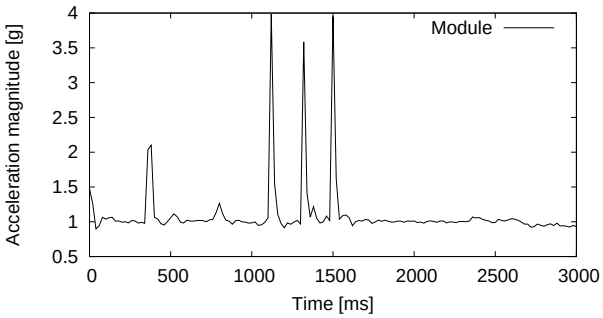


Figure 9: Hitting the sensor

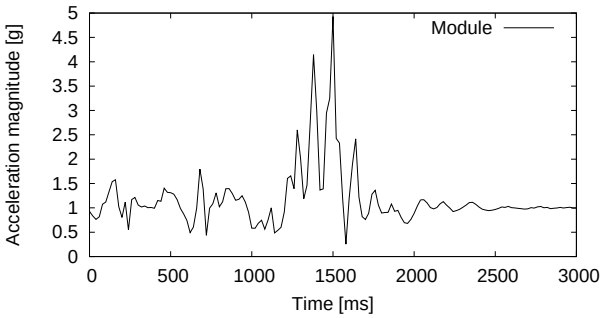


Figure 10: Fall forward

to be very common during a typical day, and may lead to a large number of false alarms. In this case, the acceleration pattern presents one or more peaks surrounded by inactivity periods (acceleration magnitude close to $1g$), and the number of samples below $1g$ is very small.

4.2. Falls

To understand the acceleration behavior during falls, several traces have been registered with the help of volunteers. Some of the falls have been performed using protective pads. Data from a dummy were not used as they showed very different dynamics, due to the material's elasticity, higher than the human tissues, and because the dummy does not react nor bend while falling.

Falls may start with an unexpected event, like toppling or slipping, or be consequence of fainting. Before impacting the ground, many persons may try to use their hands for protection, even though landing with no reaction may also happen. Different ways of falling have been considered: while walking, from a bed, landing in a prone position, on knees, etc. All falls have

in common a violent impact on a hard surface, so that the acceleration often shows two or more peaks greater than $3g$, due to different parts of the body hitting the ground at different times. As the sensor was placed at the user's waist, the highest peak usually correspond to the time the waist reaches the ground.

Falls from bed are very common among the elderly. They are characterized by a free fall phase with acceleration close to $0g$ that, after the main impact characterized by a high peak, is usually followed by smaller peaks due to the waist bouncing and then landing again on the floor.

It should be noted that, during a simulated fall, the volunteer is aware of the imminent impact and thus tends to control speed and movements. This makes experimental falls more difficult to be distinguished from other fall-like events, since impacts are slightly less violent than real falls. Thus, we argue that these falls can be successfully used for validating and analyzing the performance of a fall detection system, since the obtained results will be conservative with respect to real events.

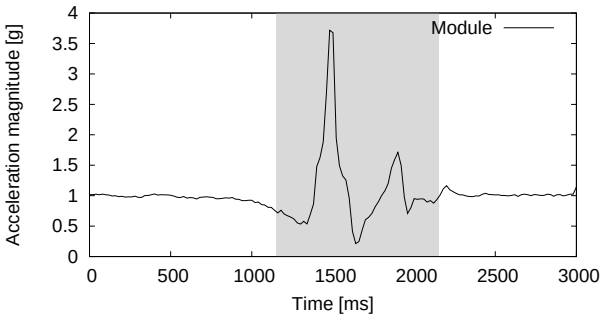
We also recorded a set of falls with a video-camera and asked a gerontologist to evaluate their level of realism. The types of falls that have been considered are: forward fall, backward fall, and faint. For every category, two different falls have been filmed and shown to the gerontologist, both at normal speed and slow motion. The falls have been performed by three different subjects; two of them used knee and wrist protections, while the third, the one who performed the backward falls, didn't use any protection. According to the gerontologist, the level of realism of the falls scored 3 points on a 1-4 scale.

Kangas et al. discuss the similarities between five real falls happened to elderly people and the intentional falls performed during laboratory sessions by middle-aged subjects [26]. To collect the five real falls, sixteen people have been monitored for a long period (up to six months). A comparison between the real-life falls and the intentional falls showed that the acceleration signals share similar features. In several cases, the real-life falls include a pattern made of a pre-impact phase and an impact phase. Also, as a consequence of protective actions (e.g. the use of hands), the impact phase of real-life falls is generally characterized by the presence of multiple acceleration peaks.

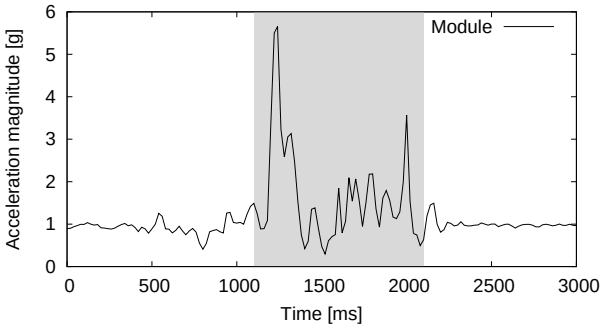
4.3. Feature extraction

In the following paragraphs we describe the eight features which are used by the classification engine to recognize the class of a fall-like event. Before describing the features used to distinguish the ADL classes from real falls, it is important to define three instants which characterize each fall-like event:

- *peak time*, time of the peak exceeding the $3g$ threshold and followed by at least 2500 ms without further peaks.
- *impact end*, time of the last acceleration magnitude value above $1.5g$, searched within the interval [*peak time*, *peak time*+1000 ms]. It represents the instant of the last significant impact with the ground.
- *impact start*, time of the first acceleration magnitude above $1.5g$ which is preceded by at least one sample below $0.8g$, searched within the interval [*impact end*-1200 ms, *peak time*]. It is the time of the first impact on the ground, after the free-fall period (when acceleration magnitude goes significantly below $1g$). If the free-fall sample is not found, as it is often the case when the peak has been caused by an unintentional hit to the sensor, *impact start* time is set to *peak time*.



(a) AAMV window in case of sitting ($0.17g$)



(b) AAMV window in case of fall ($0.59g$)

Figure 11: Accelerometric trace and AAMV

All the thresholds used have been determined by data analysis.

Average absolute Acceleration Magnitude Variation - AAMV. As previously mentioned, when sitting or lying on a soft/elastic surface, acceleration patterns show some smooth peaks, with slow variations in magnitude right after the main impact. With hard surfaces, instead, there is a single sharp peak followed by a rapid stabilization. To encompass both situations, let us define the *Average absolute Acceleration Magnitude Variation (AAMV)* [14] index as follows:

$$AAMV = \sum_{i:time(a_i)\in win} \frac{|a_{i+1} - a_i|}{\sharp\{i: time(a_i) \in win\}} \tag{1}$$

where *win* is an observation interval centered at the middle time between *impact start* and *impact end*; a_i are the samples of the observed acceleration magnitude. Experiments, where the size of *win* has been set to 1000 *ms*, showed us that average values for AAMV are $0.52g$ for the Falls

class, and $0.23g$ for the Sitting/Lying class (two examples of AAMV windows are shown in Figure 11(a) and 11(b)).

The AAMV can also be used to perform the Activity Test introduced in Section 2.3. Experiments showed us that the threshold $0.05g$, calculated in the interval $[1000, 2500]$ *ms* after *peak time*, can be used to distinguish activity from inactivity. Fall-like events exceeding the threshold are discarded as false alarms, because the person is still active after the event. The threshold has been chosen to minimize the risk of discarding real falls, while being still useful to filter long-lasting activities such as walking and running (which may produce many peaks higher than $3g$, but that are not followed by an inactivity period).

Impact Duration Index - IDI. Impact duration is the difference between *impact end* and *impact start*. This feature is useful to recognize as false alarms events characterized by a short impact duration (e.g., hitting the sensor). From our experiments, real falls have a minimum IDI of 320 *ms* and an average IDI of 780 *ms*.

Maximum Peak Index - MPI. This feature represents the highest value of acceleration magnitude found in the interval [*impact start*, *impact end*]. Based on our dataset, average values of MPI are $5.8g$ for the Falls class, $3.9g$ for Lying/Sitting class, $4.2g$ for Jumping/Running/Walking class.

Minimum Valley Index - MVI. This feature represents the lowest value of acceleration magnitude found in the interval [*impact start* - 500 ms, *impact end*]. MVI is useful to distinguish the Falls class from fall-like events due to unintentionally hitting the sensor. Experimentally, the Falls class shows an average MVI equal to $0.24g$, while hitting the sensor generally produces a MVI greater than $0.9g$.

Peak Duration Index - PDI. As previously said, according to our experiments the Falls class is characterized by the presence of sharp acceleration magnitude peaks, while sitting or lying on soft surfaces produces slower acceleration variations. PDI measures the duration of the acceleration peak exceeding the $3g$ threshold. The procedure to calculate PDI is the following:

- *peak start* is defined as the time of the last magnitude sample below $1.8g$ occurred before *peak time*;
- *peak end* is defined as the time of the first magnitude sample below $1.8g$ occurred after *peak time*;
- PDI is the difference between *peak end* and *peak start*.

According to our dataset, the average PDI value of the Falls class is 80 *ms*, while lying/sitting on soft surfaces generally have a PDI value around 120 *ms*.

Activity Ratio Index - ARI. This feature measures the "activity level" in an interval of 700 *ms* centered at the middle time between *impact start* and *impact end*. The activity level is calculated as the ratio between the number of samples not in $[0.85g, 1.3g]$ and the total number of samples in the 700 *ms* interval. From our experiments, Falls, Jumping/Running/Walking and Sitting/Lying classes have an average ARI of 0.7, while hitting the sensor generally produces an ARI significantly lower than 0.5.

Free Fall Index - FFI. As mentioned when describing the ADL classes, jumping, running and walking activities may produce a fall-like event if the user suddenly stops moving. The peak exceeding the $3g$ threshold is due to the feet hitting the ground after a step or a jump: immediately before this peak we expect to find a free-fall interval characterized by acceleration magnitude values below $1g$. In order to check the presence of this condition, we defined a feature based on the average acceleration magnitude evaluated in a small interval before *peak time*.

Such an interval lasts 200 ms and is defined as follows:

- search for an acceleration sample below $0.8g$ occurring up to 200 ms before *peak time*; if found, the sample time represents the end of the interval, otherwise the end of the interval is set 200 ms before *peak time*;
- the start of the interval is simply set to 200 ms before its end.

FFI is defined as the average acceleration magnitude evaluated within the interval.

This feature is particularly useful for the recognition of jumps, as they have a relatively long free-fall period with acceleration magnitude values close to $0g$. According to our experiments, jumps produce an average FFI of $0.1g$, while the minimum FFI experimented for the Falls class is $0.6g$ with an average of $1.1g$.

Step Count Index - SCI. Walking consists in a regular sequence of steps, while running consists in a sequence of small jumps. In both cases, the typical acceleration pattern is characterized by a sequence of valleys and peaks occurring with a quite regular period. Valleys and peaks are due to the movement of the foot before reaching the ground. SCI is the step count evaluated 2200 ms before *peak time*. In order to estimate this index we have used the number of valleys found. More precisely, a valley is defined as a region where the acceleration is below $1g$ for at least 80 ms, followed by a peak above $1.6g$ within 200 ms. Consecutive valleys are separated by at least 200 ms.

4.4. Pattern Classifier

The Pattern Classifier uses a neural network to process the features and produces four membership values. Figure 12 depicts the network architecture. The network input is a vector of the eight indexes computed by the Feature Extractor: AAMV, IDI, MPI, MVI, PDI, ARI, FFI, SCI. The output consists of four neurons, one for each of the three ADL classes and one for the Falls class.

The primary choice in the design of a neural network concerns the network structure. Since our target is pattern recognition, we used a multi-layer feed-forward network. Thanks to the reduced system complexity obtained by the preventive feature extraction, a two-layers feed-forward network suffices to our purposes. After the training and the cross-validation runs, we found that the optimal number of hidden neurons is 7.

The protocol used for training the network is the well-known *back-propagation* algorithm (with learning rate equal to 0.1 and momentum equal to 0.9). The activation function for each layer is the *log-sigmoid* function, the most suitable for classification [25].

A training phase was carried out with the purpose of making the neural network learn the complexities of the input model and understand how to classify the different patterns. Afterward, in order to have a better measurement of network's performance in the recognition of unseen inputs, a test was conducted using the cross-validation technique. Finally, a campaign of continuous data collection was performed for further validation of the fall detection system.

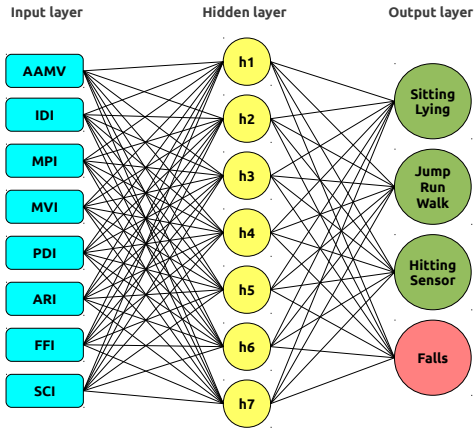


Figure 12: Neural network architecture

Gender	Age	Height (cm)	Weight (kg)
M	20	175	74
M	26	178	75
M	27	175	62
M	37	177	81
M	67	175	95
F	33	170	56
F	60	165	70

Table 2: Volunteers' characteristics

5. Performance evaluation

The observed behavior of a fall detection system is represented by four possible situations: *true positive (TP)*, a fall occurred and the system correctly detects it; *false positive (FP)*, the system declares a fall event, but it was instead a normal ADL; *true negative (TN)*, the system correctly classifies a fall-like event as an ADL; *false negative (FN)*, a fall occurred, but the system does not detect it. System reliability can then be evaluated through the following indexes [27]: $sensitivity = TP/(TP + FN)$, which measures the ability of detecting all the real falls; $specificity = TN/(TN + FP)$, which measures the ability of detecting only real falls; $accuracy = (TP + TN)/(TP + FP + FN + TN)$, which is the proportion of true results in the considered data set.

5.1. Training and re-substitution error

To train the network, we used a database of falls and fall-like ADLs gathered with the help of 7 volunteers. Gender, age and physical characteristics of volunteers are shown in Table 2. The

database includes 86 fall-like events, partitioned into 44 Falls, 14 Jumping/Running/Walking, 14 Sitting/Lying and 14 Hitting the sensor ADLs. The training set has not been split evenly among different classes: falls are more than 50% of the total number of events. Indeed, this choice comes from the consideration that sensitivity is clearly the most important performance index for a fall detection system.

The training process was carried out by randomly feeding the network with all the events from the training set, for a total of 500 epochs. The classifier achieved 100% success rate, all the samples were classified properly and the confusion matrix presented values only on the main diagonal. This means that the system achieves 100% sensitivity, specificity, and accuracy, being able to detect all the real falls and discard all the false alarms.

The error rate on a training set is also known as the *re-substitution error* and, even though useful for a preliminary evaluation, it is not a good index to predict the behavior of the classifier when new and unknown data are processed.

5.2. Cross-validation

In order to estimate the validity of our classifier to a wider extent, we used the general method known as *cross-validation*. In particular, we applied a simple generalization of the method, called *k-fold stratified cross-validation*. Let $\mathcal{D}$ be the complete database of events. The whole set of events is randomly divided into k disjoint sets of equal size n/k, where n is the cardinality of $\mathcal{D}$. The classifier is then trained and validated k times, each time with a different subset held out as a validation set, while the $k-1$ remaining subsets are used for training. For instance, if k is set to 10, this implies that each time we perform training and validation, 90% of the events are used for training, while the remaining 10% is used for validation. The term *stratified* means that each class is properly represented in the subsets, according to the proportions chosen for the original training set.

For the tests, k has been set to 10. Actually, ten-fold stratified cross-validation is widely considered to be a good choice to get an accurate estimate of error when data for training and validation is limited.

Cross-validation confirmed the results obtained using the re-substitution error: the classifier achieves 100% sensitivity and specificity (and thus also 100% accuracy) in detecting real falls, while all the ADLs are recognized properly.

5.3. Continuous data collection campaign

In order to obtain a further validation of the performance of the fall detection system, three volunteers were involved in a continuous data acquisition campaign. They were asked to follow their typical lifestyles, trying to forget they were wearing a sensor. Activities included jumping, running, walking, sitting, lying, hitting the sensor and driving a car. Volunteers were not requested to tag each fall-like event produced, but only to keep track of their movements, specifying how many hours they spent indoors, outdoors or driving a car. Accelerations were recorded and subsequently analyzed off-line. In total, we collected 82 hours of movements, distributed in: 64 hours indoors, 9 hours outdoors and 9 hours driving a car. No real falls occurred during the test.

As expected, outdoor activities generated a great number of acceleration peaks exceeding the $3g$ threshold, especially because walking and running sessions were included in the experiment. However, only a few events were not discarded by the Activity Test and were passed to the classification engine. There, they were classified as a Jumping/Running/Walking event and thus correctly discarded.

Technique	Sensitivity (%)	Specificity (%)	Accuracy (%)
Median filtering plus acc. magnitude	90.9	31.0	61.6
Vertical velocity estimation	100.0	35.7	68.6
Acc. magnitude plus timeouts	81.8	81.0	81.4

Table 3: Performances of relevant fall detection techniques

All the other fall-like events were produced indoors. The count of peaks followed by inactivity was 43, but they were all recognized by the classifier as false alarms. More precisely, 1 event was recognized as Jumping/Running/Walking, 11 events as Sitting/Lying and 31 as Hitting the sensor. In our experiment, driving a car did not produce any peak exceeding the $3g$ threshold.

As real falls did not happen, these tracks could be used only to validate the specificity of the system: the test confirmed the 100% specificity in distinguishing false alarms from real falls.

6. Comparison with relevant existing techniques

This section reports the results of the quantitative comparison of our fall detection approach with some relevant techniques described in recent literature. In particular, we selected a set of methods that have been conceived to operate using a wearable waist-mounted accelerometer. All of these methods have been re-implemented following the description available in the published papers. Then, they have been fed with our traces of intentional falls and fall-like activities, in order to evaluate their performance with respect to our classification technique on the same set of data (where our technique obtained 100% sensitivity, specificity, and accuracy). The results of this comparison are summarized in Table 3.

6.1. Acceleration magnitude threshold and median filtering

The simplest approach to detect falls using an accelerometer consists in using a single threshold on the acceleration magnitude [28]. However, it has been shown that many ADLs present acceleration magnitude peaks similar to those of falls, thus making ADLs really difficult to be distinguished from dangerous impacts on the ground [10, 14]. In particular, all the fall-like events of our training set present a peak equal or greater than $3g$ and, thus, such a simple method fails in isolating ADLs from real falls.

Median filtering of the acceleration magnitude - using a three samples window - has been suggested as a way to reduce noise and improve the specificity of the detection [10]. We applied this filtering technique to our set of data and obtained a maximum overall accuracy equal to 61.6%, with 90.9% sensitivity and 31.0% specificity.

6.2. Vertical velocity estimation

An estimation of the vertical velocity achieved by the user's waist immediately before the impact can be combined with acceleration magnitude to distinguish falls from fall-like events with a higher degree of accuracy [29, 30, 31, 32]. A simple estimation method is based on the numerical integration of the acceleration magnitude after gravity has been subtracted:
$v = \int (acc.magnitude - 1g)\, dt$.

Different techniques have been proposed to reduce the drift due to numerical integration. Degen et al. [29] suggested the application of a damping factor to the positive acceleration during integration: when acceleration magnitude is equal or above $1g$, velocity is set to the velocity of the previous sample multiplied for the damping factor. Bourke et al. [31] described the use of a low-pass filter combined with a technique to detect periods of static activity. The filter used is a 2^{nd} order Butterworth low-pass filter with $15Hz$ cut-off frequency, while static periods are found using a moving $1sec$ variance window.

In order to evaluate this approach, we implemented a velocity estimation algorithm using low-pass filtering, inactivity detection and a properly set damping factor to reduce the drift of numerical integration. An exhaustive search for the velocity threshold and the damping factor that achieve the best overall accuracy has been performed, using a reasonable range of values. On our training set we obtained the following results: accuracy equal to 68.6%, with 100% sensitivity and only 35.7% specificity.

6.3. Acceleration thresholds plus timeouts

Acceleration magnitude thresholds and properly set timeouts can be combined to characterize a typical fall and distinguish hazardous events from ADLs. One interesting example of a fall detection system using this approach is represented by iFall [33], one of the first fall detection applications developed using the Android platform (currently available on the Android market). The proposed algorithm is based on acceleration magnitude thresholds, timeouts and long-lie detection.

Two acceleration magnitude thresholds - *low th* and *hi th* - are used to detect the free-fall phase and the impact on the ground, respectively. A specific timeout - *lower timeout* - is used to ensure that the impact happens not later than $1sec$ after that free-fall has been detected. After the impact, there is a *bouncing interval* during which samples are not analyzed, waiting for stabilization of acceleration. Immediately after the bouncing interval, the algorithm starts monitoring for a long-lie phase after the fall, defined as a time period during which the measured acceleration magnitude is within a given range. The long-lie period must start not later than *upper timeout* seconds after the impact has been detected.

In order to make a quantitative comparison with our detection technique, we implemented this algorithm and ran it with our data-set. The parameters were chosen with an exhaustive search within reasonable ranges, in order to obtain the best detection accuracy. The result is an accuracy equal to 81.4%, with 81.8% sensitivity and 81.0% specificity.

The use of timeouts and acceleration threshold can effectively improve the accuracy of the detection, while requiring a relatively low computational cost.

6.4. Combining use of orientation with the previous approaches

The recognition of posture after the impact on the ground has been widely suggested as an effective method to improve the specificity of fall detection systems [30, 32]. The rationale is that while falls generally end with the user lying on the ground, most of the ADLs do not present such a significant variation in the orientation of the user's waist.

If the initial orientation of the sensor is known, then the acceleration on the vertical axis - the axis which has the same direction as gravity when the user is standing - can be analyzed to detect with good confidence if the user is lying after the impact. In [32], an angle of 60° is suggested as the minimum expected variation from the standing position during a real fall.

This approach cannot be directly applied to our training set of data, as the sensor we used to collect data was voluntarily placed without a fixed orientation. Nevertheless, we may reasonably

assume that all running, walking, jumping, sitting and hitting the sensor ADLs would have been correctly filtered using the orientation of the device, leading to a high specificity value with a relatively low effort.

However, there are some important issues that must be taken into account. First of all, such an approach has the drawback of requiring to place the sensing device with known and fixed orientation. This significantly reduces usability, as the user has to take care of how the device is oriented or periodically perform a calibration. Second, the ADLs due to quickly lying on a soft surface are not addressed by this technique and are likely to lead to false alarms. Finally, it cannot be ignored the risk of missing real falls due to calibration errors or to the fact that the user may not be perfectly parallel to the ground after the impact.

7. Related works

7.1. Smartphone-based fall detection systems

Several smartphone-based fall detectors have been developed so far. This confirms that the idea of using a smartphone as the basis for a fall detection system is sound. All the systems make use of the embedded accelerometer to detect a fall, but use different algorithms. The systems span from a simple threshold-based fall detector to a wavelet analyzer.

The authors of PerfallD [34] propose a comprehensive system to integrate fall detection and emergency communication. They designed two algorithms for fall detection. The first is based on acceleration thresholds: the total and vertical accelerations are compared in time with specific thresholds determined through experiments. The second technique uses a magnetic field sensor and a magnetic accessory. The sensor is embedded in a smartphone, which is placed in the right pocket whereas the accessory is placed slightly above the left knee. In this way the system is able to infer the distance between the smartphone and the accessory. If the distance between them is closer than normal, a fall event is detected.

In [35] the authors used discrete wavelet transform as a feature extraction technique. Then, falls and activities of daily living are distinguished through their properties in the frequency domain. By using this method, they achieved better performance (37%) compared to [33]. The system also provides location information using GPS and a warning system to alert a set of predefined contacts.

Kwapisz et al. [36] used phone-based accelerometers to perform activity recognition instead of fall detection. The work is part of the Wireless Sensor Data Mining (WISDM) project which aims to resolve research issues related to mining sensor data from smartphones. Six common activities were analyzed: walking, jogging, ascending stairs, descending stairs, sitting, and standing. Twenty-nine users performed the activities carrying an Android smartphone in their front pants leg pocket and under the supervision of WISDM team members. The sampling time was 20Hz and features such as averages, standard deviation etc. were generated from the accelerations along the three axes, in a 10 seconds window. Three classification techniques were analyzed to predict user activities: decision trees, logistic regression and multilayer neural networks. A ten-fold crossing validation was used for all the experiments and the multilayer neural network performed better than the other two techniques. Results showed that the walking, jogging, sitting, and standing activities are recognized with accuracy above 90%. The recognition of the two stairs activities, going up and down, was not accurate.

Other simple Android- and iPhone-based fall detector are nowadays available in the app stores. Besides the different user interfaces, the principle of functioning is very similar: a fall

is detected if the acceleration magnitude of the accelerometer in the smartphone reaches a given threshold; if there are no movements for a certain amount of time, the system sends an alarm (phone call, text message, email etc.) to a set of contacts.

7.2. Acceptability of m-health systems in similar domains

Here are summarized three studies where interviews have been used to evaluate the acceptability of technological solutions based on mobile devices.

In [37], an intelligent and unobtrusive reminder for taking medications system is discussed. Ten participants aged over 65 were selected from a pool of patients considered *poor adherents*, i.e. missing more than 20% of prescribed doses. The participants were selected based on some well-known scales: Alzheimer Disease Assessment Scale-Cognitive Subtest (ADAS-Cog), delayed word-list recall scores, Trail Making Test, Geriatric Depression Scale, and Cumulative Illness Rating scale.

A study on a location-based reminder application running on a mobile phone [38], has been carried out over a two-week period among ten participants (seven men, three women) aged between 18 and 45. Each participant had a different occupation and lifestyles and was recruited through mailing list posting and advertisements. Similarly to our case, none of them tested a similar system before. The study has been conducted in three steps, by providing a questionnaire about current habits, analyzing how the participants interacted with the mobile phones, and finally by asking opinions about the overall experience at the end of the two weeks. They found that eight of ten participants appreciated the consistent availability of location-based reminders through their mobile phone, and lessened their use of other reminder tools. Moreover, the types and way the the reminders were posted revealed that location itself was not always important, but just an additional information.

In [39] PDAs are used to help people with cognitive impairment to recall how to perform common activities of daily living. On the basis of the current context, a PDA is able to provide both visual and voice instructions to perform a task. In this case, individuals were recruited according to their cognitive impairments, the ability to perform a task, and the severity of loss in short-term memory. Also the ability to use a PDA was considered. Eight subjects were selected (two women) aged between 20 and 48. They were affected by syndromes such as Traumatic Brain Injury, Intellectual and Developmental Disabilities, Schizophrenia, and Epilepsy. The experiment consisted in executing three tasks including common actions to be performed at a Coffee shop. Results showed that only two subjects preferred the help of a human instructor in place of the PDA's instructions.

8. Conclusion

A fall is an ill-defined process, therefore it is not an easy task to describe it thoroughly. This implies that identifying signal patterns, and then define parameters and filters that can be applied to distinguish falls from common activities, requires a significant effort. Moreover, any technique based on analytical-only methods would require further refinement to be tailored to new users. For these reasons, a machine learning approach can be successfully applied, together with analytical methods, to provide an adaptive and easily expandable solution to the fall detection problem.

The contribution of the work presented in this paper to the body of existing literature is twofold. First, it shows that the recognition of fall-like activities of daily living (ADL), on the

basis of peculiar features of the acceleration patterns, can significantly reduce the number of false alarms. It is important to note that, in a trivial system where an alarm is raised when acceleration becomes higher than the $3g$ threshold, all the fall-like ADLs in our dataset would produce a false alarm. Second, the interviews with the group of aged people showed that usability is a fundamental factor for a real adoption of a fall detection system. In particular, forcing the users to place the phone on their belt can make a fall detection system unattractive for a large part of the population. In this context, the possibility of using a small external sensing unit can greatly reduce the intrusiveness of the system.

Acknowledgments

The work described in this paper has been partly supported by Fondazione Cassa di Risparmio di Lucca 2010 project "Fido: a fall detection alarm system for elderly people" and by MIUR-PRIN 2008 project "Cloud@Home: a New Enhanced Computing Paradigm".

References

[1] Public Health Agency of Canada, Report on Seniors' falls in Canada, 2005.

[2] S. Sadigh, A. Reimers, R. Andersson, L. Laflamme, Falls and fall-related injuries among the elderly: a survey of residential-care facilities in a Swedish municipality, Journal of community health 29 (2004) 129–140.

[3] R. Cumming, G. Salkeld, M. Thomas, G. Szonyi, Prospective study of the impact of fear of falling on activities of daily living, SF-36 scores, and nursing home admission, The Journals of Gerontology Series A: Biological Sciences and Medical Sciences 55 (2000) M299.

[4] D. Wild, U. Nayak, B. Isaacs, How dangerous are falls in old people at home?, British Medical Journal 282 (1981) 266.

[5] Philips, Lifeline - the trusted medical alert service provider, http://www.lifelinesys.com, 2009.

[6] S. Abbate, M. Avvenuti, P. Corsini, J. Light, A. Vecchio, Monitoring of Human Movements for Fall Detection and Activities Recognition in Elderly Care Using Wireless Sensor Network: a Survey, in: G. V. Merret, Y. K. Tan (Eds.), Wireless Sensor Networks: Application-Centric Design, InTech, Rijeka, Croatia, 2010, pp. 147–166.

[7] N. Noury, A. Fleury, P. Rumeau, A. Bourke, G. Laighin, V. Rialle, J. Lundy, Fall detection - principles and methods, in: Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS), pp. 1663–1666.

[8] A. K. Bourke, J. V. O'Brien, G. M. Lyons, Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm, Gait & posture 26 (2007) 194–9.

[9] A. K. Bourke, G. M. Lyons, A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor, Medical engineering & physics 30 (2008) 84–90.

[10] M. Kangas, A. Konttila, I. Winblad, T. Jamsa, Determination of simple thresholds for accelerometry-based parameters for fall detection, in: Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS), pp. 1367–1370.

[11] Q. Li, J. A. Stankovic, M. A. Hanson, A. T. Barth, J. Lach, G. Zhou, Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information, in: Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks, IEEE Computer Society, Berkeley, CA, USA, 2009, pp. 138–143.

[12] X. Yu, Approaches and principles of fall detection for elderly and patient, in: Proceedings of the 10th International Conference on e-health Networking, Applications and Services (HealthCom), pp. 42–47.

[13] M. Avvenuti, C. Baker, J. Light, D. Tulpan, A. Vecchio, Non-intrusive patient monitoring of alzheimer's disease subjects using wireless sensor networks, in: Proceedings of the 2009 World Congress on Privacy, Security, Trust and the Management of e-Business (CONGRESS 2009), IEEE Computer Society, Saint John, Canada, 2009, pp. 161–165.

[14] S. Abbate, M. Avvenuti, G. Cola, P. Corsini, J. V. Light, A. Vecchio, Recognition of false alarms in fall detection systems, in: Proceedings of the 1st IEEE International Workshop on Consumer eHealth Platforms, Services and Applications (CCNC Workshop CeHPSA), Las Vegas, NV, USA, pp. 538–543.

[15] U. Lindemann, A. Hock, M. Stuber, W. Keck, C. Becker, Evaluation of a fall detector based on accelerometers: A pilot study, Medical and Biological Engineering and Computing 43 (2005) 548–551. 10.1007/BF02351026.

[16] N. Noury, P. Barralon, G. Virone, P. Boissy, M. Hamel, P. Rumeau, A smart sensor based on rules and its evaluation in daily routines, in: Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS), volume 4, pp. 3286 – 3289 Vol.4.

[17] H. Gjoreski, M. Lustrek, M. Gams, Accelerometer placement for posture recognition and fall detection, in: Proceedings of the 7th International Conference on Intelligent Environments (IE), pp. 47–54.

[18] D. M. Karantonis, M. R. Narayanan, M. Mathie, N. H. Lovell, B. G. Celler, Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring., IEEE Transactions on Information Technology in Biomedicine 10 (2006) 156–67.

[19] Realtime Technologies Inc., http://www.shimmer-research.com, 2010.

[20] K. Lorincz, B.-r. Chen, G. W. Challen, A. R. Chowdhury, S. Patel, P. Bonato, M. Welsh, Mercury: a wearable sensor network platform for high-fidelity motion analysis, in: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, SenSys '09, ACM, New York, NY, USA, 2009, pp. 183–196.

[21] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, D. Culler, TinyOS: An Operating System for Sensor Networks Ambient Intelligence, in: W. Weber, J. M. Rabaey, E. Aarts (Eds.), Ambient Intelligence, Springer Berlin Heidelberg, Berlin/Heidelberg, 2005, pp. 115–148.

[22] M. Avvenuti, P. Corsini, P. Masci, A. Vecchio, Energy-efficient reception of large preambles in MAC protocols for wireless sensor networks, Electronics Letters 43 (2007) 300–301.

[23] M. Avvenuti, P. Corsini, P. Masci, A. Vecchio, Increasing the efficiency of preamble sampling protocols for wireless sensor networks, in: Proceedings of the Mobile Computing and Wireless Communications International Conference, pp. 117–122.

[24] M. Avvenuti, P. Corsini, P. Masci, A. Vecchio, An application adaptation layer for wireless sensor networks, Pervasive and Mobile Computing 3 (2007) 413–438.

[25] R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification (2nd Edition), Wiley-Interscience, 2 edition, 2000.

[26] M. Kangas, I. Vikman, L. Nyberg, R. Korpelainen, J. Lindblom, T. Jms, Comparison of real-life accidental falls in older people with experimental falls in middle-aged test subjects, Gait & Posture 35 (2012) 500 – 505.

[27] N. Noury, P. Rumeau, A. Bourke, G. Laighin, J. Lundy, A proposal for the classification and evaluation of fall detectors, IRBM 29 (2008) 340–349.

[28] A. Bourke, J. O'Brien, G. Lyons, Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm, Gait and Posture 26 (2007) 194–199.

[29] T. Degen, H. Jaeckel, M. Rufer, S. Wyss, Speedy:a fall detector in a wrist watch, in: Proceedings of the Seventh IEEE International Symposium on Wearable Computers, pp. 184–187.

[30] M. Kangas, A. Konttila, P. Lindgren, I. Winblad, T. Jms, Comparison of low-complexity fall detection algorithms for body attached accelerometers, Gait & Posture 28 (2008) 285 – 291.

[31] A. K. Bourke, K. J. O'Donovan, J. Nelson, G. M. OLaighin, Fall-detection through vertical velocity thresholding using a tri-axial accelerometer characterized using an optical motion-capture system, in: Proceedings of the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS), pp. 2832–2835.

[32] A. Bourke, P. van de Ven, M. Gamble, R. O'Connor, K. Murphy, E. Bogan, E. McQuade, P. Finucane, G. O and-Laighin, J. Nelson, Assessment of waist-worn tri-axial accelerometer based fall-detection algorithms using continuous unsupervised activities, in: Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS), pp. 2782 –2785.

[33] F. Sposaro, G. Tyson, iFall: An android application for fall monitoring and response, in: Proceedings of the Annual International Conference of the IEEE in Engineering in Medicine and Biology Society (EMBS), pp. 6119–6122.

[34] J. Dai, X. Bai, Z. Yang, Z. Shen, D. Xuan, PerFallD: A pervasive fall detection system using mobile phones, in: Proceedings of the 8th IEEE International Conference on Pervasive Computing and Communications Workshops, (PERCOM Workshops), 2010, pp. 292–297.

[35] G. Yavuz, M. Kocak, G. Ergun, H. O. Alemdar, H. Yalcin, O. D. Incel, C. Ersoy, A Smartphone Based Fall Detector with Online Location Support, in: Proceedings of PhoneSense 2010, pp. 31–35.

[36] J. R. Kwapisz, G. M. Weiss, S. A. Moore, Activity recognition using cell phone accelerometers, SIGKDD Explor. Newsl. 12 (2011) 74–82.

[37] T. L. Hayes, K. Cobbinah, T. Dishongh, J. Kaye, l. J. Kime, M. Labhard, T. Leen, l. J. Lundel, U. Ozertem, M. Pvael, M. Phillipose, K. Rhodes, S. Vurgun, A study of medication taking and unobtrusive, intelligent reminding., Telemed J E Health 8 (2009) 770–776.

[38] T. Sohn, K. A. Li, G. Lee, I. E. Smith, J. Scott, W. G. Griswold, Place-its: A study of location-based reminders on mobile phones, in: Proceedings of the Fifth International Conference on Ubiquitous Computing (UbiComp 2005), Springer-Verlag, 2005, pp. 232–250.

[39] Y.-J. Chang, W. C. Chang, T.-Y. Wang, Context-aware prompting to transition autonomously through vocational tasks for individuals with cognitive impairments, in: Proceedings of the 11th international ACM SIGACCESS conference on Computers and accessibility, Assets '09, ACM, New York, NY, USA, 2009, pp. 19–26.