Improving Fall Detection by the Use of Depth Sensor and Accelerometer

Bogdan Kwolek*

AGH University of Science and Technology, 30 Mickiewicza Av., 30-059 Krakow, Poland

Michal Kepski*

University of Rzeszow, Pigonia 1, 35-310 Rzeszow, Poland

Abstract

Since falls are a major cause of harm to older people, there is considerable demand for low-cost fall detection systems. To meet demands of the end-users we propose a new architecture for low cost and reliable fall detection, where an accelerometer is used to indicate a potential fall and the Kinect sensor is used to authenticate the eventual fall alert. In consequence, the depth maps are not processed frame-by-frame, but instead we download from a circular buffer a sequence of depth maps acquired prior to the fall and then process them to authenticate fall event. We determine features both in the depth maps and point clouds to extract discriminative fall descriptors. Since people typically follow typical motion patterns related to specific locations in home or typical daily activities, we propose to utilize k-nn classifier to implement an exemplar-based fall detector. We show that such a classifier is competitive on our publicly available URFD dataset in terms of sensitivity and specificity while being much more simple to implement on an embedded platform.

Keywords: Human Activity Recognition; Fall Detection; Home Care; Smart Home.

1. Introduction

People are living longer, and many forecasts make it clear that elderly people will have to live independently in their own homes for as long as possible. One of the highest risks of loss of independence for elderly persons living alone or spending much time alone is falling down [1]. Moreover, the risk of falls increases markedly with age, slower reaction and balance, and reduced muscle strength. Thus, approximately one out of every three seniors falls in any given year, and these sudden falls are the most common cause of injury and hospital admissions among this age group.

To extend the possibilities for independent living of the seniors, several smart home technologies [2] and smart cameras [3] have been proposed until now. In context of prolonged independent living, fall detection is an important task [1]. Medical alert systems with fall detection include simple push-button devices and accelerometer-based wearable systems. However, their applicability is restricted to limited markets like nursing homes rather than the broader aged communities. In context of independent living, currently available wearable systems are not acceptable by primary end-users, especially those who are not impaired. One of the most common reasons for disallowance of accelerometer-based assistive devices is their high false alarm ratio. This means that some daily activities are wrongly reported as falls, which in turn leads to frustration of the users.

A recent survey [4] demonstrates that the Kinect sensor can be very useful in detecting falls. However, the available algorithms for fall detection are not robust and do not exhibit both high sensitivity and specificity. Moreover, since such systems process, frame-by-frame, the depth maps acquired by the Kinect, they typically require a considerable processing power, and thus they usually need a PC or notebook computer. In order to keep low the number of false alarms as well as to reduce the computational burden we propose a novel architecture for fall detection, see Fig. 1. In our approach, the depth maps are stored in a circular buffer for an authentication of the fall if needed, whereas a threshold-based accelerometer module releases a depth map-based verification of the hypothesis about a potential fall. In contrast to the existing approaches [4], we extract the features not only in depth maps but we also process point clouds to extract very discriminative fall descriptor. Since people typically follow individual and distinctive at the same time motion patterns, related to specific locations in home or typical daily activities, we propose to utilize k-nn classifier to implement an exemplar-based fall detector.

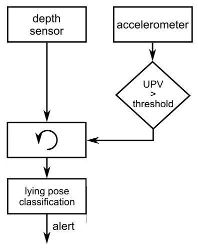

Figure 1: Proposed architecture for reliable fall detection.

The rest of the paper is organized as follows. In Section 2 we discuss related work that was accomplished in the area of fall detection. Section 3 is devoted to presentation of our embedded system for fall detection. In Section 4 we detail real-time

*Corresponding author: bkw@agh.edu.pl

and energy efficient data processing. The descriptors for distinguishing between fall and daily activities are discussed in Section 5. In Section 6 we present our URFD dataset as well as we discuss the classifier for distinguishing between daily activities and falls. The experimental results are discussed in Section 7. Section 8 provides some concluding remarks.

2. Relevant work

A range of body-attached sensors including goniometers, accelerometers, gyroscopes, pedometers, and actometers have been used to capture and analyze human movement [1]. Accelerometers offer a number of advantages in monitoring of physical human movements. Advantages of such sensors include a high accuracy even in noisy measurements as well as acceleration measurement down to zero Hertz. Rapid development in Micro-Electro-Mechanical Systems (MEMS) resulted in miniaturized and low cost accelerometers. These features have made possible development of small, lightweight, portable smart devices that can be worn without hindering physical activity. The smartphones and smartwatches are examples of mobile devices, which are equipped with accelerometers and which can be used to perform unobtrusive fall detection.

Majority of the accelerometer-based fall detection systems that were developed in the past investigates the use of a body-worn tri-axial accelerometer with a threshold algorithm [5]. A recent study [6] conducted by an international team of researchers evaluated the effectiveness of these algorithms to detect fall events within a database of real falls. The database contains accelerometer measures that capture the movements of participants, each for a period of two days. In all, it stores data from 29 real falls. Thirteen different algorithms were investigated to see if they were able to identify the real falls. Unfortunately, none of the investigated algorithms scored sufficiently high in both sensitivity (the ability to properly recognize falls that in reality occurred) and specificity (the capability to correctly identify a movement as a non-fall). Two algorithms achieved good scores on both of these measures, but each would create too many false alarms if employed as an automated fall detector. One of the main reasons for high false ratio of accelerometer-based systems is the lack of adaptability together with insufficient capabilities of context understanding. In consequence, they have difficulties in distinguishing fall events from typical daily activities, for example, lying down on the couch to relax, bending down to play with a pet, bending to pick up an object from the floor, or even just lying down to sleep.

Video monitoring systems use cameras that attempt to detect a fall acting on image-processing algorithms, which are designed to identify unusual activities. The main advantage of such systems is that the person does not need to wear any special device. However, this type of fall-monitoring is both the most expensive and most intrusive form of fall detection due to the fear of intrusion of privacy. Although many solutions for preserving privacy have been developed, people in monitored rooms still experience the feeling of being-watched, thus making the ordinary CCD cameras unacceptable in most cases, and especially in the bedroom and the bathroom. Moreover, while CCD-camera based techniques might work well in controlled environments, in order to be practically applied they ought to be adapted to non-controlled environments in which neither the lighting nor the resident tracking is fully controlled. Thus, such devices can not work in nightlight or low light conditions. Additionally, the lack of depth information might lead to lots of false alarms. Nevertheless, due to recent developments in smart camera [3] and smart home [7] technology, the CCD camera-based solutions have some potential to be utilized in smart fall detectors.

As demonstrated several years ago, the cameras delivering in real-time the depth information can be very helpful in detecting and tracking faces and heads [8]. The head trajectory is a very useful source of information for behavior recognition and can be greatly advantageous for video surveillance applications, especially for fall detection [9]. Another promising research direction in this domain is the use of multiple omnidirectional cameras to observe and to track the inhabitants of a room [10]. Overall, the omnidirectional cameras are very useful in areas where large visual field coverage is needed, whereas stereo-pairs deliver very advantageous 3D information. Thermal imaging cameras, also called infrared cameras, which detect the heat given off by an object or human can also deliver very valuable source of information for detecting falls [11].

Recently, Kinect's depth camera has been proposed to be utilized in fall detection [12] [13]. As demonstrated in the discussed work, depth information is sufficient to detect person undergoing monitoring. Since Kinect uses infrared light sensors to illuminate the objects in front of it and an infrared camera to observe them in invisible light, the fall detection can be done any time. In contrast to the discussed work, the algorithms presented in [14][15] rely on the 3D skeleton, which is automatically extracted by Kinect for Windows SDK/OpenNI-NITE framework. However, given that a person can be in any pose prior to a fall, it is very likely that the skeleton extraction may fail to acquire the skeletal model, or be unreliable in the period of the fall motion [16]. In [17] a network of multiple Kinect sensors is installed in different areas of a house monitoring the individual. However, it is unclear how a collaboration between the cameras under the communication and latency constraints, which is an important problem in camera networks [3], has been solved in the discussed work.

One way to improve the reliability of detection of emergency situations is to combine video/depth and accelerometer signals, as proposed recently [18] [13] [19]. Recent work [20][21] demonstrates that combining the depth with inertial sensors improves the human activity recognition. In this work, we demonstrate that the thresholded accelerometer signal can be used to reduce the computational overheads, whereas the combined data from the accelerometer and the depth sensor allows us to obtain lower false alarm ratio. We also propose a very discriminative fall descriptor and show that a k-nn classifier achieves very good results on our publicly available URFD dataset[1]. The system has been designed to consume least amount energy to achieve reliable fall detection.

[1]`http://fenix.univ.rzeszow.pl/~mkepski/ds/uf.html`

3. The system

Having on regard that the system for fall detection should be inexpensive, we developed an energy-efficient data processing architecture, see Fig. 1, and implemented the algorithms on a low-cost PandaBoard ES, which is a development platform for mobile applications. It features a dual-core 1 GHz ARM Cortex-A9 MPcore processor with Symmetric Multiprocessing (SMP) and a programmable C64x DSP. The board contains 1 GB of DDR2 SDRAM, dual USB 2.0 ports as well as wired 10/100 Ethernet along with wireless Ethernet and Bluetooth connectivity. It supports various Linux-based operating systems such as Android, Chrome and Linux Ubuntu, which can be bootloaded from a SD memory card.

The fall detection is done on the basis of body worn accelerometer, which wirelessly transmits the motion data to the embedded system, and processing of the depth maps, which are acquired by a Kinect sensor directed towards the monitoring area. The person movement is sensed by an x-IMU inertial device [22], which contains triple axis 12-bit accelerometer. The motion data are acquired with 256 Hz and transmitted wirelessly via Bluetooth to the processing device, whereas the Kinect sensor is connected to it via USB, see Fig. 2. The depth images are acquired using OpenNI (Open Natural Interaction) library.

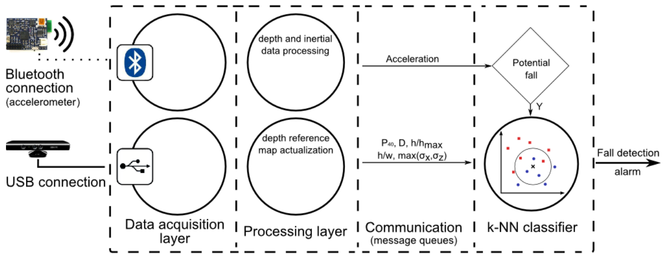
Bluetooth connection (accelerometer)
USB connection
depth and inertial data processing
depth reference map actualization
Acceleration
Potential fall
Y
P40, D, h/hmax h/w, max(σX,σZ)
Fall detection alarm
Data acquisition layer
Processing layer
Communication (message queues)
k-NN classifier

Figure 2: Data acquisition, processing and communication in the embedded system for fall detection.

The fall detection application runs under Linux and inherits all advantages of Unix-like operating systems. It consists of five main concurrent processes that communicate via message queues, see also Fig. 2. They supply an asynchronous communication protocol, meaning that the sender and receiver of the message do not need to interact with the message queue at the same time. The messages placed onto the queue are stored until the receiver retrieves them. The first process is accountable for acquiring data delivered by the wearable device, the second one acquires depth data from the Kinect sensor, third process continuously updates the depth reference image, fourth one is responsible for data processing and feature extraction, whereas the fifth process is in charge for data classification and alarm triggering. The dual-core processor permits parallel execution of data acquisition and processing processes. The person is extracted on the basis of the depth reference maps since he/she can be delineated reliably with relatively low computational cost [19].

The Kinect sensor consists of three main ingredients, namely, a color camera, an IR laser emitter, and another IR camera. The IR laser emitter and IR camera are utilized to create a depth map on the basis of a structured light technique. The laser emits a known pseudorandom dot pattern, which is then observed by the IR camera. The detected dots are compared against the known pattern. This is analogous to how a stereo-camera system works, but with one of the cameras replaced with a static virtual image of the dot pattern. Because the IR camera has a certain horizontal shift from the IR emitter, the projected dots will end up at miscellaneous image locations depending on the depth. The baseline of such a stereo-pair is approximately 75 mm. The angular field of view is fifty-seven degrees horizontally and forty-three degrees vertically. The minimum range for the Kinect is about 0.6 m and the maximum range is somewhere between 4-5 m. Pixels in the provided depth images indicate the calibrated depth in the scene. The depth resolution is about 1 cm at 2 m distance. The depth is supplied on 11 bits in maps with 640×480 resolution. Owing to the Kinect's ability to extract the depth images in unlit room and since we process only depth maps, our system is capable of detecting falls any time, i.e. it works on a 24/7 hours and days basis.

The fall event is identified using a k-nn classifier that takes decisions acting on a pool of representative examples, which had been collected in advance, see Fig. 3. The pool of examples representing typical activities of daily living (ADLs) and falls was extracted on the basis of human activities from our URFD dataset. Each example is a five dimensional vector and stores the activity/fall descriptors, which we discuss in Section 5. The activity/fall descriptors are extracted by the data processing module, which is discussed in detail in the subsequent Section.

During the algorithm evaluation or in on-line mode, see Fig. 3 and part of the figure on the right side of the dotted line, the system processes sequences of the depth maps. In this mode of the system we employ both the accelerometric and depth data. The motion data are thresholded to decide if the extraction of activity descriptors is needed, see also Fig. 1. If yes, the algorithm downloads from the circular buffer the recent depth map in order to extract the person and then to calculate the activity descriptors. In the training phase the motion data are not used since the activity exemplars are extracted on the basis of a collection of labeled human activities, see Fig. 3 and block `known human activities`. In the evaluation/on-line mode of fall detection we utilize a sequence of known activities to evaluate the detection performance of the system, see block `sequence of known human activities` on the discussed figure, or depth sequences acquired with 30 Hz by the Kinect sensor, see also block `sequence of unknown activities`.

4. Data processing

At the beginning of this Section we discuss how the accelerometric data are employed to trigger the processing of the depth maps. Afterwards, we present processing of depth data.

4.1. Triggering the processing of depth images

On the basis of the data acquired by the IMU (Inertial Measurement Unit) device the algorithm indicates a potential fall. In the flow chart of the algorithm depicted on Fig. 5, a block

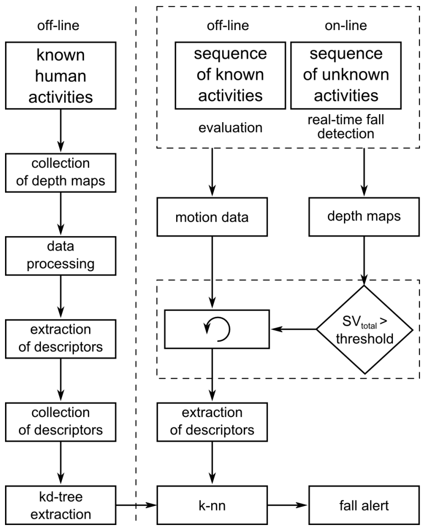

Figure 3: Extraction of k-nn decision rules (part of figure on the left side of the vertical line) and evaluation/on-line fall detection (right part of the figure).

`SVtotal > threshold`, see also Fig. 3, represents the detection of the potential fall using data from the inertial device. The value of SV_{total} has been calculated in the following manner:

$$SV_{total}(t) = \sqrt{A_x^2(t) + A_y^2(t) + A_z^2(t)} \qquad (1)$$

where $A_x(t)$, $A_y(t)$, $A_z(t)$ is the acceleration in the $x-$, $y-$, and $z-$axes at time t, respectively. The SV_{total} contains both the dynamic and static acceleration components, and thus it is equal to 1 g for standing, see also plots of acceleration change curves on Fig. 4. The discussed figure depicts sample plots of the acceleration for falling along with daily activities like going down the stairs, picking up an object, and sitting down – standing up. The sensor signals were acquired at a frequency of 256 Hz and resolution of 12 bits.

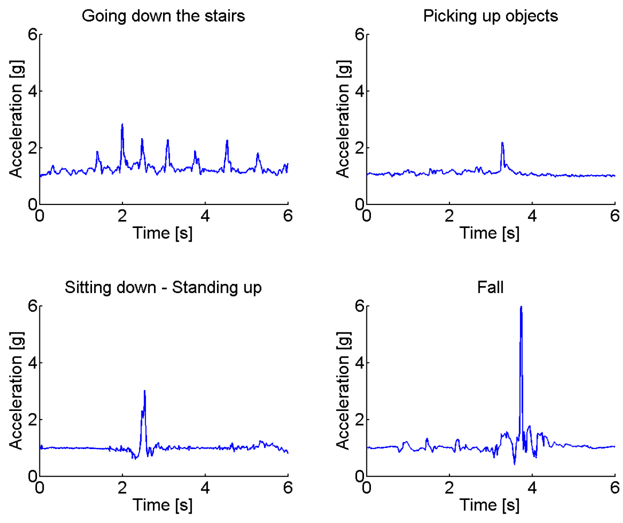

Figure 4: Acceleration over time for walking downstairs, picking up an object, sitting down, standing up and falling.

As we can observe on the discussed plots, during the falling phase the acceleration attained the value of 6 g, whereas during walking downstairs it attained the value of 2.7 g. It is worth noting that the data were acquired by x-IMU device [22], which was worn by a middle aged person. Such placement of the inertial device has been chosen since this body part represents the major component of body mass and undergoes movement in most activities.

In practice, it is not easy to construct a reliable fall detector with almost null false alarms ratio using the inertial data only. Thus, our system employs a simple threshold-based detection of falls, which are then verified on the basis of analysis of the depth images. The critical issue in threshold-based approach is the selection of a appropriate threshold since if the value is too high the system (having sensitivity < 100%) might miss some real falls but never triggers false alarms (with 100% specificity), while if the threshold value is too low the system will detect all actual falls (100% sensitivity) but, at the same time, it may trigger some false alarms (specificity < 100%). Thus, choosing the threshold for accelerometric data to be utilized in a fall detector is a compromise between sensitivity and specificity. In our approach, if the value of SV_{total} is greater than 3 g then the system starts the extraction of the person and then executes the classifier responsible for the final decision about the fall, see also Fig. 5.

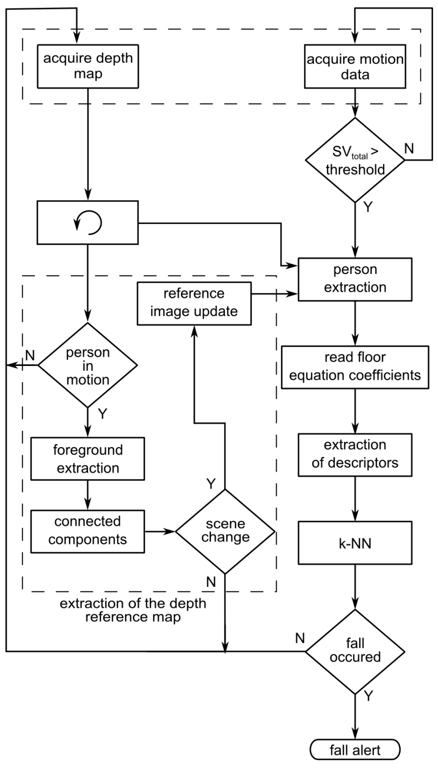

Figure 5: Flow chart of the algorithm for fall detection.

It is worth mentioning that in opposition to threshold-based accelerometer fall detectors [6][5], the change in some range of the threshold value does not have considerable influence on the specificity and sensitivity of our system. For instance, if we select a lower value of the threshold, say 2.7, the system will consider more typical ADLs as potential falls, and in consequence the processing overheads connected with fall authentication will be larger. As pointed out in [5], the trunk fall data has a large spread of UPVs (signal upper peak values), ranging from 3.5 to 12 g, whereas some ADL peak values of UPV are close to exceeding the 3.5 g threshold. In consequence, in real conditions such threshold-based algorithms generate considerable number of false alarms [6]. Moreover, our research findings reveal that the UPVs strongly depend not only on fall direction, type of substrate (wet/dry, parquet, carpet), or whether in proximity of the individual is an object or not, but primarily they depend on the way how a falling person is trying to save or to minimize the fall consequences. However, such state transitions are highly unpredictable and thus there is considerable overlap between the UPV values of ADLs and falls. This motivated our research on combining accelerometric and depth data to achieve reliable fall detection in home [13]. Moreover, early approaches to fall detection using Kinect assumed the use of simple cues [4], like distance of the person's gravity center to the floor [12], and in consequence they generate too much false alarms. The research results presented in [16] indicate that the body velocity prior to the occlusion, which was utilized in the discussed work of Rougier et al., can trigger a huge number of false alarms (on the order of 20 per day) for people walking out of the scene. They also showed that after disabling the velocity component, the discussed algorithm triggered a large number of false alarms at low detection rates. False alarms were caused by a variety of everyday occurrences, including pets moving on the floor, items dropped or moved on the floor, and residents and visitors lying or playing on the floor. Such everyday occurrences are filtered reliably by our algorithm at low computational cost.

4.2. Processing of depth data

The depth maps acquired by the Kinect sensor are stored in a circular buffer. In the current implementation, the size of the circular buffer is set to 15. The depth sequence is utilized to extract a depth reference image, which is in turn employed to delineate the person. The extraction of the person is achieved through differencing the current depth map from such a depth reference image. As we can observe on Fig. 5, the extraction of the person is executed only when the condition `SV`$_{\texttt{total}}$ `> threshold` is true. After the person extraction, the floor equation coefficients are uploaded to extract descriptors of the activities. The descriptors are then employed to take decision if a fall occurred. If the SV_{total} is smaller or equal to the assumed threshold then new data from the accelerometer is acquired.

In the current implementation the depth reference map is updated on-line, see also Fig. 5 and the block `reference image update`, which makes possible to utilize the fall detection system in a wide range of scenarios. In the depth reference image each pixel assumes the median value of several pixels values from the past images. In the set-up stage we collect a number of the depth images, and for each pixel we assemble a list of the pixel values from the former images, which is then sorted in order to determine the median. Given the sorted lists of pixels the depth reference image can be updated quickly by removing the oldest pixels and updating the sorted lists with the pixels from the current depth image and then extracting the median value. We found that for typical human motions, satisfactory results can be obtained using 15 depth images. For the Kinect sensor acquiring the images at 30 Hz we take every fifteenth depth map. This means that when the person is in motion, the reference image is updated at 2 Hz. On the basis of the reference image accommodated with such a frequency, the person is extracted at 30 Hz, see also the connection of the circular buffer with blocks `extraction of the depth reference map` and `person extraction`. As we already mentioned, in order to preserve the privacy of the user as well as to make the system ready to work any time, the RGB images corresponding to the depth maps are not acquired by our fall detection system.

In order to prevent disappearance of the person (on the binary image indicating the foreground objects) if he/she is not in motion for a while, i.e. to avoid assigning the person to be extracted to the depth reference image, we perform updating of the depth reference map only when the person is in motion, see Fig. 5 and the block `person in motion`. If the person is not at rest during an assumed in advance period of time, the algorithm extracts the foreground and then it determines the connected components to decide if a scene change took place, see Fig. 5 and the block `scene change`. If no substantial scene change is detected then there is no necessity to update the scene reference depth map, and in such a case the algorithm acquires from the Kinect a new depth map. It is worth noting that the block called `extraction of the depth reference map` can be replaced by other algorithm for person delineation, for instance, through a block based on the well known Gaussian Mixture Models, which are frequently used in the background subtraction. The scene change can be inferred on the basis of energy maps [19], or eventually on the basis of differencing the consecutive depth maps, which is in fact the simplest method of motion detection. Since the person is the most important subject in the fall detection, we consider also motion data from the accelerometer to sense the person's state and scene changes.

Figure 6 demonstrates a situation when in addition to the monitored person an additional object, i.e. a moved chair, compare also color images #610 and 810, appears in the binary image indicating the extracted objects. As we can observe on image #1010, after a while the chair is included into the depth reference map of the observed scene and the only delineated object in the binary image is the person undergoing monitoring. A sporadic appearance of items other than the observed person on the binary images does not degrade the performance of our algorithms since they typically appear for a few seconds. In such a period of time the accelerometer allows us to filter out such situations, including situations reported in [16], which degraded the performance of the algorithm proposed in [12].

5. Descriptors

At the beginning we explain the extraction of the person in the depth maps. In the next subsection we discuss how points belonging to floor are determined. Then we show the equation describing the floor. The next subsection is devoted to depth features, whereas the last one is devoted to a description of the proposed fall descriptor.

5.1. Person extraction

The person undergoing monitoring is extracted through differencing the current depth map from the depth reference image, see Fig. 5. As illustrated on the discussed figure, optionally he/she can be delineated on the basis of relevant background subtraction-like algorithms. Fig. 7 depicts some example binary images with the extracted person. In the middle column

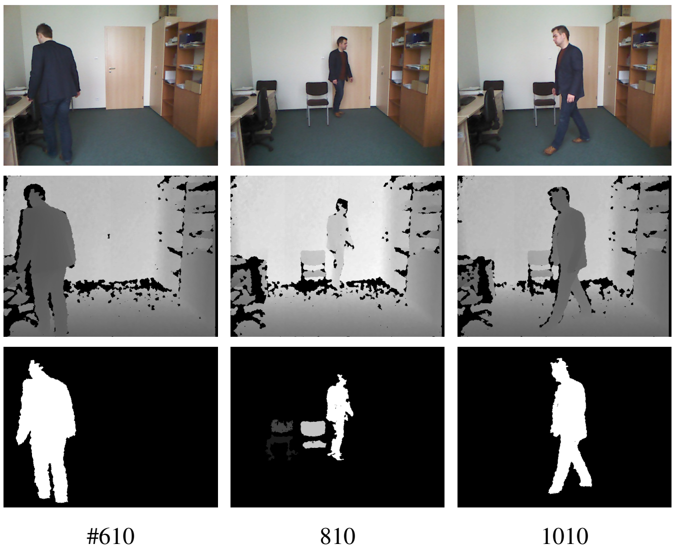

Figure 6: Accommodation of the depth reference image to scene change. RGB images (upper row), depth (middle row) and binary images depicting the delineated person (bottom row).

there are example raw depth maps, whereas in the left one there are the corresponding RGB images. The RGB images are not processed by our system and they are only depicted for illustrative purposes. It is well known that the precision of Kinect measurements decreases in strong sunlight. In order to investigate the influence of sunlight on the performance of fall detection we analyzed the person extraction in the depth maps acquired in strong sunlight. We found that the in-home depth measurements on a person being in sunlight, i.e. in sunlight that passes a closed window, can be made with limited complications. Despite that some body parts of a person in sunlight may not return measurements, the most of the body gives the depth measurements and the performance of the fall detection does not degrade noticeably.

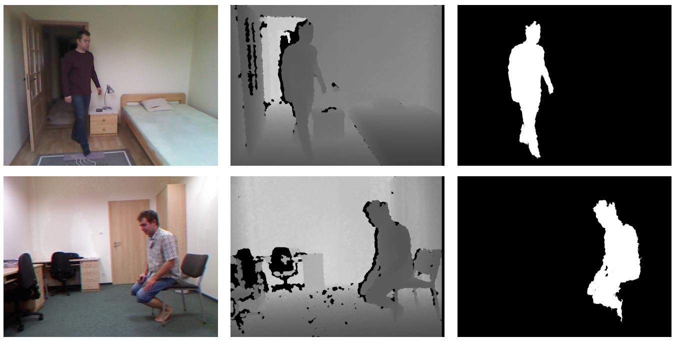
Figure 7: Delineation of person using depth reference image. RGB images (left), depth (middle) and binary images depicting the delineated person (right).

5.2. *Ground plane extraction*

After the transformation of the depth pixels to the 3D points cloud, the ground plane described by the equation $ax + by + cx + d = 0$ was recovered. Assuming that the optical axis of the Kinect camera is almost parallel to the floor, a subset of the points with the lowest altitude has been selected from the entire points cloud and then utilized in the estimation. For non-parallel Kinect set-up a method for extracting the points belonging to the floor can be used instead [9][19]. The parameters a, b, c and d were estimated using the RANdom SAmple Consensus (RANSAC) algorithm. RANSAC is an iterative algorithm for estimating the parameters of a mathematical model from a set of observed data, which contains outliers [23]. The distance to the ground plane from the 3D centroid of points cloud corresponding to the segmented person has been determined on the basis of the following equation:

$$D = \frac{|aX_c + bY_c + cZ_c + d|}{\sqrt{a^2 + b^2 + c^2}} \qquad (2)$$

where X_c, Y_c, Z_c stand for the coordinates of the person's centroid. The parameters should be re-estimated subsequent to each change of the Kinect location or orientation.

5.3. *Depth features*

The following features were extracted in a collection of the depth images in order to authenticate the fall hypotheses, which are altered by the threshold-based procedure:

- H/W - a ratio of width to height of the person's bounding box in the depth maps
- H/H_{max} - a proportion expressing the height of the person's surrounding box in the current frame to the physical height of the person, projected onto the depth map
- D - the distance of the person's centroid to the floor
- $max(\sigma_x, \sigma_z)$ - standard deviation from the centroid for the abscissa and the applicate, respectively.

Given the delineated person in the depth image along with the automatically extracted parameters of the equation describing the floor, the aforementioned features are easy to calculate.

Figure 8 depicts a person in depth images together with the H/W and H/H_{max} features. As we can observe on the discussed depth maps with graphically marked features, the depth features assume quite different values during an example fall event and typical daily activities like walking and sitting on a chair.

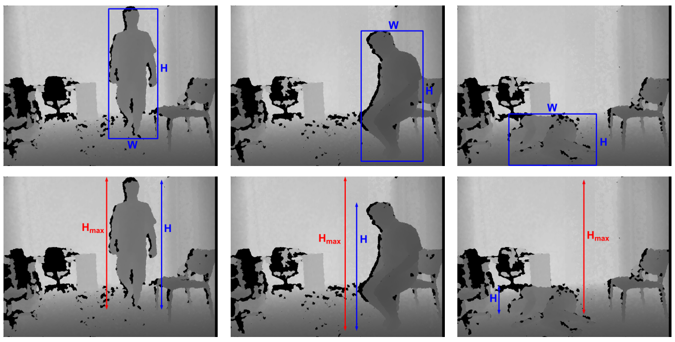

Figure 8: Person on depth images with the marked H/W and H/H_{max} features.

Figure 9 shows bar plots of the utilized depth features for discrimination between ADLs and fall events. As we can observe, the depth features are quite discriminative on the considered events. However, very rarely, such bag of features can have insufficient discrimination power. For instance, for a standing man with outstretched arms the H/W feature can assume different values in comparison to normal standing position. Therefore, we developed a very discriminative feature, which further diminishes the number of false alarms.

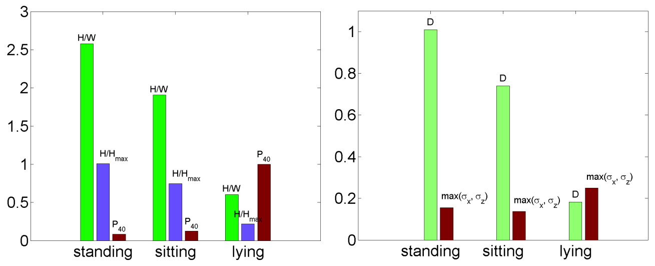

Figure 9: Depth features for isolation of fall events from ADLs.

5.4. *The proposed descriptor to identify fall events*

The proposed P_{40} descriptor is calculated on 3D point clouds. On the basis of the extracted person in the depth map a corresponding person's point cloud is determined in 3D space, see Fig. 10. Afterwards, a ratio of the number of the point clouds belonging to the cuboid of 40 cm height and placed on the floor to the number of the point clouds belonging to the cuboid of height equal to person's physical height is calculated. The distance of each point to the floor is calculated on the basis of (2). The 40 cm height of the cuboid has been chosen experimentally to include all 3D points belonging to a lying person on the floor.

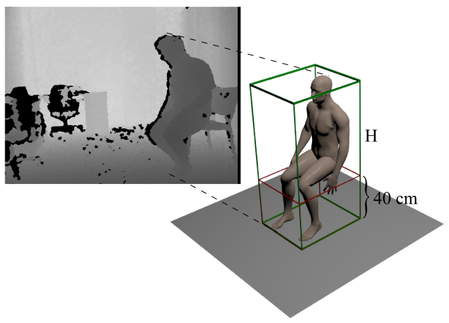

Figure 10: Proposed P_{40} descriptor.

6. The classifier for fall detection

At the beginning of this Section we discuss the dataset that was recorded in order to extract the features for constructing as well as evaluating the fall classifiers. After that, we overview the k-nn classifier for separating in real-time the ADLs from fall events.

6.1. *The dataset*

A dataset consisting of depth maps with typical activities like walking, sitting down, crouching down and lying has been composed in order to construct the classifiers responsible for examination whether a person is lying on the floor and to evaluate their detection performance. In total 2395 images were selected from our UR Fall Detection Dataset (URFD) and other image sequences, which were recorded in typical rooms, like office, classroom, etc. The selected image set consists of 1492 images with typical ADLs, whereas 903 images depict a person lying on the floor. The aforementioned depth images were employed to determine the features discussed in Subsection 5.3 and 5.4.

The URFD dataset consists of thirty image/acceleration sequences with falls, thirty image/acceleration sequences with typical ADLs like sitting down, crouching down, picking-up an object from the floor, and ten sequences with fall-like activities as quick lying on the floor and lying on the bed/couch. The number of images in the sequences with falls is equal to 3000, whereas the number of images from sequences with ADLs is equal to 10000. Two kinds of falls were performed by five persons, namely from standing position and from sitting on the chair. The data were acquired at a sampling rate of 30 Hz. All RGB and depth images are synchronized with motion data, which were acquired by the x-IMU inertial device. The motion data contains the acceleration over time in the $x-$, $y-$, and $z-$axes together with the precalculated SV_{total} values.

6.2. *k-nn classifier*

The k-Nearest Neighbor (k-nn) classifier is a very simple classifier that works well on many classification problems. Despite its simplicity, the k-nn has been found to be successful in a large number of classification and regression problems, including biometrics, handwritten digits recognition or fall detection [24]. Being a non-parametric method, it is very useful in classification tasks, where the decision boundary is very irregular. The principle behind k-nn methods is to find a predefined number of training samples closest in a distance to the classified example, and then to predict the label from these. No model needs to be built, i.e. k-nn does not learn anything from the training data and simply uses the training data itself for the classification. Thus, the cost of the learning process is equal zero and all the cost is dedicated to determining the decision. The decision is the most common label among the k closest neighboring points. The parameters of the algorithm are the number k of neighbors and the procedure for combining the predictions of the k examples. Changing k can alter the decision of the classifier. The most naive neighbor search implementation involves the brute-force computation of distances between the current example and all instances in the dataset. To cope with the computational inefficiencies of the brute-force neighbor search, the kd-tree data structure has been used to store the ADL and fall examples, see also Fig. 3. Once constructed, the nearest neighbor of an example in question can be determined with only $O(\log(N))$ distance computations. The utilized features were scaled to have the same range of values.

7. Experimental results

We evaluated the k-nn classifier and compared it with a SVM classifier. The classifiers were evaluated in 10-fold cross-validation. To examine the classification performance we calculated the sensitivity, specificity, precision and classification accuracy. The sensitivity is the number of true positive (TP) responses divided by the number of actual positive cases (number of false negatives (FN) plus number of TP). It is the likelihood of fall, given that a fall took place, and hence it is the

Table 1: Performance of lying pose classification using H/W, H/H_{max}, D, $max(\sigma_x, \sigma_z)$ and P_{40} descriptors.

			True		
			Fall	No Fall	
Estimated	k-nn, SVM	Fall	903	0	Accuracy=100%
		No fall	0	1492	Precision=100%
			Sens.=100%	Spec.=100%	

classifier's capability of recognizing the considered occurrence properly. The specificity is the number of true negative (TN) decisions divided by the number of actual negative cases (number of false positives (FP) plus number of TN). It is the likelihood of non-fall, given that a non-fall ADL occurred, and in consequence it demonstrates how good a classifier is at avoiding the false alarms. The classification accuracy is the number of correct decisions divided by the full amount number of the cases, i.e. the amount of true positives plus sum of true negatives divided by total number of instances in the population. In other words, the accuracy is the ratio of true results (both TP and TN) in the population. The precision or positive predictive value (PPV) is equal to true positives divided by sum of true positives and false positives. Therefore, it illustrates how many of the positively classified falls were relevant.

In Tab. 1 are presented experimental results that were obtained in 10-fold cross-validation by classifiers responsible for the lying pose detection on image set discussed in Subsection 6.1. They were obtained using $c = 1$ in the linear SVM classifier and 3, 5, 7 and 9 neighbors in the k-nn classifier. As we can see, both specificity and precision are equal to 100%, i.e. the ability of the classifier to avoid false alarms and its exactness assume perfect values. The results were obtained using H/W, H/H_{max}, D, $max(\sigma_x, \sigma_z)$ and P_{40} descriptors.

Table 2 shows the performance of fall detection that was obtained by k-nn with 3 and 5 neighbors as well as by the linear SVM with c set to 1, and which operated on H/W, H/H_{max}, D and $max(\sigma_x, \sigma_z)$ descriptors. As we can observe, on the discussed features the linear SVM classifier [25] achieves worse results than the k-nn with three neighbors since four false alarms are generated. The results achieved by the k-nn with five neighbors are slightly worse since two false alarms are generated and additionally two falls are not altered. Comparing the experimental results in Tab. 1 and 2 we see that the proposed P_{40} fall descriptor allows us to identify the fall events with better accuracy and the classifiers using it have better sensitivity. The features used by both classifiers were scaled to have the same range of values between zero and one. To avoid square root computation and thus to accelerate the k-nn we calculated squared Euclidean distances.

Table 3 shows results that were obtained in 10-fold cross-validation on depth image sequences from the URFD dataset. It demonstrates the performance of fall detection for SV_{total} equal to 3, three neighbors in the k-nn and $c = 1$ in the SVM classifiers, which were selected in hyperparameter tuning. The results were obtained on thirty image/acceleration sequences with falls and forty image/acceleration sequences with typical ADLs like sitting down, crouching down, picking-up an object from the floor and lying on the sofa. That means that they were obtained on 30 data sequences with falls and 40 data sequences with ADLs, i.e. 13000 data examples. In the case of incorrect response of the system the remaining part of the sequence has been omitted. This means that the detection scores were determined on the basis of the number of the correctly/incorrectly classified sequences. As we can observe, the k-nn algorithm using both motion data from accelerometer and depth maps for verification of IMU-based alarms achieves better performance in comparison to SVM. In [24], where a ceiling-mounted depth sensor has been utilized in fall detection, the k-nn also achieved superior results over a SVM classifier.

In Tab. 4 are presented results, which were obtained on independent test data set with k set to 3 in the k-nn and $c = 1$ in the linear SVM. The data were split into two sets, one for model building and one for model validation. Due to limited training data the data sequences were split in proportion 66% and 33%. As we can observe, the performance of the SVM classifier is slightly worse in comparison to performance from Tab. 3. The experimental results reported in Tab. 3 and 4 were obtained using H/W, H/H_{max}, D, $max(\sigma_x, \sigma_z)$ and P_{40} descriptors.

The algorithms for fall detection that were evaluated in such a way have been implemented in C/C++ on the PandaBoard-ES platform. The code profiler reported about 50% utilization of the CPU power by the module responsible for update of the depth reference map. The SVM classifier has been trained off-line on a PC using LIBSVM software. The SVM model obtained in such a way has been used to implement the fall predictor, executed on the PandaBoard.

Five volunteers with age over 26 years attended in an evaluation of the developed algorithm and the embedded system for fall detection in real-time. Intentional falls were performed in an office by five persons towards a carpet with thickness of about 2 cm. The x-IMU device was worn near the pelvis. Each individual performed three types of falls, namely forward, backward and lateral at least three times. Each individual performed also ADLs like walking, sitting, crouching down, leaning down/picking up objects from the floor, as well as lying on a settee. All intentional falls have been detected appropriately. In particular, quick sitting down, which is not easily distinguishable ADL from an intentional fall when only an accelerometer is used, has been correctly classified as an ADL. It has been observed that while performing such activities, the acceleration value equal to 3 g, which is often used in accelerometer-based algorithms has been exceeded several times. We noticed, that different locations of the accelerometer, for instance on chest or

Table 2: Performance of lying pose classification using H/W, H/H_{max}, D and $max(\sigma_x, \sigma_z)$ descriptors.

			True		
			Fall	No Fall	
Estimated	k-nn (3)	Fall	901	0	Accuracy=99.92%
		No fall	2	1492	Precision=100%
			Sens.=99.78%	Spec.=100%	
	k-nn (5)	Fall	901	2	Accuracy=99.83%
		No fall	2	1490	Precision=99.78%
			Sens.=99.78%	Spec.=99.87%	
	SVM	Fall	903	4	Accuracy=99.83%
		No fall	0	1488	Precision=99.56%
			Sens.=100%	Spec.=99.73%	

Table 3: Performance of fall detection on URFD data sequences.

		Method	
		k-nn + acc.	SVM + acc.
Results	Accuracy	95.71%	94.28%
	Precision	90.90%	88.24%
	Sensitivity	100.00%	100.00%
	Specificity	92.50%	90.00%

Table 4: Performance of fall detection on independent test data set.

		Method	
		k-nn + acc.	SVM + acc.
Results	Accuracy	95.83%	91.67%
	Precision	90.91%	83.33%
	Sensitivity	100.00%	100.00%
	Specificity	92.86%	85.71%

back do not diminish noticeably the detection performance.

8. Conclusions

In this paper we presented how to improve fall detection by the use of depth and accelerometric data. In the proposed architecture an accelerometer is utilized to indicate an eventual fall and the Kinect sensor is used to authenticate the fall. We demonstrated that through the thresholding of accelerometric signal we can filter out a considerable number of non-fall events. We then showed that a depth sensor can reliably distinguish between such filtered events and the falls. We then demonstrated that owing to thresholding of the motion data we can considerably reduce the computing overheads for processing the depth data. In consequence, the depth maps are not processed frame-by-frame, but instead a circular buffer is used to store the depth maps for processing them in the case of possible fall. We demonstrated that on our publicly available dataset for evaluation of fall detection, the k-nn classifier achieves better classification performance in comparison to linear SVM. We also demonstrated that the proposed fall descriptor contributed towards better fall detection performance. The presented embedded system works 24/7 hours and days, permits reliable and unobtrusive fall detection as well as preserves privacy of the user.

Acknowledgements

This work was partially supported by Faculty of Computer Science, Electronics and Telecommunications of AGH Univ., by the Interdisciplinary Centre for Computational Modelling of Univ. of Rzeszow as well as by Polish National Science Center (NCN) under a research grant conducted over years 2015-2018.

References

[1] R. Igual, C. Medrano, and I. Plaza, "Challenges, issues and trends in fall detection systems," *BioMedical Engineering OnLine*, vol. 12, 2013.

[2] J. Zhang, Y. Shan, and K. Huang, "ISEE Smart Home (ISH): Smart video analysis for home security," *Neurocomputing*, vol. 149, Part B, pp. 752 – 766, 2015.

[3] C. Wu and H. Aghajan, "Real-time human pose estimation: A case study in algorithm design for smart camera networks," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1715–1732, Oct 2008.

[4] D. Webster and O. Celik, "Systematic review of Kinect applications in elderly care and stroke rehabilitation," *Journal of NeuroEngineering and Rehabilitation*, vol. 11, 2014.

[5] A. Bourke, J. O'Brien, and G. Lyons, "Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm," *Gait & Posture*, vol. 26, no. 2, pp. 194–199, 2007.

[6] F. Bagala, C. Becker, A. Cappello, L. Chiari, K. Aminian, J. M. Hausdorff, W. Zijlstra, and J. Klenk, "Evaluation of accelerometer-based fall detection algorithms on real-world falls," *PloS ONE*, vol. 7(5), e37062, 2012.

[7] L. C. D. Silva, C. Morikawa, and I. M. Petra, "State of the art of smart homes," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 7, pp. 1313 – 1321, 2012.

[8] B. Kwolek, “Face tracking system based on color, stereovision and elliptical shape features,” in *Proc. of the IEEE Conf. on Advanced Video and Signal Based Surveillance*. Washington, DC, USA: IEEE Computer Society, 2012, pp. 21–26.

[9] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, “3D head tracking for fall detection using a single calibrated camera,” *Image Vision Comput.*, vol. 31, no. 3, pp. 246–254, 2013.

[10] L. A. B. E. Demiroz, A. A. Salah, “Coupling fall detection and tracking in omnidirectional cameras,” in *Human Behavior Understanding*, ser. Lecture Notes in Computer Science. Springer Int. Publ., 2014, vol. 8749, pp. 73–85.

[11] M. V. Sokolova, J. Serrano-Cuerda, J. C. Castillo, and A. Fernández-Caballero, “A fuzzy model for human fall detection in infrared video,” *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology*, vol. 24, no. 2, pp. 215–228, Mar. 2013.

[12] C. Rougier, E. Auvinet, J. Rousseau, M. Mignotte, and J. Meunier, “Fall detection from depth map video sequences,” ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, vol. 6719, pp. 121–128.

[13] M. Kepski and B. Kwolek, “Fall detection on embedded platform using Kinect and wireless accelerometer,” in *Proc. of the 13th Int. Conf. on Computers Helping People with Special Needs*. Berlin, Heidelberg: Springer-Verlag, 2012, pp. II:407–414.

[14] G. Parra-Dominguez, B. Taati, and A. Mihailidis, “3D human motion analysis to detect abnormal events on stairs,” in *Second Int. Conf. on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, Oct 2012, pp. 97–103.

[15] T.-T.-H. Tran, T.-L. Le, and J. Morel, “An analysis on human fall detection using skeleton from Microsoft Kinect,” in *IEEE Fifth Int. Conf. on Communications and Electronics (ICCE)*, July 2014, pp. 484–489.

[16] E. E. Stone and M. Skubic, “Fall detection in homes of older adults using the Microsoft Kinect,” *IEEE J. of Biomedical and Health Informatics*, 2015.

[17] M. Vela and D. Ruiz-Fernandez, “Automatic detection of health emergency states at home,” in *XIII Mediterranean Conf. on Medical and Biological Engineering and Computing*, ser. IFMBE Proc., vol. 41. Springer Int. Publ., 2014, pp. 1209–1212.

[18] K. Cuppens, C.-W. Chen, K. B.-Y. Wong, A. Van de Vel, L. Lagae, B. Ceulemans, T. Tuytelaars, S. Van Huffel, B. Vanrumste, and H. Aghajan, “Integrating video and accelerometer signals for nocturnal epileptic seizure detection,” in *Proc. of the 14th ACM Int. Conf. on Multimodal Interaction*, ser. ICMI ’12. New York, NY, USA: ACM, 2012, pp. 161–164.

[19] B. Kwolek and M. Kepski, “Fall detection using Kinect sensor and fall energy image,” in *Hybrid Artificial Intelligent Systems*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, vol. 8073, pp. 294–303.

[20] B. Delachaux, J. Rebetez, A. Perez-Uribe, and H. F. Satizbal Mejia, “Indoor activity recognition by combining one-vs.-all neural network classifiers exploiting wearable and depth sensors,” ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, vol. 7903, pp. 216–223.

[21] C. Chen, R. Jafari, and N. Kehtarnavaz, “Improving human action recognition using fusion of depth camera and inertial sensors,” *IEEE Trans. on Human-Machine-Systems*, 2014.

[22] x IMU. (2014) The x-IMU Inertial Measurement Unit. [Online]. Available: http://www.x-io.co.uk/products/x-imu/

[23] M. A. Fischler and R. C. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[24] M. Kepski and B. Kwolek, “Fall detection using ceiling-mounted 3d depth camera,” in *Proc. of the 9th Int. Conf. on Computer Vision Theory and Applications*. Scitepress, 2014, pp. II:640–647.

[25] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for Support Vector Machines,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1 – 27, May 2011.