electronics

MDPI

Article

Fallen People Detection Capabilities Using Assistive Robot

Saturnino Maldonado-Bascón, Cristian Iglesias-Iglesias, Pilar Martín-Martín * and Sergio Lafuente-Arroyo

Signal Theory and Communications Department, Alcalá University, Alcalá de Henares, 28805 Madrid, Spain
* Correspondence: p.martin@uah.es; Tel.: +34-918-856-735

Received: 24 July 2019; Accepted: 18 August 2019; Published: 21 August 2019

Abstract: One of the main problems in the elderly population and for people with functional disabilities is falling when they are not supervised. Therefore, there is a need for monitoring systems with fall detection functionality. Mobile robots are a good solution for keeping the person in sight when compared to static-view sensors. Mobile-patrol robots can be used for a group of people and systems are less intrusive than ones based on mobile robots. In this paper, we propose a novel vision-based solution for fall detection based on a mobile-patrol robot that can correct its position in case of doubt. The overall approach can be formulated as an end-to-end solution based on two stages: person detection and fall classification. Deep learning-based computer vision is used for person detection and fall classification is done by using a learning-based Support Vector Machine (SVM) classifier. This approach mainly fulfills the following design requirements—simple to apply, adaptable, high performance, independent of person size, clothes, or the environment, low cost and real-time computing. Important to highlight is the ability to distinguish between a simple resting position and a real fall scene. One of the main contributions of this paper is the input feature vector to the SVM-based classifier. We evaluated the robustness of the approach using a realistic public dataset proposed in this paper called the Fallen Person Dataset (FPDS), with 2062 images and 1072 falls. The results obtained from different experiments indicate that the system has a high success rate in fall classification (precision of 100% and recall of 99.74%). Training the algorithm using our Fallen Person Dataset (FPDS) and testing it with other datasets showed that the algorithm is independent of the camera setup.

Keywords: assistive robot; fall detection; lying-pose recognition; deep learning; mobile robot; convolutional neural network; support vector machine

1. Introduction

Falls are considered one of the most serious issues for the elderly population [1]. In general, those falls cause injury, loss of mobility, fear of falling and even death. Some studies suggest that falls where the patient has been waiting a long time on the ground before help arrives are associated with bigger health problems [2]. Reliable fall detection systems are an essential research topic for monitoring the elderly and people with disabilities who are living alone [3].

Many approaches have been proposed using many different kinds of devices and methodologies and some of them are summarized by Noury [4], Mubashir [5], Igual [6] and Khan [7]. Principally, all proposed approaches can mostly be divided into two big groups—wearable-based and vision-based devices methods.

Studies based on wearable devices are growing fast and they rely on sensors that are attached to the person's body as accelerometers, gyroscopes, interface pressure sensors and magnetometers [8–11]. Although these approaches have provided high detection rates by using small and cheap technology,

they require active cooperation by wearing the sensors. As a consequence, for long-term use, they are not a practical solution by themselves.

On the contrary, vision-based devices do not require any support from the elderly. At the same time, cameras nowadays are increasingly used in our daily lives. Vision-based fall detection systems analyze in real-time the position and shape of the person using different kinds of algorithms that combine standard computing platforms and low-cost cameras. Compared with other methods, the vision-based methods promise results due to the fast advances in computer-vision and video-camera technologies, such as the economical Microsoft Kinect [12–14]. The combination of video-based and ambient sensor-based systems (external sensors embedded in the environment, such as infrared, pressure and acoustic sensors [15]) also provide excellent results.

Mobile robots are the right solution for keeping a single person in view when compared to static cameras [14,16,17]. To avoid terrain difficulty, Máthé et al. [18] and Iuga et al. [19] proposed methods that use uncrewed aerial vehicles (UAVs) as mobile robots. A useful aspect of patrol robots, instead of robots that keep the person continuously in view, is the integration of privacy protection and real-time algorithms. As the person is not under supervision all the time, especially in particular locations like the bathroom, the elderly feel more relaxed because their privacy is less invaded.

In this work, we deal with the fall detection problem in the case of having one, two, or more people in the same environment. We used our multifunctional and low-cost mobile robot equipped with a 2D image-based fall detection algorithm as a patrol robot. The assistive robot autonomously patrols an indoor environment, and when it detects falls, it activates an alarm. The system was designed to recognize lying-poses in single images without any knowledge about the background. Additionally, the robot relocates itself in case of doubt in detection. However, we assume that it is improbable that a patrol robot takes an image during the falling; therefore, this work focuses on detecting falls in a short interval after the event of falling.

Additionally, to analyze the effectiveness of the approach, we provide a new dataset to be used in fall detection algorithms. The main features of this dataset are:

- several scenarios with variable light conditions,
- different person sizes,
- images with more than one actor,
- persons wearing different clothes,
- several lying-position perspectives and
- resting and fallen persons.

The remainder of the paper is organized as follows. Section 2 describes the needs of and challenges for fall detection systems and reviews the work related to fall detection vision-based approaches. Section 3 describes the design and methodology of the proposed fall detection method in detail. We describe the system architecture in Section 3.1. Section 3.2 focuses on person detection and fall classification is analyzed in Section 3.3. In Section 4, a new dataset is described and the method is evaluated. Section 4.1 describes the Fallen Person Dataset (FPDS) in detail. Section 4.2 presents the used metrics for measuring the effectiveness of the technique. The following three Sections 4.3–4.5, outline the carried-out experiments to evaluate the proposed approach from different points of view. Two evaluations of the method, relocation of the patrol robot and performance verification over other datasets were done and they are outlined in the last two Sections 4.6 and 4.7. Finally, in Section 5, conclusions and future research directions are identified.

2. Vision-Based System Overview

Vision-based systems offer many advantages over wearable sensor-based systems. Mainly, they are more robust and once they are installed, the person can forget about them. In these systems, cameras play an important role. If we consider the number and type of cameras, there are mainly three groups [20]—single camera, multicamera and depth cameras. For 2D-vision systems, only

one uncalibrated camera is required but for 3D vision systems, we need a calibrated single camera or multicamera.

The most extensive systems are based on a single camera due to their simplicity and price. Particularly in the case of fixed cameras, since cameras are static, and background subtraction can mainly be applied to find the person in the image [21]. Kim et al. [22] proposed one of the more used real-time foreground-background methods. However, the person could be integrated into the background when they have been sitting on a couch for long. Several approaches show that it is possible to achieve good results using a single camera. Charfi et al. [23] proposed a technique based on feature extraction, an SVM-based classifier and a final decision step. Liu et al. [24] used a k-nearest neighbor classifier and Wang [25] performed multi-viewpoint pose parsing based on part-based detection results.

Fixed cameras are efficient only if the camera is placed in the ceiling of the room to avoid occlusion objects. However, the camera does not have good access to the vertical parameter of the body, which provides essential information for fall detection [26]. Another intelligent solution consists of using an assistive robot equipped with a single camera. In that case, occlusion or doubtful cases can be solved using different viewpoints that can be taken from the moving robot.

On the other hand, a good solution for solving the problem of occlusion would use a system with multiple cameras. However, the main issues in those cases are time-consuming calibration to compute reliable 3D information and the synchronization process between the different cameras. Some studies have been working out these problems, such as Rougier et al. who, in Reference [27], proposed a method based on Gaussian Mixture Model (GMM) classification and human-shape deformation for uncalibrated single- and multicamera systems.

Depth cameras, such as Kinect, provide several advantages, for example, independence from light conditions, silhouette ambiguity of the human body, simplification of background-subtraction tasks and reduction of the time needed for calibration [12,13].

In general, vision-based fall detectors have some challenges to resolve for good performance in the different situations that the person can be found:

- high variability of possible body orientations on the floor,
- different person sizes,
- wide range of background structures and scenarios and
- occlusions being frequent cases in the fall detection context.

Based on all previously mentioned reasons, our proposal is a vision-based learning solution for fall detection by using a single RGB camera mounted on an assistive patrol robot. The robot patrols around the indoor environment and, in case of fall detection, activates an alarm. The proposed method deals with three of the previous four points, as is shown in the Experiment Results section. How to improve our work with the occlusions is further investigated.

3. Proposed Fall Detection Approach

Our approach solves the fall detection problem in an end-to-end solution based on two steps—person detection and fall classification. The person detection algorithm aims to localize all persons in an image. Its output is the enclosing bounding boxes and the confidence scores that reflect how likely it is that the boxes contain a person. Fall classification estimates if the detected person is in a fall or not.

In this approach, we propose to combine the YOLOv3 algorithm based on a Convolutional Neural Network (CNN) for person detection and a Support Vector Machine (SVM) for fall classification. The main steps of our detection system (Figure 1) are as follows:

- Take a single image.
- Person detection. Results are the coordinates of the bounding box of the detected human body.

- Feature extraction from the bounding box coordinates.
- Fall identification.
  - Nonfall detection—continue taking new images.
  - Fall detection—ask for confirmation of the fall.
  - Doubt detection—the bounding box is too small, too big, or is located at the edges of the image. The robot needs to relocate itself to center the possible fall detection with the proper dimensions.

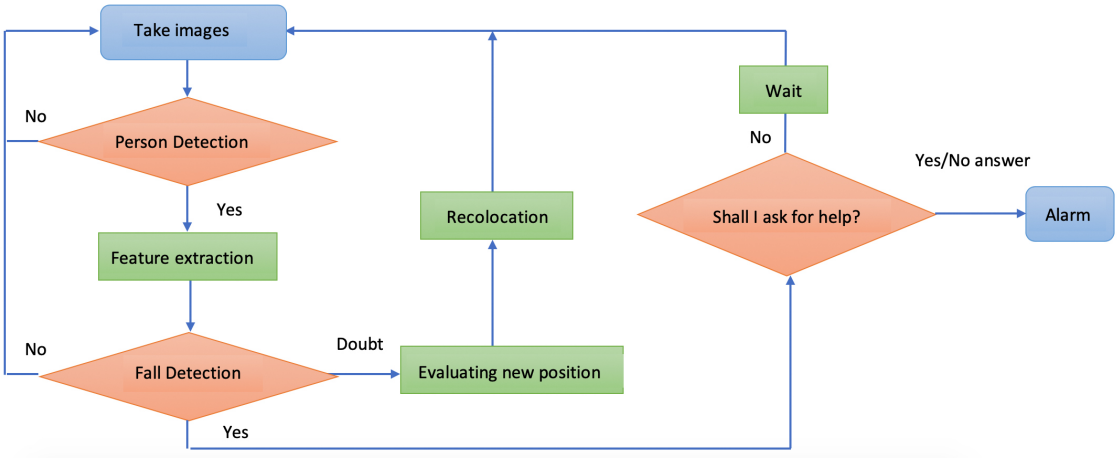


Figure 1. Flowchart of fall detection approach.

3.1. System Architecture

As a base for the fall detection approach, we used the assistance robot LOLA, designed entirely by our research team to monitor and help the elderly and any other person with a functional disability who lives alone. The main idea behind the LOLA robot was to be an assistive robot that could also work as a rollator for helping to walk or as a table to transport objects due to its shape—80 cm height, 58 cm width and 70 cm depth (Figure 2).

Figure 2. LOLA assistive robot.

The system is equipped with an Arduino Mega board, various sensors, a single RGB camera and Raspberry V3 B+. We also needed a connection to a server to perform the heavy workload—image processing and the fall detection algorithm. This connection could be WIFI to a remote server or

ethernet to a laptop located in the robot. The camera is located 76 cm above the floor and takes images of 640 × 480 pixels (Figure 3).

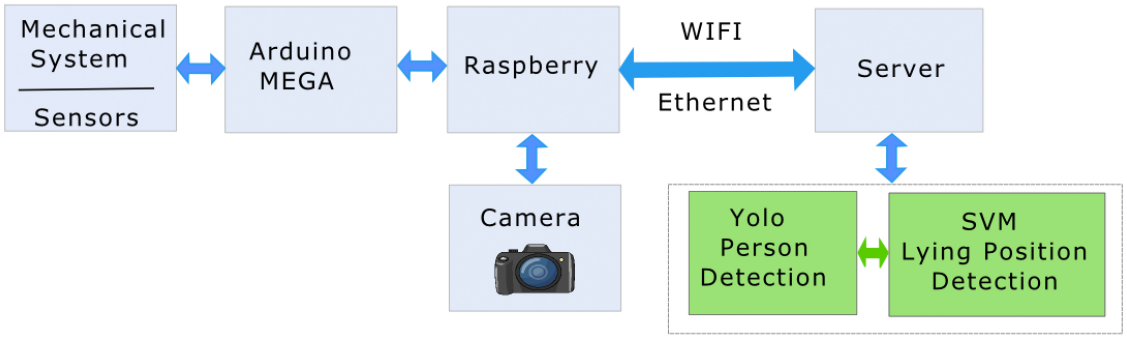


Figure 3. System-architecture overview.

3.2. Deep Learning-Based Person Detection

CNNs are one of the most popular machine-learning algorithm types at present and it has been decisively proven over time that they outperform other algorithms in accuracy and speed for object detection [28].

Algorithms for object detection using CNN can be broadly categorized into two-stage and single-stage methods. The two-stage algorithm based on classification first generates many proposals or interesting regions from the image (body) and then those regions are classified using the CNN (head). In other words, the network does not check the complete image; instead, it only checks parts of the image with a high probability of containing an object. Region-CNN (R-CNN) proposed by Ross Girshick in 2014 [29] was the first of this series of algorithms that was later modified and improved, for example, fast R-CNN [30], faster R-CNN [31], R-FCN [32], Mask R-CNN [33] and Light-Head R-CC [34]. However, single-stage algorithms based on regression do not use regions to localize the object within the image; the predict bounding boxes and class probabilities at the whole image. The most known examples of this type of algorithm are Single Shot Detector (SSD), proposed by Liu et al. [35] and 'you only look once' (YOLO) proposed by Joesph Redmon et al. in 2016 [36]. YOLO has been updated to versions YOLOv2, YOLO9000 [37] and YOLOv3 [38]. In this paper, we decide to apply real-time object detection system YOLOv3 for person detection, which has proven to be an excellent competitor to other algorithms in terms of speed and accuracy.

The YOLO network takes an image and divides it into S × S grids. Each grid predicts B bounding boxes $\{bi\}, i = 1, \ldots, B$ and provides a confidence score for each of them $Conf_{bi}$, which reflects how likely the box contains an object. Bounding boxes with this parameter above a threshold value are selected and used to locate the object, a person in our case. The bounding box position is the output of this stage for our algorithm.

3.3. Learning-Based Fall/Nonfall Classification

The effectiveness of SVM-based approaches for classification has been widely tested [39–41]. The SVM algorithm defines a hyperplane or decision boundary to separate different classes and maximize the margin (maximum distance between data points of the classes). Support vectors are training data points that define the decision boundary [42]. To find the hyperplane, a constrained minimization problem has to be solved. Optimization techniques such as the Lagrange multiplier method are needed.

In the case of nonlinearly separable data, data points from initial space R_d are mapped into a higher dimensional space Q where it is possible to find a hyperplane to separate the points. With this, the classification-decision function becomes

$$f(x) = sgn(\sum_{i=1}^{N_s} y_i \alpha_i K(x, s_i) + b) \tag{1}$$

where training data are represented by $\{x_i, y_i\}$, $i = 1, \ldots, N$, $y_i \in \{-1, 1\}$, b is the bias, α_i, $i = 1, \ldots, N$ are the Lagrange multipliers obtained during the optimization process [43] and s_i, $i = 1, \ldots, N_s$ are the support vectors, for which $\alpha_i \neq 0$ and $K(x, x_i)$ is a kernel function. A Radial Basis Function (RBF) was used as a kernel in this study:

$$K(x, x_i) = e^{-\gamma ||(x - x_i)||^2} \tag{2}$$

where γ is the parameter controlling the width of the Gaussian kernel.

The accuracy of the SVM classifier depends on regularization parameter C and γ. C is the parameter that controls the penalization associated with the training samples that are misclassified and γ defines how far the influence of a single training point reaches. So, both parameters must be optimized for every different task in particular, for example, by using cross-validation.

The selection of the right features or input parameters to the SVM plays an important role in having a high-performance classification algorithm. Some features are most widely used in the literature as aspect ratio (AR), change in AR (CAR), fall angle (FA), center speed (CS) or head speed (HS) [21,44,45]. However, after analyzing the parameters that provide the best trade-off performance for goals to achieve in our approach, using the bounding box data of a detected person, we defined the input feature vector for the SVM classifier as

- Aspect ratio of bounding box, AR_i:

$$AR_i = \frac{W_{bi}}{H_{bi}} \tag{3}$$

- Normalized bounding box width, NW_i:

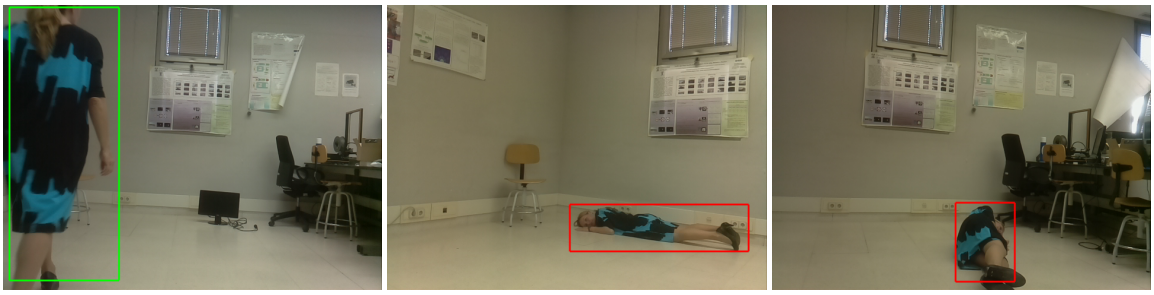
$$NW_i = \frac{W_{bi}}{W_{image}} \tag{4}$$

- Normalized bounding box bottom coordinate, NB_i:

$$NB_i = 1 - \frac{Ydown_{bi}}{H_{image}} \tag{5}$$

where $W_{bi} = Xright_{bi} - Xleft_{bi}$, $H_{bi} = Ydown_{bi} - Xtop_{bi}$ are the width and height of bounding box $\{bi\}$, respectively, calculated from the bounding box position provided by YOLOv3 $\{Xleft_{bi}, Xright_{bi}, Ytop_{bi}, Ydown_{bi}\}$ and W_{imagen}, H_{imagen} are the width and height of the overall image. Point $(0, 0)$ is at the top-left corner of the overall image. Parameter NB_i defines the distance from the bottom of the image to the lower part of the normalized bounding box. As the values of the NB_i and NW_i parameters are between 0 and 1, in order to give a similar weight to AR_i, we needed to adjust its value as input to the SVM. We analyzed the data and W_{bi} was lower than $10H_{bi}$ for all cases, so we normalized AR_i by 10 in order to get a feature in [0,1]. Therefore, we considered detection if $W_{bi} < 10H_{bi}$.

Parameter AR_i is the most significant feature that characterizes the fall. As can be seen from the examples in Figure 4a,b, a person standing upright has a small AR_i, while this ratio is large in the case of a person lying in a horizontal body orientation position. However, this parameter alone is not enough. There are some cases where the person is in a lying-position but this parameter does not show it; this is the case of lying in a vertical body orientation position, as we show in Figure 4c.



(**a**) (**b**) (**c**)

Figure 4. Aspect ratio. (**a**) Standing person $AR_i = 0.402$. (**b**) Fallen person in horizontal-pose orientation position $AR_i = 3.810$. (**c**) Fallen person in vertical-pose orientation position $AR_i = 0.751$.

One of the main goals of the algorithm is the ability to differentiate between fallen people and resting situations. Figure 5 shows one example of how the optical perspective in the cameras works. The object size in the image (in pixels) depends on the real image size (in mm) and the distance from the camera to the object [46]:

- Objects with the same size at different distances from the camera (object planes) appear with a different size (pixels) in the image plane; the closest one is visible in a larger size (Figure 5a);
- objects with the same size at the same distance to the camera (object planes) appear with the same size (pixels) in the image plane (Figure 5b). If objects are at different heights in the object plane, the same happens in the image plane.

When we compare a fallen person and a resting person at the same distance from the camera, the situation is the one observed in Figure 5b. The resting person is the person in the higher position. As shown in Figure 6a, the AR_i and NW_i parameters in both cases were the same (same size of bounding box); however, the NB_i parameter was different NB_1, NB_2. For the same value, NB_1, the bounding box size for a fallen person should be the red one (see Figure 6b).

Therefore, proposed parameters AR_i, NW_i and NB_i provide needed information for differentiating those situations and, during the training stage, the SVM learns the relation between them in both cases (fall and resting position).

Figure 7 shows the previous explanation with real images. It contains three pairs of images where fallen and resting persons are at the same distance from the camera (1.5, 2 and 3 m away). Table 1 shows the parameters provided to the SVM in those situations. As can be seen in the table, each pair of images have a similar NW_i parameter (slight differences are due to not being precisely at the same position from the camera). However, parameter NB_i had a larger value in the nonfall situation because the body was in a higher position in the image.

Table 1. Input parameters to the support vector machine (SVM) from images in Figure 7.

1.5 m			**2 m**			**3 m**		
AR$_i$	**NW$_i$**	**NB$_i$**	**AR$_i$**	**NW$_i$**	**NB$_i$**	**AR$_i$**	**NW$_i$**	**NB$_i$**
5.33	0.85	0.014	4.54	0.61	0.11	3.47	0.39	0.22
4.64	0.69	0.25	4.41	0.56	0.30	3.06	0.37	0.32

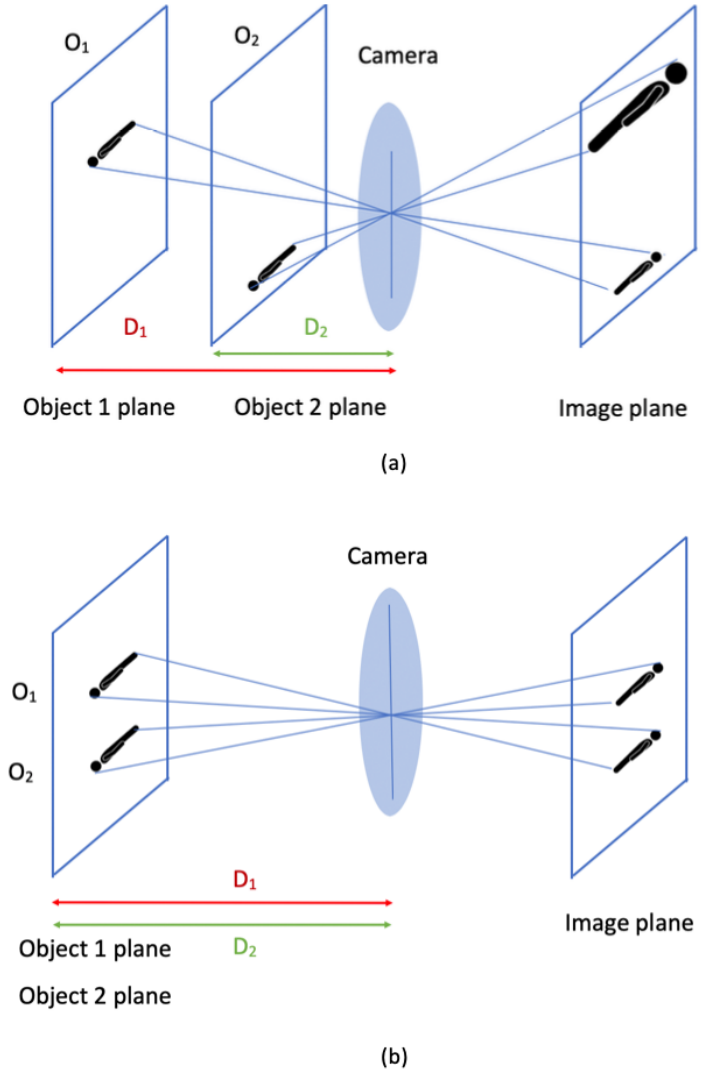


Figure 5. Optical perspective. Image plane for same object at (**a**) different distances and (**b**) different heights.

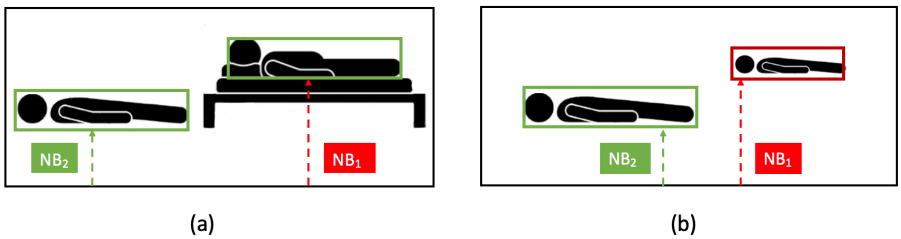


Figure 6. Relation between the NB_i parameter and the bounding box size. (**a**) Fallen and resting persons at same distance from camera. (**b**) Two fallen persons at different distances from camera.

Figure 7. Fall/nonfall detection 1.5, 2 and 3 m away (each pair of images are in the same column).

4. Experiment Results

4.1. FPDS Dataset

Analysis and comparison of different fall detection algorithms is a real problem due to the lack of public datasets with a large number of people in lying-positions [21,47,48]. ImageNet [28] and MS-COCO [49] are examples of those large image datasets. Some fall detection datasets provide images or videos with the camera situated in different positions but most of them in simulated environments [23,48,50–52]. However, they are neither large enough nor have all the required image variations for testing our experiments—several environments, more than one person in each image, persons in resting positions, falls with a variety of body orientations and persons with different sizes and clothes.

For all those reasons, in this paper, we present our own dataset (FPDS) to be used in fall detection algorithms. All images were taken by using a single camera inserted in a robot at 76 cm above the floor. This dataset consisted of a total of 2062 manually labeled images with 1072 falls and 1262 people standing up, sitting in a chair, lying on the sofa, walking and so forth. Images could have more than one actor and were recorded from different perspectives (Figure 8). An essential feature of this dataset compared with other datasets was having actors with a height range of 1.2–1.8 m (see Figure 9).

Figure 8. Fallen Person Dataset (FPDS) images with different lying-body orientations.

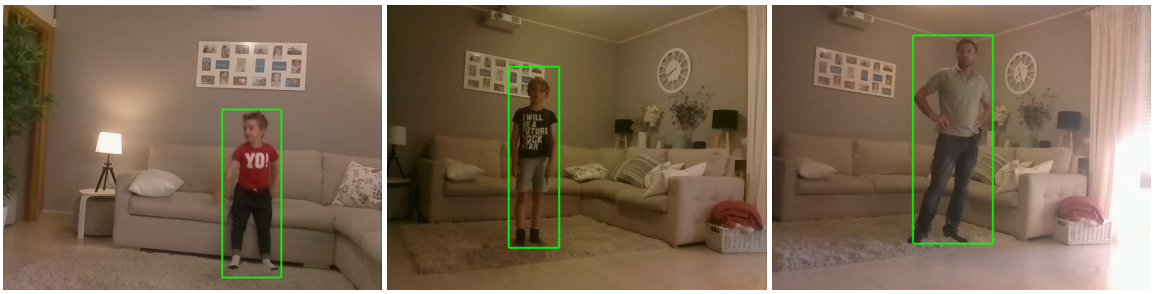


Figure 9. FPDS images with different person sizes—1.2, 1.4 and 1.8 m height.

Images were taken in eight different environments with variable illumination, as well as shadows and reflections, defining eight splits. Figure 10 and Table 2 show sample images and the characteristics of the FPDS, respectively.

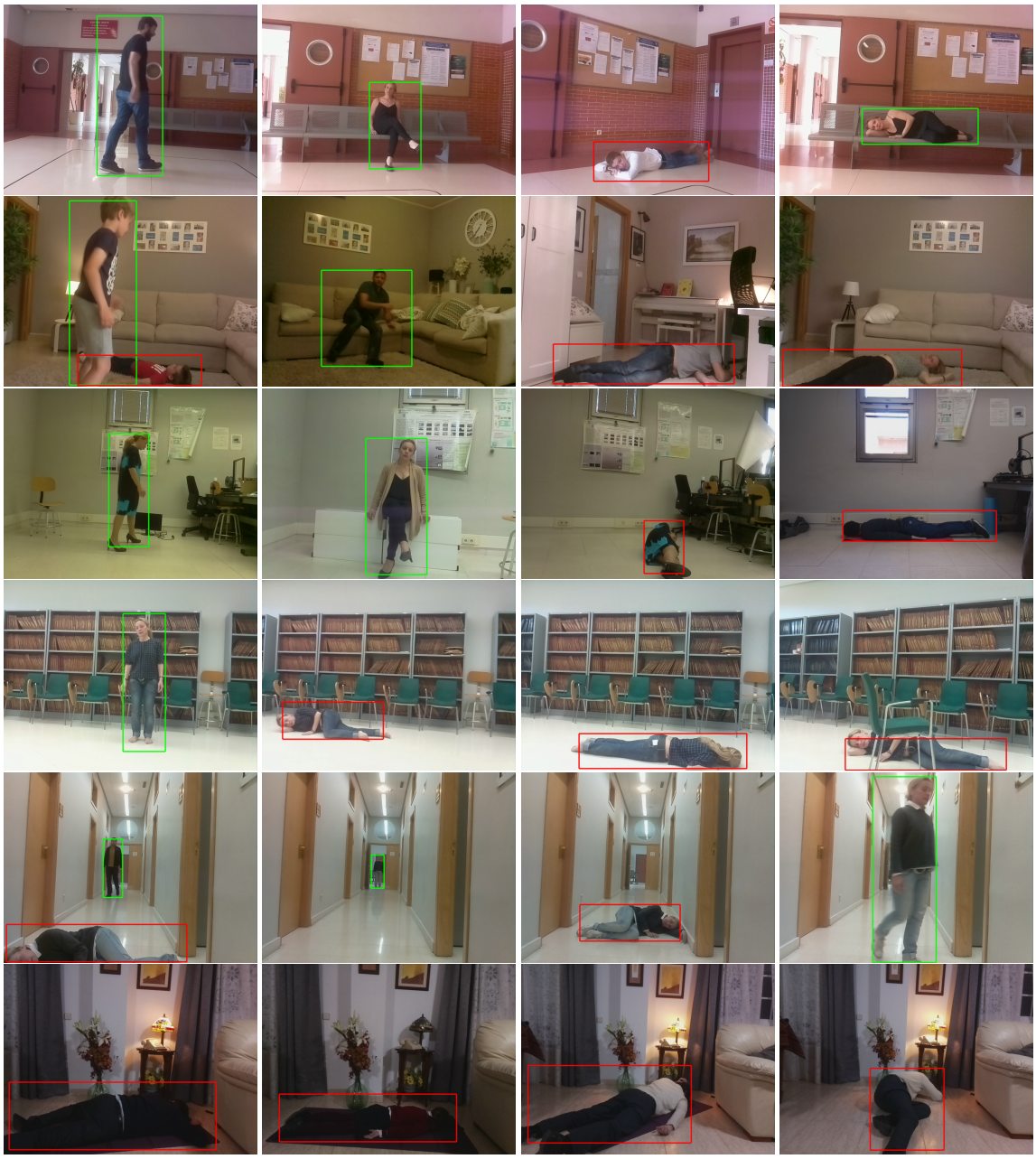


Figure 10. *Cont.*

Figure 10. Ground-truth image examples of the FPDS. Each row belongs to a different split. Bounding boxes are red/green in case of fall/nonfall detection.

Table 2. FPDS dataset characteristics.

	Split 1	Split 2	Split 3	Split 4	Split 5	Split 6	Split 7	Split 8	Total
Number of falls	278	223	180	104	49	42	15	181	1072
Number of nonfalls	175	82	175	3	704	0	39	84	1262
Number of images	400	323	368	117	553	42	51	210	2064

FPDS dataset consists of images and txt files with the same name. These files contain five parameters per bounding box in the image— bounding box coordinates $\{Xleft_{bi}, Xright_{bi}, Ytop_{bi}, Ydown_{bi}\}$ and the classification label y ($y = 1$ fall, $y = -1$ nonfall). Additionally, in the dataset we provided some sample images of a well-defined pattern (chessboard) taken with the camera from different perspectives for calibration purposes. FPDS dataset is public and available at http://agamenon.tsc.uah.es/Investigacion/gram/papers/fall_detection/FPDS_dataset.zip.

For all experiments, training and testing images belonged to different splits to correctly evaluate the ability of the algorithm to learn. We built training set L with splits 1, 2 and 3, by using 681 falls and 432 nonfalls in a total number of 1084 images. Testing set T was built by using from 4 to 8 splits, with 391 falls and 830 nonfalls in a total number of 973 images.

4.2. Metrics

To investigate the effectiveness of the method, we evaluate fall detection at the classifier-output level by measuring error rates, computed from good and misclassified images. However, to fully evaluate the algorithm, we needed to measure the precision and recall parameters [23]. Precision provides information about the proportion of positive fall identifications that are actually falls and recall the proportion of falls that were identified correctly. Unfortunately, these parameters work in different directions, meaning that improving precision typically reduces recall and vice versa:

$$Pr = \frac{TP}{TP + FP} \tag{6}$$

$$Re = \frac{TP}{TP + FN} \tag{7}$$

being

- True positives (TP)—number of falls correctly detected,
- false negatives (FN)—number of falls not detected and
- false positives (FP)—number of nonfalls detected as falls.

4.3. Experiment 1: Fall Classification

In this first experiment, we evaluated the performance of the learning-based fall/nonfall classification algorithm by itself without considering the person detection part. We use the ground-truth hand-labeled bounding boxes from our dataset as inputs. Cross-validation was performed on the training set to find the optimal C and γ values in the RBF SVM classifier. Figure 11 shows the accuracy-level curves for both parameters. We selected $\gamma = 2$ and $C = 128$ with an accuracy of 99.55%. These values were also established for Experiments 2 and 3.

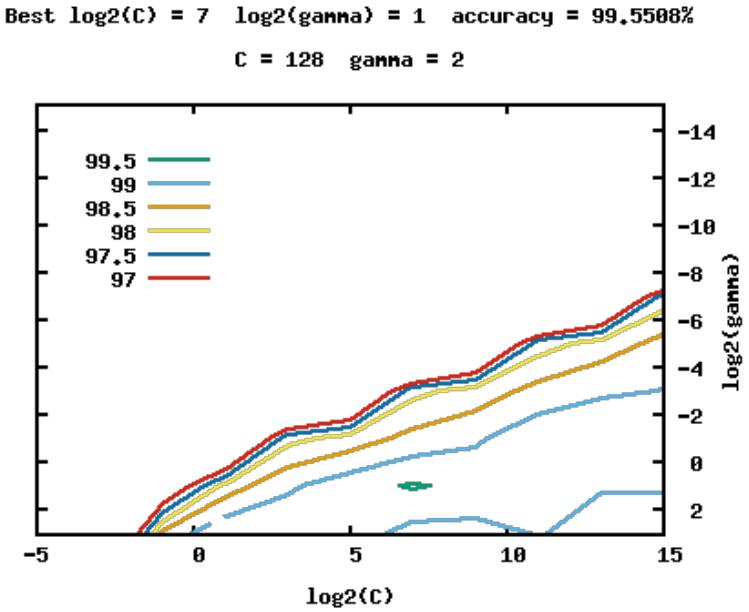


Figure 11. Accuracy-level curves during cross-validation for γ and C parameters in Experiment 1.

We summarize the experiment results in only one table to help with comparisons (Table 3). The first row of this table are the results of this experiment. The fall classifier detected 390 true positives, 1 false negative and 0 false positives, which means precision and recall of 100% and 99.74%, respectively. These results confirm a great selection of the selected input parameters to the SVM classifier.

Table 3. Performance over testing set T in the FPDS dataset.

	TP	*FN*	*FP*	*Pr* (%)	*Re* (%)
Experiment 1: Fall classification	390	1	0	100	99.74
Experiment 2: Fall detection algorithm	304	87	9	97.12	77.74
Experiment 3: Fall detection with pose correction	360	31	17	95.49	92.07

Several approaches have been proposed to detect falls, with good results. However, only a few of them take into account realistic datasets with different normal daily situations. One of the more complicated situations to solve is not detecting falls versus standing but rather falls versus resting situations where the person has a similar pose orientation. Our fall classifier can detect both situations in all cases that were tested. Figure 12 shows two examples.

Figure 12. Example images from testing test where algorithm differentiates between fallen person and resting person.

4.4. Experiment 2: Fall Detection Algorithm

The next experiment evaluated the performance of the overall end-to-end fall detection algorithm—person detection and fall classification. In this case, the person detection part was done by using deep learning method YOLOv3.

To maximize performance, the confidence score of bounding box conf_{bi} provided by YOLOv3 should have been above than a certain threshold Conf_i. We selected threshold value $\text{Conf}_i = 0.2$ for having good trade-off performance between recall and precision. Figure 13a shows this point by '*'.

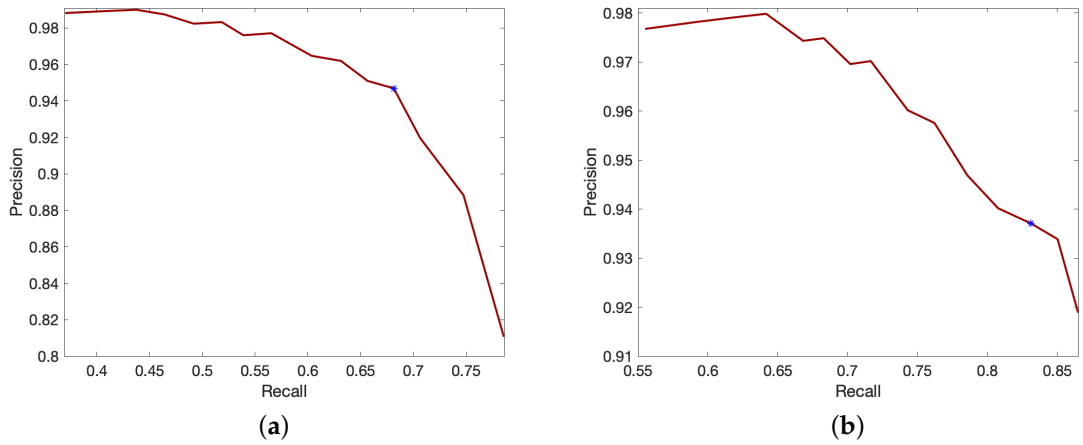


Figure 13. Recall and precision metrics for different thresholds. (**a**) Experiment 2, conf_i. (**b**) Experiment 3, conf_r.

Note the terminology—subindex "i" is used for parameters assigned to the "image directly from the camera" to differentiate them from the parameters assigned to the "rotated images" with subindex "r" that is explained in the next subsection.

We used Intersection over Union (IoU) as an evaluation metric to compare the bounding boxes provided by the fall detection algorithm and the ground-truth hand-labeled images from our dataset. To set a threshold value for the IoU, called IoU_i, we analyzed how this value affects the precision and recall parameters. It was observed that the values of these metrics were almost independent of the selected threshold, setting; in that case, value to $\text{IoU}_i = 0.2$. Values $\text{Conf}_i = 0.2$ and $\text{IoU}_i = 0.2$ were also established in Experiment 3.

As in the preceding subsection, the second row of Table 3 shows the results of testing set T for this experiment. It detected 304 true positives, 87 false negatives and 9 false positives. The values of precision and recall, in this case, were 97.12% and 77.74%, respectively. The false alarms were

mainly caused by errors in the person detection step of the overall algorithm. Therefore, if we compare with the performance of the SVM classifier itself, overall performance is worse. YOLOv3 was trained using the Common Objects in Context (COCO) dataset [49], which did not have enough lying-position persons for training the CNN to recognize persons in that position with high accuracy.

4.5. Experiment 3: Fall Detection with Pose Correction

YOLOv3 performance to detect persons in lying-positions improves with customized training using a dataset with a large number of persons in that position. However, to build this kind of dataset is costly and time-consuming. Due to the lack of public datasets with this characteristic at the moment, this training is not possible. The FPDS dataset proposed in this paper is useful for evaluating the robustness of the algorithm in different situations but does not have enough images for the customized training of YOLOv3.

The smallness of the training set represents a significant problem to the overall algorithm, as we analyzed in the previous experiment. Many have, therefore, tried to reduce the need for large training sets. In this article, we investigated how person pose position affects the efficiency of the approach. The experiments show that adding simple pose correction to YOLOv3 improves performance without the need for new customized training. The pose correction algorithm is explained in Figure 14. We ran three separate YOLOv3 networks, one for the initial image and two more for the rotated images at 90 and 270 degrees.

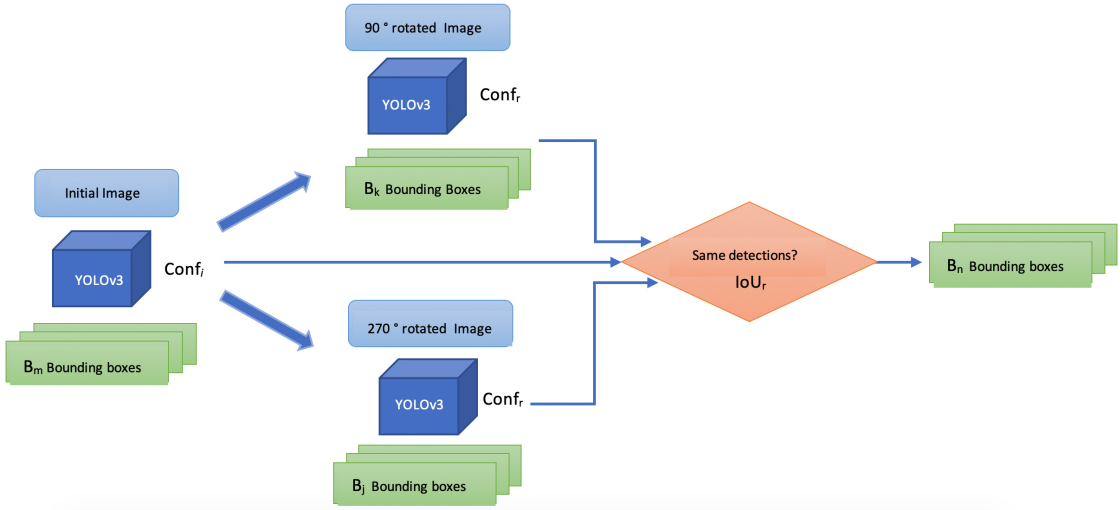


Figure 14. Flowchart of fall detection with pose correction.

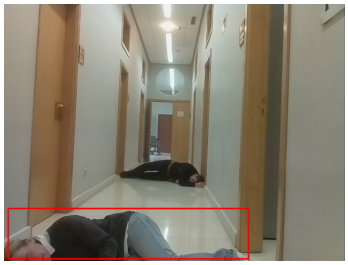
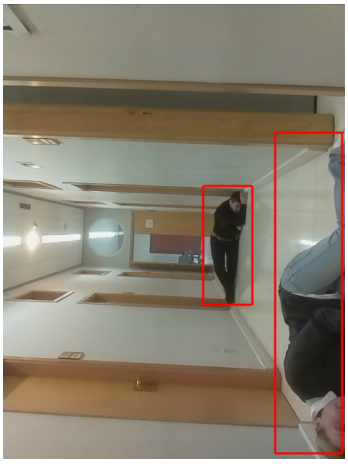
For better optimization, we analyzed whether the correct threshold of the confidence score applied to the rotated images, called conf_r, was the same as the one used for the image directly from the camera, conf_r. Figure 13b shows the precision and recall metrics for different thresholds. In this case, we obtained the best trade-off for a value of $\text{conf}_r = 0.15$, keeping the value of $\text{conf}_i = 0.2$ for the image directly from the camera.

This modified person-detector algorithm could detect the same fall more than once. To identify if the bounding boxes belonged to the same fall, we needed to establish a new threshold for the IoU parameter, called IoU_r. In case the bounding boxes are the same, the algorithm keeps only one; otherwise, it keeps both of them. A threshold of $\text{IoU}_r = 0.1$ provides a good trade-off between the precision and recall metrics.

Table 4 shows three examples from the testing set of the FPDS dataset with its detections in the initial and rotated images. In the first row, we can observe how the lying-person was detected in the two rotated images but not in the initial one. However, in the second example, the person was only

detected in the 270°-rotated image. In the last example, with two fallen persons, one of the falls was detected in the three images but the other one was only detected in the 270°-rotated image.

Table 4. Fall detection examples with pose correction.

Initial Image	90° Rotated Image	270° Rotated Image
		
		
		

Thanks to pose position correction, the overall method improved considerably in recall while keeping almost the same precision. Results are shown in the third row of Table 3. It has detected 360 true positives, 31 false negatives and 17 false positives with values of precision and recall of 95.49% and 92.07%, respectively.

4.6. Evaluation 1: Relocation for Doubtful Cases

One of the main points of the proposed approach is the ability of the robot to relocate itself when fall detection is doubtful. The relocation algorithm moves the robot depending on the size and position of the detected bounding box. In all cases, the robot moves to center the possible fall detection with proper dimensions. Figure 15 shows three different cases where the robot needs to move to get a better picture of the person.

Figure 15. Robot relocation in three different testing examples of the FPDS dataset.

4.7. Evaluation 2: Other Datasets

To evaluate the detection effectiveness of our algorithm, we needed to test the proposed approach with alternative algorithms described in the literature. In this case, we decided to use the public Intelligent Autonomous Systems Laboratory Fallen Person Dataset (IASLAB-RGBD) [52], close enough to our dataset. This dataset was generated by using a Kinect One V2 camera mounted on a mobile robot 1.16 m above the floor. We used static dataset with 374 images, 363 falls and 133 nonfalls. Despite our camera being 76 cm above the floor and the training set having been built by using the same splits of the FPDS dataset as in the other test experiments, results were quite satisfactory in Experiment 1, with precision and recall of 99.45% and 100%, respectively. However, detection was not so good in Experiments 2 and 3, as can be observed in Table 5. These results indicate the good selection of the input feature vector to the SVM, which makes the classifier almost independent of the camera setup. Giving the impossibility to compare the results directly, the comparison is proof of the good performance of our method with other datasets that contain images that considerably differ from the examples in the training set.

Table 5. Performance over the Intelligent Autonomous Systems Laboratory Fallen Person Dataset (IASLAB-RGBD).

	TP	*FN*	*FP*	*Pr* (%)	*Re* (%)
Experiment 1: Fall classification	363	0	2	99.45	100
Experiment 2: Fall detection algorithm	212	151	43	83.13	58.40
Experiment 3: Fall detection with pose correction	271	92	53	83.69	74.72

5. Conclusions and Future Work

In this paper, we presented a low-cost system for detecting falls in elderly populations and people with functional disabilities who are living alone. The system is based on an assistive patrol robot that

can be used for one, two, or more people. Our objective was to implement a vision-based fall detector system in our robot that acquires image data, detects falls and alerts emergency services. In our attempts to detect falls with an easy, fast and flexible end-to-end solution, we proposed a two-step algorithm. We combined a CNN to be used for person-detection and an SVM for fall classification. One of the main contributions of this paper was to find the combination features for the SVM-based classifier that provide the best performance for the design requirements. Results obtained from the different experiments indicate that the system had a high success rate in fall detection and could correct the position of the robot in case of doubt.

It is important to remark that, compared with existing fall detection approaches that show weakness in distinguishing between a resting position and a real fall scene, our fall classification algorithm could correctly detect both situations in all tested cases. Another important result to highlight is the ability to work correctly and detect fall situations with persons of different heights.

Since one of the goals of the work was to run a fall detection algorithm in real-time, it was needed to evaluate time implementation. In our case, the only time-consuming task was due to YOLOv3 person detection, which is more than acceptable for a real-time fall detection system.

We evaluated the robustness of the method using a realistic dataset called FPDS, which is publicly available and a contribution of this paper. The main features of this dataset are eight different scenarios, various person sizes, more than one person in an image, several lying-position perspectives and resting persons.

Additionally, we tested our algorithm using other datasets (training was done using the FPDS dataset). The results are quite satisfactory in fall classification, which showed us the almost-independence of the algorithm with the camera setup.

Future works to investigate are improvement in occlusion detection and the possibility to merge person detection and fall classification into a single CNN by using one or two different classes.

Author Contributions: Conceptualization, S.M.-B.; methodology, S.M.-B. S.L.-A., and C.I.-I.; software, C.I.-I.; validation, C.I.-I., S.M.-B., and P.M.-M.; writing—original-draft preparation, P.M.-M.; supervision, S.M.-B.

Funding: This work was partially supported by the Alcalá University research program through projects CCG2018/EXP-061 and TEC2016-80326-R from the "Ministerio de Economía, Industria y Competitividad".

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional Neuronal Network
RCCN	Region-based Convolutional Neuronal Network
YOLO	You Only Look Once
SVM	Support Vector Machine
RBF	Radial Basis Function
FPDS	Fallen Person DataSet
TP	True Positive
FN	False Negative
FP	False Positive

References

1. Ambrose, A.F.; Paul, G.; Hausdorff, J.M. Risk factors for falls among older adults: A review of the literature. *Maturitas* **2013**, *75*, 51–61. [CrossRef]
2. Rubenstein, L.Z. Falls in older people: Epidemiology, risk factors and strategies for prevention. *Age Ageing* **2006**, *35*, ii37–ii41. [CrossRef] [PubMed]
3. World Health Organization. WHO Global Report on Falls Prevention in Older Age. 2007. Available online: https://www.who.int/violence_injury_prevention/publications/other_injury/falls_prevention.pdf?ua=1 (accessed on 18 August 2019)

4. Noury, N.; Fleury, A.; Rumeau, P.; Bourke, A.; Laighin, G.; Rialle, V.; Lundy, J. Fall detection-principles and methods. In Proceedings of the 2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Lyon, France, 22–26 August 2007; IEEE: Piscataway, NJ, USA, 2007; pp. 1663–1666.
5. Mubashir, M.; Shao, L.; Seed, L. A survey on fall detection: Principles and approaches. *Neurocomputing* **2013**, *100*, 144–152. [CrossRef]
6. Igual, R.; Medrano, C.; Plaza, I. Challenges, issues and trends in fall detection systems. *Biomed. Eng. Online* **2013**, *12*, 66. [CrossRef]
7. Khan, S.S.; Hoey, J. Review of fall detection techniques: A data availability perspective. *Med. Eng. Phys.* **2017**, *39*, 12–22. [CrossRef]
8. Tamura, T.; Yoshimura, T.; Sekine, M.; Uchida, M.; Tanaka, O. A wearable airbag to prevent fall injuries. *IEEE Trans. Inf. Technol. Biomed.* **2009**, *13*, 910–914. [CrossRef]
9. Bianchi, F.; Redmond, S.J.; Narayanan, M.R.; Cerutti, S.; Lovell, N.H. Barometric pressure and triaxial accelerometry-based falls event detection. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2010**, *18*, 619–627. [CrossRef]
10. Tmaura, T.; Zakaria, N.A.; Kuwae, Y.; Sekine, M.; Minato, K.; Yoshida, M. Quantitative analysis of the fall-risk assessment test with wearable inertia sensors. In Proceedings of the 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Osaka, Japan, 3–7 July 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 7217–7220.
11. Rucco, R.; Sorriso, A.; Liparoti, M.; Ferraioli, G.; Sorrentino, P.; Ambrosanio, M.; Baselice, F. Type and location of wearable sensors for monitoring falls during static and dynamic tasks in healthy elderly: A review. *Sensors* **2018**, *18*, 1613. [CrossRef] [PubMed]
12. Mastorakis, G.; Makris, D. Fall detection system using Kinect's infrared sensor. *J. Real-Time Image Process.* **2014**, *9*, 635–646. [CrossRef]
13. Stone, E.E.; Skubic, M. Fall detection in homes of older adults using the Microsoft Kinect. *IEEE J. Biomed. Health Inform.* **2015**, *19*, 290–301. [CrossRef]
14. Sumiya, T.; Matsubara, Y.; Nakano, M.; Sugaya, M. A mobile robot for fall detection for elderly-care. *Procedia Comput. Sci.* **2015**, *60*, 870–880. [CrossRef]
15. Zigel, Y.; Litvak, D.; Gannot, I. A method for automatic fall detection of elderly people using floor vibrations and sound—Proof of concept on human mimicking doll falls. *IEEE Trans. Biomed. Eng.* **2009**, *56*, 2858–2867. [CrossRef]
16. Martinelli, A.; Tomatis, N.; Siegwart, R. Simultaneous localization and odometry self calibration for mobile robot. *Auton. Robot.* **2007**, *22*, 75–85. [CrossRef]
17. Zhang, T.; Zhang, W.; Qi, L.; Zhang, L. Falling detection of lonely elderly people based on NAO humanoid robot. In Proceedings of the 2016 IEEE International Conference on Information and Automation (ICIA), Ningbo, China, 1–3 August 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 31–36.
18. Máthé, K.; Buşoniu, L. Vision and control for UAVs: A survey of general methods and of inexpensive platforms for infrastructure inspection. *Sensors* **2015**, *15*, 14887–14916. [CrossRef]
19. Iuga, C.; Drăgan, P.; Buşoniu, L. Fall monitoring and detection for at-risk persons using a UAV. *IFAC-PapersOnLine* **2018**, *51*, 199–204. [CrossRef]
20. Zhang, Z.; Conly, C.; Athitsos, V. A survey on vision-based fall detection. In Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments, Corfu, Greece, 1–3 July 2015; ACM: New York, NY, USA, 2015; p. 46.
21. Debard, G.; Mertens, M.; Deschodt, M.; Vlaeyen, E.; Devriendt, E.; Dejaeger, E.; Milisen, K.; Tournoy, J.; Croonenborghs, T.; Goedemé, T.; et al. Camera-based fall detection using real-world versus simulated data: How far are we from the solution? *J. Ambient. Intell. Smart Environ.* **2016**, *8*, 149–168. [CrossRef]
22. Kim, K.; Chalidabhongse, T.H.; Harwood, D.; Davis, L. Real-time foreground–background segmentation using codebook model. *Real-Time Imaging* **2005**, *11*, 172–185. [CrossRef]
23. Charfi, I.; Miteran, J.; Dubois, J.; Atri, M.; Tourki, R. Optimized spatio-temporal descriptors for real-time fall detection: comparison of support vector machine and Adaboost-based classification. *J. Electron. Imaging* **2013**, *22*, 041106. [CrossRef]
24. Liu, C.L.; Lee, C.H.; Lin, P.M. A fall detection system using k-nearest neighbor classifier. *Expert Syst. Appl.* **2010**, *37*, 7174–7181. [CrossRef]

25. Wang, S.; Zabir, S.; Leibe, B. Lying pose recognition for elderly fall detection. *Robot. Sci. Syst. VII* **2012**, *29*, 345.
26. Nait-Charif, H.; McKenna, S.J. Activity summarisation and fall detection in a supportive home environment. In Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, 26 August 2004; IEEE: Piscataway, NJ, USA, 2004; Volume 4, pp. 323–326.
27. Rougier, C.; Meunier, J.; St-Arnaud, A.; Rousseau, J. Robust video surveillance for fall detection based on human shape deformation. *IEEE Trans. Circuits Syst. Video Technol.* **2011**, *21*, 611–622. [CrossRef]
28. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
29. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
30. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 1440–1448.
31. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 91–99.
32. Dai, J.; Li, Y.; He, K.; Sun, J. R-fcn: Object detection via region-based fully convolutional networks. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 379–387.
33. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
34. Li, Z.; Peng, C.; Yu, G.; Zhang, X.; Deng, Y.; Sun, J. Light-head r-cnn: In defense of two-stage object detector. *arXiv* **2017**, arXiv:1711.07264.
35. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2016; pp. 21–37.
36. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
37. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
38. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
39. Laptev, I.; Caputo, B. Recognizing human actions: A local SVM approach. In *Null*; IEEE: Piscataway, NJ, USA, 2004; pp. 32–36.
40. Zhang, H.; Berg, A.C.; Maire, M.; Malik, J. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, 17–22 June 2006; IEEE: Piscataway, NJ, USA, 2006; Volume 2, pp. 2126–2136.
41. Ebrahimi, M.; Khoshtaghaza, M.; Minaei, S.; Jamshidi, B. Vision-based pest detection based on SVM classification method. *Comput. Electron. Agric.* **2017**, *137*, 52–58. [CrossRef]
42. Burges, C.J. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.* **1998**, *2*, 121–167. [CrossRef]
43. Belousov, A.; Verzakov, S.; Von Frese, J. A flexible classification approach with optimal generalisation performance: Support vector machines. *Chemom. Intell. Lab. Syst.* **2002**, *64*, 15–25. [CrossRef]
44. Rougier, C.; Meunier, J.; St-Arnaud, A.; Rousseau, J. Fall detection from human shape and motion history using video surveillance. In Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07), Niagara Falls, ON, Canada, 21–23 May 2007; IEEE: Piscataway, NJ, USA, 2007; Volume 2, pp. 875–880.
45. Willems, J.; Debard, G.; Vanrumste, B.; Goedemé, T. A video-based algorithm for elderly fall detection. In Proceedings of the World Congress on Medical Physics and Biomedical Engineering, Munich, Germany, 7–12 September 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 312–315.

46. Svoboda, T.; Pajdla, T.; Hlaváč, V. Epipolar geometry for panoramic cameras. In Proceedings of the European Conference on Computer Vision, Germany, 2–6 June 1998; Springer: Berlin/Heidelberg, Germany, 1998; pp. 218–231.
47. Igual, R.; Medrano, C.; Plaza, I. A comparison of public datasets for acceleration-based fall detection. *Med. Eng. Phys.* **2015**, *37*, 870–878. [CrossRef] [PubMed]
48. Martínez-Villaseñor, L.; Ponce, H.; Brieva, J.; Moya-Albor, E.; Núñez-Martínez, J.; Peñafort-Asturiano, C. UP-fall detection dataset: A multimodal approach. *Sensors* **2019**, *19*, 1988. [CrossRef]
49. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 740–755.
50. Auvinet, E.; Rougier, C.; Meunier, J.; St-Arnaud, A.; Rousseau, J. *Multiple Cameras Fall Dataset*; DIRO-Université de Montréal, Tech. Rep; Université de Montréal: Montreal, QC, Canada, 2010; Volume 1350.
51. Kwolek, B.; Kepski, M. Human fall detection on embedded platform using depth maps and wireless accelerometer. *Comput. Methods Programs Biomed.* **2014**, *117*, 489–501. [CrossRef]
52. Antonello, M.; Carraro, M.; Pierobon, M.; Menegatti, E. Fast and robust detection of fallen people from a mobile robot. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 4159–4166.