FALL DETECTION USING LOW-RESOLUTION THERMAL SENSOR

Yu Xiao, Xun Lin

Boston University
Department of Electrical and Computer Engineering
8 Saint Mary's Street
Boston, MA 02215
`www.bu.edu/ece`

Apr. 10, 2019

Technical Report No. ECE-2019-03

Summary

This research was performed as a term project within EC520 course entitled ”Digital Image Processing and Communication”. We developed a machine learning algorithm to detect the fall of a person using a low-resolution thermal sensor. The motivation for the use of such sensor was preservation of privacy so that the system could be used in bathrooms, changing rooms, etc. The algorithm we developed is based on a pipeline composed of the following steps: pre-processing, feature extraction and classification, widely used in computer vision. We collected our own dataset using a thermal sensor with resolution of 24×32 pixels. We include quantitative results for two sensor-mounting scenarios: overhead and sideways.

Contents

List of Figures

List of Tables

1 Introduction

Falling is a main cause for injury at home, especially for elder citizens. In this project, an intelligent surveillance system is developed for fall detection using low-resolution thermal images. The motivation of using low-resolution sensor is to preserve privacy of the users, since it doesn't capture much detail. On the other hand, typical RGB cameras capture too much details, so it's not proper to use them in some household situations. Our work in this project can be divided into two parts: 1) Acquire images of different human postures with the thermal sensor; 2) Design and implement an algorithm to process and detect human falling from thermal images.

To formulate the problem in a trackable way, we made several key assumptions to our task. We assumed that 1) One person appears in every image. ; 2) The person in the image takes one of the following three postures: a) upright, b) squat and c) fall; 3) There is a difference between the temperature of the human body and the temperature of the environment. These assumptions simplify the complicated task of detecting falling action into classifying different kinds of human body poses, which is the critical part in determining whether a person is fallen or not. Further work may involve relaxing these assumptions. Based on these assumptions, we developed an algorithm based on shape information of human body. The algorithm takes multiple stages of processing, including pre-processing, feature extraction and feature classification. This report is structured in the following way: in section two we briefly review the literature of fall detection; in section three we formulate the problem addressing our current assumptions; detailed information about the proposed algorithm is discussed in section four; section five is about the result of our experiments; in the last section, several conclusions drawn from experiments and directions for further investigation is presented.

2 Literature Review

In order to perform fall detection, [5] surveys different methods in the literature using different sensor settings, such as RGB camera, thermal sensor and inertial sensor on wearable devices. When camera is the only sensors, many ideas have been used to develop recognition algorithms, such as spatial-temporal feature, inactivity/change of shape, human postures and etc. Recently, several fall detection methods using thermal sensors are developed aiming at preserving privacy of people. In [3], researchers captured falling using thermal camera on side and developed a heuristic algorithm to perform recognition.

Fall detection is one kind of the more general human activity recognition problem. Many approaches have been developed to recognize human activity from videos. [4] used three 8-by-8 Panasonic thermal sensors running at 10 frames per second to acquire a dataset, which consists of different number of people performing several kinds of actions(standing, sitting and walking). Several classification algorithms were applied directly on the dataset. Quantitative results with respect to different kinds of combinations of sensor data(single sensor, two sensors and three sensor altogether) and different kinds of classification algorithms(support vector classifier, random forest and k-nearest-neighbor) were presented. As a continuation of [4], [6] developed a recognition pipeline works on image sequences, which consists of background subtraction, feature extraction and classification. In this work, background is modeled by taking the average of a long sequence of raw data. After background subtraction, a DCT based feature vector is calculated from image sequence utilizing spatial and temporal information. The feature vector is then fed into a support vector classifier. We would like to explore further in this framework by designing filters, features and classifier for data acquired by our 24-by-32 thermal sensor.

3 Problem Formulation

The problem we considered in this project is detecting the falling action of human body. Specifically, in this stage of the project, we considered the problem of distinguishing the posture of the person using image data from a thermal sensor. To simplify the fall detection problem, We make the following three assumptions:

1. One person appears in every image.
2. The person in the image takes one of the following three postures:
   (a) upright: standing still or walking around.
   (b) squat: performing a squat.
   (c) fall: falling on the floor.
3. There is a difference between the temperature of the human body and the temperature of the environment.

The first assumption we made is to avoid detecting whether there is any person appeared in the image, since we want to focus on detecting the actual pose of the human body. However, detecting whether there is human in the image is important for real world deployment, which will be considered later. The second assumption allow us to narrow down all possibilities of human postures to three distinctive kinds. The third assumption we made is critical for applying the thermal sensor. The thermal sensor will work only if there is difference between temperature of human body and the environment.

Assuming all the conditions above, we want to develop an algorithm which can classify human poses using thermal images. The input to the algorithm is a single frame of thermal image with 24*32 resolution. The program will output a number representing one of the following three types of postures: upright, squat and fall.

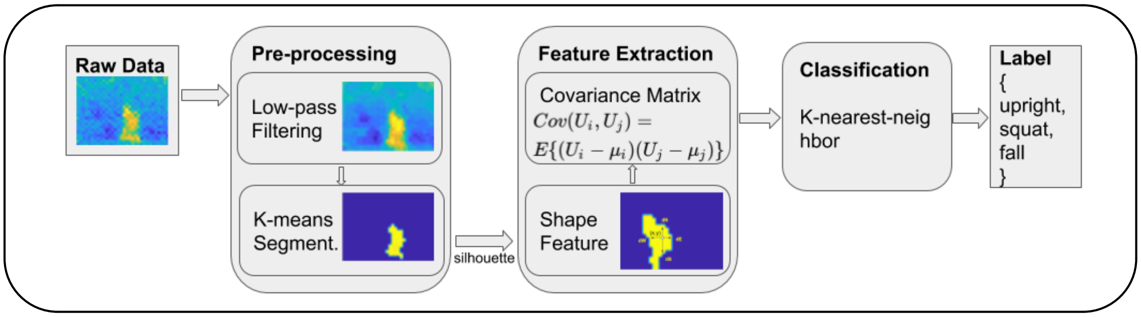


Figure 1: System diagram of proposed method.

4 Proposed Method and Implementation

Our methods are based on the framework proposed by [6], which consists of multiple cascaded modules. The algorithm classifies thermal images based on shape information of human body in the scene. The intuition we take is that the shape of human bodies should be distinguishable when people take different poses. For example, from the side view, a person standing still should have roughly a rectangular shape with vertical long edges, while a person fallen on the floor may also have a rectangular shape but with horizontal long edges. Due to our low resolution sensor, the main challenge for us is that body shapes of different poses can be sometime similar. We address this challenge by improving on our feature vector and classifier.

The block diagram of our method is shown in Fig.1. We decomposed our task of classifying human poses from thermal images into three parts: 1) image preprocessing, 2) feature extraction and 3) feature classification. Then, our proposed method has one stage for each of these parts, which are represented by oval boxes in the middle part of the system diagram.

The pre-processing stage deals with the fact that our raw data is noisy in most cases. The noise in the raw data is largely due to the natural scanning of the sensor we use but not the environment, since temperature won't change rapidly in most situations. Before trying to extract feature from the image, we want an intermediate representation which contains most information about the shape of human body with as little as possible noise. The pre-processing stage consists of several cascaded modules including low-pass filter and k-means segmentation. The cascaded pre-processing modules output a binary image of the same size as input, which contains a silhouette of the human body. The silhouette image contains most part of the shape information about the human body.

Although the pre-processing stage yields a rather clean image with much information, the image cannot be classified directly because it is high-dimensional.The second stage constructs lower dimensional feature from silhouette image, which is suitable for classifiers. In this stage, a hand-crafted feature vector is extracted from the silhouette image for each pixel, which results in a 24*32*6 dimensional tensor.

To reduce the dimensionality of the feature, we compute the covariance matrix over the image, which has 6*6 entries. The covariance matrix computed from silhouette images is then used for classification in the next stage.

In the final classification stage, the covariance matrix is classified using K-nearest-neighbor model.

In the rest part of this section, detailed information for each of the stages in provided.

4.1 Image Pre-processing

4.1.1 Noise Removal

The noise in the raw data is mostly in the high frequency components of the signal. One way to deal with it is applying the raw data with a low pass filter. The first filter we designed is a low pass DCT filter. DCT has the property of power concentration. The most power of the thermal images are concentrated in the low frequency components of its DCT. Our DCT filter concentrates all spectral-coefficients on the first half of the spectrum and eliminates all the other coefficients. But the result Fig.5.2.1 has too much blur on the body, which indicates huge information loss caused by the DCT filter. The second filter we tried is Gaussian low pass filter. Although Gaussian filter blurs the image to reduce the noise, the result has much more details on the body than our DCT filter.

Another filter we tried is the median filter. The median filter sets the value of the output pixel to the median of the pixel values in the neighborhood around the corresponding input pixel. Instead of using the mean value of the neighborhood, the median is much less sensitive than the mean to extreme values. The result shows that median filter removes noise while keeping edges relatively sharp. However, as shown in Fig.4, the median filter loses some information(the gap between the legs and the hip) but is better for background noise reduction (more blur in the background) compared to the Gaussian filter.

4.1.2 K-means Clustering Based Segmentation

To capture shape information about human body in the image, we want to obtain a binary silhouette image, where all pixels corresponding to the body have value 1 and other have value 0. In order to obtain the contour of the human body, we need to apply image segmentation. Here we use K-means clustering to segment human body, which is fairly robust and cluster the pixels into K categories. In our case, the K-means Clustering problem is defined as flowing:

Input: $x_1, x_2 ... x_i ... x_m \in \Re$, x_i is the temperature of one pixel, target cardinality k=3

Output: 3 centers c_1, c_2, c_3, labeled pixel sets L

Objective: $\sum_{i=1, j \in (1,2,3)}^{m} min \left\| x_i - c_j \right\|$

The reason we choose $k = 3$ is because the temperature of the wall and window is slightly higher than the ground and ceiling. Our solution is choosing the cluster with the highest mean temperature between the 3 clusters created by K-means as the result.

4.2 Feature Extraction

Inspired by the approach of [2], we constructed feature vectors that describes the shape of the silhouette. The feature extraction step assigns a vector at each pixel location that contains local information about the silhouette. The feature vector shown in Figure.2 at each point is defined as

$$U[x, y] = [x, y, d_E, d_W, d_N, d_S],$$

where $[x,\ y]$ is the location of the pixel in image coordinate. The value d_E is the distance from the pixel to the boundary of the silhouette on the right side. Value d_W d_N and d_S are defined similarly but for other three directions: left, up and down.

Then, we compute the covariance matrix of the 3-D tensor to reduce its dimensionality. The computation of covariance matrix of feature vector at each pixel location use the following formulas:

$$\mu_i = 1/|M| * \textstyle\sum_{(x,y)\in M} U_i[x, y], i = 1, ..., 6,$$
$$Cov(U_i, U_j) = 1/|M| * \textstyle\sum_{(x,y)\in M} (U_i[x, y] - \mu_i)(U_j[x, y] - \mu_j), i, j = 1, ..., 6,$$

where M denotes the set of pixels lie inside the silhouette. after extracting the feature, the resulting covariance matrix is then feed into a K-nearest-neighbor classifier.

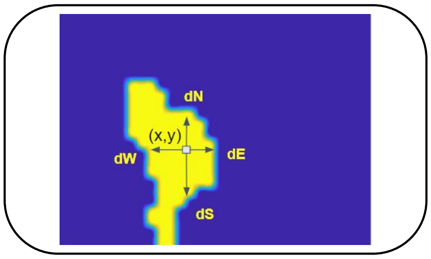


Figure 2: Silhouette feature

4.3 Feature Classification

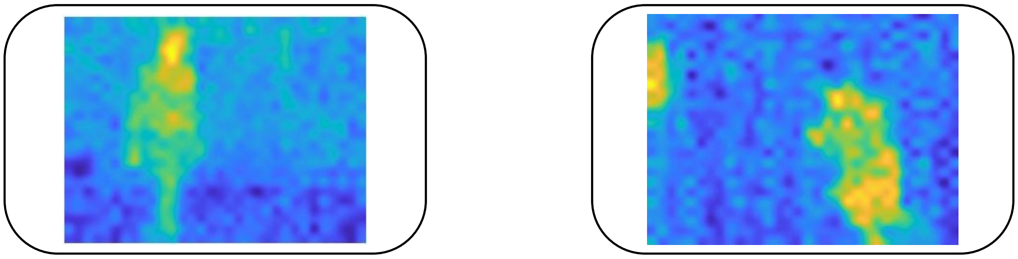
The classifier is the final module in our pipeline. The covariance matrix of silhouette image is fed into the classifier to generate prediction. Here, we used a K-Nearest-Neighbor classifier to complete this task. For a given data point, the classifier works by conducting a majority vote among k neighbors around it with respect to some distance metric, where k is a parameter picked manually.

The distance metric for covariance matrices is the critical part for the KNN classifier. The usual Euclidean distance doesn't work well with covariance matrices due to the fact that the set of real symmetric positive definite matrices forms a non-Euclidean manifold. Distance metric based on eigenvalues([1] and [2]) are used to compare covariance matrices from different frames. [2] proposed a matrix logarithm metric based on eigenvalue decomposition. [1] proposed a metric using the generalized eigenvalues. We tried both metrics and decided to use the generalized-eigenvalue metric to classify covariance matrices in our implementation. Euclidean distance was tested in the experiments and served as the baseline.

5 Experimental Results

In this section, qualitative and quantitative results from our experiments are present. First, we describe our data acquisition procedure and some detailed information about the dataset. Then, we present some qualitative results of our image preprocessing algorithms to demonstrate their capability. Finally, quantitative results from classification stage are provided to justify our conclusions.

5.1 Data Acquisition



(a) side view: a person standing

(b) top view: a person fell on the floor

Figure 3: Example visualizations of raw data from different camera angles.

In our experiments, we acquire our own data using the Melexis MLX90640 thermal sensor at an apartment's living room, which is a typical in-door environment. The sensor captures temperature value of objects in the scene, which is returned in degree Celcius. To give a feeling for the raw data, several thermal images that came from our sensor are shown in Fig. 3. These color images are visualized using raw data with *colormap* command in MATLAB.

Our dataset consists of three different types of human postures using two kinds of camera angles. We found that camera angle plays an important role in the performance of our classification algorithm. The first kind of camera angle is side view. We set the thermal sensor on a dining table and point it to the open space of a living room. The second kind of view is top view. We acquired data from this camera angle by attaching the sensor to the ceiling of the room, where the camera pointed down to the floor.

Fig.3 shows two example frames in our dataset. One of them came from label 'upright' of side view data, while the other came from label 'fall' of top view data.

Table.1 shows the number of frames in our dataset with respect to different human poses and camera angles.

Pose Name	Upright	Squat	Fall
# Frames (side view)	340	300	430
# Frames (top view)	230	280	160

Table 1: Dataset information

5.2 Image Pre-processing

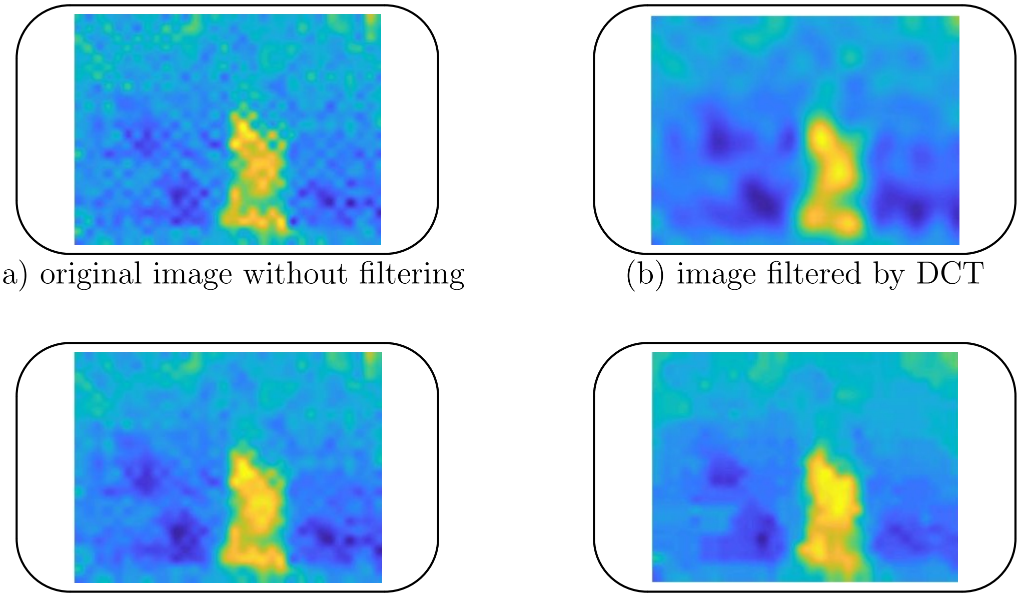
In this section, we provided some illustrative results from the image preprocessing stage of our algorithm. We observed that the quality of the preprocessing steps is critical to obtain high classification accuracy in later steps. While designing the algorithm, we measured the performance of different pre-processing modules by eyeballing.

5.2.1 Noise Removal

In this section, we compare the performance of three filtering methods. Figure.4 shows the original image under "falling" label and the original image filtered by three different filters respectively. The original image (Figure.4(a)) is extracted from one clip of falling. The DCT filter passes only spectral-coefficients from the top-left quarter of the spectrum (Figure.4(b)). The Gaussian filter has the standard deviation of 0.5 (Figure.4(c)). The median filter sets the output pixel with the median value in the 3-by-3 neighborhood and pads the image by extending the image at the boundaries symmetrically (Figure.4(d)).

5.2.2 Image Segmentation

Image Segmentation is the critical step to get silhouette images. The performance of k-means based segmentation algorithm is related to the filtering technique applied before. In this section, we show the k-means segmentation result of an image with different filters. Figure.5 shows the segmentation result of the human body generated by applying K-means clustering to Figure.4. Figure.6 shows the segmentation result of applying K-means to sitting posture with different K=2 and K=3. We use *imsegkmeans*() function in MATLAB and choose the highest mean temperature cluster to generate silhouette image.



(a) original image without filtering

(b) image filtered by DCT

(c) image filtered by Gaussian

(d) image filtered by median filter

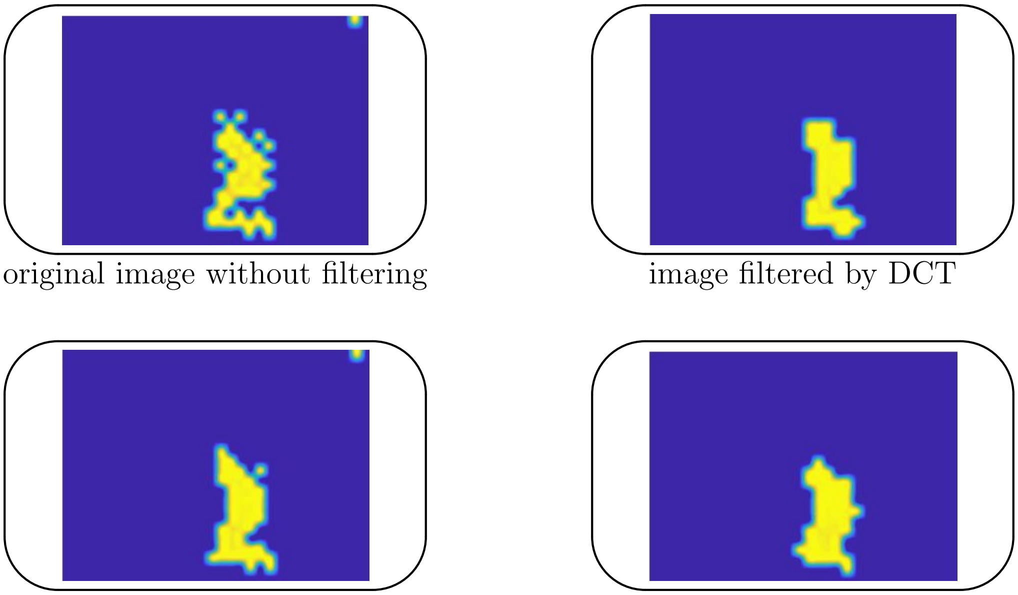
Figure 4: Comparison between three filters

5.3 Classification Result

In this section, a set of quantitative results generated by the K-nearest-neighbor classifier is presented. In order to study the effect of each module on the classifier's accuracy, we tried to feed raw data, silhouette images and covariance matrices into the classifier and measured the confusion matrices. Raw data refers to 32-by-24 temperature data directly acquired using the thermal sensor, on which no processing algorithm is applied. Silhouette images are calculated based on raw data, using filtering and segmentation techniques, which is output of the pre-processing. module we described. Silhouette images are fed into feature extraction module to calculate covariance matrices. Thus, covariance matrices are the final output of cascading pre-processing and feature extraction modules.

Experimental results are presented in the form of confusion matrices. The confusion matrices presented in this section were calculated from 20 independent trials of testing. During each trail, the dataset was randomly splitted into 80% training data and 20% testing data. The numbers in each entry are mean and standard deviation of classification rate from 20 trials.

The rows in the tables are labels (each row in the table add up to one) and the columns are predictions. For example, we can see that the (1,2) entry in table 2 shows the rate of predicting image of label "Upright" as "Squat", which is 15.21%. Also, entry (2,1) indicates rate of classifying label "Squat" as "Upright" is 2.11%. These results lead to our conclusions presented in the next section.



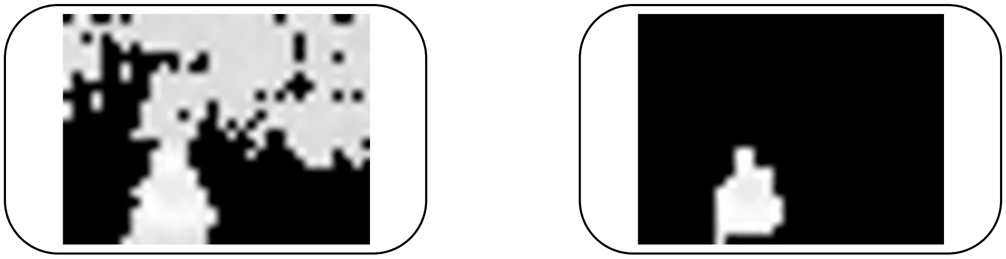
original image without filtering

image filtered by DCT

image filtered by Gaussian

image filtered by median filter

Figure 5: K-means clustering



k=2

k=3

Figure 6: K means with different K

5.3.1 Classification with Raw Data

Since the size of thermal images is rather small, it can be directly fed into KNN. The resulting confusion matrices for top view(Figure 2) and side view(3) data is presented in this section.

5.3.2 Classification with Silhouette Images

The silhouettes are binary images, which are calculated using the pre-processing module. Silhouettes contain shape information of human body captured by the thermal sensor. To study the effect of pre-processing module on our classifier, silhouettes were fed into KNN and experimental results are presented in Table 4 for top view data and Table 5 for side view data.

label vs. prediction	Upright	Squat	Fall
Upright	82.02%(6.12%)	15.21%(5.45%)	2.77%(2.59%)
Squat	2.11%(2.52%)	97.81%(2.54%)	0.09%(0.39%)
Fall	1.67%(2.30%)	2.58%(2.83%)	95.76%(4.65%)

Table 2: Confusion Matrix of Raw Data Classification (top view)

label vs. prediction	Upright	Squat	Fall
Upright	98.84%(2.08%)	1.16%(2.08%)	0.00%(0.00%)
Squat	0.57%(1.10%)	99.02%(1.12%)	0.41%(0.73%)
Fall	0.17%(0.56%)	0.00%(0.00%)	99.83%(0.56%)

Table 3: Confusion Matrix of Raw Data Classification (side view)

5.3.3 Classification with Covariance Matrix of Silhouettes

After the image pre-processing and feature extraction modules, the resulting covariance matrix were fed into the classifier. Experimental results from top view and side view data are presented in Table 6 and Table 7.

5.3.4 2-class Classification: Upright/Fall

We finished our experiments by letting the classifier discriminating only 2 classes of postures: stand and fall. We tested the classifier with respect to different choice of view angle, distance metric and parameter K for the classifier. Table.**??** shows the prediction accuracy on the test set between only 2 labels: stand and fall.

6 Conclusions and Further Work

Interesting conclusions can be drawn from the experimental results. From the quantitative results of the classifier, we found that top view data is harder for our algorithm to classify than side view data. This result is probably due to larger amount of noise is presented in top view data. Also, the shape of human body for many postures are likely to be similar in the top view, for example standing and squatting. Also, from the results of image segmentation step, we found that the background temperature of side view data might be complicated, probably due to common temperature difference between floor and walls. By comparing the different features we used to feed the classifier, we found that feeding raw data actually gives the best result in terms of correct classification rate. Feeding silhouette images gives accuracy worse than raw data, as well as feeding covariance matrices of silhouettes.

There are still much thing to explore in the next stage of the project. One major direction to dig deeper is to collect more samples with better variety. Since we only have around 1500 sample images in the dataset, the dataset is likely to be highly biased. This fact may lead to some incorrect evaluation results of our algorithms.

label vs. prediction	Upright	Squat	Fall
Upright	74.68%(6.76%)	24.79%(7.18%)	0.53%(0.95%)
Squat	2.37%(2.07%)	97.37%(2.17%)	0.26%(0.64%)
Fall	2.73%(2.76%)	1.82%(1.81%)	95.45%(2.31%)

Table 4: Confusion Matrix of Silhouettes Classification (top view)

label vs. prediction	Upright	Squat	Fall
Upright	98.70%(1.32%)	0.65%(0.88%)	0.65%(0.99%)
Squat	0.49%(1.08%)	97.38%(2.34%)	2.13%(1.93%)
Fall	2.76%(1.37%)	3.16%(1.90%)	94.08%(2.09%)

Table 5: Confusion Matrix of Silhouettes Classification (side view)

Also, finding ways to relax our current assumptions may also be interesting. For example, using an extra module to detect the presence of human may be crucial for real world deployment of the system. Also, using multiple frames to capture dynamics of the falling process is interesting.

References

[1] W. Fosterner and B. Moonen, "A metric for covariance matrices," 1999.

[2] K. Guo, P. Ishwar, and J. Konrad, "Action recognition in video by sparse representation on covariance manifolds of silhouette tunnels," *International Conference on Pattern Recognition*, 2010.

[3] A. Hayashida, V. Moshnyaga, and K. Hashimoto, "The use of thermal ir array sensor for indoor fall detection," *IEEE International Conference on Systems, Man and Cybernetics*, 2017.

[4] Y. Karayaneva, S. Baker, B. Tan, and Y. Jing, "Use of low-resolution infrared pixel array for passive human motion movement and recognition," *Proceedings of British HCI 2018*, 2018.

[5] M. Mubashir, L. Shao, and L. Seed, "A survey on fall detection: Principles and approaches," *Neurocomputing*, 2013.

[6] L. Tao, T. Volonakis, B. Tan, Y. Jing, K. Chetty, and M. Smith, "Home activity monitoring using low resolution infrared sensor array," *arXiv18*, 2018.

label vs. prediction	Upright	Squat	Fall
Upright	62.55%(7.02%)	31.70%(7.75%)	5.74%(4.03%)
Squat	7.02%(3.86%)	92.02%(4.00%)	0.96%(1.06%)
Fall	1.52%(1.84%)	1.21%(2.28%)	97.27%(2.93%)

Table 6: Confusion Matrix of Covariance Matrices Classification (top view)

label vs. prediction	Upright	Squat	Fall
Upright	96.30%(1.52%)	2.25%(1.52%)	1.45%(1.49%)
Squat	3.85%(2.39%)	92.13%(4.00%)	4.02%(2.94%)
Fall	2.47%(1.98%)	4.66%(1.95%)	92.87%(3.00%)

Table 7: Confusion Matrix of Covariance Matrices Classification (side view)

Feature/Angle	Upright	Fall
raw/side	100.0%(0.00%)	99.83%(0.42%)
raw/top	93.19%(4.76%)	97.27%(2.18%)
silhouette/side	100.0%(0.00%)	99.89%(0.35%)
silhouette/top	96.49%(3.61%)	97.27%(2.93%)
covariance/side	99.71%(0.76%)	97.70%(1.29%)
covariance/top	89.36%(4.63%)	90.61%(6.44%)

Table 8: Correct Classification Rates for Upright/Fall