DOI: 10.1109/JSEN.2017.2723840

User-Centric Coordinates for Applications Leveraging 3-Axis Accelerometer Data

Alexandre Ferreira, *Member, IEEE*, Geise Santos, Anderson Rocha, *Senior Member, IEEE*, and Siome Goldenstein, *Senior Member, IEEE*

***Abstract*— Mobile devices are becoming ubiquitous and, sometimes, even extensions of ourselves. These devices are growing fast in terms of delivered computational power, storage capacity, battery duration, and built-in sensors. Time and again, we see headlines advertising new unforeseen applications leveraging this power, especially the sensors, for solving diverse problems, including fall detection, user's activity recognition, location identification, or even user authentication based on the way of walking (gait). In this paper, we focus on motion sensors and discuss how the provided data can be interpreted and transformed to better serve different purposes. We propose a method to process the data from such sensors that reduces the acquisition noise and possible artifacts, and turns the data invariant to the device's position and the user's movement direction. We introduce a new coordinate system referred to as user-centric, as opposed to the two most common coordinate systems used—the device and world-coordinate systems. For testing, we design and develop a user recognition system based on gait, in which the three coordinates systems are compared and discussed considering a one-versus-rest authentication setup. The results show the importance of properly pre-processing the acquired data to enable more reliable applications underpinned by mobile sensors.**

***Index Terms*— Accelerometers, mobile applications, gait recognition, data preprocessing, digital filters, signal analysis.**

I. INTRODUCTION

MOBILE devices are rapidly reshaping the very notion of computation, the perception of connectivity and social networking as well as the concepts of online privacy. Nowadays we see an unprecedented number of mobile devices available on the market with unrivaled processing capabilities [1]. It is not only the processor that is becoming better, but also the device as a whole, which involves cameras, motion sensors, the display, the battery, internal storage, among other items. With such powerful devices in our hands, it is only natural to think of its many possible applications that could go far beyond calling, e-mailing and chatting.

As a matter of fact, this is already happening with the development of ubiquitous solutions, in which the device becomes part of ourselves instead of just close to ourselves. Allied with groundbreaking advances in machine learning and cloud computing, such small pieces of hardware are far more powerful than one could imagine. Given this potential, several players in the academy and in the industry are constantly seeking to design and deploy tools that make the maximum use of such devices, in terms of generated data and available computation power. The ultimate objective is to empower such devices with reasoning capabilities upon existing/collected data, allow the design of powerful inference models for dealing with unseen situations, and to facilitating decision making. With these devices becoming parts of our beings, they are swiftly turning themselves into our "confidants" and storing virtually all information associated with our daily activities. By consequence, such devices need to be fully protected and understood.

In this vein, in this article, we focus our attention on motion sensors, which comprehend the mobile device's accelerometer, gyroscope, and compasses. While at first glance one could deem such sensors as less important than their more expensive cousins (processors, mobile GPUs or even the displays), they are responsible for a series of essential small tasks and, more importantly, they have potential to improving several of our daily activities and way of life.

Each of the aforementioned sensors captures some specific information about motion. The accelerometer, for example, measures the device linear acceleration in m/s^2 in all three physical axes (x, y and z) by gauging the forces applied to the sensor itself (including gravity). The simplest use of this sensor is to infer the device's orientation (portrait or landscape), which can change according to how the user holds the device (e.g., a smartphone).

The accelerometer and other motion sensors from mobile devices have been used in many situations, in particular, when combined with machine-learning techniques including: (i) action recognition, which involves the recognition of human movements and the classification of the different activities such as walking, jogging, climbing stairs, going down stairs, sitting, and standing [2]–[4]; (ii) fall detection, which focuses on the risks involved with elderly people [5]–[7]; (iii) gait

Manuscript received May 24, 2017; revised June 21, 2017; accepted June 26, 2017. Date of publication July 5, 2017; date of current version July 21, 2017. This work was supported in part by Brazilian National Council for Scientific and Technological Development, in part by the São Paulo Research Foundation–FAPESP under Grant 2015/19222-9, DéjàVu, in part by the Brazilian Coordination for the Improvement of Higher Level Education Personnel–CAPES (DeepEyes project), in part by the Microsoft Research, and in part by the European Union H2020 Marie Sklodowska-Curie under Grant 690907 - Identity Project. The associate editor coordinating the review of this paper and approving it for publication was Dr. Arindam Basu. *(Corresponding author: Alexandre Ferreira.)*

The authors are with the RECOD Laboratory, Institute of Computing, University of Campinas, Campinas 13083-970, Brazil (e-mail: melloferreira@ic.unicamp.br; geise.santos@ic.unicamp.br; anderson@ic.unicamp.br; siome@ic.unicamp.br).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. The Supplementary Material contains Java source code and some detailed explanation of the authors' proposed method. This material is 88.3 KB in size.

Digital Object Identifier 10.1109/JSEN.2017.2723840

recognition, in which the user's gait (way of walking) characteristics are used to improve the device's authentication mechanism in a user-friendly fashion [8]–[14]; and (iv) location identification and strengthening, in which we can improve location services indoors and in dense urban areas for more reliable mapping and direction services [15]–[17].

To properly use the raw sensors data in such applications, it is highly important to convert such data into a resilient and unbiased device- and environment-independent coordinate system. The most common way of accomplishing this conversion involves adopting the so-called world-coordinate system. This coordinate system, although effective in some situations, might lead to a strong bias in which unwanted characteristics are not properly decoupled from the problem. An example of such bias can be the environment itself, in which the user's direction impacts the raw data. As such biases can impair data inference later on (for different applications), in this paper, we set forth the objective of discussing the problems associated with the device coordinate system (in which the data is captured) and the commonly-used world coordinate system. In doing so, we also design a robust user coordinate system aimed at eliminating possible biases associated with the device and its environment. The new coordinate system can be invaluable for the aforementioned applications allowing the design of more effective machine-learning solutions using less training data and device's resources when deploying embedded solutions.

We hereinafter refer to the proposed coordinate system as **User-centric Coordinates**, which defines a set of unique coordinates for each user at different time quanta, being resilient to different acquisition and environmental conditions. When presenting this technique, we adopt a didactic approach, describing all the details associated with the new coordinate system and discussing its different benefits from different vantage points. Moreover, we present in a Supplementary Material the necessary steps and source-code to implement the discussed methods. In summary, the main contributions are:

- A formal discussion of the pros and cons of adopting a more resilient and unbiased coordinate system independent of the sensor-related application envisioned;
- The introduction of a new user-centric coordinate system to better separate the environment and user characteristics, isolating possible biases associated with the collected raw sensor data;
- A step-by-step description of the proposed method implemented through spherical interpolation and quaternions, reducing the likelihood of numerical problems and making the calculation more efficient;
- The implementation of the new proposed coordinate system with a gait recognition application, in which one user is confronted against possible impostors seeking to gain unauthorized access to a mobile device.

We organize the rest of paper into five sections. Sec. II presents previous work in the literature, which rely upon multiple sensor data for different applications. Sec. III details the mobile accelerometer sensor and the three-axial data representation. Sec. IV shows how we can transform accelerometer data into different coordinate systems and introduces a robust user-centric coordinate system underpinned by quaternion rotations and a suitable interpolation method called spherical linear interpolation. Sec. V presents a case-study of the proposed coordinate system for gait authentication, showing its robustness to noise and different acquisition conditions through a series of experiments. Finally, Sec. VI concludes the paper and sheds light on some future investigations.

II. Related Work

In this section, we discuss some recent publications describing methods leveraging mobile devices' sensor data for different applications, with particular interest in accelerometer data. We are not disputing their adopted methodology nor their achieved results. Rather, we focus on bringing to bear some aspects related to the data acquisition process itself and the data representations used in the proposed solutions.

An example of the accelerometer sensor wide adoption is in the health and wellness research areas, in which the data serve as input to more specific analytic models responsible for several tasks from fall detection to activity recognition. Alsheikh *et al.* [18] collected data using a mobile device's accelerometer sensor to recognize user activities. The authors argue that the classification step is the most important in their solution (in this case, they rely upon data-driven models). However, in their work, there are no details of the data collection process, pre-processing or even coordinate system transformations to deal with possible biases and acquisition artifacts. This is an example for which the input data were considered clean enough, without supporting evidence, for further processing.

Matthews *et al.* [19] used a specific type of accelerometer sensor (AM-7164; ActiGraph) in order to analyze the relationship among mortality, sedentary time and physical activity factors. In the experimental description, different subjects were evaluated during a seven-day period. The authors performed a series of statistical analysis to cope with the posed research goals, however, there is almost no mention about any pre-processing step and how the collected data were manipulated in terms of filters, rotations and interpolations to eliminate acquisition biases and artifacts. Sanchez *et al.* [20] proposed a model capable of describing activities and emotional aspects of a person based on smartphone accelerometer data. Upon a close inspection of their work, we could not find details regarding the data acquisition process nor any coordinate system transformation. Instead, the authors opted to calculate the acceleration module and standard deviation within a 20s-moving window to make it independent from the device's orientation and position. The main issue of this approach, however, is that important information is lost due to the adoption of the motion vector's magnitude for representing the data instead of the 3-axis traditional representation. The same problem also happens in [21], where the authors calculate the signal vector magnitude of a smartphone accelerometer data and use them into a fall predict system.

Thiemjarus [22] aimed at making the accelerometer data, from a specific sensor device, independent to the device orientation. For that, a k-NN classifier is trained considering four classes, i.e., four different positions. Based on the identified position, the accelerometer data are rotated accordingly through a rotation matrix. Although the solution has kept all original data, which is important to properly identify one activity, the solution has the following differences with our work: firstly, in the referenced work, the used sensor is very specific and placed in the subject's body without any free movement, which leads to a very controlled experiment. In our case, we use a smartphone sensor and the device is freely placed in the subject's pocket. Secondly, the three-axis rotation in that work is performed directly with the rotation matrix, however, as we show in this work, the use of quaternions to perform such rotation seems to be more advantageous: the representation is more compact (only 4 numbers instead of 9 from the traditional rotation matrix), it avoids the Gimbal Lock situation, it allows a proper interpolation in 3-D data through `Slerp` and it reduces the numerical error after iterated rotations. Finally, the proposed solution in that work is device independent only, which is suitable and sufficient for their setup involving user activity recognition, for instance. However, their method is still dependent on the subject's direction with respect to the Earth's magnetic field.

Another research field in which the utilization of smartphone sensors have been increasing is digital forensics and biometrics. Seeking to protect the device data from prying and unauthorized eyes, many authentication mechanisms now focus on providing the user with more than one way of authentication (the so-called 2-way or n-way authentication factors). This strategy may lead to higher security levels, but also impacts directly on usability. One way of having the best of both worlds is to keep verifying the user without requiring its immediate attention [23]–[26]. These mechanisms normally rely upon different device sensors, including motion and touch-screen sensors, to capture behavioral traits. Referred to as continuous behavioral authentication methods [25], these systems use a set of behavioral traits to calculate a score between the observed and the expected behavior of a user enrolled in a mobile device. Examples of such traits are the users' gait [12], [13], [27]–[29] and gestures [30]–[32]. The main problem with the aforementioned methods is that most of them do not have a special treatment for cleaning and sanitizing the acquired data and eliminating possible acquisition biases and artifacts. Sometimes, a simple low-pass filter is used when, for instance, the authors are dealing with gait tracking scenarios [33], [34].

Table I summarizes the main aspects of some of these approaches, which adopt accelerometer data to identify some user characteristic or activity. The first column denotes the work reference, the second shows the sensor type used (mobile or wearable), the third column brings the adopted coordinate system (it also states whether the magnitudes are considered with Euclidean norm) and, ultimately, the fourth column shows the interpolation technique employed, if any.

To the extent of our knowledge, the user-centric coordinate system we propose in this paper is completely new in the literature and could be used in several of the mentioned applications, providing the users with more reliable acquired sensor data. This coordinate system is independent of the device's orientation/position and also considers all three-axial values in an independent fashion, which leads to more accurate raw data than a traditional representation based on acceleration vector magnitudes. Moreover, the described calculation process that involves interpolation and rotation is more efficient than the traditional vector-magnitude representation, as the adopted methods are more robust to numerical errors.

TABLE I

COMPARISON OF RELATED WORK CONSIDERING THE SENSOR TYPE, ADOPTED COORDINATE SYSTEM, AND THE USED ROTATION METHOD TO OBTAIN A GIVEN COORDINATE SYSTEM

Reference	**Sensor type**	**Coordinate system**	**Interpolation**
Fuentes et al.[2]	Mobile	Device/magnitude	Linear
Khan et al.[3]	Wearable	Device/magnitude	Linear
Catal et al.[4]	Mobile	Device/magnitude	Linear
Valcourt et al.[5]	Mobile	Device/magnitude	Linear
Hoang et al.[35]	Mobile	Device/magnitude	Linear
Gowda et al.[33]	Mobile	Device/magnitude	N.A.
Lee at al.[36]	Mobile	World	Linear
Bojja et al.[37]	Custom	World	N.A.

Aiming at providing a clear overview of the increasing number of research works that rely upon accelerometer data in physical-related activities, Montoye *et al.* [38] surveys the literature with about three hundred papers discussed. The evaluation divides the papers into three key areas: accelerometer information, data processing and interpretation and protocol non-compliance. The results indicate an urgent need for improvement on how future publications should deal with accelerometer data and the acquisition processes. The goal of this work is completely aligned with the aforementioned demand, not only with physical-related activities but also all research fields relying upon wearable accelerometer sensor data and, in particular, Android smartphone accelerometer data.

III. MOBILE SENSORS

Most of the smartphones available on the market comprise sensors capable of operating in the background without blocking user activities on the devices. Examples of popular sensors include the accelerometer, which measures the device acceleration; the magnetometer, to estimate the North pole; and the gyroscope, which gives the device's rotation rate.

A. Accelerometer

The accelerometer sensor measures the acceleration of an object (in meters per second squared – m/s^2), i.e., the rate of change in velocity. The forces applied to the object that can change its acceleration can be either static (e.g., gravity), or dynamic (e.g., motion). These forces are measured on a 3-physical axes (x, y and z), in which each axis points to a different direction of the physical world.

With the accelerometer becoming a commodity in virtually all smartphones, its applications are endless. Its primary function, however, has been to detect the device's rotation and the user motion-associated gestures, such as shaking and tilting.

Its simplest application use is to determine the screen auto-orientation changing between portrait and landscape. More sophisticated developments are present in gaming or music playing [39]–[41].

Due to the wide range of uses of this sensor, in particular, when built-in on smartphones, it is important to understand the sensor's delivered data. First of all, the accelerometer adopts the standard coordinate system, which is referred to as device coordinate and it is explained in Section IV-A. The generic model is presented as:

$$\hat{f} = Mf + b_a + n_a$$

where $\hat{f}$ is the delivered data, M represents the cross-axis sensitivity matrix, usually called *Misalignment Matrix*, b_a is the accelerometer bias, and n_a is a random noise vector that is assumed to be close to zero. Besides the variables in this model, we also have to consider the sampling rate of the sensor, which in smartphone-related setups is affected by other existing sensors and the involved applications. According to the Android documentation [42], it is possible to determine the minimum time interval (in milliseconds) the data can be sensed; however, there is no guarantee that all data will come at the same rate. To solve this problem, most applications need to rely upon an *interpolation* process.

The typical interpolation method used to cope with unsteady sampling is the linear interpolation. It considers a straight line between two known points, which is represented by a linear function. This function can be written as $f(p) = mp + b$, where m (slope) and b (ordinate intercept) are calculated using two known points in order to find a new one, p. The main advantage of this method is its low-computational requirements, a key aspect for mobile devices' setups.

B. Device's Orientation

The device's orientation can be obtained directly from some existing sensors or even from the combination of these sensors. The magnetometer, for instance, returns the magnetic North Pole, while the gyroscope returns the rotation rate in rad/s around the device's axes. We can obtain and represent the device's orientation relative to the Earth's frame of reference orientation by the following ways:

- **Rotation matrix**, which is obtained either with gyroscope sensor (estimating the rotation around each axis) or accelerometer data joint with magnetic field sensor (estimating the device positioning regarding to North and East orientations);
- **Euler rotation angles**, which represent rotation angles upon the three axes calculated using the gyroscope. Although Euler angles represent the easiest way to rotate a three-axial coordinate system, it has limitations such as: (a) the interpolation of two orientations may be difficult or unreliable; (b) the estimation becomes imprecise after a given number of rotations; (c) it is unprotected from the Gimbal-Lock problem [43], in which there is a degenerate rotation of two dimensions when two axes of the triaxial rotate over the same vector; (d) intrinsic inconsistency, as some angles with the same values but with different signals (e.g., positive and negative values) may produce the same rotation; and, finally, (e) the rotation over only one axis can be complicated as it is necessary to determine a simple steady rotation between two independent orientations [44].
- **Unit quaternions**, which can be associated directly to the rotation vector data provided by the Android API. The rotation vector is the aggregation of different information coming from the magnetometer, accelerometer and/or gyroscopy sensors. The unit quaternions describe the rotation by an axis $\vec{e} = [\, e_x \; e_y \; e_z \,]^T$ and an angle α. A quaternion is represented by $\vec{q} = [\, q_r \; q_i \; q_j \; q_k \,]^T$, where:

$$\begin{cases} q_r = \cos(\frac{\alpha}{2}) \\ q_i = e_x \sin(\frac{\alpha}{2}) \\ q_j = e_y \sin(\frac{\alpha}{2}) \\ q_k = e_z \sin(\frac{\alpha}{2}) \end{cases}$$

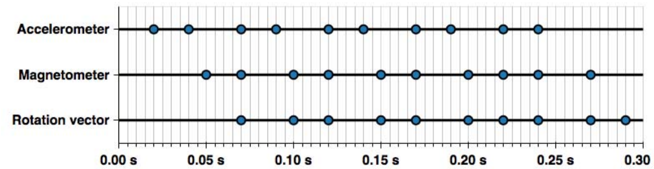


Fig. 1. Sensor data collection over time.

The adoption of unit quaternions has some advantages, from which we highlight: (i) the representation is more concise with respect to the rotation matrix; (ii) it is more numerically stable as less computation is required; and (iii) it avoids the Gimbal-Lock problem. Considering these advantages, the unit quaternions is the method we have chosen to transform the data from the accelerometer (device coordinates) into world coordinates.

C. Alignment and Interpolation

The Android API does not guarantee the same acquisition time interval between two sensor samples [42] or even a certain start and final time of the sensors data collection. Fig. 1 depicts an example, based on real data, showing how accelerometer, magnetometer and rotation vector samples are not aligned nor are they within the same time interval.

Usually, to obtain a more accurate device's orientation calculation, it is necessary to combine data coming from different sensors and, for this purpose, the data of such sensors must be aligned. It is important to use reliable alignment methods such as, for instance, adaptive filters [45], to predict the missing samples. The main advantages of these methods are that they can estimate an underlying model for the available data, leading to an improved prediction and final independent samples over time. In this work, we have used the rotation vector data provided by the Android API and, therefore, we do not perform this type of alignment. However, for the alignment between the rotation vector and accelerometer data, we have applied a linear interpolation method due its simplicity, as previously discussed.

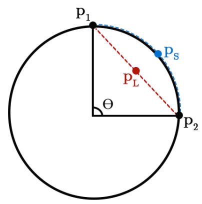


Fig. 2. Slerp and linear interpolation comparison.

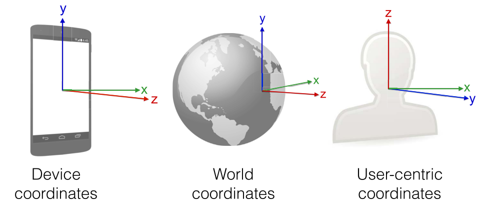


Fig. 3. Different coordinate systems for representing data acquired with the mobile devices' accelerometers.

After the linear interpolation of the accelerometer data, the beginning and ending samples are defined for both accelerometer and rotation vector sensors. The next step is to interpolate the rotation vector samples, represented as quaternion units to guarantee that the accelerometer and rotation vector data are aligned. Given that the rotation vector data are hyperspherical coordinates, the linear quaternion interpolation will produce a secant between both quaternions, which is not reliable as these coordinates are not independent with one another. The solution is to interpolate the quaternion units with Spherical linear interpolation (`Slerp`), which is suitable for rotations in the three-dimensional space [46].

The `Slerp` algorithm [47], when applied to unit quaternions, creates an interpolated great arc on the four dimensional quaternion sphere S^3 (Lie group structure) connecting two quaternion units $\vec{q}_1$ e $\vec{q}_2$.

Taking an interpolation fraction t between two points, in which $0 \leq t \leq 1$, we have:

$$\mathtt{Slerp}(\vec{q}_1, \vec{q}_2, t) = \vec{q}_1((\vec{q}_1)^{-1}\vec{q}_2)^t, \tag{1}$$

where $(\vec{q}_1)^{-1}$ is the inverse of the quaternion $\vec{q}_1$. Given a quaternion $\vec{q}$, the inverse of $\vec{q}$ is obtained by $(\vec{q})^{-1} = [\, q_r \ -q_i \ -q_j \ -q_k \,]^T$. In Fig. 2, we have a tangent plane of a sphere S^3 to compare the `Slerp` and linear interpolation in the case of hyperspherical coordinates. Given two points, p_1 and p_2 in the sphere, the linear interpolation draws a straight line between them and creates a point in the middle of this line, p_L. The `Slerp`, however, draws an arc on the sphere and picks the arc's midpoint, p_S.

In the Supplementary Material, we show the Java implementation of the techniques presented in this section for all device's orientation transformations.

IV. Proposed Coordinate System

In this section, we start with a description of the two most common coordinate systems (device and world) in the literature. Most of related work using accelerometer sensor data, for different applications, rely upon such coordinate systems [5], [35], [36]. These two coordinate systems are not very robust to possible acquisition biases and artifacts such as those involving the device position and the movement direction. Therefore, we propose a new coordinate system referred to as *user-centric coordinates* with the very objective of providing the users and different applications with a more robust representation of the acquired accelerometer data. Fig. 3 depicts these three different coordinate systems.

The *device coordinate system* is the one obtained directly from raw acceleration and it is totally dependent on the device orientation. The *world coordinate system*, in turn, is obtained from a linear transformation of the accelerations represented in the device coordinate system. It solves the orientation variation of the accelerometer data but, with this representation, the data become dependent on the user's movement direction given that there is a projection of the raw data onto a new coordinate system according to the North Pole direction.

Compared to the previous coordinate systems, the one we propose herein is independent of both the device's orientation/position and from the movement direction of the user. These are key advantages, reducing possible external interference on the data (acquisition telltales), providing applications with more accurate data with respect to what is happening with the device (bias minimization) and constant change of directions on the movement (direction independence).

A. Device Coordinate System

In this coordinate system, the three axial data provided by the smartphone accelerometer are collected relative to the device's front and thus, called device coordinate system. Looking at the device's screen, the x axis points to the right, the y axis points up, and the z axis points directly to the observer. Fig. 3 (left) shows this device coordinate system.

This representation is dependent on the device positioning, thus it becomes mandatory that it remains in the same position during its utilization depending on the application. In order to deal with this limitation, it is common, in different applications, to transform (rotate) these coordinates into a more stable coordinate system. The most common way of doing this is to consider the world coordinate system [42], in which the x axis points to East, the y axis points to the North pole, and z points toward the sky, as Fig. 3 (center) depicts.

B. World Coordinate System

The adoption of the world coordinate system, which is also referred to as real-world coordinate system, is the easiest way of turning the signals from the accelerometer independent of the device positioning while preserving the three-axial information. It represents a coordinate system in which one direction (in the accelerometer setup, the y axis) points toward the Earth's North Pole. This is essential when the user (or the device itself) is the focus of the collected data such as, for instance, in activity recognition related applications. Obtaining the world coordinate system is relatively simple as we know where the North Pole is located (this task can be supported by other sensors such as the magnetic field). Nonetheless, it captures information about the environment as the Earth is the reference point, turning the data dependent on the user's movement direction.

The process of transforming the device coordinates into world coordinates starts with the association between each accelerometer sample with a quaternion unit based on a certain timestamp. In this way, it is possible to rotate the accelerometer data from the device's coordinate system to the world's coordinate one such that x becomes tangent to the Earth and points to East; y also becomes tangent to the Earth but points to the Earth's North Pole; and z becomes perpendicular to the Earth and points toward the sky. Fig. 3 (center) depicts an example of this coordinate system. Given the accelerometer sample $\vec{a} = [\, x \; y \; z \,]^T$, it can be rotated using the corresponding quaternion $\vec{q}$ of $\vec{a}^{\,w}$ by:

$$\vec{a}^{\,w} = (\vec{q})^{-1}\vec{v}\vec{q}, \tag{2}$$

where $\vec{v} = [\, 0 \; \vec{a} \,]^T$ and the vector multiplications are unit quaternion operations [46], [47].

Whenever the coordinate system data are needed for an outside-app purpose, such as user activity recognition, for instance, it is highly important to make these signals independent from the device's positioning. As previously mentioned, there is related work that have assumed a fixed position for the device, which surely is unrealistic. Another fairly common approach consists of computing the magnitude of the acceleration vector through the Euclidian formula. In this case, important information about individual axes are lost and the whole application might be in peril. At this point, it is worth stressing again that we are not discussing the signal pre-processing issues nor their implications in the specific applications, but rather the importance of having the richest and most accurate input data possible from the accelerometer sensor ready to be used in the application level.

Although the world coordinates system represents a remarkable improvement over the device coordinate system, the idea of fixing the axis directions toward the Earth's North Pole may represent a drawback onto itself if we are trying to analyze specific signal nuances. For instance, this might be the case when we are not only aiming at recognizing an activity but also who is performing it through behavioral biometrics. The signal input data, from the accelerometer, have to be independent from the device positioning and also from the movement direction. Moreover, some applications might require the input data to be also robust/independent from right/left- and up/down-turns as in gait authentication, activity recognition, steps counter, among others. This is why we now turn our attention to the introduction of a user-centric coordinate system aimed at addressing such problems.

C. User-Centric Coordinate System

In this section, we introduce the user-centric coordinate system, which might be more appropriate to several applications, mainly when the used data refer to some action performed by the user. Such applications may include, but are not limited to, activity recognition [18], identification of emotional aspects [20], gait authentication [11], [48], [49], fall detection [5], and others. The underlying idea is to capture the coordinates according to the user orientation, making it independent from device positions and movement directions relative to the Earth.

To project the data onto the user-centric coordinate system, the first step consists of obtaining the user direction. This can be done by calculating the instantaneous velocity, which is the integral of the acceleration data projected onto the world coordinate system. However, the main issue here is that the acceleration might be noisy and biased, as we explained in Sec. III-A and Eq. 1, leading to unreliable velocity data. To minimize these problems, we propose two digital filters:

- A low-pass filter to remove additional movements that can impair the calculation of the directions; and
- A band stop filter, whose objective is to remove random low-frequency components from the accelerometer data that can also interfere with the estimation of the correct directions, according to the velocity analyses.

1) The Low-Pass Filter: The first digital filter is contingent on the application, as the cutoff frequency and the desirable filter features depends on the type of movements to which the accelerometer data refer. In spite of that, this filter must have linear phase so it does not compromise the directions. Considering the setup in which we want to analyze actions related to step change / walking movements of the user, we designed a low-pass finite impulse response (FIR) filter using a windowing method because it is simpler, more popular and easier to adopt when compared with another FIR methods. We evaluated two possible windows, both with cutoff frequency set to 2 Hz as it is usually on the upper frequency of a normal walking activity:

- **Hamming window**: it has desirable features such as low frequency response on secondary lobes, which have low energy.
- **Gaussian window**: we verified that the collected data have Gaussian distribution, which contributes to noise reduction without corrupting the preserved data.

2) The Band-Stop Filter: Digital filters can also be developed using a technique called the z-transform [50], in which we have the z-plane, an unit circle, and the z-domain that uses a polar notation. The transfer function is expressed as poles and zeros placed on the z-plane. In this case, we aim at designing a band reject (*notch*) filter that removes 0 Hz (bias) with a small gap to eliminate very low frequencies. We start locating the poles and zeros in the z-plane:

$$z = re^{i\omega}$$

where ω is the cutoff frequency in rad/s and r is the radius.

We have used only two coefficients to design this filter, placing one zero on $z_0 = 1+0j$ and one pole on $z_p = 0.9+0j$. Considering the frequency response of the Notch filter, the band-rejection width allows us to remove components on 0 Hz and on the very low frequencies (below 1 Hz).

3) Instantaneous Velocity Calculation: After the filtering process, the instantaneous velocity can be obtained from the integration of the accelerations in world coordinates. For that, we apply the trapezoidal rule, Eq. 3.

$$v[n] = \begin{cases} 0, & \text{for } n = 0 \\ \vec{v}[n-1] + \frac{\Delta t}{2}.(\vec{a}^{\,w}[n] + \vec{a}^{\,w}[n-1]), & \text{for } n \neq 0 \end{cases} \tag{3}$$

where $\vec{a}^{\,w}[n]$ is the n^{th} triaxial accelerometer sample, $\vec{v}[n]$ is the n^{th} obtained velocity vector, which represents the direction, and Δt is the period between each two samples.

Once we have the user directions, we need to create the rotation matrix to project data onto the user-centric coordinate system using the direction obtained from the instantaneous velocity calculation. This can be done by the following vectors:

- $\vec{e}_x[n]$, which takes the three axial vector, divides by its norm and multiplies by the previous one:

$$\vec{v}[n] = \vec{v}[n-1], \text{ if } \|\vec{v}[n]\| \leq \varepsilon$$
$$\vec{e}_x[n] = \frac{\vec{v}[n]}{\|\vec{v}[n]\|}$$

- $\vec{e}_y[n]$, which calculates the cross product between $\vec{e}_x[n]$ and the gravity vector and normalizes it:

$$\vec{e}_y[n] = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \times \vec{e}_x[n]$$
$$\vec{e}_y[n] = \frac{\vec{e}_y[n]}{||\vec{e}_y[n]||}$$

- and $\vec{e}_z[n]$, which is an orthogonal vector to $\vec{e}_x[n]$ and $\vec{e}_y[n]$:

$$\vec{e}_z[n] = \vec{e}_x[n] \times \vec{e}_y[n]$$

Thus, the rotation matrix ${}^{u}\mathbb{R}[n]^{w}$ is given as:

$${}^{u}\mathbb{R}[n]^{w} = \left[\, \vec{e}_x[n] \; \vec{e}_y[n] \; \vec{e}_z[n] \,\right]$$

The next stage consists of rotating each acceleration sample in the world coordinate system $\vec{a}^{\,w}[n]$ according to the obtained rotation matrix and thus, obtaining the user's coordinates. First, the acceleration sample is smoothed to reduce noises from the collection process and sensor. In this filter, we use a triangular Weighted Moving Average (WMA), keeping the bandwidth until the Nyquist frequency. Each filtered acceleration sample $\vec{a}^{\,w}[n]$ is rotated as:

$$\vec{a}^{\,u}[n] = {}^{u}\,\mathbb{R}[n]^{w} \,.\, \vec{a}^{\,w}[n],$$

where $\vec{a}^{\,u}[n]$ is the n^{th} obtained acceleration in the user-centric coordinate system.

In the Supplementary Material along with this submission, we present the Java implementation of all methods discussed in this paper.

V. Case-Study on Gait Authentication, Experiments and Validation

In this section, we show the advantages of adopting the user-centric coordinate system with a case-study on gait authentication. We start describing an Android App developed for collecting input data. Then we explain the acquired dataset and its characteristics analyzing the differences of collected samples represented with world and user-centric coordinate systems. In particular, we discuss a case-study for user authentication based on the gait behavioral trait. In this setup, a user seeks authentication for further use of the mobile device through its specific way of walking.

A. Experimental Protocol

To motivate the difference and advantages of adopting the proposed user-centric coordinate system representation, we have defined a gait authentication setup, in which accelerometer and rotation vector data are obtained from an LG Nexus 5 smartphone. Having the proper institutional review board authorization in our university[1] to collect biometric data from volunteers, each one had to walk with the smartphone in a front pocket jeans for about five minutes. For data collection, we developed an Android App, which samples the accelerometer and rotation vector from the device. The accelerometer is a physic sensor and the rotation vector is a fusion sensor, provided by the Android API, including accelerometer, magnetometer and, in some cases, gyroscope.[2]

After collecting the data, an interpolation and alignment process takes place. This ensures the sampling rate at 40 Hz and guarantees the data from both sensors have the same beginning and ending timestamps and, consequently, the same number of examples. As described in Sec. III, we used linear interpolation for the accelerometer and `Slerp` for the rotation vector. Therefore, the data can be combined to project the accelerometer data onto the world's coordinate system, according to the process described in Sec. IV-B.

The next step consists of guaranteeing the invariance with respect to the movement direction. This is achieved by transforming the acceleration, represented in world coordinates, to the user-centric coordinates. This process was described in Sec. IV-C, which involves applying a low-pass filter and a notch filter on the accelerations (in world coordinates) to remove the low-frequency components and the ones from other movements besides walking. Afterwards, the velocities can be calculated by integrating the acceleration. The obtained velocities are important to provide the movement direction and, therefore, to obtain the rotation matrix. The obtained rotation matrix is used to project the accelerometer data (in world coordinates) onto the chosen user-centric coordinate system. Fig. 4 shows the complete data transformation process.[3]

The collected gait dataset, referred to as *RecodGait*, comprises data from 50 volunteers. For each volunteer, we collect their accelerometer data over two sessions of five minutes each under different acquisition conditions and in different days.

[1] Authorization number 1.459.131 and CAAE 53035216.6.0000.5404 from March, 2016.

[2] The Rotation Vector was introduced in the Android API level 9. Currently, the Android API provides the Game Rotation Vector and the Geomagnetic Rotation Vector. The former does not use the geomagnetic field, thus Y does not point to north but to some other reference, and the latter uses a magnetometer instead of a gyroscope.

[3] The complete code in Java and a walk-through associated documentation is available on https://github.com/amellof/user-centric-coordinates

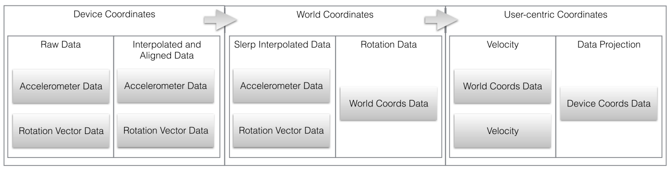


Fig. 4. The data transformation step-by-step.

We further slice the acquired data into frames of 256 samples each. Considering we acquire 40 samples per second, each considered frame comprises 6.4 seconds. Each frame is sampled with a 75%-overlap with the previous one in such a way we can collect more training data for each user.

For the authentication application, we assume a single-user setup for the device as this is a fairly typical scenario in real-world — most mobile devices have just one core authorized owner/user. We train the classifier for an user considering a One-vs-Rest class binarization approach [51], which consists of training a classifier for each positive class of interest (user) against a sample of existing available negative classes (other users). As an example, consider a validation setup with three users Alice, Bob and Charlie in the dataset. With this validation policy, for user Alice, we train a One-vs-Rest classifier with Alice's samples as the positive class and a sampling of the available negative users (Bob's and Charlie's). When focusing on Bob, we would train a classifier taking his samples as the positive class and a sample of the remaining classes (Alice's and Charlie's) as the negative class.

Although it is possible to consider all the training samples from the remaining classes/users available, it often leads to an unbalanced training as we usually have more negative samples than positive samples for a given user. In our setup, the positive class comprises 150 training samples while the negative one consists of 150 samples. Other proportions are possible but following an initial set of experiments, we found out that having balanced sets lead to more accurate results.

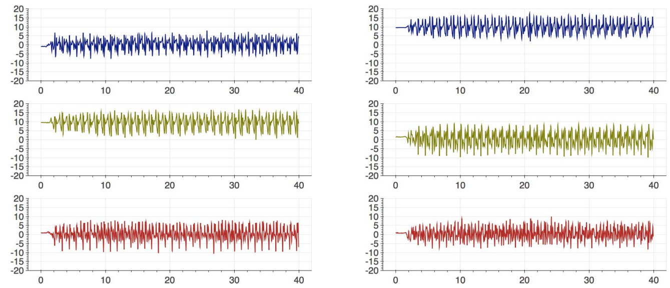
Taking advantage of available temporal data from the accelerometer, we have adopted some methods for temporal fusion in order to improve the user authentication process (continuous authentication). We combine the classification results of the last nine frames, which contain 768 samples in total, with majority voting and max probability [52].

Finally, we perform the experiments considering a 2-fold cross-validation protocol [53]. For finding the best classifier parameters for each user, we deploy an additional 5-fold cross-validation protocol within the training data.

B. Results

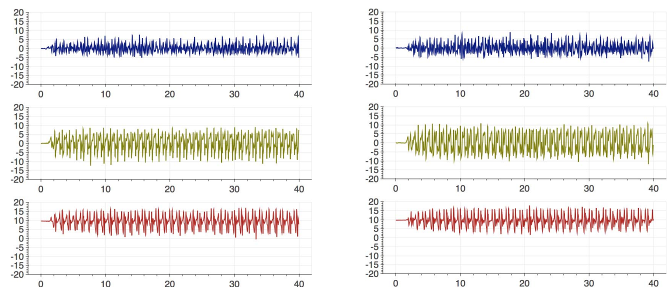
We divided the experiments into two parts. In the first part, we show the the proposed approach is independent from the device position and the environment. The environment is represented by the user left-turns during the data collection process. In the second part, we show the effectiveness of the proposed user-centric coordinate system through a gait authentication setup.

1) Qualitative Analysis of the Method's Steps: To check the coordinate system's position invariance, we show a simple



(a) (b)

Fig. 5. Device coordinates obtained with the device in different positions: (a) device's default position and (b) device's lying down position.



(a) (b)

Fig. 6. Accelerations in world coordinates obtained with the device in default (a) and lying down (b) positions.

Fig. 7. Walking path of the experiment (red-dashed line).

example of a person walking in a straight line and holding the smartphone in her front pocket. The experiment has two parts: first, the device is placed in the default position (see Fig. 3). Then the the device is placed in the lying down position (rotating 90 degrees to the right of the default position). Fig. 5 depicts the accelerometer data in the device coordinate system, in both positions. For reference, the blue line denotes the x coordinate, the grayish-green line the y coordinate and the red line shows the z coordinate, for all the plots presented in this paper. Figs. 5a and 5b show the results of these two experiments, respectively.

Comparing Figs. 5a and 5b, from the x and y point of view, we notice the acceleration, which corresponds to the g acceleration (around 10 m/s^2), has been shifted from y in 5a to x in 5b. This shifting is caused by the device positioning change, in which the acceleration is more evident in the axis parallel to the Earth due to the walking movement.

After transforming the data from the device coordinate system to world coordinates (see Fig. 6), the shifting does

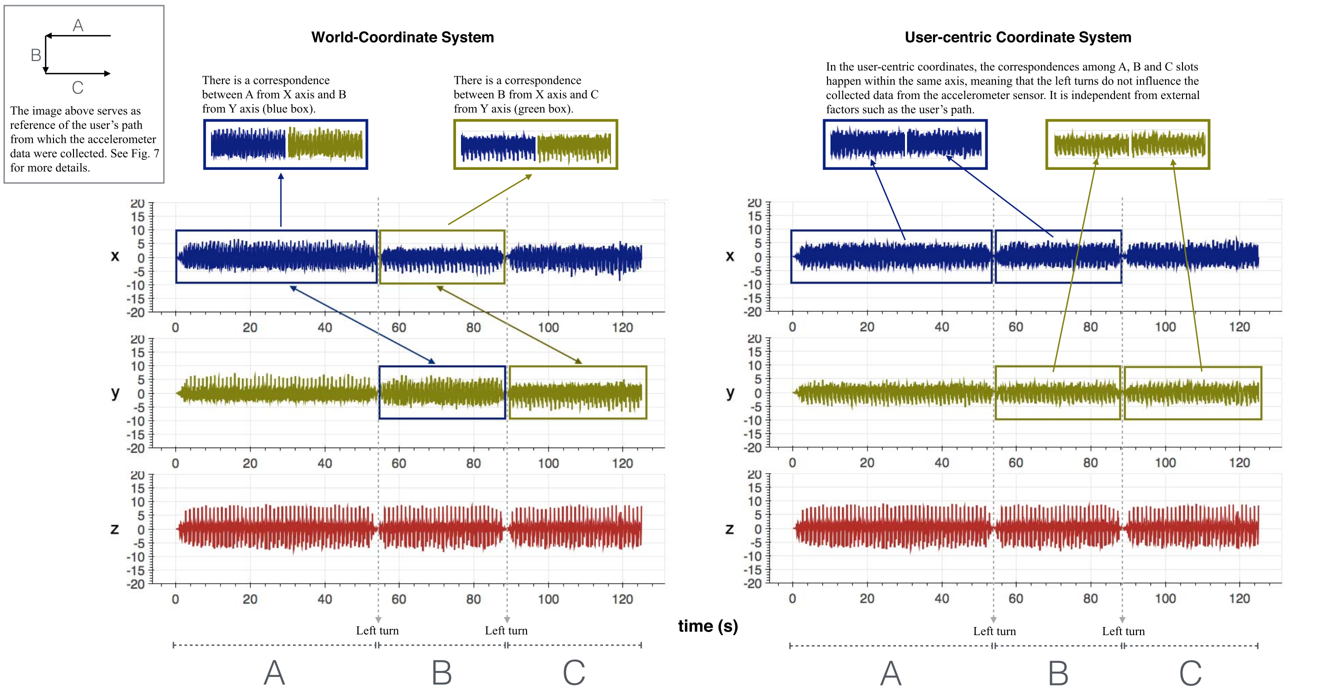
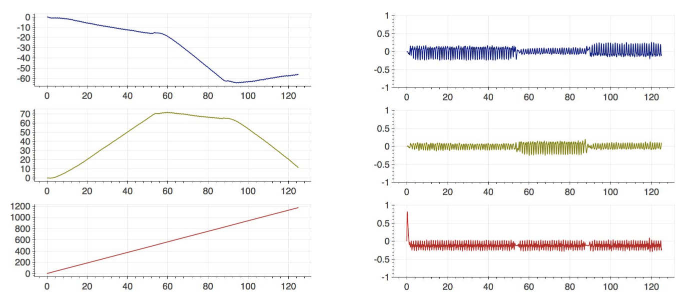


Fig. 8. Accelerations from axis x, y and z obtained from the experiment of Fig. 7 and represented in both coordinate systems: world coordinate system (left) and user-centric coordinate system (right). In this example, we clearly demonstrate that the user's left turns are captured inside axis x and y in the world coordinate system, what do not happen in the user-centric coordinate system.

not appear anymore because the g acceleration is rotated to the axis z, which is the vector perpendicular to the ground plane. Figs. 6a and 6b show the results of applying the transformations to the world coordinate system considering the device in the default and lying positions, respectively.

Nevertheless, when the person walks taking some left/right turns, the accelerations in the world coordinate system suffer variations. To address this problem, the user-centric coordinate comes into play. Thus, if the user is walking through a straight line or has to take some left/right turns, the provided data should not suffer from noise or any other factor that represents this environmental changing. Based on that, we performed another small experiment with the purpose of showing the importance of the user-centric coordinate system. A volunteer was asked to walk for about five minutes around a building. The complete path is highlighted with a red-dashed line in Fig. 7, from which it is important to note the two corners in the path, representing two left-turns.

Fig. 8 depicts acceleration values obtained following the path depicted in Fig. 7. The figure shows the acceleration data represented in the world-coordinate system (left side) and in the user-centric coordinate system (right side). The samples are divided into three straight lines, represented by A, B, and C slots, and two left turns. In the world-coordinate system, after the first user's left turn, we can notice the acceleration data from axis x in slot A moving to axis y in slot B, as blue box shows. This phenomenon happens again after the second user's left turn, from slot B to slot C. It indicates that the direction the user is following is somehow represented in the acceleration data. Because we aim at being as much independent as possible from external factors — such as the



(a) (b)

Fig. 9. Velocities calculated from accelerations in world coordinates, in which we have the data: (a) without notch filter and (b) with notch filter.

user's path — the coordinate systems shall not capture such variations. This is exactly what happens in the proposed user-centric coordinate system, in which the acceleration samples are kept in the same axis even after the user's left turns.

In Sec. IV-C, we mentioned it is necessary to remove very low-frequency components from accelerations represented in the world's coordinate system because they can disturb the calculation of the velocity vectors. Figs. 9a and 9b illustrate the impact of this noise, where the velocity values are changed if the low-frequency components are not removed using the proposed Notch filter.

If the velocities are not correctly obtained, it is not possible to properly project the data onto the user-centric's coordinate system as the movement directions would be incorrect. We have calculated the walking path for both cases, with and without the filter (see Figs. 9a and 9b). This is achieved by calculating the integral of the velocities to obtain the positions.

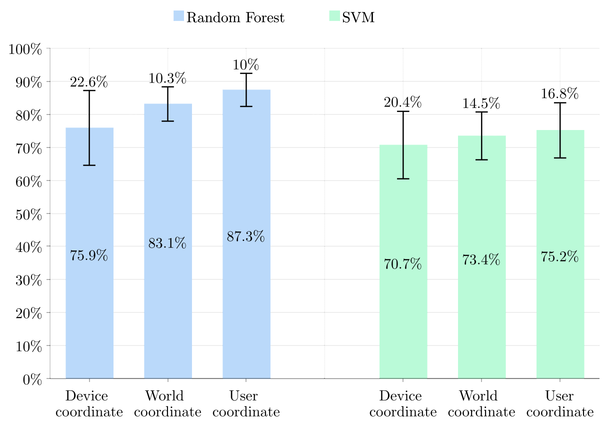


Fig. 10. Summarized results for gait authentication.

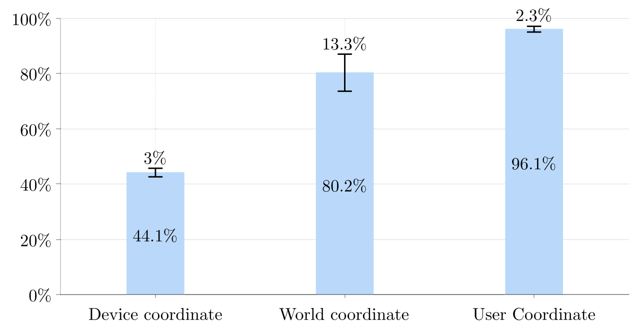


Fig. 11. Analysis with the device on different positions.

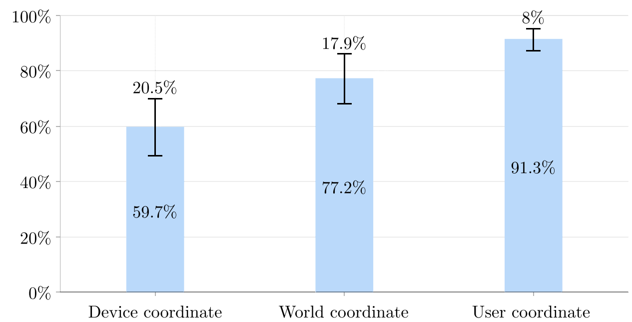


Fig. 12. Analysis with users using different shoes.

This idea is based on the same logic presented in Eq. 3, in which the first user's position is given.

2) Continuous Gait Authentication: In the second part of the experiments, we show the user-centric coordinates provide better input data for feature extraction and classification considering the continuous gait authentication scenario. In this experiment, the data were collected using the protocol described in Sec. V-A, for which 50 volunteers were free to select their own paths, in an arbitrary number of left/right turns during 5-minute acquisition sessions. The only constraint was regarding to stairs and high-slope planes, which could not be part of the walking path at this collection. Fig. 10 shows the obtained results for the authentication considering the device, world and user-centric coordinate systems. We evaluated two classifiers: Support Vector Machines (SVMs) and Random Forests. Each classifier contains the best result considering both majority voting and max probability as the late fusion methods. The results shown in Fig. 10 were obtained using major voting for SVM and max probability for Random Forest. This work does not focus on finding the best possible features that could be extracted/calculated on top of the collected transformed signals. Rather, we opt to train the authentication signals directly with the transformed input signals to show the very impact of each of the coordinate systems.

We present experimental results in terms of average classification accuracy and standard deviation among the 50 users in the dataset. The user-centric coordinate system not only leads to a more accurate classification rate but also reduces the classification variation among different users, which shows a more reliable data description. This is critical in real-world setups when a user collects data in two or more different days, making it difficult to guarantee the device is placed exactly the same way. We also note the results obtained with world coordinates outperform the ones obtained with device coordinates, once they are not invariant to the device's placement. However, the world coordinates fail when the users change directions during the collection process. Again, it is not possible to guarantee the same path in different data collection moments. This problem is solved by the user-centric coordinates, which provide the best results in terms of classification accuracy for both classifiers.

To complement these results, we investigated some case studies to show the importance of the user-centric coordinates. These cases take into consideration the device position and shoes changing. The first set of experiments analyzes the influence of different positions the device can take (default/upright or lying down) inside the user's pocket during different data collection processes. The assumption for this scenario relies on the fact that we cannot guarantee that the device's position is the same during the training and testing phases. To check the impact of the device positioning, we asked five subjects to perform two different walking collections in the same day: the first one with the device in the default position and the second one with the device in the lying position. The goal is to evaluate how the device position influences the different coordinate systems in the user's recognition effectiveness. We used only the Random Forest classifier for this experiment as it has shown better results in all previous experiments. Fig. 11 depicts the results obtained with this case-study in terms of normalized accuracy and standard deviation.

Fig. 11 shows the device position has substantial effect on the device coordinates, which renders the gait authentication process more difficult. The results obtained with the proposed user-centric method have shown higher normalized accuracy and the smallest standard deviation, which indicates our coordinate system is more robust against the device direction changes like the device changing position. Best results are obtained using the user-centric coordinates, as expected.

Another important factor that can influence the success of the authentication process is related to the user's clothes and shoes. More specifically, the next experiment aims at evaluating the user's shoes changes. The idea behind this evaluation relies on the assumption that the user may change some pieces of clothes or the set of shoes during the training and testing collection process. Thus, we carried an additional case-study with five subjects walking during two sessions in the same day, but using different shoes in each session. Fig. 12 depicts the obtained results from this experiment regarding normalized accuracy and standard deviation. In this case, the user-centric coordinates also outperform its counterparts due to the advantages of the invariance to the device's positioning and walking direction. It is important to point out that the standard deviation was considerably reduced, an important indicator of the results variability reduction.

VI. Concluding Remarks

The obtained results with the three different coordinates systems indicate the user-centric coordinate system contains richer and more accurate information about what is happening with the data across the x, y and z axes. The device's coordinate system is sensitive to the smartphone's positioning during the collection process, which renders its use applicable to only a reduced set of controlled setups. The world coordinates, in turn, solves many of the problems presented by the device coordinates. In this work, we detailed how to calculate these coordinates using unit quaternion rotation — more accurate and easier to calculate than traditional methods. Nonetheless, data represented in world coordinates might still contain significant noise from the environment, such as right/left-turns. Other environmental factors could also contribute to degrade the data, such as atmospheric pressure, slopes, steps and so on. The proposed user-centric coordinates come into play to remove this interference and provide the application with cleaner (and more reliable acquired data).

The experiments we carried out with a gait authentication setup show that a deeper analysis of the input signal along with a comprehensive understanding of the device motion sensor data, such as the accelerometer, are key aspects to obtaining reliable results in the posterior steps of an application, such as the feature extraction stage. This is particularly newsworthy when real-case scenarios are taken into consideration with many variables that can not be controlled, such as the device's position or the user's path. Such uncontrolled environment is also present in the introduced *RecodGait* dataset, which provides accelerometer data from 50 volunteers under several different acquisition circumstances. This dataset is an important contribution to the community as there is no available dataset that simulates gait real-case setups with data from several associated sensors.

Considering the results of this research, we can conclude it advances the state of the art of mobile sensors applications in the following aspects: (i) it raises and discusses a key issue of existing device- and world-coordinate systems: their weaknesses when we move to a more realistic scenario encompassing uncontrolled variables; (ii) it introduces a detailed solution through a new coordinate system, referred to as user-centric coordinates, that has shown to be more robust to such conditions by applying a set of filters, data transformation and numerical efficient calculations (`Slerp`) on the accelerometer and rotation vector data considering a gait authentication scenario; and (iii) it is easy to reproduce and to apply in different setups through a step-by-step supplementary material, with code and data fully and freely available to the research community.

Future explorations of this work could include modifications of the sensors fusion stage. Instead of using the Android API rotation vector, one could consider all source sensors directly, such as the magnetic field, and create her own sensor fusion technique to improve some desired features, such as a more accurate North Pole estimation. Another possible improvement is the adoption of predictive models as a replacement technique for the interpolation. Nonetheless, all of these paths come with a price and need to be carefully considered as they may add extra compute time, may slow down the entire proposed pipeline or may require more powerful devices.

In addition, the adopted filters are also an important step that can be changed according to different setups and applications. To the purpose of this work, we have used a windowing low-pass filter to remove frequency components above 2 Hz as we focused on the action of walking. However, it can be replaced by other filters that are more suitable for other applications, according to the kind of movement important to keep/discard.

Finally, we plan to validate the proposed methodology in different applications using smartphone sensors, including fall detection, identification of diseases affecting the way of walking and emotion analyses. We also envision this work to contribute with future research when dealing with mobile sensors, regardless of their applications.

References

[1] *Smartphone Vendor Market Share, 2016 Q2*, accessed on Nov. 10, 2016. [Online]. Available: http://www.idc.com/prodserv/smartphone-market-share.jsp

[2] D. Fuentes, L. Gonzalez-Abril, C. Angulo, and J. A. Ortega, "Online motion recognition using an accelerometer in a mobile device," *Expert Syst. Appl.*, vol. 39, no. 3, pp. 2461–2465, 2012.

[3] A. Khan, N. Hammerla, S. Mellor, and T. Plötz, "Optimising sampling rates for accelerometer-based human activity recognition," *Pattern Recognit. Lett.*, vol. 73, pp. 33–40, Apr. 2016.

[4] C. Catal, S. Tufekci, E. Pirmit, and G. Kocabag, "On the use of ensemble of classifiers for accelerometer-based activity recognition," *Appl. Soft Comput.*, vol. 37, pp. 1018–1022, Dec. 2015.

[5] L. Valcourt, Y. De La Hoz, and M. Labrador, "Smartphone-based human fall detection system," *IEEE Latin Amer. Trans.*, vol. 14, no. 2, pp. 1011–1017, Feb. 2016.

[6] R. Y. W. Lee and A. J. Carlisle, "Detection of falls using accelerometers and mobile phone technology," *Age Ageing*, vol. 40, no. 6, pp. 690–696, 2011.

[7] I. C. Lopes, B. Vaidya, and J. J. P. C. Rodrigues, "Towards an autonomous fall detection and alerting system on a mobile and pervasive environment," *Telecommun. Syst.*, vol. 52, no. 4, pp. 2299–2310, 2013.

[8] M. Derawi and P. Bours, "Gait and activity recognition using commercial phones," *Comput. Secur.*, vol. 39, pp. 137–144, Nov. 2013.

[9] T. Hoang and D. Choi, "Secure and privacy enhanced gait authentication on smart phone," *Sci. World J.*, vol. 2014, May 2014, Art. no. 438254.

[10] Y. Ren, Y. Chen, M. C. Chuah, and J. Yang, "User verification leveraging gait recognition for smartphone enabled mobile healthcare systems," *IEEE Trans. Mobile Comput.*, vol. 14, no. 9, pp. 1961–1974, Sep. 2015.

[11] Y. Zhong, Y. Deng, and G. Meltzner, "Pace independent mobile gait biometrics," in *Proc. IEEE Int. Conf. Biometrics Theory, Appl. Syst.*, Sep. 2015, pp. 1–8.

[12] T. T. Ngo, Y. Makihara, H. Nagahara, Y. Mukaigawa, and Y. Yagi, "The largest inertial sensor-based gait database and performance evaluation of gait-based personal authentication," *Pattern Recognit.*, vol. 47, no. 1, pp. 228–237, 2014.

[13] H. J. Ailisto, M. Lindholm, J. Mantyjarvi, E. Vildjiounaite, and S.-M. Makela, "Identifying people from gait pattern with accelerometers," *Proc. SPIE*, vol. 5779, pp. 7–14, Apr. 2005.

[14] M. R. Hestbek, C. Nickel, and C. Busch, "Biometric gait recognition for mobile devices using wavelet transform and support vector machines," in *Proc. IEEE Int. Conf. Syst., Signals Image Process.*, Apr. 2012, pp. 205–210.

[15] W. Waqar, Y. Chen, and A. Vardy, "Smartphone positioning in sparse Wi-Fi environments," *Comput. Commun.*, vol. 73, pp. 108–117, Jan. 2016.

[16] Q. D. Vo and P. De, "A survey of fingerprint-based outdoor localization," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 491–506, 1st Quart., 2016.

[17] C. Bo, T. Jung, X. Mao, X. Li, and Y. Wang, "Smartloc: Sensing landmarks silently for smartphone-based metropolitan localization," *EURASIP J. Wireless Commun. Netw.*, vol. 2016, no. 1, p. 111, Apr. 2016.

[18] M. A. Alsheikh, A. Selim, D. Niyato, L. Doyle, S. Lin, and H.-P. Tan. (2015). "Deep activity recognition models with triaxial accelerometer." [Online]. Available: https://arxiv.org/abs/1511.04664

[19] C. E. Matthews *et al.*, "Accelerometer-measured dose-response for physical activity, sedentary time, and mortality in us adults," *Amer. J. Clin. Nutrition*, vol. 104, no. 5, pp. 1424–1432, 2016.

[20] D. Sanchez-Valdes and G. Trivino, "Linguistic and emotional feedback for self-tracking physical activity," *Expert Syst. Appl.*, vol. 42, no. 24, pp. 9574–9586, 2015.

[21] R.-K. Shen, C.-Y. Yang, V. R. L. Shen, and W.-C. Chen, "A novel fall prediction system on smartphones," *IEEE Sensors J.*, vol. 17, no. 6, pp. 1865–1871, Mar. 2017.

[22] S. Thiemjarus, "A device-orientation independent method for activity recognition," in *Proc. IEEE Int. Conf. Body Sensor Netw. (BSN)*, Jun. 2010, pp. 19–23.

[23] S. Peisert, E. Talbot, and T. Kroeger, "Principles of authentication," in *Proc. ACM Workshop New Secur. Paradigms*, 2013, pp. 47–56.

[24] S. Mondal and P. Bours, "Continuous authentication in a real world settings," in *Proc. IEEE Int. Conf. Adv. Pattern Recognit.*, Jan. 2015, pp. 1–6.

[25] I. Deutschmann, P. Nordstrom, and L. Nilsson, "Continuous authentication using behavioral biometrics," *IT Prof.*, vol. 15, no. 4, pp. 12–15, Jul./Aug. 2013.

[26] A. E. de Oliveira and G. H. M. B. Motta, "A security API for multimodal multi-biometric continuous authentication," in *Proc. IEEE Int. Conf. Comput. Intell. Secur.*, Dec. 2011, pp. 988–992.

[27] M. O. Derawi, P. Bours, and K. Holien, "Improved cycle detection for accelerometer based gait authentication," in *Proc. IEEE Int. Conf. Intell. Inf. Hiding Multimedia Signal Process.*, Oct. 2010, pp. 312–317.

[28] J. Mantyjarvi, M. Lindholm, E. Vildjiounaite, S.-M. Makela, and H. A. Ailisto, "Identifying users of portable devices from gait pattern with accelerometers," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 2. Mar. 2005, pp. 973–976.

[29] M. Boyle, A. Klausner, D. Starobinski, A. Trachtenberg, and H. Wu, "Gait-based user classification using phone sensors," in *Proc. ACM Conf. Mobile Syst., Appl. Services*, 2011, pp. 1–11.

[30] T. Feng, J. Yang, Z. Yan, E. M. Tapia, and W. Shi, "TIPS: Context-aware implicit user identification using touch screen in uncontrolled environments," in *Proc. ACM Workshop Mobile Comput. Syst. Appl.*, Feb. 2014, p. 9.

[31] A. Roy, T. Halevi, and N. Memon, "An HMM-based behavior modeling approach for continuous mobile authentication," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2014, pp. 3789–3793.

[32] X. Zhao, T. Feng, W. Shi, and I. A. Kakadiaris, "Mobile user authentication using statistical touch dynamics images," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 11, pp. 1780–1789, Nov. 2014.

[33] A. K. S. N. Gowda, S. R. Babu, and D. C. Sekaran, "UMOISP: Usage mode and orientation invariant smartphone pedometer," *IEEE Sensors J.*, vol. 17, no. 3, pp. 869–881, Feb. 2017.

[34] R. Zhang, H. Yang, F. Höflinger, and L. M. Reindl, "Adaptive zero velocity update based on velocity classification for pedestrian tracking," *IEEE Sensors J.*, vol. 17, no. 7, pp. 2137–2145, Apr. 2017.

[35] T. Hoang, D. Choi, and T. Nguyen, "Gait authentication on mobile phone using biometric cryptosystem and fuzzy commitment scheme," *Int. J. Inf. Secur.*, vol. 14, no. 6, pp. 549–560, 2015.

[36] H. Lee, J. Lee, J. Cho, and N. Chang, "Estimation of heading angle difference between user and smartphone utilizing gravitational acceleration extraction," *IEEE Sensors J.*, vol. 16, no. 10, pp. 3746–3755, May 2016.

[37] J. Bojja, J. Collin, M. Kirkko-Jaakkola, M. Payne, R. Griffiths, and J. Takala, "Compact north finding system," *IEEE Sensors J.*, vol. 16, no. 8, pp. 2554–2563, Apr. 2016.

[38] M. Ahk *et al.*, "Reporting accelerometer methods in physical activity intervention studies: A systematic review and recommendations for authors," *Brit. J. Sports Med.*, pp. 1–10, Aug. 2016, doi: 10.1136/bjsports-2015-095947.

[39] M. Joselli and E. Clua, "gRmobile: A framework for touch and accelerometer gesture recognition for mobile games," in *Proc. Brazilian Symp. Games Digit. Entertainment*, 2009, pp. 141–150.

[40] M. S. Lyons and B. M. Kelly, "System and method for bonus gaming using a mobile device," U.S. Patent 8 550 903, Oct. 8, 2013.

[41] S. Strachan, R. Murray-Smith, and S. O'Modhrain, "BodySpace: Inferring body pose for natural control of a music player," in *Proc. ACM Hum. Factors Comput. Syst.*, 2007, pp. 2001–2006.

[42] *Android Online Documentation*, accessed on Nov. 1, 2016. [Online]. Available: https://developer.android.com/

[43] F. S. Grassia, "Practical parameterization of rotations using the exponential map," *J. Graph. Tools*, vol. 3, no. 3, pp. 29–48, 1998.

[44] A. Watt and M. Watt, *Advanced Animation and Rendering Techniques*. Reading, MA, USA: Addison-Wesley, 1992.

[45] A. Sayed, *Fundamentals of Adaptive Filtering*. Hoboken, NJ, USA: Wiley, 2003.

[46] E. B. Dam, M. Koch, and M. Lillholm, "Quaternions, interpolation and animation," Dept. Comput. Sci., Univ. Copenhagen, Copenhagen, Denmark, Tech. Rep. DIKU 98/5, 1998, vol. 2.

[47] K. Shoemake, "Animating rotation with quaternion curves," *ACM SIGGRAPH Comput. Graph.*, vol. 19, no. 3, pp. 245–254, 1985.

[48] S. Sprager and M. B. Juric, "An efficient HOS-based gait authentication of accelerometer data," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 7, pp. 1486–1498, Jul. 2015.

[49] Z. Yuting, P. Gang, K. Jia, M. Lu, Y. Wang, and Z. Wu, "Accelerometer-based gait recognition by sparse representation of signature points with clusters," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1864–1875, Sep. 2015.

[50] S. Smith, *Digital Signal Processing: A Practical Guide for Engineers and Scientists*. Boston, MA, USA: Newnes, 2013.

[51] A. Rocha and S. K. Goldenstein, "Multiclass from binary: Expanding one-versus-all, one-versus-one and ecoc-based approaches," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 289–302, Feb. 2014.

[52] L. I. Kuncheva, "A theoretical study on six classifier fusion strategies," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 2, pp. 281–286, Feb. 2002.

[53] P. Zhang, "Model selection via multifold cross validation," *Ann. Statist.*, vol. 21, no. 1, pp. 299–313, 1993.

Alexandre Ferreira received the B.Sc. degree in computer science from the Engineering School of Piracicaba (EEP) in 2004, the M.Sc. degree in computer science from the Federal University of São Carlos (UFSCar) in 2008, and the Ph.D. degree in computer science from the Politecnico di Milano, Italy, in 2013. He is a Post-Doctoral Research Fellow with the Institute of Computing, University of Campinas, Brazil, since 2015. His main interests include machine learning applied to computational forensics and biometrics, green computing, adaptable systems, and education.

Geise Santos received the B.Sc. degree in computer science from the Federal University of Goiás in 2014, and the M.Sc. degree in computer science from the University of Campinas (Unicamp) in 2017. She is currently pursuing the Ph.D. degree in computer science with the Institute of Computing, Unicamp, Brazil. Her main interests include machine learning applied to computational forensics and biometrics.

Anderson Rocha (SM'15) received the B.Sc. degree in computer science from the Federal University of Lavras, Brazil, in 2003, and the M.Sc. and Ph.D. degrees in computer science from the University of Campinas (Unicamp), Brazil, in 2006 and 2009, respectively. He is currently an Associate Professor with the Institute of Computing, Unicamp. His main interests include digital forensics, reasoning for complex data, and machine intelligence. He is a two-term elected member of the IEEE Information Forensics and Security Technical Committee. He has been an Associate Editor of numerous top-tier journals, such as the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and the IEEE *Security and Privacy Magazine*. He is a Microsoft Research and a Google Research Faculty Fellow. He has been the Principal Investigator of a number of research projects with public and private sponsors in Brazil and abroad having already filed and licensed several patents.

Siome Goldenstein (SM'16) received the degree in electronic engineering from the Federal University of Rio de Janeiro in 1995, the M.Sc. degree in computer science from the Pontifical Catholic University of Rio de Janeiro in 1997, and the Ph.D. degree in computer and information science from the University of Pennsylvania in 2002. He is an Associate Professor with the Institute of Computing, University of Campinas, Brazil. In 2003, he was a Post-Doctoral Fellow with the CBIM Center, Rutgers University, and from 2010 to 2012, he was a Visiting Associate Professor at the Division of Engineering, Brown University. He is an Area Editor of the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, Elsevier's *Computer Vision and Image Understanding*), and Elsevier's *Graphical Models*. His interests lie in computational forensics, computer vision, computer graphics, and machine learning.