Fall simulator for supporting supervised Machine Learning techniques in wearable devices

1st Armando Collado-Villaverde
Computer Engineering
University of Alcala
Alcalá de Henares, Spain
0000-0003-3554-9645

2nd Mario Cobos
Computer Engineering
University of Alcala
Alcalá de Henares, Spain
0000-0003-3981-6245

3rd Pablo Muñoz
Computer Engineering
University of Alcala
Alcalá de Henares, Spain
0000-0003-0581-5383

4th Maria D. R-Moreno
Computer Engineering
University of Alcala
Alcalá de Henares, Spain
0000-0002-7024-0427

***Abstract*—Falls are the predominant cause of injury for older people. How to detect them is being in the last decade the focus of attention of many projects and researchers.**

This paper presents a simulator for recreating triaxial accelerometer measures of people falling in two circumstances: as a consequence of a loss of conscience or due to bumping into an obstacle. The objective of the simulator is to generate falling instances to train Machine Learning algorithms that can be incorporated into wearable devices. The developed simulator can generate triaxial accelerometer measures that exhibit similar patterns compared to falls recorded with real people and mannequins using a commercial device.

***Index Terms*—Machine Learning, Supervised Learning, wearable devices, triaxial accelerometer**

I. Introduction

Falls in the elderly are a public health problem [14]. They are not only a significant source of problems associated with the elderly for their direct consequences (such as trauma), but also because falls are a symptom of infirmity (such as heart attack). For that reason, falls detection has gained importance and currently, several approaches have been developed to detect falls by means of image and video processing [3] or audio detection [6] among others. In our case, in the context of the LARES project [13][1], we focus on fall detection by using the information provided by wireless wearable devices, such as bracelets and watches. We are currently exploiting triaxial accelerometers incorporated in those devices for motion detection tasks. We are especially interested in being able to detect falls in order to, using the LARES architecture, enable a fast response to fall events of elderly persons in their homes.

Fall detection is a common Machine Learning (ML) task [11], both for its relevance in general motion recognition and health monitoring applications. One of the most important issues that this task faces is the scarce amount of recorded falls that could be used to train a ML algorithm. Thus, we propose the use of a video game engine with a realistic physics engine to simulate falls, obtaining a large amount of simulated, yet realistic data, tailored to the problem that we are trying to solve, thanks to the large amount of customization allowed by the engine. Game engines have been used previously in ML tasks due to their physics simulation capabilities, namely, the CARLA simulator [7], used to validate autonomous driving systems, made using Unreal Engine.

Then, in this paper, we propose a fall simulation environment implemented in a free to use game engine that allows to automatically generate falls using body dynamics and configurable physics based on tensors. The proposed simulator implements the two most commons types of falls that can occur at home: syncope and forward. The first type is the consequence of loss of conscience and the second type usually happens while walking. Based on the information generated with this simulator and our previous experiments [5], our intention is to improve our fall detection technique [6] for its implementation into a wearable device.

The rest of the paper is structured as follows. The next section presents the related work. Section III provides an overview of the proposed simulator, presenting the physics model and the implementation. Following, the experimental section shows a comparison between the data created using the simulator and the real data retrieved from an accelerometer (worn by both a mannequin with human-like characteristics and real humans). Finally, some conclusions are outlined.

II. Related works

Biomechanics simulation is a common approach to help investigators in recreating the circumstances that led to an injury [15], for instance, investigating the consequence of a fall [2] or a car hit [1]; or to generate useful information without the risk of injuring people [16].

From the point of view of the physical analysis, we can see the study of Lau et al. [8] where a model to relate the height of a fall can be derived by the severity and extent of the injuries is proposed. The work of Barbuceanu et al [4] presents a physical analysis of falls associated with sport activities. In this case, a model of the bones and the muscular groups of the legs are mathematically modeled and validated through the assessment of high-speed recordings videos of a falling sportswoman.

Focusing on computer-based simulations, we can highlight the reconstruction of real-life head injuries as a consequence of a fall [12]. To do that, the fall is recreated using a multibody

978-1-7281-6799-2/20/$31.00 ©2020 IEEE

[1]An AI-based solution designed to monitor people at home. It is composed of (i) a Wireless Sensor Network; (ii) a low-cost robot; (iii) a Web-based system to provide telecare assistance and telepresence via the robot

modeling software with varying initial conditions in order to find the best fit with the real conditions. The work of Wach and Unarski [17] presents a simulation for helping forensic investigators to determine from where a person fell in a stairwell. They implemented the simulation in PC-Crash[2], reconstructing the fall using the final point of the body by means of a multibody dynamics model.

Meanwhile, human motion and fall assessment are mostly covered by mathematical models and simulations using licensed software, to our knowledge, there are no free to use solutions that allow to easily generate multiple instances of falls. Also, current simulations do not provide enough information about the acceleration during the fall, with the exception of the work of O'Riordain et al. [12], which provides the acceleration at the head.

However, there are some exceptions that have used physics simulations to recreate fall events, such as the work by Mastorakis et al. [10] started in 2007 using OpenSim to simulate a myoskeletal model, further developed in a later work [9] which added the usage of ML to simulate the fall with a higher degree of precision.

Nevertheless, since our objective is to train ML algorithms to detect falls that not necessarily start from a static position, those simulators do not fit our purposes, we need a highly customizable fall simulator from which we can record the measurements that wearable devices such as bracelets or watches would have.

III. The Fall Simulator

Since the most important issue for most ML applications may be the gathering of high quality data to train and test the classifiers, our goal is to simulate falls in order to obtain artificial, yet realistic, accelerometer values to circumvent the inherent difficulties in the acquisition of real falls data. This is due to the health risks that implies the involvement of people falling numerous times. Also, from our previous experience, using dummies to generate this data is not feasible due to the time required and personnel involved.

We have simulated two types of falls, which we have named *syncope* and *forward falls*. Those kinds of falls follow different dynamics in their development, making their implementation needs significantly different in both the starting points and the human reaction during the fall.

- **Syncope falls** are those falls consequence of a loss of conscience or heart condition that prevents using the limbs to control the fall. It results in a vertical motion and a two stages fall: first, the knees impact on the ground, and then the trunk moves forward until it hits the ground. For this kind of falls, we did some testing on a dummy [5] under the supervision of infirmary experts that helped us to understand how this kind of fall would develop on a real person. We established the need to apply a small forward force onto the dummy for it to fall properly in case the person was standing still when the loss of conscience happened. Thus we applied a similar kind of force onto the simulator, introducing some noise in both the direction of the force and its intensity to achieve a higher variability over our recorded data, yet discarding every instance that would be unnatural. The development of this kind of falls on the dummy can be observed in Fig. 1. In this scenario, in our opinion, is better to obtain the data by the use of a non-human source, either a dummy or our simulator, since human reflexes could contaminate the data recordings by trying to protect themselves from the damage of the fall. The use of our approach makes the application of the necessary force more consistent and greatly reduces the required human effort to record the data compared to the use of a dummy.
- **Forward falls** are originated by the collision of a foot with an object while the person is walking, losing balance and falling over. The trunk in this type of falls moves forward, while the arms and hands try to cushion the fall in order the prevent further damage.

Fig. 1: Recreation of a syncope fall using a dummy.

The next subsections describe the physics and the implementation of these falls in our simulator.

A. Physics

The simulation of the process of tripping and falling was provided by the already built physics engine employed. The result said simulation was a set of 3D coordinates at different points in time. It was, then, necessary to establish the physical significance of this data in order to properly parse and use it.

For that purpose, we first studied the analytical implications of the generated data. The simulated accelerometer, in charge of providing these discretized positions, could be interpreted as a particle moving in a 3D space. Given that this movement is continuous in time and space, we can describe it using the generalized tensor function presented in Eq. 1. Therefore, we establish a correlation between the position of the particle or accelerometer at any given point in time with its position in the time step immediately preceding it and the variation of said position between both times, always expressed as a 3-dimensional tensor.

$$\vec{P}(t) = \vec{P}(t-1) + \frac{\partial \vec{P}}{\partial t}(t) \tag{1}$$

Following this reasoning, it stands to reason that we can use the physical definition of velocity in order to further simplify

[2]https://pc-crash.com/

the problem. We know it to equal the variation of position over time, and can, therefore, describe it as the gradient of the function $\vec{P(t)}$. Moreover, provided that velocity is not static, we can approximate its value by adding its variation at any given time, $\frac{\partial \vec{V}}{\partial t}(t)$, to its value in the previous time step, resulting in the relationship described in Eq. 2.

$$\vec{V}(t) = \frac{\partial \vec{P}}{\partial t}(t) = \vec{V}(t-1) + \frac{\partial \vec{V}}{\partial t}(t) \tag{2}$$

Finally, classical dynamics describe the variation of velocity over time as acceleration. We can, therefore, apply the same reasoning we used to mathematically describe acceleration, by assuming its value to be the variation in speed. Given the nature of the physical system in question, this value will not be constant, so we must, therefore, take into account its variation in regards to time, as well as its value in the immediately preceding time step, as described in Eq. 3.

$$\vec{A}(t) = \frac{\partial \vec{V}}{\partial t}(t) = \vec{A}(t-1) + \frac{\partial \vec{A}}{\partial t}(t) \tag{3}$$

$\vec{P}(t)$ may be approximated using an interpolation over a sufficiently large dataset, and serve as a base for further analysis. Whilst this method will inherently cause an error to appear, said error is minute enough to be affordable for the purpose of this simulation, that is, to generate significant enough data to extract acceleration patterns related to the process of a human being tripping and falling down.

Having established these considerations, it is now possible to analytically describe the state of the system using discrete data, sampled from the accelerometer's state. It is important to consider that the simulation provides us with the location of the accelerometer at a given point in time, and not the values actually registered for the device. In order to extract this information, we can use the properties described in Eqs. 1-3, as presented in Eq. 4.

$$\begin{aligned} \vec{P}(t) = \vec{P}(t-1) + \frac{\partial \vec{P}}{\partial t}(t) = \\ \vec{P}(t-1) + \vec{V}(t) = \vec{P}(t-1) + \vec{V}(t-1) + \frac{\partial \vec{V}}{\partial t}(t) = \\ \vec{P}(t-1) + \vec{V}(t-1) + \vec{A}(t) \\ \Rightarrow \vec{A}(t) = \vec{P}(t) - \vec{P}(t-1) - \vec{V}(t-1) = \vec{V}(t) - \vec{V}(t-1) \end{aligned} \tag{4}$$

Having established these theoretical considerations, it was possible to devise a proper, computationally efficient software approximation, that would provide reliable, accurate data, without sacrificing excessive resources in the process.

B. Implementation

For the implementation of the simulator[3] we have decided to use Unity3d[4] game engine which incorporates NVidia's Physx System Software[5] for complex physics calculations.

[3]https://github.com/ISG-UAH/ISG_FallSimulator

[4]https://unity.com

[5]https://developer.nvidia.com/gameworks-physx-overview

We have used a human-like 3D model, downloaded from mixamo[6] and imported into Unity. Then, Unity's ragdoll wizard was used as a starting point; however, some adjustments were needed, such as the addition of colliders in the feet, as well as the hands, with their respective articulations and the subsequent redistribution of the humanoid weight. Finally, the rotation capabilities of each articulation were restricted so they were similar to a human's.

To simulate both types of falls, we have started from an animation not influenced by physics. In the case of *forward falls*, the starting animation was one of several possible walking animation towards an obstacle; and for *syncope falls* the starting animation could be of any kind. Used animations range from walking animations to usual everyday actions such as talking or manipulating objects, and idle animations where the humanoid is mostly standing still with minimal motions. Since Unity's physics engine works in a non-deterministic way, each fall is different from the rest even if the starting parameters, such as the position of the obstacle with which the humanoid will collide or the last pose of the animation before the physics are activated, are the same.

In the case of a *forward fall*, while the animation is being played, we simulate the fall. We start playing one of several different walking animations until the humanoid collides with the obstacle placed in front of it, randomizing its distance, height, and orientation in each fall. At that moment the physics engine takes over. We apply forces on the arm and hands, considering their natural movement capabilities, to simulate the response that a person would have in this situation trying to cushion the impact. Those forces are applied for a few milliseconds (randomized between two limits) after the foot collides with the obstacle in order to simulate the reaction time of the person. In addition, since the walking animations are all different from each other and the starting point of each fall is also affected by the randomized position of the obstacle, we can be sure that each simulated fall will be different from the rest. In order to simulate the force that the person would have in the moment of the fall due to walking, a forward force is applied to each limb. An example sequence of the forward fall in the simulator can be observed in Fig. 2.

Regarding the simulation of *syncope falls*, we started from animations of different activities, such as talking, walking, jumping, moving objects, and several other usual tasks as well as instances where the humanoid was standing still playing an idle animation. Then, we stop playing the animation and let the physics engine take over, simulating the loss of conscience, and the subsequent fall of the humanoid. In this case, it is required to apply a small forward force in order for the humanoid to fall like a person would in this kind of situation. An example of a syncope fall recreated in the simulator can be seen in Fig. 4.

To obtain a higher degree of variability in the recorded data we decided to add some randomness to every force that was applied to the humanoid with the objective to make the falls

[6]https://www.mixamo.com

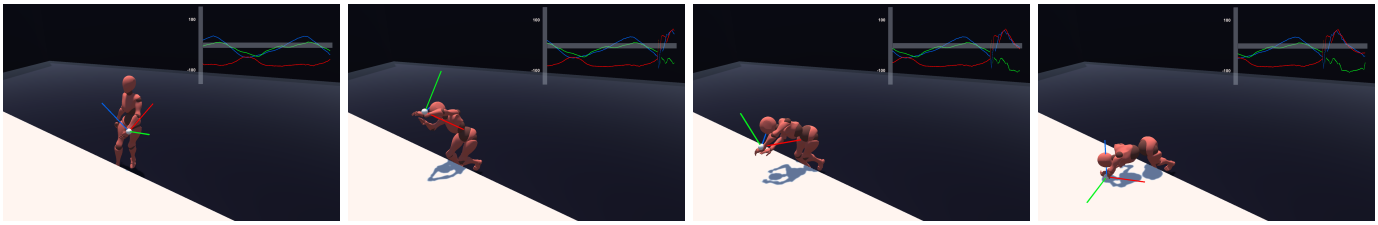

Fig. 2: Recreation of a forward fall in the simulator. Triaxial accelerometer measures are provided in the top-right.

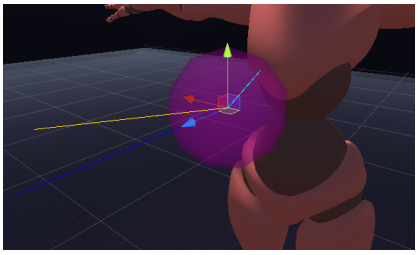

Fig. 3: Forces decomposition in the simulator. In *blue*, original force; *cyan*: random point added to the original force; *yellow*: normalized addition of previous forces (final force applied).

more natural. We slightly modified both the direction and the intensity of the forces, which can be thought of as a vector. The module of the vector will be multiplied by a random value between two limits, and the direction will be modified by adding a random point inside a small sphere and normalizing it afterwards as shown in Fig. 3, where the blue vector represents the original force, the cyan one the random force applied, and the yellow one the final force applied to the fall. This modification, in addition to the non-deterministic nature of Unity's physics engine, gives us a rich variety of recorded falls.

For data acquisition, in both cases, we have recorded the position that a smartwatch would have when placed on the wrist of the humanoid at fixed time steps of 20ms. This is the usual maximum frequency that a commercial-grade accelerometer operates at. In addition, to simulate gravity, we have calculated the direction of a vector pointing towards the ground in the local space of the wrist in each time step. Those positions and directions are later processed as explained in III-A and transformed into the values that a real-life accelerometer would have obtained if it followed the same motions.

In order to perform said values transformation, it is necessary to adapt the logic applied in Eq. 1 through 4 to a discrete, sampled system. Provided that the simulation would be outputting specific positions, as measured by the simulated accelerometer, as well as the time between different measures being taken, it is possible to calculate the displacement vector between individual time steps and obtain the velocity the simulated accelerometer was subjected to during it.

Let $\vec{v}$ represent the resulting velocity vector, $\Delta\vec{p}$ represents the displacement vector of the accelerometer between time steps, and dt represents the time difference between time steps, then Eq. 5 can be used to calculate the resulting velocity vector from the simulator's raw data.

$$\vec{v} = \frac{\Delta\vec{p}}{dt} \tag{5}$$

Likewise, provided that we can obtain the velocity vector between time steps, let $\vec{a}$ represents the acceleration vector for the accelerometer in a given time step, $\Delta\vec{v}$ the velocity difference between time steps, represented as a vector, and dt the time difference between time steps, we can calculate the acceleration vector, and therefore the simulated accelerometer output, for any given time step using Eq. 6.

$$\vec{a} = \frac{\Delta\vec{v}}{dt} \tag{6}$$

It should be noted that the resulting dataset will present clean data. Therefore, actual sensor readings must be filtered in order to be properly compared. Likewise, in order to simulate actual, noisy sensor data, a noise function, such as a Gaussian noise function, must be applied over the result.

IV. Experimental results

In this section, we show a comparison of our simulated data falls and the real data falls used in a previous work [5]. Those falls were supervised by geriatric experts that ensured the correct development of the fall (from an elder perspective), discarding those that were not natural.

Fig. 5 presents a comparison between a *syncope fall* using a realistically built dummy humanoid to simulate falls of an elderly person (left) and one recreated in our simulator (right). Numerous factors impact the readings gathered from both accelerometers. On the one hand, the force exerted on the dummies, as well as their not fully reacting as a human body, cause readings to look more abrupt. On the other hand, the fact that the simulated fall presents an initial pose causes additional differences in the readings prior to the fall proper. Moreover, the initial rotation of the accelerometer further alters the resulting readings. Despite these differences, it is our belief that a sufficient amount of samples generated by the simulator may still provide insight into the acceleration patterns involved in a *syncope fall*.

In Fig. 6 we can see that the synthetic data gathered from the simulator (right) presents some common patterns to that obtained from an actual accelerometer (left) during a *forward fall*. Nevertheless, there are some notable differences between

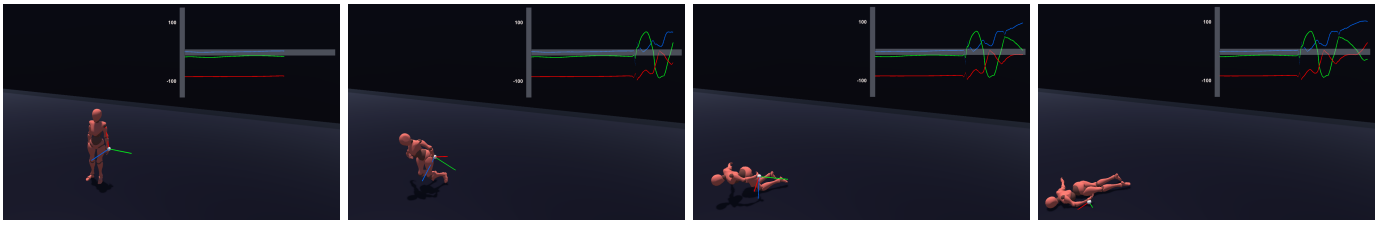

Fig. 4: Recreation of a syncope fall in the simulator. Triaxial accelerometer measures are provided in the top-right.

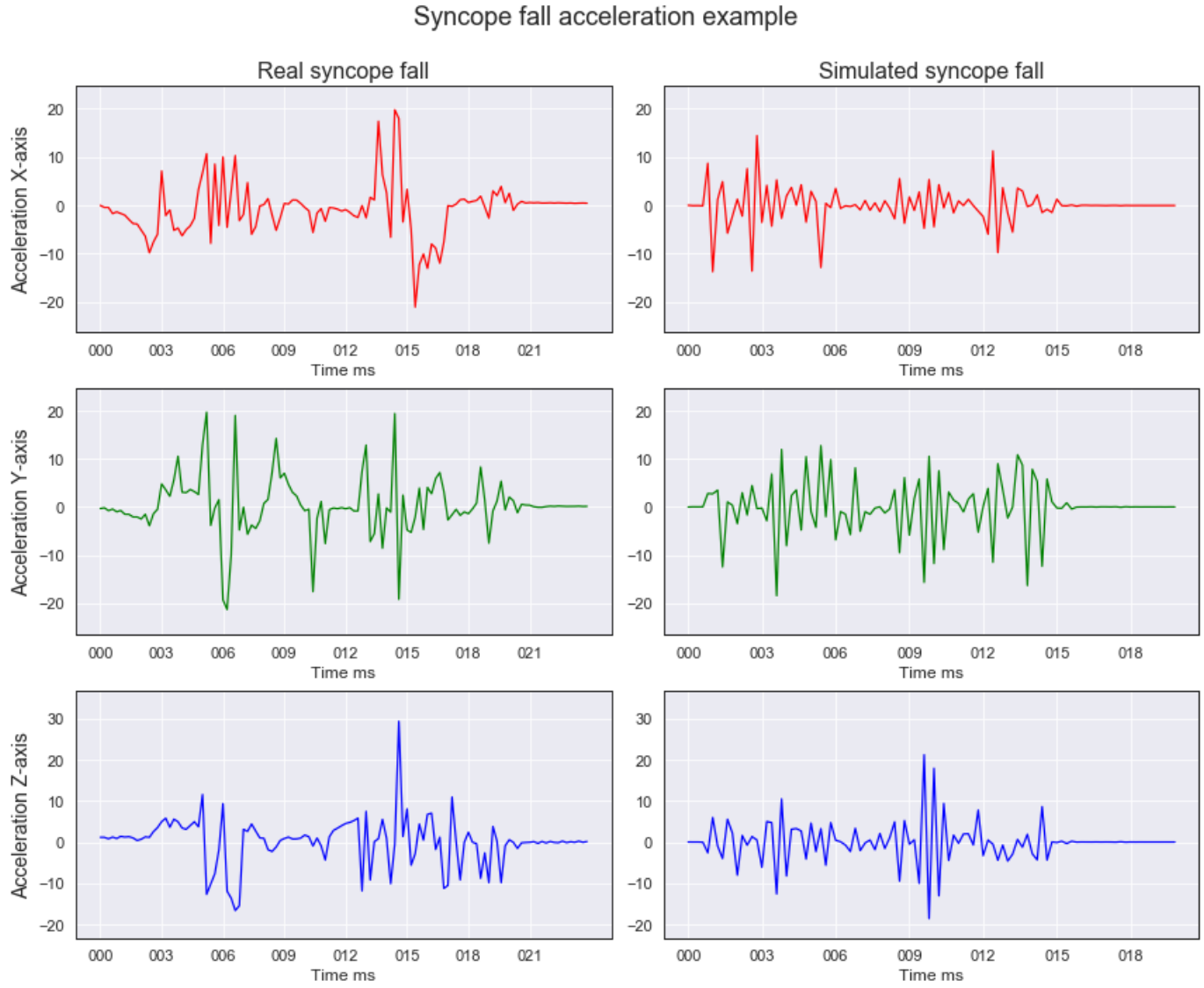

Fig. 5: Comparison between triaxial accelerometer data of a syncope fall retrieved from our experiments with dummies (left) and the generated in the simulator (right) for a syncope fall. Y axis has been normalized for all figures in order to facilitate results comparison.

real and simulated data. This is due to the method employed to gather said data.

Namely, these differences come from the inability to fully replicate human gait and reactions when tripping. It should be noted that the data from real sensors was gathered in a cushioned environment. It is to be expected that some major differences will surface in the readings after the fall is complete. Another notable source of differences is the presence of random forces exerted by the falling subject instinctively in response to falling. Finally, human gait presents a wider swing

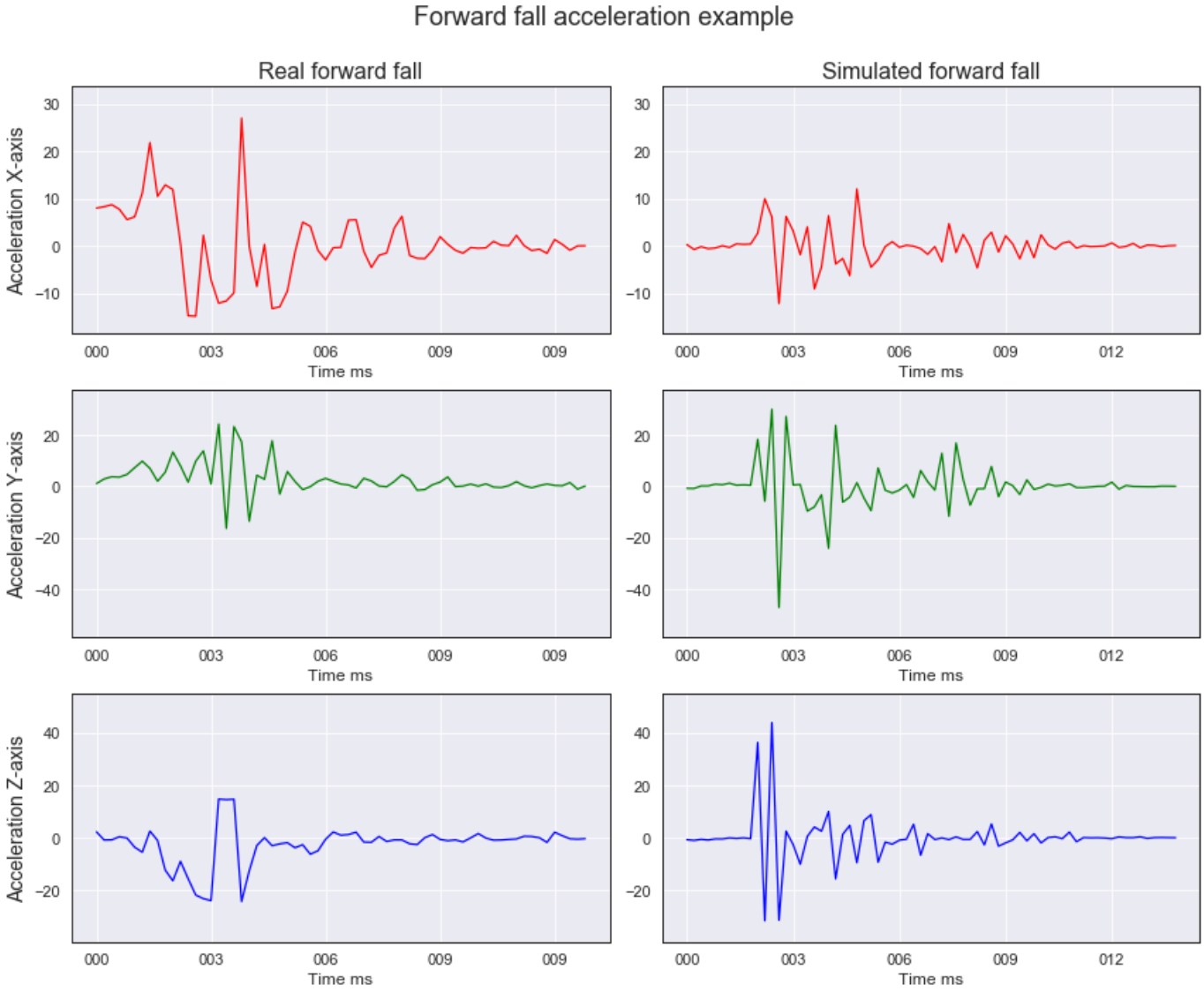

Fig. 6: Comparison between triaxial accelerometer data of a forward fall retrieved from our experiments with real people falling over a soft ground (left) and the generated in the simulator (right) for a forward fall. Y axis has been normalized for all figures in order to facilitate results comparison.

than the animations present in the simulator, thus resulting in the more ample data present prior to the fall when compared to the simulated readings.

The nature of a fall as a time series might be enough to offset the deviations in simulated data. Further refinement of the simulator itself ought to lead to a more robust simulation platform to generate synthetic fall data.

There are some important considerations regarding the specific problem of working with fall events and using accelerometers as the only source of information. Namely, fall events do not have a fixed duration, and the starting pose plays a major role in the resulting series of vectors. These characteristics, make the usage of traditional time series similarity metrics, such as the Pearson Correlation Coefficient or Dynamic Time Warping, that evaluate each axis separately of little value. Instead, the usage of ML classifiers offers a better approach to evaluate the degree of fidelity achieved by the simulator.

In order to validate the simulator's performance, we classified 40 fall instances (30 forward and 10 syncope) using the classifiers from our previous experiment [6] trained with real fall data. Over 94% accuracy was achieved running a Random Forest classifier, asserting its performance.

V. Conclusions

In this paper, we have presented a fall simulator capable of recreating accelerometer fall samples of two of the most common and important types of falls, syncope and forward. Those simulated samples are pretty similar to real falls recorded in order to use them later as input for ML applications.

This can be considered a starting point for simulating other motion detection tasks using the same procedure, as well as expanding further in the usage of accelerometers by either increasing the number of sensors or attaching them to other parts of the body. By increasing the customization possibilities of the dummy, other kinds of falls, or behaviors during them, could be simulated, further improving the simulator capabilities. Likewise, the physics engine could be further refined in order to provide more accurate, realistic data to be used in training algorithms.

ACKNOWLEDGMENT

The authors thank the contribution of Isabel Pascual Benito, Francisco López Martínez and Helena Hernández Martínez, from the Department of Nursing and Physiotherapy of the University of Alcalá, for their help designing and supervising the simulated falls procedure. Armando Collado is co-supported by the European Social Fund and Junta de Comunidades de Castilla-La Mancha under the *Garantía Juvenil* (SBPLY/18/180501/000019). Marío Cobos is co-supported by Comunidad de Madrid *Garantía Juvenil* (PEJD-2018-PRE/TIC-8176).

This work is supported by the JCLM project SBPLY/19/180501/000024 and the Spanish Ministry of Science and Innovation project PID2019-109891RB-I00, both under the European Regional Development Fund (FEDER).

REFERENCES

[1] A. Moser, H.S., Kasanický, G.: The pedestrian model in pc-crash – the introduction of a multi body system and its validation. SAE Transactions **108**(1), 794–802 (1999)

[2] Adamec, J., Jelen, K., Kubovy, P., Lopot, F., Schuller, E.: Forensic biomechanical analysis of falls from height using numerical human body models. Journal of Forensic Sciences **55**(6), 1615–1623 (Nov 2010). https://doi.org/10.1111/j.1556-4029.2010.01445.x

[3] Auvinet, E., Multon, F., Saint-Arnaud, A., Rousseau, J., Meunier, J.: Fall detection with multiple cameras: an occlusion-resistant method based on 3-D silhouette vertical distribution. IEEE transactions on information technology in biomedicine **15**(2), 290–300 (2011). https://doi.org/10.1109/TITB.2010.2087385

[4] Barbuceanu, M., Bărbuceanu, D., Giosanu, D., Iorga-Simăn, I.: The biomechanical analysis of standing fall. Romanian Journal of Physics **51**(3-4), 379–389 (Dec 2006)

[5] Collado-Villaverde, A., D. R-Moreno, M., F. Barrero, D., Rodríguez, D.: Detección de caídas mediante un acelerómetro de tres ejes ubicado en la muñeca en personas de tercera edad. In: Actas de la XVII Conferencia de la Asociación Española para la Inteligencia Artificial. pp. 29–38. CAEPIA 2016, Salamanca, Spain (Sep 2016)

[6] Collado-Villaverde, A., D. R-Moreno, M., F. Barrero, D., Rodríguez, D.: Triaxial Accelerometer Located on the Wrist for Elderly People's Fall Detection. In: Procs. of the 17th International Conference of Intelligent Data Engineering and Automated Learning – IDEAL 2016. pp. 523–532. Cham, Yangzhou, China (Oct 2016). https://doi.org/10.1007/978-3-319-46257-8_56

[7] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An open urban driving simulator. In: Proceedings of the 1st Annual Conference on Robot Learning. pp. 1–16 (2017)

[8] Lau, G., Ooi, P.L., Phoon, B.: Fatal falls from a height: The use of mathematical models to estimate the height of fall from the injuries sustained. Forensic Science International **93**(1), 33–44 (Apr 1998). https://doi.org/10.1016/S0379-0738(98)00027-9

[9] Mastorakis, G.: Human fall detection methodologies: from machine learning using acted data to fall modelling using myoskeletal simulation. PQDT - UK & Ireland (July) (2018)

[10] Mastorakis, G., Hildenbrand, X., Grand, K., Makris, D.: Customisable fall detection: a hybrid approach using physics based simulation and machine learning. IEEE Transactions on Biomedical Engineering **54**(11), 1940–1950 (2007). https://doi.org/10.1145/2769493.falls

[11] Noury, N., Fleury, A., Rumeau, P., Bourke, A.K., Laighin, G.O., Rialle, V., Lundy, J.E.: Fall detection - principles and methods. In: Procs. of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. pp. 1663–1666. Lyon, France (Aug 2007). https://doi.org/10.1109/IEMBS.2007.4352627

[12] O'Riordain, K., Thomas, P., Phillips, J., M.D.Gilchrist: Reconstruction of real world head injury accidents resulting from falls using multibody dynamics. Clinical Biomechanics **18**(7), 590–600 (Aug 2003). https://doi.org/10.1016/S0268-0033(03)00111-6

[13] Ropero, F., Vaquerizo-Hernández, D., Barrero, D.F., R-Moreno, M.D., Muñoz, P.: LARES: An AI-based teleassistance system for emergency home monitoring. Cognitive Systems Research **56**, 213–222 (Aug 2019). https://doi.org/10.1016/j.cogsys.2019.03.019

[14] Sadigh, S., Reimers, A., Andersson, R., Laflamme, L.: Falls and fall-related injuries among the elderly: a survey of residential-care facilities in a swedish municipality. Journal of Community Health **24**, 129–140 (Apr 2004). https://doi.org/10.1023/b:johe.0000016717.22032.03

[15] Sloan, G., Talbott, J.: Forensic application of computer simulation of falls. Journal of Forensic Sciences **41**(5), 782–785 (Sep 1996). https://doi.org/10.1520/JFS13997J

[16] Snyder, R.G., Foust, D.R., Bowman, B.M.: Study of impact tolerance through free-fall investigations. Tech. rep., Insurance Institute for Highway Safety, Washington, D.C., USA (Dec 1997)

[17] Wach, W., Unarski, J.: Fall from height in a stairwell – mechanics and simulation analysis. Forensic Science International **244**, 136–151 (Nov 014). https://doi.org/10.1016/j.forsciint.2014.08.018