Privacy Preserving Fall Detection Based on Simple Human Silhouette Extraction and a Linear Support Vector Machine

Velislava Spasova[1,*], Ivo Iliev[1], Galidiya Petrova[2]

[1]*Department of Electronics*
Technical University of Sofia
8 Kliment Ohridski Blvd., Sofia 1000, Bulgaria
E-mails: vgs@tu-plovdiv.bg, izi@tu-sofia.bg

[2]*Department of Electronics*
Technical University of Sofia, Plovdiv Branch
25 Tzanko Djustabanov Str., Plovdiv 4000, Bulgaria
E-mail: gip@tu-plovdiv.bg

[*]*Corresponding author*

Received: December 01, 2016 **Accepted: March 15, 2016**

Published: June 30, 2016

Abstract: *The paper presents a novel fast, real-time and privacy protecting algorithm for fall detection based on geometric properties of the human silhouette and a linear support vector machine. The algorithm uses infrared and visible light imagery in order to detect the human. A simple real-time human silhouette extraction algorithm has been developed and used to extract features for training of the support vector machine. The achieved sensitivity and specificity of the proposed approach are over 97% which match state of the art research in the area of fall detection. The developed solution uses low-cost hardware components and open source software library and is suitable for usage in assistive systems for the home or nursing homes.*

Keywords: *Ambient assisted living, Fall detection, Support vector machine.*

Introduction

Falls are among the most frequent causes of injuries at the elderly with one third of the people aged 65 and over falling every year. Falls pose significant health risk for old adults, especially if they live on their own and cannot signalize for help. A situation in which the fallen person is lying on the floor without the ability to get up for one hour or more is known as the 'long lie' and can result in dehydration, hypothermia, etc. and has a very negative impact for the physical and emotional recovery of the fallen person.

Ambient assisted living (AAL) systems are technological solutions that integrate sensors, actuators, processing and telecommunication units in or around the user's home environment. The AAL domain's main focus is to ensure higher quality of life to the independently living elderly. Due to the aforementioned reasons, automatic fall detection is one of the required modules for almost any AAL system and as such has been in the researcher's focus for the past ten years.

A recent trend that emerges in the field of automatic fall detection is the use of computer vision (CV) algorithms. Images and video provide a rich source of information that could be used to build a more robust and generic fall detection systems. The drawback of using computer vision for the purposes of fall detection is that computer vision algorithms are often

slow and resource-heavy and it is not always possible to run them in real-time, especially on embedded platforms. In addition to that they might be perceived by the users of an AAL system as too intrusive and may raise privacy concerns.

Privacy protection is one of the important challenges that the designers of AAL systems need to fulfill. There are a number of additional requirements that have to be taken into account in order to ensure financial feasibility, maintainability and acceptance by the users. Table 1 presents the most important of these requirements.

Table 1. Specific requirements to AAL systems

Requirement	Description
Privacy	Users value privacy very high. There can be no constant video monitoring.
Low-cost	The price of the whole system should be low so that large scale adoption is possible.
Real-time	Systems should operate in real time, especially in cases of emergency detection.

As it has been mentioned, privacy is very important to the users of AAL systems and consequently, the use of video in AAL systems is strongly discouraged. The other important requirements reflect some of the technological challenges to AAL systems – response time and suitability for large-scale home adoption. AAL systems, especially emergency detection (such as fall detection) systems, have to run in real-time in order to ensure that timely response and adequate action could be taken. In addition to that the cost of the system should be low so that mass adoption could be facilitated. All of these requirements have been reflected in the presented fall detection solution.

The remainder of the paper is organized as follows: the next section discusses relevant work in the area of computer vision based fall detection, the third section details the presented system's architecture; the forth section presents the human detection module of the system; the fifth and sixth sections provide overview of the silhouette extraction and fall detection modules; and the final section concludes the paper.

Related work

Fall detection

In recent years there has been promising research applying CV techniques for fall detection. Support vector machines in particular have been used to classify postures and falls [4, 10, 15]. The approach presented by Qian et al. [10] trains a cascade of support vector machines (SVM) on features extracted from the minimal bounding rectangle (MBR) of a human in order to classify different postures, such as walking, sitting, crawling and lying. Foroughi et al. [4], and Yu et al. [15] extract features from the ellipse fitted around the human silhouette in order to train a multi-class SVM into classifying different postures. All three approaches use video sequences for the classification purposes.

Another emerging trend has been the use of Microsoft Kinect™ and its RGB and depth sensors [6, 8, 14]. All of these approaches also use video sequences in the fall detection procedures. A completely different approach is presented by Feng et al. [2]. They use

silhouettes of humans from video frames in order to train a deep neural net to classify into fall and non-fall frames.

As it can be seen, most of the CV based fall detection approaches use videos which poses a problem to their adoption in an AAL system. Some of the algorithms use videos because it simplifies the human detection and silhouette extraction components. In fact, most of the background subtraction techniques used for segmentation between foreground and background rely on building a model of the background from video frames. As the use of videos in AAL systems is discouraged, the solution presented in this paper uses as input only single images and consequently a silhouette extraction algorithm had to be developed to accommodate this limitation.

Human detection

Human detection is a necessary step from a CV based fall detection algorithm, especially for on that does not use videos and background subtraction. However, the majority of the efforts in human detection domain have been concentrated in the development of algorithms that detect humans in videos and to a lesser extent in images. The images or videos are typically captured by standard digital or web cameras.

Lately researchers have started combining other sensor modalities with visible spectrum imagery in order to achieve better accuracy of detection and faster algorithms runtime. One strong direction of research, particularly in the robotics domain, is the fusion between visible light and thermal or depth imagery in order to enhance computer vision algorithms [5, 9]. Depth images are often obtained through the range camera of the Microsoft Kinect™ kit [5, 9, 11, 13]. Other researchers use thermal cameras that provide very high-resolution thermal images but have prohibitively high prices for use at home environments [1, 3]. An interesting alternative is the approach presented in [7], which is most similar to the idea adopted in the presented fall detection system. The authors in [7] use simple low-cost infrared (IR) sensor which is spatially and temporally aligned with the color imagery from a digital camera in order to track multiple humans in a home environment.

As it can be seen from the review of related research, a lot of the approaches to fall detection and human detection use video and/or expensive hardware components. The approach presented in this paper is targeted towards a low-cost solution which nevertheless ensures high reliability of the provided results and at the same time uses single images. The architecture of the presented system is detailed in the following section.

System architecture

The proposed solution is a novel, reliable, real-time, low-cost, and privacy preserving fall detection system based on computer vision. The system uses a standard web camera for obtaining visible light imagery and an inexpensive infrared sensor array for gathering infrared data. The architectural overview of the system is presented at Fig. 1.

As it can be seen from the figure, the system is split into two components: a home component and a remote component. The home component comprises the sensors, home gateway and software installed in the user's home, whereas the remote component includes a remote medical server and the accompanying infrastructure for notification of interested parties in case of a fall alert. The two components communicate over the internet using a secure cryptographic channel.

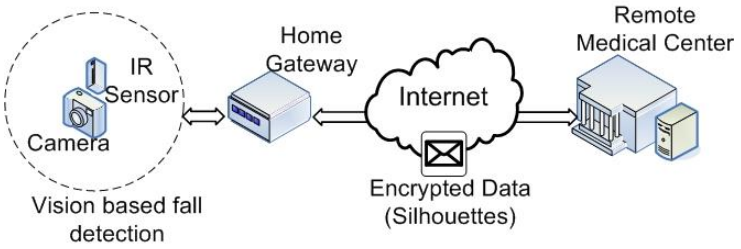


Fig. 1 System architecture of proposed fall detection solution

The fall detection algorithm executes on the home gateway which serves as a control unit for the home component and is also responsible for the communications to the medical server. The algorithm processes standalone still images only – video is not captured by the system. In addition to that all data processing of the images is done locally, only silhouettes are transmitted over the Internet to the remote medical server. This confines the processing of sensitive information to the user's home and enables better protection of his/her privacy. Moreover, the user receives sound notification prior to image capturing and is able to switch off the module if he/she is not in a state of helplessness. This is measure to prevent unnecessary breach of the user's privacy.

For the purposes of fall detection every pose in which the user is located on the floor such as lying, kneeling, sitting or crawling on the floor is considered a fall. Positions in which the user is upright or lying on a bed or sofa are considered non-fall positions due to activities of daily living.

The computer vision based module can be used as a standalone fall detection module or as a fall verification tool for fall detection based on other less reliable sensor modalities. A flow chart of the CV module is presented at Fig. 2.

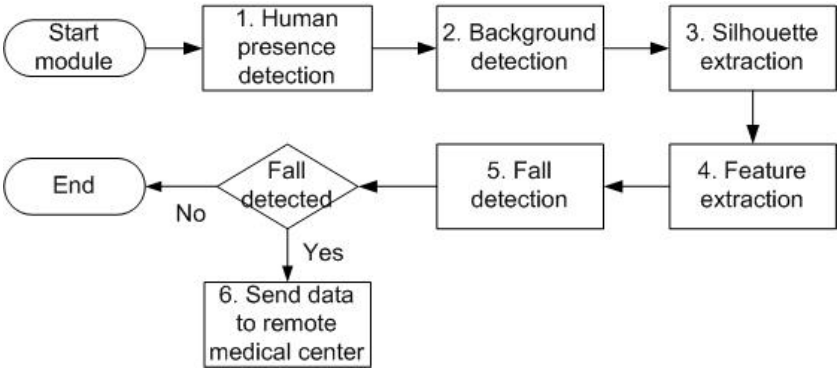


Fig. 2 Flow chart for computer vision based module

The first step of the module is to detect the user (Human presence detection). The IR sensors array is used in order to detect human thermal emissions. Once the user has been detected a picture of him/her can be captured by the web camera. The system captures images of the background regularly throughout the day when the room is not occupied for later usage during fall detection. In order to use these images in the silhouette extraction step, the system has to know which background image corresponds to the image with the user (step 2 – Background matching). An innovative approach for background detection based on keypoints matching has been developed. Once the matching background has been determined, the system moves on to the third step – Silhouette extraction. A simple silhouette extraction algorithm has been designed and evaluated as part of this step. The next step is Feature extraction, during which

the features used for detection are extracted (geometric properties of the silhouette), followed by step 5 – Fall detection. A machine learning approach based on linear support vector machines has been adopted. Finally, if a fall has been detected the system sends an alarm and other significant data, such as an image of the extracted silhouette to the remote medical server.

Hardware setup

The proposed system uses ARM-based board A13-OlinuXino-WIFI with A13 microcontroller by Allwinner which runs Debian Linux distribution and has a variety of interfaces such as universal serial bus (USB) ports and inter-integrated circuit (I2C). There is one visual unit (web camera + infrared sensor) per room. The unit is wall mounted and rotational. A standard web camera connected to the board through USB is used. It captures grayscale images with resolution of 640×480 pixels. For gathering IR data a low-cost, low-resolution IR array MLX90620 by Melexis has been used. The IR sensor has 4×16 pixels and is connected to the development board via I2C. All implementations of image processing algorithms are used from the free and open source software library OpenCv 2.4.9.

Human detection

Spatial alignment

Before the two computer vision signal inputs, namely the inputs from the web camera and from the IR array, can be used, they have to be aligned spatially. The IR sensor has a field of view of 60 degrees, whereas there isn't any information about the field of view of the web camera. The spatial alignment procedure used in the presented solution is illustrated at Fig. 3.

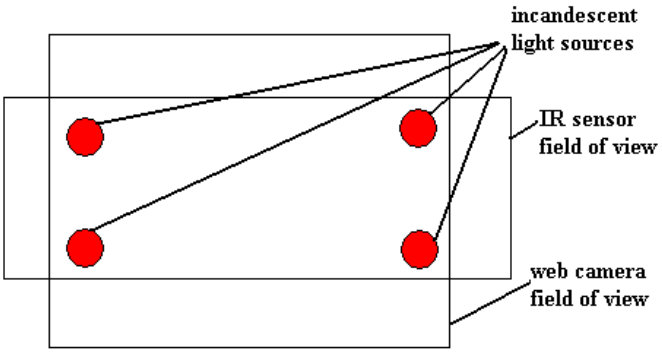


Fig. 3 Spatial alignment of the web camera and IR sensor

The lens of the web camera and the IR sensor are positioned as close as possible so that they can have maximum overlap of their fields of view. Four incandescent light sources placed as shown at the figure have been used. As incandescent light sources emit light in the visible as well as in the IR spectrum, they can be detected in both images and consequently the two sensors can be aligned. It has been discovered that only 14 out of the 16 IR pixels per row are needed and that the height of the area covered by the IR sensor corresponds to approximately 2/3 of the height of the area covered by the web camera.

Human presence detection

In this step the algorithm evaluates whether there is a human present in the aligned field of view of the IR sensor. The IR sensor and web camera are placed on a platform which is automatically rotated by 10° until it covers the whole room. The algorithm which has been developed checks whether there is a source of IR emission in the temperature interval that is

representative for the temperature of the human body. The temperature of the human body is typically measured in the range of 35-38 °C. However, due to the presence of clothing, partial occlusions, differences in ambient temperature and the uneven distribution of temperature along the body (e.g. limbs have typically lower temperature than the head), the algorithm has to account for significant variations to the lower interval boundary. Experimental evaluation of 4 different temperature ranges has been performed in order to determine the best range for human presence detection. In order to conduct the experiments 20 positions for which a human is in the field of view of the infrared sensors and 15 in which there is no human in view are used per temperature range. Results of the experiment are presented in Table 2.

Table 2. Human presence detection results

Interval, °C	*Sensitivity*, %	*Specificity*, %	*Accuracy*, %
24-38	88.24	87.5	88
25-38	100	94.44	98.08
26-38	100	87.5	95.83
27-38	100	94.12	97.78

The results are evaluated in terms of sensitivity, specificity and accuracy. Sensitivity in this case can be defined as the percentage of the successfully detected human presences and is calculated as:

$$sensitivity = \frac{TP}{TP + FN}, \tag{1}$$

where *TP* is true positives, and *FN* are false negatives.

Specificity indicates the percentage of non-presences that are detected as non-presence and is calculated as:

$$specificity = \frac{TN}{TN + FP}, \tag{2}$$

where *TN* is true negatives, and *FP* is false positives.

Accuracy gives us the percentage of the correctly classified human presence vs. non-presence emissions and is calculated as:

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}. \tag{3}$$

The most important metric to maximize is sensitivity, e.g. no human should be left undetected. As it could be seen from the results in Table 2, the interval from 25 °C to 38 °C gives the best results along these terms. Once a human is detected, the web camera takes a picture and passes it to the second stage of the algorithm – human silhouette extraction.

Human silhouette extraction

The aim of the human silhouette extraction phase is to confirm the result from the human presence phase, and to extract the silhouette so that any subsequent processing could use only the silhouette instead of the whole image. This module has three distinct inputs: the image captured by the web camera in the human presence stage; the coordinates of the possible human IR emission in this image; and a set of background images of the room. The background images are taken at every 10° of the room through rotation of the web camera at fixed times throughout the day when the room was empty.

Background detection

From the above inputs the exact background image that corresponds to the image obtained from the human presence module has to be found. In order to do this, a novel background matching approach has been proposed [12]. It uses keypoints matching between the potential human image and all the backgrounds in order to find the pair with the highest number of matching keypoints. Different combinations between keypoints detectors, descriptors and matchers have been tested. The best combination in terms of accuracy was found to be Features from Accelerated Segment Test keypoints detector, Binary Robust Independent Elementary Features keypoints descriptor and Fast Library for Approximate Nearest Neighbors keypoints matcher. With the upgrade to the OpenCV 2.4.8 software library, this is also the fastest combination (with 1.8 seconds for processing of an image pair). An example for the matching between correct backgrounds is presented at Fig. 4.

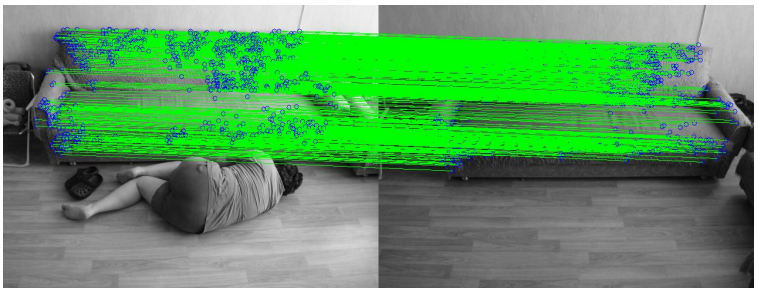


Fig. 4 Keypoints matching for background detection

Simple silhouette extraction

Once the correct background has been identified, the algorithm proceeds with the silhouette extraction module. As it has been mentioned above, there are a number of foreground/background segmentation algorithms, however the majority of them rely on a video input in order to build a model of the background. The proposed module implements a simple human extraction silhouette which takes as an input the image with the user, captured by the human presence detection module and the matching background detected by the background detection module. A flow chart for the algorithm is presented at Fig. 5.

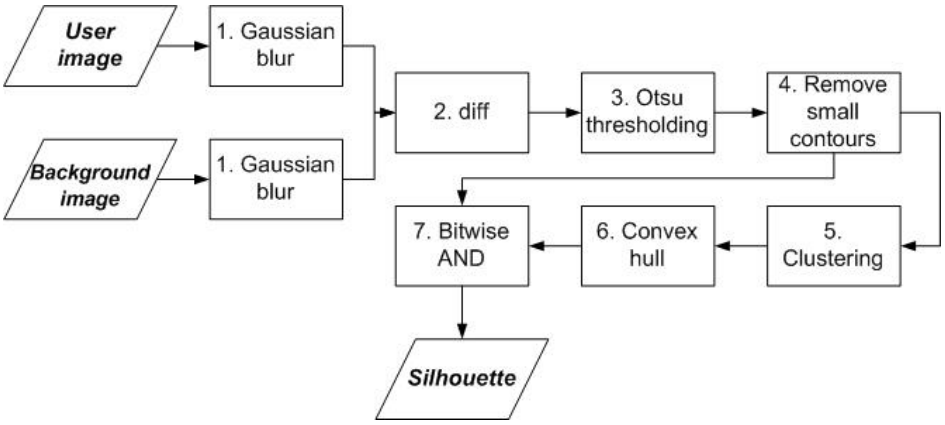


Fig. 5 Simple human silhouette extraction

The silhouette extraction algorithm starts by blurring both input images with a Gaussian filter which aims to reduce high frequency noise from the images. The Gaussian function used to calculate a kernel with which to convolve the image is defined in Eq. (4):

$$G(x,y)=\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4}$$

where $G(x, y)$ is the Gaussian function for two dimensions, x and y are the distances from the origin of the horizontal and vertical axes (which is the center of the kernel), and σ is the standard deviation of the Gaussian (normal) distribution. After experimentation with various Gaussian kernels calculated with the above function, a 21×21 kernel obtained for σ = 1.1 was determined as the one producing optimal results.

After the blurring, a difference image is formed taking the difference of the two blurred input images. The resulting grayscale difference image roughly reflects the changes in the scene introduced by the presence of the human, i.e. the human silhouette. In order to be able to calculate the geometric properties of this silhouette it has to be transformed into a binary image. Otsu thresholding, which is an algorithm for automatic detection of the threshold value, has been used for this task. Its application to the difference image produces a binary image in which the human silhouette is white and the background is black.

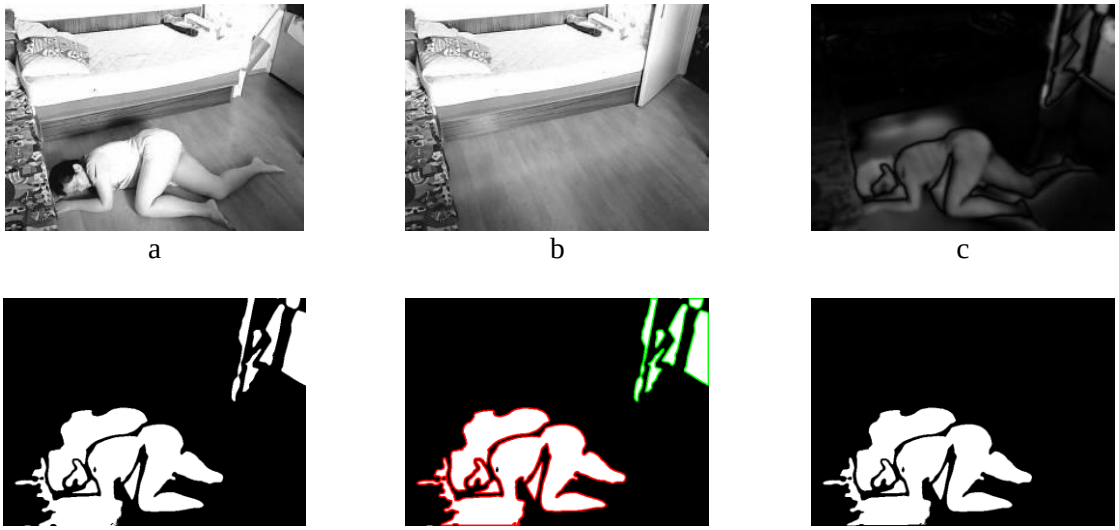
At this stage of the algorithm the produced silhouette contains a lot of small white regions even though the high frequency noise has been removed. These regions are produced by furniture and objects displacement, shadows due to illumination changes, etc. In order to reduce the number of these regions two approaches can be adopted. The first approach relies on morphological operations such as erosion and dilation to remove the unnecessary regions from the binary image. This approach is effective for small regions removal but it introduces considerable distortions in the shapes of the remaining regions. The second approach is to calculate the area of all white regions in the image and to remove small regions. The latter approach has been adopted in the proposed algorithm. The area of an image region is given by its order image spatial moment M_{00}:

$$M_{00}=\sum_{x,y} I(x,y), \tag{5}$$

where x and y are the coordinates within the image coordinate system and $I(x, y)$ is the value of the pixel at position (x, y).

After the removal of small regions the resulting image contains larger white regions which are grouped into clusters with the larger cluster representing the human silhouette. In order to leave only the regions that belong to the human silhouette a distance based clustering algorithm has been designed and implemented. It separates the white regions into clusters based on the distance between the regions. Once the image is separated into clusters, the convex hulls of the clusters are calculated and only the cluster whose convex hull has the highest area is preserved.

An example for the original user and background images can be seen at Fig. 6a and Fig. 6b, for the difference image – at Fig. 6c and for the image after Otsu thresholding and small regions removal – at Fig. 6d. The clusters of the clustering algorithm are illustrated at Fig. 6e and the final result from silhouette extraction – at Fig. 6f. As it can be seen from the figure, the resulting silhouette is imperfect – there are shadows and part of the legs are removed during the extraction process. Despite this, the result is a good approximation of the human silhouette, especially taking into account that it is extracted from a single image and without explicit model of the background.



a b c d e f

Fig. 6 Silhouette extraction phases: (a) original user image; (b) original background image; (c) difference image; (d) silhouette after removal of small regions; (e) clusters of the silhouette image; (f) final image.

The runtime of the silhouette extraction algorithm has been measured on the A13-OlinuXino board. The average runtime is 0.18 seconds which make this algorithm is particularly well suited for real-time scenarios.

Fall detection

The fall detection algorithm that is presented in this paper is based on a linear support vector machine. It uses features derived from the human silhouette obtained from the human silhouette extraction module. Similar approaches have been presented in literature but with varying success. The presented approach is novel in its privacy protecting use of silhouettes and single images instead of the more intrusive video.

Feature extraction

The input to the fall detection module is the silhouette image produced by the silhouette extraction module. The silhouette is appropriate for human inspection but in order to be used in a computer vision algorithm it has to be approximated by a simpler geometric shape. The MBR and the fitted ellipse are two good candidates for silhouette approximation. The MBR and fitted ellipse for an example silhouette image are presented at Fig. 7.

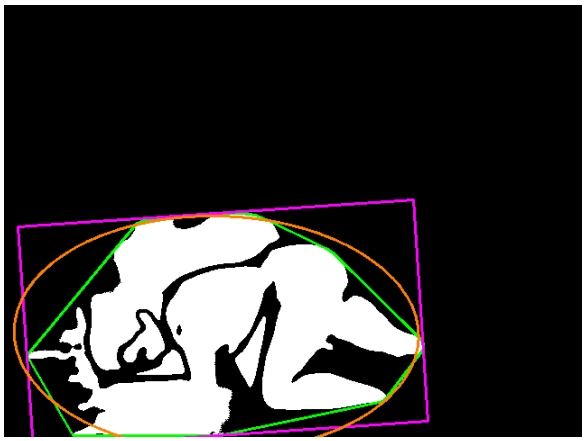


Fig. 7 Hull (green), fitted ellipse (orange), and minimal bounding rectangle for a silhouette image

The MBR is the minimal rectangle that encloses the silhouette. Its height *h*, width *w*, and tilt angle θ are obtained by the iterative rotated calipers method. The parameters of the general quadratic curve representation of the fitted ellipse are obtained through the direct least squares algorithm. The general quadratic curve has the form:

$$ax^2 + 2bxy + cy^2 + 2dx + 2fy + g = 0, \tag{6}$$

where *x* and *y* are the coordinates of a point lying on the ellipse, and (*a*, *b*, *c*, *d*, *f*) are the parameters returned by the algorithm. In order to calculate the major and minor axes as well as the tilt angle of the ellipse the Eqs. (7)-(9) are used:

$$a' = \sqrt{\frac{2(af^2 + cd^2 + gb^2 - 2bdf - acg)}{(b^2 - ac)\left[\sqrt{(a-c)^2 + 4b^2} - (a+c)\right]}}, \tag{7}$$

where a' is the major semi-axis;

$$b' = \sqrt{\frac{2(af^2 + cd^2 + gb^2 - 2bdf - acg)}{(b^2 - ac)\left[-\sqrt{(a-c)^2 + 4b^2} - (a+c)\right]}}, \tag{8}$$

where b' is the minor semi-axis;

$$\theta = \begin{cases} 0; \ (b = 0, \ a < c) \\ \frac{\pi}{2}; \ (b = 0, \ a > c) \\ \frac{1}{2}\cot^{-1}\left(\frac{a-c}{2b}\right); \ (b \neq 0, \ a < c) \\ \frac{\pi}{2} + \frac{1}{2}\cot^{-1}\left(\frac{a-c}{2b}\right); \ (b \neq 0, \ a > c) \end{cases}, \quad (9)$$

where θ is the tilt angle of the fitted ellipse.

In order to train the SVM the following features derived from the silhouette image have been used: the area of the human silhouette; the coordinates of the topmost, bottommost, leftmost and rightmost points of the silhouette; the width, height and tilt angle of the MBR along with the aspect ration (the ratio of the width to the height of the MBR); the major and minor axis along with the tilt angle and the aspect ratio of the fitted ellipse. These features form a 17-dimensional feature vector which is prepared for feeding into the classifier.

However, before the SVM can be trained, the training and testing data have to be standardized. The process of standardization, also known as *Z*-score normalization, yields features that are rescaled in the interval [–1; 1] and centered at 0, thus having the properties of a standard distribution with mean $\mu' = 0$ and standard deviation $\sigma' = 1$. The standard scores (also called *z*-scores) of the features are calculated as:

$$z_i = \frac{x_i - \mu_i^j}{\sigma_i^j}, \quad (10)$$

where z_i, $i = \{1, \ldots, n\}$ is the standardized feature, n is the number of features in the feature vector (in this case 17), x_i is the value of the feature prior to standardization, μ_i^j is the mean of the feature x_i over all j image samples, and σ_i^j is the standard deviation for the feature x_i over all j image samples. The standardization is a general pre-processing requirement for many machine learning algorithms, including SVMs. In order to have correct classification results both training and test data have to be standardized.

Algorithm

The algorithm that is used for fall detection is a linear SVM which has as input 17-dimensional standardized feature vectors. The support vector machine transforms these vectors to a higher dimensional space in which the vectors are linearly separable, and then tries to build an optimal separating hyperplane which separates the training feature vectors into classes. Any new test feature vector is classified based on its position with regards to the hyperplane. There may be infinitely many hyperplanes that separate the training set but the optimal hyperplane is the one that has the largest functional margin, i.e. the largest minimum distance to the training examples. The optimal hyperplane separated the classes with the highest degree of generalization.

Let us have a training set defined as:

$$\left\{(x_i, y_i),\ i = 1,...,n,\ x_i \in R^d,\ y_i \in \{+1,-1\}\right\}, \tag{11}$$

where x_i are the input feature vectors of the training set, and y_i are the labels corresponding to these feature vectors. There are only two classes of samples – class –1 and class +1 (in this case corresponding to non-fall and fall). Let H denotes the new feature space and Φ denotes the mapping of the feature vectors from R^d to H, so that:

$$\Phi : R^d \rightarrow H\,. \tag{12}$$

Thus, a training example (x_i, y_i) becomes ($\Phi(x_i)$, y_i). Then, we search for a hyperplane in H, so that a transformed feature vector $\Phi(x_i)$ lies on one side of the hyperplane if $y_i = -1$ and on the other side of the hyperplane if $y_i = +1$. The equation of this hyperplane in H can be represented in terms of a vector ω and a scalar b as:

$$\omega \cdot \Phi(x) + b = 0\,, \tag{13}$$

where $\cdot$ is the dot product.

It is proven than ω is the normal to the hyperplane and $|b|/||\omega||$ is the distance of the hyperplane from the origin. Since there are finite number of training samples, if a hyperplane separates the training set, than each training sample must be at least β away from the hyperplane for some $\beta > 0$. Then, we can renormalize (13) to require that:

$$y_i(\Phi(x_i) \cdot \omega + b) - 1 \geq 0 \quad \text{for } i = 1, \ldots, n. \tag{14}$$

In general, it may be impossible to separate linearly the training data with a hyperplane, even in H. However, it can be searched for a hyperplane that separates the data as much as possible while also trying to maximize the margin. In order to do this, a relaxation variable ξ_i, $\xi_i \geq 0$ for $i = 1, \ldots, n$ is introduced so that the following is satisfied:

$$y_i(\Phi(x_i) \cdot \omega + b) - 1 + \xi_i \geq 0 \quad \text{for } i = 1, \ldots, n. \tag{15}$$

The relaxation variables ξ_i, also called slack variables, measure the degree of misclassification of the data x_i, i.e. the distance between the wrongly classified x_i and its correct classification region. In essence they add a penalty for violating the classification constraints.

If d_+ is the distance from the support vectors to the hyperplane for the class +1 and d_- is the distance form the support vectors to the hyperplane for the –1 class, then $d_+ = d_- = 1/||\omega||$ and the margin between the two classes is $d_+ + d_- = 2/||\omega||$. In order to maximize the margin, thus to have a better generalization of the classification, $||\omega||$, or equivalently $||\omega||^2$ has to be minimized. In order to account for the relaxation variables, a penalty term of the form $C\sum_i \xi_i$, where C is some appropriate constant, has to be added. Thus, searching for an optimal hyperplane results in solving the following optimization problem:

$$\text{minimize } \|\omega\|^2 + C\sum_i \xi_i$$

$$\text{subject to } y_i(\Phi(x_i) \cdot \omega + b) - 1 + \xi_i \geq 0 \quad \text{for } \xi_i \geq 0,\ i = 1, \ldots, n. \tag{16}$$

Through solving the optimization, the values of ω and b are returned and the equation of the hyperplane is found. This process of solving the optimization and obtaining the hyperplane is the process of training. Once the SVM is trained, new (unseen during training) feature vectors x, thus new images, are classified by checking on which side of the hyperplane they fall. During training an optimal value of the regularization constant C has to be established through experimentation. Typically, large values of C produce solutions to the optimization margin that result in smaller margins and less misclassification errors. An opposite – small values of C result in solutions with large margins and more classification errors. Thus, a balance should be looked for on a case by case basis.

The fall classification model is trained on a server and once trained is transferred to the home gateway situated in the user's home. During fall detection, the module forms a feature vector out of the current silhouette image and fits it into the model in order to check on which side of the hyperplane it is. If the image is recognized as a fall image, a fall alarm is issued.

Experimental evaluation

The algorithm has been trained and tested on a dataset of fall and non-fall images. The dataset consists of 1829 labeled images of 5 volunteers (ages 27 to 81 years old) in different poses. 784 images represent falls in different positions and orientations while the remaining 1045 images represent different activities of daily living. Any image in which the person is lying, kneeling, sitting or crawling on the floor is considered as a fall image. Images in which the person is lying on a couch or bed are considered as no-fall images.

Before the SVM is trained an appropriate value for the regularization constant C has to be determined – this is the so called hyperparameter tuning. During the tuning the goal is to maximize the important evaluation metrics, in this case the sensitivity, while also getting as high specificity and accuracy as possible. It was experimentally determined that the best value for C is 1.0 as can be seen from Fig. 8.

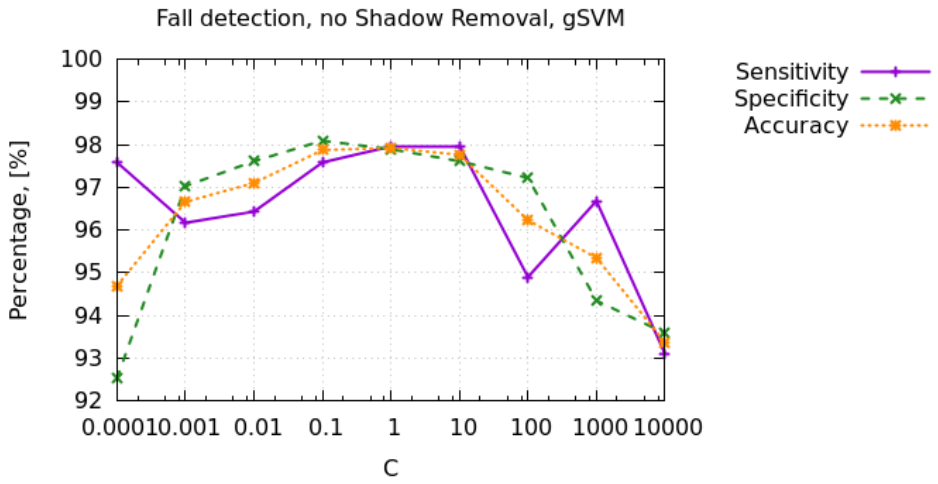


Fig. 8 Hyperparameter tuning for SVM based fall detection

In order to evaluate the SVM performance and how well it generalizes to independent (previously unseen) data, the technique of cross-validation has been used. The goal of cross-validation is to use three datasets in order to obtain the highest performing and most unbiased classifier. The datasets that are used are: one training dataset which is used for training; one validation dataset which is used to tune the hyperparameters of the classifier;

and one test dataset to evaluate the performance of the classifier on previously unseen data. In order to divide the initial dataset, first some of the data samples are randomly selected for the test dataset. The remaining training data samples are split into training and validation datasets. There are various methods to split the training dataset into training and validation sets. In this algorithm was chosen stratified 10-fold cross-validation which splits the training set into 10 equal folds with roughly the same class distribution. Every fold participates exactly once as a validation fold in a total of 10 rounds of the cross-validation run.

In the evaluation of the presented algorithm, 20% of the data samples have been used as a test dataset with the remaining 80% for 10-fold stratified cross-validation. The initial dataset contains 1829 data samples, 784 out of which represent falls and the remaining 1045 – activities of daily living (ADLs). After the split 366 samples are randomly left for testing (156 falls and 210 ADLs) and the remaining 1463 for cross-validation (628 falls and 835 ADLs).

The SVM is first trained on the training set, and then tested on the testing set. The achieved results are:

- *sensitivity* – 97.96%;
- *specificity* – 97.89%;
- *accuracy* – 97.92%;
- average runtime (A13-OlinuXino) – 0.068 seconds.

These results are very good – sensitivity of almost 98% while also having very high specificity of over 97.5% matches state of the art fall detection algorithms. Moreover, these results are achieved by a machine learning algorithm which is much more generic than a threshold based solution. Additionally, the runtime for fitting is under 1 second. Overall, the whole algorithm – human presence detection, background detection, silhouette extraction and fall detection, runs in under 1 minute on an embedded device like the A13-OlinuXino which makes it suitable for real-world emergency detection systems.

Conclusion

This paper presents a novel machine learning based fall detection solution with particular focus on privacy protection. Several original contributions have been presented in the area of personal assistive systems. The proposed approach makes use of a fusion between visible light and infrared imagery in order to detect humans. In addition to that a new method for background detection has been proposed and developed. The paper presents a simple human silhouette extraction algorithm which is based only on single still images. The silhouettes obtained from this module have been integrated in a linear SVM algorithm in order to build a robust and reliable fall detection system.

The reliability in terms of sensitivity, specificity and accuracy as well as the average runtimes on an embedded platform have been experimentally evaluated. The results from the experiments show that the proposed system as a whole, and each of its components, achieve very high sensitivity and specificity of detection. In addition to that the runtimes allow the usage of the system in real time solutions. The proposed solution is build with low-cost hardware components and uses free and open source software which makes it suitable for mass scale home installation.

Moreover, the whole system is focused on delivering reliable results without compromising the privacy of its users. This would allow better acceptance by the elderly and faster adoption of the end product.

References

1. Correa M., G. Hermosilla, R. Verschae, J. Ruiz-del-Solar (2012). Human Detection and Identification by Robots Using Thermal and Visual Information in Domestic Environments, Journal of Intelligent and Robotic Systems, 66(1-2), 223-243.
2. Feng P., M. Yu, S. M. Naqvi, J. Chambers (2014). Deep Learning for Posture Analysis in Fall Detection, Proc. of the 19th Intl. Conf. on Digital Signal Processing, Hong Kong, China, August 2014, 12-17.
3. Fernandez-Caballero A., J. C. Castillo, J. Serrano-Cuerda, S. Maldonado-Bascon (2011). Real-time Human Segmentation in Infrared Videos, Expert Systems with Applications, 38(3), 2577-2584.
4. Foroughi H., A. Rezvanian, A. Paziraee (2008). Robust Fall Detection Using Human Shape and Multi-class Support Vector Machine, Proc. of the 6th Indian Conf. on Computer Vision, Graphics and Image Processing, Bhubaneswar, India, December 2008, 413-420.
5. Gritti A. P., O. Tarabini, J. Guzzi, G. A. Di Caro, V. Caglioti, L. M. Gambardella, A. Giusti (2014). Kinect-based People Detection and Tracking from Small-Footprint Ground Robots, Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS 2014), Chicago, USA, September 2014, 4096-4103.
6. Han J., L. Shao, D. Xu, J. Shotton (2013). Enhanced Computer Vision with Microsoft Kinect Sensor: A Review, IEEE Trans. Cybernetics, 43(5), 1318-1334.
7. Kumar S., T. Marks, M. Jones (2014). Improving Person Tracking Using an Inexpensive Thermal Infrared Sensor, Proc. of the Int. Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW 2014), Columbus, USA, June 2014, 217-224.
8. Mastorakis G., D. Makris (2014). Fall Detection System Using Kinect's Infrared Sensor, Journal of Real-Time Image Processing, 9(4), 635-646.
9. Murano M., E. Menegatti (2014). Fast RGB-D People Tracking for Service Robots, Autonomous Robots, 37(3), 227-242.
10. Qian H., Y. Mao, W. Xiang, Z. Wang (2009). Home Environment Fall Detection System based on a Cascaded Multi-SVM Classifier, Proc. of the 10th Int. Conf. on Control, Automation, Robotics and Vision, Hanoi, Vietnam, December 2009, 1567-1572.
11. Salas J., C. Tomasi (2011). People Detection Using Color and Depth Images, Proc. of the 3rd Mexican Conference on Pattern Recognition (MCPR 2011), Cancun, Mexico, June-July 2011, 127-135.
12. Spasova V. (2014). Experimental Evaluation of Keypoints Detector and Descriptor Algorithms for Indoors Person Localization, Annual Journal of Electronics, 2014, 85-87.
13. Spinello L., K. Arras (2011). People Detection in RGB-D Data, Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS 2011), San Francisco, USA, September 2011, 3838-3843.
14. Stone E., M. Skubic (2015). Fall Detection in Homes of Older Adults Using the Microsoft Kinect, IEEE Journal of Biomedical and Health Informatics, 19(1), 290-301.
15. Yu M., A. Rhuma, S. M. Naqvi, L. Wang, J. Chambers (2012). Posture Recognition based Fall Detection System for Monitoring an Elderly Person in a Smart Home Environment, IEEE Trans. Information Technology in Biomedicine, 16(6), 1274-1286.

Velislava Spasova, Ph.D. Student
E-mail: vgs@tu-plovdiv.bg

Velislava Spasova is a Ph.D. student at the Department of Electronics, Technical University of Sofia. She is working actively in the area of ambient assisted living, and is particularly interested in fall detection for the elderly. She holds an M.Sc. in Computer Engineering.

Prof. Ivo Iliev, D.Sc.
E-mail: izi@tu-sofia.bg

Ivo Iliev graduated from the Technical University – Sofia, Faculty of Electronic Engineering and Technology, division of Biomedical Engineering in 1989. He is presently with the Department of Electronics of the Technical University – Sofia, working on methods and instrumentation for bio-signal registration and analysis, telemetry and wireless monitoring of high-risk patients, assistive systems for elderly and disable people.

Prof. Galidiya Petrova, Ph.D.
E-mail: gip@tu-plovdiv.bg

Galidiya Petrova, Ph.D. in Biomedical Engineering, is a Professor in the Department of Electronics, Faculty of Electronics and Automation, Technical University of Sofia, Plovdiv branch. Her research interests are within data acquisition systems, applications of distributed systems in medicine, personalized healthcare and ambient assisted leaving systems, wireless body sensor networks.