Journal of Engineering Science and Technology
Vol. 13, No. 8 (2018) 2587 - 2605

INDOOR HUMAN FALL DETECTION SYSTEM BASED ON AUTOMATIC VISION USING COMPUTER VISION AND MACHINE LEARNING ALGORITHMS

KISHANPRASAD GUNALE[1,*], PRACHI MUKHERJI[2]

[1]Department of Electronics and Telecommunications, Sinhgad College of Engineering, Survey No. 44/1, Off Sinhgad Road, Vadgaon Budruk, Pune-411041, Maharashtra, India
[2]Department of Electronics and Telecommunications, Cummins College of Engineering, Survey No. 11, 2,Karve Nagar, Pune-411052, Maharashtra, India
*Corresponding Author: kishanprasad.gunale@mitwpu.edu.in

Abstract

The detection of unnatural falls among older adults living alone at home is an essential requirement of the day. Injury after a fall is common, and immediate medication is often necessary. A fall detection system using the following features such as orientation angle, aspect ratio, Motion History Image and object below the threshold line is proposed. The combination of these features can lead to a robust fall detection system. These features are used as an input for fall detection systems using machine-learning techniques such as Support Vector Machine, K-Nearest Neighbors, Stochastic Gradient Descent, Decision Tree and Gradient Boosting. The effect of using different classifiers and feature sets for the performance of fall detection is observed. This approach provided a promising *F*-measure value of 96% for Decision Tree algorithm on Coffee room environment.

Keywords: Computer vision, Fall detection, Feature extraction, Motion history image, Support vector machine classifiers, Video surveillance.

.

1. Introduction

Visualizing activity in the home environment has gained much interest in the research community. Recently, modelling of human motion and behaviours for activity patterns and event detection has attracted significant research interest. Various places like airport, railway station, shopping mall, sports stadium and hospital are potential locations for monitoring events. Anomalous event detection techniques can be easily extended towards the healthcare industry for fall detection. In older persons, falls lead to a more risk and cause significant injuries.

Given that the frequency of falls increases with age, fall detection systems based on computer vision will be beneficial for the elderly. According to Chan et al. [1], one-third of the elderly population are injured due to falls each year. Among them, 3% have fractures, 56% have soft tissue injuries, 5.3% are hospitalized, and 90% have hip injuries that can be attributed to unexpected falls. The effects of falling can get worse and can result in the death of a person. If corrective measures are not taken at the proper time, then the number of elderly affected can reach to be 100% higher by the year 2030 [2]. In this context, there is a need to come up with a robust system for vision-based fall detection.

Fall detection is classified into two main categories; vision and non-vision based. A non-vision attributed method based on a wearable sensor that employs inertial sensors, such as accelerometers; for fall detection, is widely used [3]. The sensor-based approaches are mostly intrusive while the vision-based system is non-intrusive. The motion of the object is captured and analysed with respect to its position associated with lying on the floor, standing or sitting. The features used in the vision-based system can be posture, shape, pixel information, silhouette distance, head position, orientation angle, etc. Audio-based features are combined with non-vision-based and vision-based methods to improve the accuracy [4].

In this paper, the authors propose vision-based algorithms to detect the object while machine-learning techniques are used for the classification of the fall activities. Most of the fall monitoring systems are supported by the functional algorithm of detection, tracking and human motion analysis with low level (background subtraction, pre-processing, object detection, tracking, and classification), middle level (machine learning algorithms), and high-level components (semantic interpretation). The aim of this research is to integrate different features for designing a robust fall detection system suitable for various environments and investigate the effect of the feature set on various classifiers.

2. Literature Survey

Vision-based methods are used significantly due to the ease of event monitoring, as no wearable devices are attached to the body. Different approaches have been adapted for fall detection using effective features.

Ali et al. [5] categorized fall and non-fall events based on the extracted feature such as centroid, head position, a motion vector, aspect ratio, head speed, center speed, and fall angle. These features were classified using the Support Vector Machine (SVM) classifier. This approach achieved an accuracy of 86.36%. The authors recommended using night vision cameras to improve accuracy. Chen et al. [6] presented fall detection based on the combination of outline features and human shape variations. The curve approximation, Delaunay triangulation, and skeleton extraction were carried out by the Douglas-Peuker algorithm. This approach reached 90.9% accuracy. The elliptical bounding box gives more exact information than the rectangular bounding box-like shape, falling angle, eccentricity, and aspect ratio. In another research, Chen et al. [7] proposed a shadow assistant fall detection system. They classified the falling and non-falling events by linear SVM. Cheng et al. [8] proposed a framework for the accurate detection of falling. In the first step, data samples were reduced by transforming data into data vectors. Finally, the postures were classified by using the Hidden Markov Model (HMM) and One-Class Support Vector Machine (OCSVM). The advantage of this system is a good balance between a false alarm and false negative rates. The combination of the three approaches achieved a 90.5% accuracy rate.

Colon et al. [9] proposed a system where all sensors information were gathered and built into the smartphone. The smartphone was placed in different pockets such as chest pouch, trousers side pocket, and cell phone holster. The authors obtained an accuracy of 83.3%, 87.5%, and 81.3% for chest pouch, side pocket and cell phone holster, respectively. Delgado et al. [10] proposed a video surveillance system to monitor and identify events in crowded scenes. In this method, they mostly focused on detecting unusual activity on the train platform such as jumping or falling off. The experimental evaluation was conducted using a recorded video dataset. This method achieved a precision of 90%. In the future, this approach can be extended by estimating the depth information of an object.

Jiang et al. [11] presented a robust method for detecting falls based on human posture and motion of the silhouette. This system detects all-directional falls. The proposed system performs operations with good accuracy and high processing speed. The author indicates the scope of work in fall detection for crowded and complex situations. Lin et al. [12] presented an accidental fall detection system in the real-time environment. Aspect ratio and inclination angles were extracted as features, and K-Nearest Neighbors (KNN) was used for classification purposes. This system achieved an accuracy of 86.11%.

Qian et al. [13] proposed a fall detection system based on human silhouettes evolution. In this approach, firstly, the silhouettes of a moving object are detected and then the Poisson ratio is calculated by solving the 2D Poisson equations defined in the spatial-temporal accumulative image. Secondly, the Histogram of Oriented Gradients (HOG) is collected and stored as an action descriptor. Finally, the action descriptor is fed to the nearest neighbour classifier to classify the fall action, which achieved an average recognition accuracy of 98.88%. Rougier et al. [14] described the tracking of the silhouette. The shape deformation was quantified using shape analysis techniques. The Gaussian mixture model was used to separate fall and non-fall events. The evaluation was performed on the real-time dataset. Bian et al. [15] proposed a fall detection approach by tracking body parts individually. The videos are fed into this system using depth-based cameras. The key joint extraction is carried out by the Randomized Decision Tree (RDT) algorithm.

A silhouette shape-based system has been proposed by Lin et al. [16]. In this system, a fall is detected using a Gaussian mixture background model. Motion History Image (MHI) is applied to analyse the fall behaviour and improve fall detection accuracy, using acceleration and angular acceleration. Ma et al. [17] extracted the shape-based features from depth videos. In this system, the Bag of

Curvature Scale Space (BoCSS) word was built using CSS features and classified using Extreme Learning Machine (ELM) classifier. Yu et al. [18] have proposed a semi-unsupervised fall detection system. The author mentioned online OCSVM-based human fall detection system. Features like elliptical properties, shape and position are given as input to an OCSVM. Madhubala and Umamakeswari [19] extracted the features using context-aware feature extraction techniques. Shape and mean value of the image are used to categorize the activity into fall and non-fall events, and in the case of unusual activity, an alert is sent through the GSM modem. The system is implemented on a Raspberry Pi platform. According to Nguyen et al. [20], falls are detected based on the analysis of orientation, magnitude and human shape. By analysing the speed of change in motion magnitude, motion orientation and human shape falls are detected. The authors achieved *F*-measure of 93% for coffee room and 95% for the home environment.

After studying various fall detection methods, it is observed that there is a vast scope for the development of an efficient fall detection system by combining different fall description features.

3. Overview of Fall Detection System

Figure 1 describes the state of art of the proposed system. It is separated into two main parts: training and testing. The database selection for the proposed technique is an important task. The videos for evaluation have been selected from the Le2i fall database (coffee room, home environment, lecture room, and office). The building block of the proposed system is explained in Sections 3.1 to 3.5.

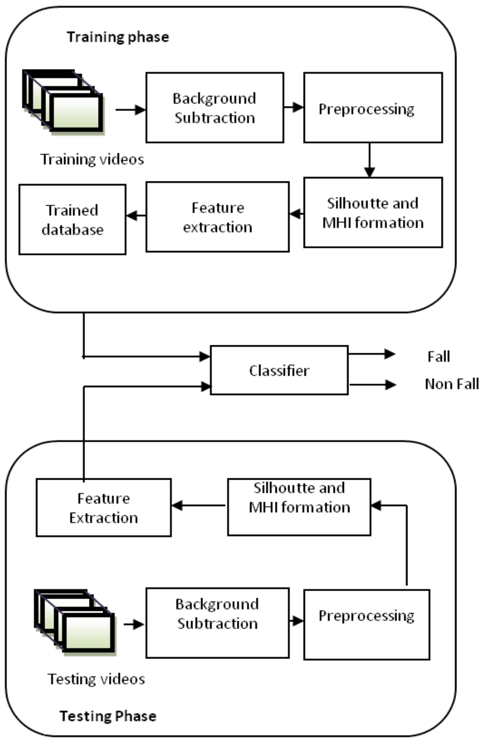


Fig. 1. Schematic diagram of fall detection system.

3.1. Video acquisition from Le2i database

To evaluate fall detection activities, the videos of this database are recorded with single camera views in different indoor scenarios like the coffee room, home, lecture room, and office, as shown in Fig. 2. Videos are recorded at 25 fps speed and with the resolution of 320 × 240 pixels. This database contains 191 videos with respective ground truth [21]. In this approach, 142 videos from the coffee room, home, lecture room and office environment are considered for training, and the proposed system is tested on the basis of 54 videos from all the four databases. The detailed information of the database is tabulated in Table 1. It contains the number of training and testing videos along with the fall and non-fall events.

(a) Home.

(b) Coffee room.

(c) Lecture room.

(d) Office.

Fig. 2. Le2i database fall activity videos in different scenarios.

Table 1. Fall database information.

Le2i Database	Training videos	Training frames	Training database fall events	Training database non-fall event	Testing videos
Coffee room	53	16213	4891	11322	17
Home	45	10201	3705	6496	15
Lecture room	13	9239	1247	7992	06
Office	31	12182	7232	4950	16

3.2. Background subtraction

Initially, the input video is converted into frames. For moving object detection, absolute frame differencing method is implemented. The absolute difference (I_d) between the current frame at time t and previous frame at a time $(t-1)$ is calculated as given in Eq. (1).

$$I_{d\,(t,t-1)} = |I_t - I_{t-1}| \quad (1)$$

where I_t is the pixel intensity of the current frame, and I_{t-1} is the pixel intensity of the previous frame. Movement is detected by recognizing large motions, which may or may not be attributed to a fall because different motions like running and jumping also attribute to large motions.

3.3. Pre-processing

In the first phase of fall detection, pre-processing of the frame is used to remove the noise from an image. Pre-processing makes the video appropriate for the system. Filtering is the most commonly used smoothing operation. The most widely used filter is a linear filter, and the output pixel value is defined in Eq. (2).

$$f(x,y) = \sum_{k,j}^{n} f(x-k, y-t)h(k,l) \quad (2)$$

where $f(x,y)$ the linear filter is output, $h(k,l)$ is called the kernel or coefficients of the filter and $f(x-k, y-t)$ is the time delay sequence. This section contains two main steps: Binarization and morphological operation. The salt and pepper noisy frames were firstly filtered using the median filter and further converted into binary using Otsu's global thresholding method [22]. In the median filter, the mask of size 5×5 is used as the kernel. The mask is convolved with the original image, and the middle pixel value is replaced by a median of the convolved mask.

3.4. Feature extraction

The transformation of visual features into statistical measures is called feature extraction.

3.4.1. Motion history image

When a fall occurs, the motion of the object might be large and that may be a key to detect the fall event. Therefore, the large motion is an important factor in detecting the fall. The MHI is a popular way to detect the large motion of the moving object. Bobick and Davis [23] proposed the technique of MHI where the latest movement in an image sequence is denoted by pixel intensity, and the most recent movement is commonly used for activity identification. The larger movement was taken to remove the small change, and it eliminates all other small motions. To calculate the motion coefficient, the proportion of bright pixels W in MHI is calculated as shown in Eq. (3).

$$W(motion) = \left(\frac{Number\ of\ white\ pixels}{Total\ Number\ of\ pixels} \right) \times 100 \quad (3)$$

3.4.2. Change in the human shape

The second feature considered for the human fall detection system is a human silhouette shape. The shape of the silhouette changes after a fall. Instead of applying a rectangular bounding box, the ellipse is fitted over the silhouette because, of which its noise immunity increases and it extracts exact human body information [24]. An ellipse is fitted around the foreground object to extract information of the falling person. The aspect ratio and orientation angle play an important role in fall activity detection. The detailed explanation is shown in Fig. 3.

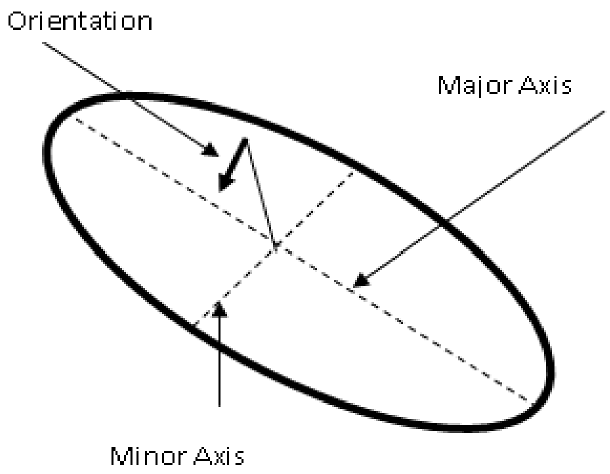


Fig. 3. Ellipse feature.

- **Orientation angle**

  Orientation gives the global direction of the object. It is the angle between the horizontal *x*-axis and the major axis of the ellipse.

After fitting an ellipse around the foreground object, the orientation angle is defined as the angle made by the major axis to the *X*-axis of an image, and it has the same second moment in the given region. It can be calculated with a central moment of second order as given in Eq. (4).

$$\theta = \frac{1}{2}\arctan\left(\frac{2\mu_{11}}{\mu_{20}-\mu_{02}}\right) \tag{4}$$

where the moments μ_{20} and μ_{02} are the variance of *x* and *y*, respectively. The moment μ_{11} is the covariance between *x* and *y*.

- **Aspect ratio**

  The proportion of the major axis and its corresponding minor axis of the ellipse present on the object are called the aspect ratio as in Eq. (5).

  If a person falls, the major and minor axis are changed, and it becomes easy to separate out a fall and non-fall incident [25].

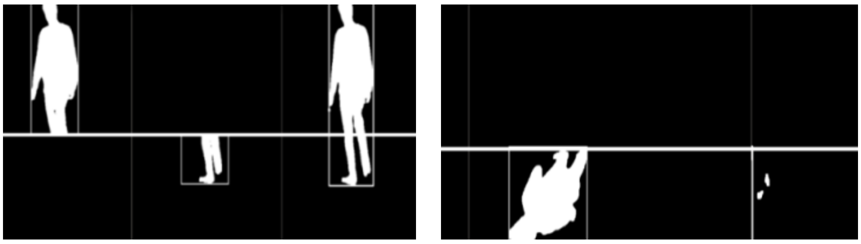
$$Aspect\ Ratio = \frac{mA}{ma} \tag{5}$$

where mA = major axis length and ma = minor axis length of an ellipse.

- **Silhouette detection below threshold line**

  When a fall occurs, the silhouette of the foreground object is recognized on the ground level. If a threshold line is set by, considering a particular height from the ground level, then falling and non-falling of the object can be based on the height of the silhouette. This feature enhances the accuracy of the system.

In Fig. 4, the bold horizontal line represents the threshold line. Figure 4(a) shows that 22.26% of the silhouette is present below the threshold. This means, that, the maximum part is above the threshold line, and so, it cannot be considered as a fall. In Fig. 4(b), 100% part of the silhouette is found below the threshold line indicating that fall activity has occurred.



(a) 22.26% silhouette below reference line.

(b) 100% silhouette below reference line.

Fig. 4. Percentage of silhouette below threshold line.

The method implemented is a combination of features such as aspect ratio and MHI. If the person is sitting or lying down, then there is no abrupt motion detected by MHI even if the aspect ratio changes. Hence, MHI supports the decision of falls or non-falls.

- **Dataset preparation**

  As discussed earlier, the features are MHI, orientation angle, aspect ratio and percentage of silhouette below reference line. These features are extracted from the frames of the videos in the training set. As the features have a correspondence with frames (such as feature row number i corresponds to frame number i of the current video), it can be label the feature rows as either fall or non-fall (say class 0 for fall and class 1 for non-fall) by looking at the frame. This process gives us a labelled data set. Different classifiers are then trained using this labelled dataset, as described in Section 3.5; the same process applies to the preparation of the test set as well.

3.5. Classification using SVM

Classification is the concluding step of the proposed system. According to the decision of a classifier, the activity is classified into fall and non-fall.

The selection of the classification technique largely depends on the characteristics of the problem at hand. Depending on the problem, one technique may be better than the others. So, various techniques applicable to the fall detection problem were evaluated. This system evaluates SVM, KNN, Stochastic Gradient

Descent (SGD), Decision Tree (DT), and Gradient Boosting (GB), on the basis of their performance (measured using f-score). As discussed in the results section, DT classifier turns out to be the best classifier for this problem; however, the best classifier also depends on the testing environment (e.g., GB is better in complex environments such as lecture room). Nevertheless, there is a need to establish the generalizing capability of DT through extensive cross-validation prior to proposing it as the only technique for use in practice, when the system is in operation. This is discussed in detail in the conclusions section. The formulations of each of the classification techniques are presented in the following sections.

3.5.1. SVM

SVM was first developed by Vapnik [26]. SVM is a supervised binary classifier, which is a superior classifier among the supervised category. The SVM classifies the features into two classes using the hyperplane in high-dimensional feature space. On the basis of the kernel, SVM classifies the features. SVM algorithm has steps of training, validation and performance evaluation. In the training phase, the feature sets are provided with respective labels to create a model. In the testing phase, the model evaluates the performance of test features.

The kernels are of different types: linear, Radial Basis Function (RBF) and polynomial. The kernel function decides the boundary of the hyperplane. The following formulation is proposed for building an SVM model for fall detection.

- **Linear kernel**

  In the proposed system, the focus is on two-class problems, i.e., fall and non-fall. The training data is represented by (x_i, y_i), i = 1, 2 …T, and the linear kernel for separable training data is given by Eq. (6)

$$f(x) = W^T X + b \tag{6}$$

where W is the unit vector, b is constant, and X is a function, which gives,

$$f(x_i \geq 0) \qquad \text{for } y_i = +1$$

$$f(x_i < 0) \qquad \text{for } y_i = -1$$

For each training vector, some hyperplanes can be drawn to separate the classes but the SVM classifier chooses the hyperplane having the maximum margin. The data vectors of class 1 and class 2, represented by "●" and "○" respectively, are separated by an optimum hyperplane as shown in Fig. 5.

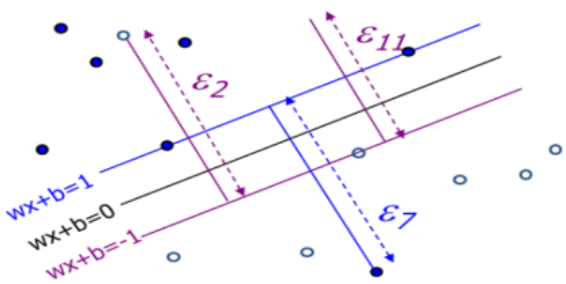


Fig. 5. Linear SVM.

- **RBF kernel**

  RBF is a non-linear kernel, which realizes the mapping of the nonlinear operator with an input feature set X . Eq. (7) gives the nonlinear kernel for separable training data.

$$f(x) = W^T \phi(x) + b \tag{7}$$

To satisfy this equation, the function $J(\omega, \xi)$ must satisfy the criterion subject Eqs. (8) and (9).

$$\min J(\omega, \xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l} \varepsilon_i \tag{8}$$

$$k(x_i, x_j) = \exp\left[\frac{-\|X_i - X_j\|^2}{2\sigma^2}\right] \tag{9}$$

where x_i and x_j are the samples that represent feature vectors in the same input space, the term $\|X_i - X_j\|^2$ that defines the $L2$-norm between x_i and x_j is the weight vector, b is an offset of the hyperplane, σ is a free parameter and ξ is a slack variable used when separating hyperplane does not exist.

As discussed in Section 3.4, the training dataset is first prepared. This data gives us the (x_i, y_i) pairs. The SVM classifier is trained using this dataset. Both linear kernel as well as RBF kernel is used in this classifier. The hyper-parameters of the SVM classification algorithm, such as C (the multiplier in Eq. (8) controls regularization) and ϵ (the margin), are tuned through multiple runs. Finally, the trained classifier model is run on the test set to obtain its precision and recall.

3.5.2. KNN

It is a supervised learning classifier [27]. It finds k-nearest samples from the training data and assigns the label of the highest votes of the nearest neighbor to the test instance. The distance of nearest neighbors is calculated by the Euclidean Distance(D), which is given by Eq. (10).

$$D = \sqrt{\sum_{i=1}^{v} (P_{1j} - P_{2j})^2} \tag{10}$$

where P_{1j} are the features of query data, and P_{2j} are the features of data from the training database.

As the feature set increases, the KNN performance decreases. Hence, there is a challenge of the selection of the best value of k.

The KNN is a lazy algorithm in that it defers all the training to the point of testing. In other words, there is no explicit training phase, except a number K-Nearest Neighbours to be used needs to be defined. The training data is stored by the algorithm for use in the testing phase. In the testing phase, for each test sample, the following steps are taken to classify it:

- Find K-number of the nearest neighbors in the training data close to the test sample.
- Find out the label of the most number of nearest neighbors.
- Assign the label of the most number of K-Nearest Neighbour to the testing sample.

3.5.3. Stochastic Gradient Descent (SGD)

SGD is a linear classifier having a convex type of loss function. It is a supervised, discriminative and efficient classifier. It offers large-scale learning, and hence, it attracts the researchers. It is mostly used for large-scale and sparse machine learning problems such as natural language processing, character recognition, etc. It has good efficiency and is easy to implement. It is useful to our approach because the extracted feature set of the proposed system is large [28]. SGD performs a parameter modernize for each training example $x^{(i)} and\ y^{(i)}$ as in Eq. (11)

$$\theta = \theta.\nabla_{\theta} J(\theta; x^{(i)}; y^{(i)}) \tag{11}$$

where η is the learning rate, $\nabla_{\theta}J(\theta)$ is an objective function and θ is the model parameter.

This is a straightforward application of the SGD, where one iteration of stochastic gradient descent consists of going through the entire dataset (x_i, y_i) and updating the weights as per Eq. (11) for each example and finally learns a set of weights θ that give us the classification boundary. In the testing phase, for each instance x, one calculates $\theta^T x;$ if it is negative, then it predicts fall (say, class -1), and if it is positive, it predicts non-fall (say, class +1).

3.5.4. Decision Tree

DT is a supervised learning algorithm used for classification as well as regression. A DT consists of a node and root sections. At the root section, no classification is possible, whereas the node sections are classified into classes called leaves. Leaves are represented as class labels while branches represent a conjunction with the feature that further leads to the respective classes.

The decision tree algorithm, when trained on the labelled dataset created as described in Section 3.4, produces a decision tree that has, at each node, a condition testing based on one of the four features (e.g., percentage of silhouette below reference line is greater than 50% and orientation angle is less than 45, etc.). If the condition evaluates true, it traverses to left, otherwise to the right subtree and so on until it reaches the leaf nodes. The leaf nodes represent classes. Therefore, in this application, there will be two root nodes (fall and non-fall). In the testing phase, just put each test instance through the tree starting from the root. The leaf node that we end up in is predicted as the class.

3.5.5. Gradient Boosting (GB)

GB is a combination of two methods, i.e., gradient descent method and Adaboost. It builds the model in a forward fashion and optimizes the differential loss function. The algorithm is highly customizable for the particular application. Adaboost has an advantage as it boosts the outliers near classification boundaries. GB helps to increase the accuracy of the classifier.

Although Gradient Boosting is an ensemble-based technique, meaning it is based on learning multiple models, the idea is to combine several weak models into a single, strong model. So depending upon the form of the week models used (e.g., most commonly decision trees), the final model is of the same type, i.e., in this example, a single decision tree comes out of the training process. Now using a decision tree for classification into fall/non-fall, it can proceed as discussed in Section 3.5.4. Similarly, if the week classifiers have any other form, then the evaluation method of that particular form applies.

4. Results and Discussion

The projected system is implemented using the OpenCV library with python. The system is intended for real-time environments using USB/Raspberry Pi camera. The processing time of the algorithm is more than 25 frames per second, which is sufficient to work in real-time applications. The flow diagram of the presented system is described in Fig. 6.

The algorithm has been tested in four different environments (coffee room, home, lecture room and office) of Le2i database. The results are presented in qualitative and quantitative ways.

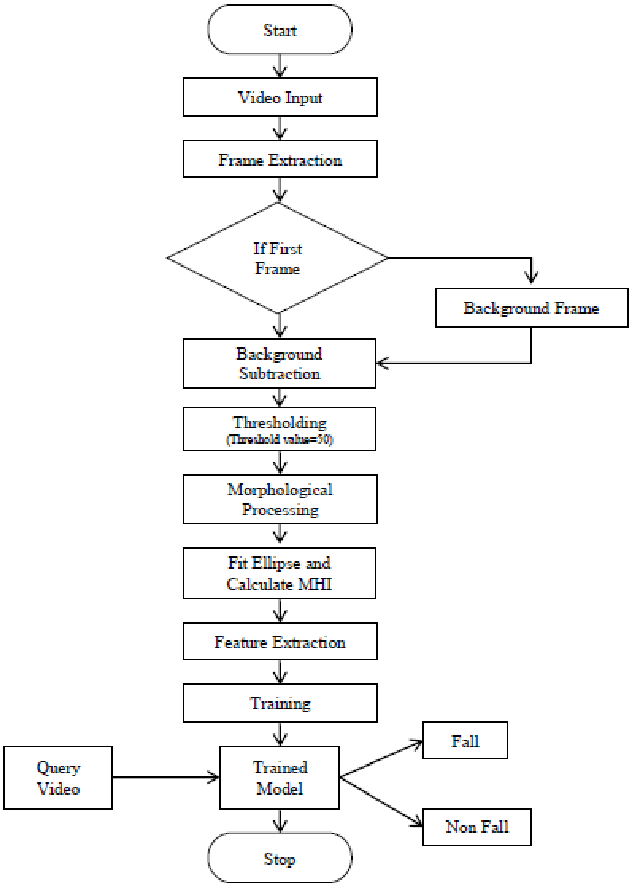
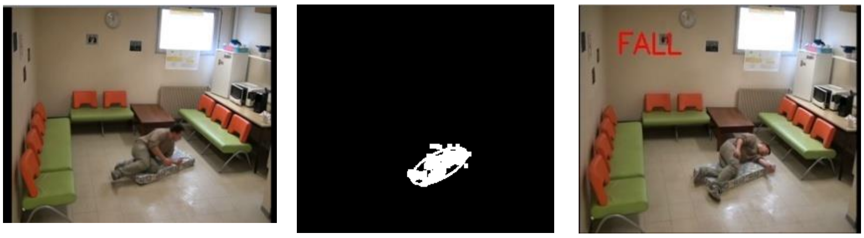


Fig. 6. Flowchart of the proposed system.

4.1. Qualitative analysis

Qualitative analysis is the non-statistical and pictorial representation of the research. The results of the proposed system for each stage are shown in Fig. 7.



(a) Input video frame. **(b) Bounding box.** **(c) Fall detection.**

Fig. 7. Qualitative analysis.

4.2. Quantitative analysis

The quantitative analysis of the proposed system is calculated with precision, recall, and *F*-measure. Eqs. (12-14) mathematically represent them respectively.

$$precision\ (\%) = \frac{TP}{TP + FP} \times 100 \tag{12}$$

$$recall\ (\%) = \frac{TP}{TP + FN} \times 100 \tag{13}$$

$$f - measure\ (\%) = \frac{precision \times recall}{precision + recall} \times 100 \tag{14}$$

where *TP* is the true positive value, which defines the occurred movement as a fall. *TN* is the true negative value, which defines non-fall detected as non-fall. *FP* is a false positive value, which defines occurred fall detected as non-fall. *FN* is a false negative value, which defines non-fall detected as fall.

This system is extensively tested on four environments of the Le2i dataset using five different classifiers (SVM, KNN, SGD, DT, and GB) for investigating precision, recall and *F*-measure. Results for the coffee room, home, lecture room and office environments of Le2i are shown in Table 2 and plotted in Fig. 8. Based on the *F*-measure value, it is observed that the linear kernel gives better result compared to RBF. For the KNN classifier, the selection of the value of *k* decides the performance of the classifier. It has been empirically observed that lower values of *k* result in the improvement of the *F*-measure. SGD classifier proves to be inefficient for all the experimentation of the proposed system. Exhaustive investigation results also effectively demonstrate that the GB and DT classifiers provide the best value of *F*-measure. In the case of GB, higher values of estimator results in favourable *F*-measure.

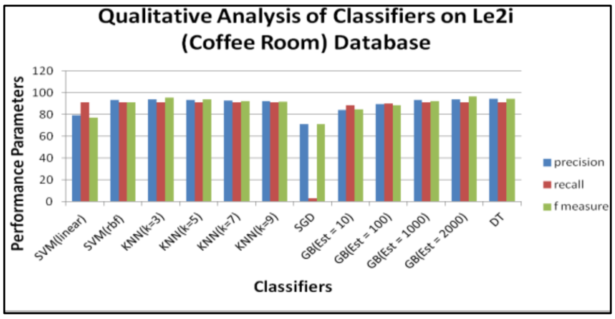
In video processing, a major challenge is to work in dynamic environments. In real life, only one classifier can be used, but in this system, for experimentation, various classifiers have been implemented to check for suitable classifiers based on the problem characteristics and datasets.

The results of the proposed system have been compared with the solo camera based fall detection using movement and human silhouette features method explained by Nguyen et al. [20]. The experimentation of this method is carried out on the Le2i database having different activities. This system uses a combination of motion features and body shape features. The performance comparison of this method and the proposed method is tabulated in Table 2. From the comparative analysis, it is observed that the proposed system gives improved results for all environments of Le2i database, and DT and GB algorithms provide satisfactory results. In comparison, the SGD classifier fails to achieve the expected performance for all the environment of Le2i fall dataset.

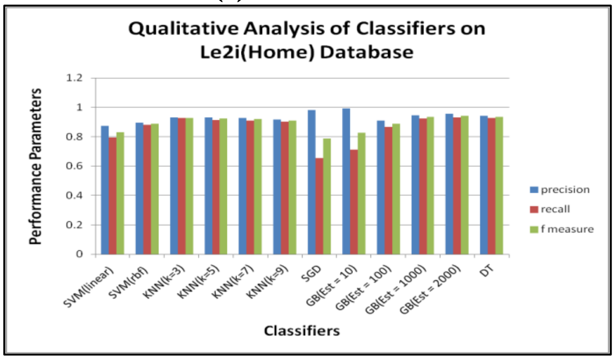
Execution time is one of the metrics, which defines the performance of the system. From Fig. 9, it is observed that RBF function takes more time to execute than the linear kernel. For the KNN classifier, the computational time goes on increasing as the value of the k increases. Likewise, as the estimator of GB classifier increases, the computational time also increases. DT algorithm consistently achieves the lower computational time for all environments of the Le2i dataset. The execution time of each classifier on the different environment of Le2i database is tabulated in plots as shown in Fig. 9.

Table 2. Performance comparison on the Le2i dataset.

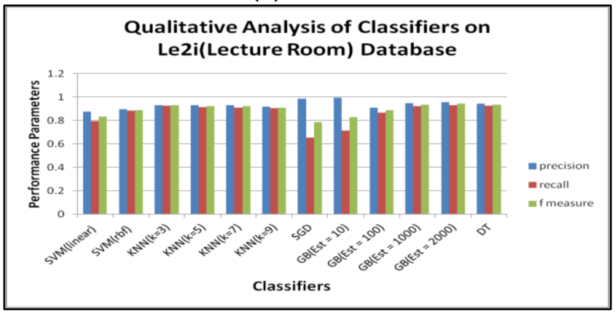
Environment	**Classifier**	**Parameter**	**Precision**	**Recall**	***F*-measure**
Coffee room	SVM	Linear	0.92	0.74	0.92
	KNN	*K*=3	0.95	0.73	0.94
	SGD	-	0.99	0.99	0.83
	DT	-	0.95	0.96	**0.96**
	GB	Estimator =2000	0.96	0.73	0.95
	SVM [20]		0.90	0.95	0.93
Home	SVM	Linear	0.89	0.88	0.88
	KNN	*K*=3	0.93	0.92	0.92
	SGD	-	0.98	0.65	0.78
	DT	-	0.94	0.92	0.93
	GB	Estimator =2000	0.95	0.93	**0.94**
	SVM [20]		0.94	0.97	0.93
Lecture Room	SVM	*Linear*	1	0.80	0.89
	KNN	*K=3*	0.94	0.94	0.94
	SGD	-	0.31	0.90	0.60
	DT	-	0.94	0.94	0.94
	GB	Estimator =2000	0.97	0.95	**0.96**
	SVM [20]		1	0.93	97
Office	SVM	Linear	0.90	0.92	0.91
	KNN	*K*=3	0.92	0.93	0.93
	SGD	-	0.67	0.92	0.78
	DT	-	0.94	0.95	**0.95**
	GB	Estimator =2000	0.95	0.94	0.94
	SVM [20]		0.94	0.97	0.95



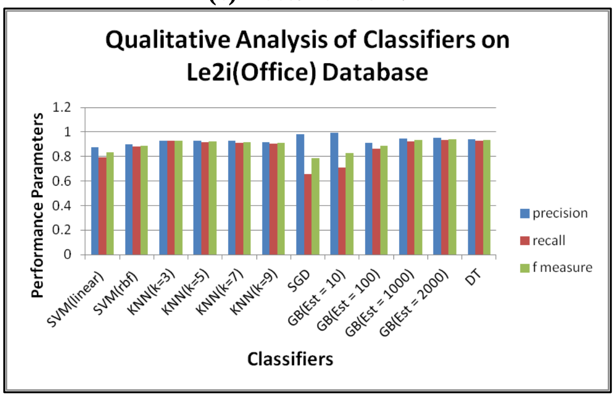
(a) Coffee room.



(b) Home.



(c) Lecture room.



(d) Office.

Fig. 8. Qualitative analysis of classifier on Le2i database with a different environment.

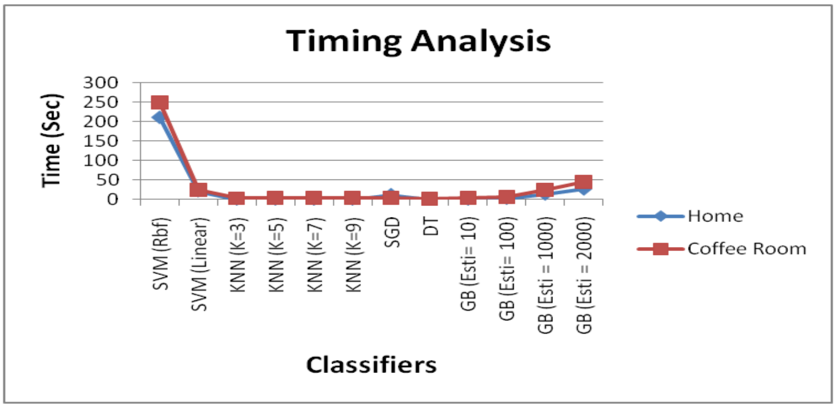


Fig. 9. Execution time analysis of different classifier for different environment of Le2i databases (home and coffee room).

5. Conclusion

In this paper, a system for human fall detection has been proposed. The proposed fall detection system extracts four visual features: MHI, aspect ratio, orientation angle and threshold below the reference line. The blend of motion gradient and change in silhouette gives crucial information about the fall.

The extracted features set are fed to the machine learning algorithms and experimented with five different classification techniques (i.e., SVM, KNN, SGD, GB, and DT) to find out the most suitable technique for the fall detection problem. From the experimental results, it is observed that the DT algorithm provides better *F*-measure value with minimum computational time. The performance of a classifier also depends on the testing environments. For complex environments such as lecture room, GB may be required to increase variance. In normal testing environments, however, this system proposes using the high-bias low-variance DT classifier as it is the most parsimonious and efficient model. This bias-variance trade-off has long been a dilemma for machine learning researchers and is an interesting research area in [29].

For now, this method chooses to base the decision on the test-set performance. The proposed algorithm uses 142 training videos and 54 testing videos of the Le2i dataset having different environments. The proposed system provides *F*-measure of 96% for DT classifier on the Coffee Room environment of Le2i database. The fall detection system proposed here is fast, robust, and computationally efficient. This work is significant as a robust algorithm with multiple features is used for classification, and work is tested on a huge dataset for generalizing the algorithm.

An important direction for future work is to establish the wide applicability and generalization capability of the DT classifier. Extensive parameter tuning using cross-validation may be required to achieve this. Preliminary results indicate that this is indeed possible. Moreover, the DT model is desirable as it is a simple and efficient model. Another direction for future work is accuracy improvement by considering more features for fall detection. The system may be trained and tested with the deep learning architectures for the betterment of results.

Nomenclatures

I_d	Absolute difference between the current and previous frame
I_t	Intensity of current frame
$I_{(t-1)}$	Intensity of the previous frame

Greek Symbols

Σ	Free parameter

Abbreviations

CSS	Curvature Scale Space
DT	Decision Tree
ELM	Extreme Learning Machine
GB	Gradient Boosting
GSM	Global System for Mobile Communication
HMM	Hidden Markov Model
HOG	Histogram of Oriented Gradients
KNN	K-Nearest Neighbors
MHI	Motion History Image
OCSVM	One-Class Support Vector Machine
RBF	Radial Basis Function
RDT	Randomized Decision Tree
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine

References

1. Chan, B.K.; Marshall, L.M.; Winters, K.M.; Faulkner, K.A.; Schwartz, A.V.; and Orwall, E.S. (2007). Incident fall risk and physical activity and physical performance among older men: The osteoporotic fractures in men study. *American Journal of Epidemiology*, 165(6), 696-703.
2. Mane, A.B.; Sanjana, T.; Patil, P.R.; and Sriniwas, T. (2014) . Prevalence and correlates of fear of falling among elderly population in urban area of Karnataka, India. *Journal of Mid-Life Health*, 5(3), 150-155.
3. Zhang, Z.; Conly, C.; and Athitsos, V. (2015). A survey on vision-based fall detection. *Proceedings of the 8th ACM International Conference on Pervasive Technologies Related to Assistive Environments.* New York, USA, Article 46.
4. Cheffena, M. (2016). Fall detection using smartphone audio features. *IEEE Journal of Biomedical and Health Informatics,* 20(4), 1073-1080
5. Ali, S.F.; Muaz, M.; and Fatima, A.; Idrees, F.; and Nazar, N. (2013). Human fall detection. *Proceedings of the IEEE International Multi-Topic Conference* (*INMIC*), Lahore, Pakistan, 101-105.
6. Chen, Y.-T.; Lin, Y.-C.; and Fang, W.-H. (2010). A hybrid human fall detection scheme. *Proceedings of the IEEE International Conference on Image Processing*. Hong Kong, China, 3485-3488.

7. Chen, Y.-T.; Lin, Y.-R.; and Fang W.-H.(2012). Detection of human fall in the video using shadow information. *Proceedings of the IEEE Biomedical Circuits and Systems Conference* (*BioCAS*), Hsinchu, *Taiwan*, 284-287.
8. Cheng, H.; Luo, H; and Zhao F. (2011). A fall detection algorithm based on pattern recognition and human posture analysis. *Proceedings of the IET International Conference on Communication Technology and Application* (*ICCTA 2011*), Beijing, China, 853-857.
9. Colon, L.N.V.; DeLaHoz, Y.; and Labrador, M. (2014). Human fall detection with smartphones. *Proceedings of the IEEE Latin-America Conference on Communications* (*LATINCOM*), Cartagena de Indias, Columbia, 1-7.
10. Delgado, B.; Tahboub K.; and Delp, E.J. (2014). Automatic detection of abnormal human events on train platforms. *Proceedings of the IEEE National Aerospace and Electronics Conference* (*NAECON* 2014), Dayton, Ohio, United States of America, 169-173.
11. Jiang, M.; Chen, Y.; Zhao, Y.; and Cai A. (2013). A real-time fall detection system based on HMM and RVM. *Proceedings of the IEEE Conference* on *Visual Communications and Image Processing* (*VCIP*), Kuching, Malaysia, 1-6.
12. Lin, B.-S.; Su J.-S.; Chen, H.; and Jan C.Y. (2013). A fall detection system based on human body silhouette. *Proceedings of the 9th International Conference on Intelligent Information Hiding and Multimedia Signal Processing*. Beijing, China, 49-52.
13. Qian, H.-m.; Zhou, J.; and Yuan, Y. (2015). Visual-based fall detection using a histogram of oriented gradients of Poisson distance image. *Proceedings of the Chinese Automation Congress* (*CAC*), Wuhan, China, 657-662.
14. Rougier, C.; Meunier, J.; St-Arnaud, A.; and Rousseau, J. (2011). Robust video surveillance for fall detection based on human shape deformation. *IEEE Transactions on Circuits and Systems for Video Technology,* 21(5), 611-622.
15. Bian, Z.-P.; Hou, J.; and Chau, L.-P.; and Magnenat-Thalmann, N. (2015). Fall detection based on body part tracking using a depth camera. *IEEE Journal of Biomedical and Health Informatics*, 19(2), 430-439.
16. Lin, C.-Y.; Wang, S.-M.; Hong J.-W.; Kang, L.-W.; and Huang, C.-L. (2016). Vision-based fall detection through shape features. *Proceedings of the IEEE 2nd International Conference on Multimedia Big Data* (*BigMM*), Taipei, Taiwan, 237-240.
17. Ma, X.; Wang, H.; Xue, B.; Zhou, M.; Ji, B.; and Li, Y. (2014). Depth-based human fall detection via shape features and improved extreme learning machine. *IEEE Journal of Biomedical and Health Informatics*, 18(6), 1915-1922.
18. Yu, M; Yu, Y.; Rhuma, A.; Naqvi, S.M.R.; Wang, L.; and Chambers, J.A. (2013). An online one class support vector machine-based person-specific fall detection system for monitoring an elderly individual in a room environment. *IEEE Journal of Biomedical and Health Informatics*, 17(6), 1002-1014.
19. Madhubala, J.S.; and Umamakeswari, A. (2015). A vision based fall detection system for elderly people. *Indian Journal of Science and Technology*, 8(S9), 167-175.

20. Nguyen, V.A.; Le, T.H.; and Nyuyen T.T. (2016). Single camera based fall detection using motion and human shape features. *Proceedings of the 7th Symposium on Information and Communication Technology* (*SoICT* '16), New York, United States of America, 339-344.
21. Charfi, I.; Miteran, J.; Dubois, J.; Atri, M.; and Tourki, R. (2013). Optimised spatio-temporal descriptors for real-time fall detection: comparison of SVM and Adaboost based classification. *Journal of Electronic Imaging*, 22(4), 17 pages.
22. Otsu, N.A. (1979). A threshold selection method from gray-level histogram. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(1), 62-66.
23. Bobick, A.F.; and Davis, J.W. (2001). The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3), 257-267.
24. Yu M.; Rhuma, A.; Naqvi, S.M.; Wang L.; and Chambers J. (2012). A posture recognition-based fall detection system for monitoring an elderly person in a smart home environment. *IEEE Transactions on Information Technology in Biomedicine*, 16(6), 1274-1286.
25. Vaidehi, V.; Ganapathy, K.; Mohan, K.; Aldrin, A.; and Nirmal, K. (2011). Video based automatic fall detection in indoor environment. *International Conference on Recent Trends in Information Technology* (*ICRTIT*), Chennai, Tamil Nadu, 1016-1020.
26. Vapnik, V.N. (1995). *The nature of statistical learning theory.* New York: Springer Science + Business Media.
27. Duda, R.O.; Hart, P.E.; and Stork, D.G. (2000). *Pattern classification* (*2nd ed.*). New York: John Wiley & Sons Incorporation.
28. Tsuruoka, Y.; Tsujii J.; and Ananiadou, S. (2009). Stochastic Gradient Descent training for L1-regularized log-linear models with a cumulative penalty. *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, Suntec, Singapore, 477-485.
29. Geman, S.; Bienenstock, E.; and Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1), 1-58.