Syracuse University

SURFACE

Dissertations - ALL

SURFACE

June 2017

Computer Vision Algorithms for Mobile Camera Applications

KORAY OZCAN

Syracuse University

Follow this and additional works at: https://surface.syr.edu/etd

Part of the Engineering Commons

Recommended Citation

OZCAN, KORAY, "Computer Vision Algorithms for Mobile Camera Applications" (2017). *Dissertations - ALL*. 704.
https://surface.syr.edu/etd/704

This Dissertation is brought to you for free and open access by the SURFACE at SURFACE. It has been accepted for inclusion in Dissertations - ALL by an authorized administrator of SURFACE. For more information, please contact surface@syr.edu.

ABSTRACT

WEARABLE and mobile sensors have found widespread use in recent years due to their ever-decreasing cost, ease of deployment and use, and ability to provide continuous monitoring as opposed to sensors installed at fixed locations. Since many smart phones are now equipped with a variety of sensors, including accelerometer, gyroscope, magnetometer, microphone and camera, it has become more feasible to develop algorithms for activity monitoring, guidance and navigation of unmanned vehicles, autonomous driving and driver assistance, by using data from one or more of these sensors. In this thesis, we focus on multiple mobile camera applications, and present lightweight algorithms suitable for embedded mobile platforms. The mobile camera scenarios presented in the thesis are: (i) activity detection and step counting from wearable cameras, (ii) door detection for indoor navigation of unmanned vehicles, and (iii) traffic sign detection from vehicle-mounted cameras.

First, we present a fall detection and activity classification system developed for embedded smart camera platform CITRIC. In our system, the camera platform is worn by the subject, as opposed to static sensors installed at fixed locations in certain rooms, and, therefore, monitoring is not limited to confined areas, and extends to wherever the subject may travel including indoors and outdoors. Next, we present a real-time smart phone-based fall detection system, wherein we implement camera and accelerometer based fall-detection on Samsung Galaxy S™ 4. We fuse these two sensor modalities to have a more robust fall detection system. Then, we introduce a fall detection algorithm with autonomous thresholding using relative-entropy within the class of Ali-Silvey distance measures. As another wearable camera application, we present a footstep counting algorithm using a smart phone camera. This algorithm provides more accurate step-count compared to using only accelerometer data in smart phones and smart watches at various body locations.

As a second mobile camera scenario, we study autonomous indoor navigation of unmanned vehicles. A novel approach is proposed to autonomously detect and verify doorway openings by using the Google Project Tango™ platform.

The third mobile camera scenario involves vehicle-mounted cameras. More specifically, we focus on traffic sign detection from lower-resolution and noisy videos captured from vehicle-mounted cameras. We present a new method for accurate traffic sign detection, incorporating Aggregate Channel Features and Chain Code Histograms, with the goal of providing much faster training and testing, and comparable or better performance, with respect to deep neural network approaches, without requiring specialized processors. Proposed computer vision algorithms provide promising results for various useful applications despite the limited energy and processing capabilities of mobile devices.

COMPUTER VISION ALGORITHMS FOR MOBILE CAMERA APPLICATIONS

By

Koray Özcan
B.S., Bilkent University, Ankara, Turkey, 2011

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Electrical and Computer Engineering

Syracuse University
May 2017

Copyright © 2017 Koray Özcan

All rights reserved

For all the loving support, motivation, and reasoning:

this thesis is dedicated to my wonderful family Sultan Özcan,
Zeki Özcan, and Gözde Özcan.

Acknowledgements

To begin with, I would like to thank my dear advisor DR. SENEM VELIPAŞALAR for mentoring, guiding my research endeavors and generously supporting my education at Syracuse University. We have been through many challenging projects, publications, academic conferences, and pressuring deadlines. I would not be able to find such immense motivation and encouragement without her continuous support for my degree, progress, and education. I am also thankful to DR. PRAMOD K. VARSHNEY for guiding my research and providing beneficial recommendations for my research ideas and projects. Moreover, I am delighted to have DR. CLIFF I. DAVIDSON serve as the oral examination chair for my defense and DRS. QINRU QIU, M. CENK GÜRSOY, AND JIAN TANG for being part of my defense committee. Their input helped this thesis to become its' current from.

I am thankful for all the support and help I received from our research group members MARIA CORNACCHIA, BURAK KAKILLIOĞLU, YANTAO LU, ANVITH K. MAHABALAGIRI, CHUANG YE, AND YU ZHENG. We collaborated for many projects, publications, and experiments for our group's research purposes. I received influential support from every one of them while performing experiments and progressing with our research projects. Special thanks are extended my notable friends HAO HE, DANUSHKA BANDARA, RAGHED A. EL BARDAN, ABHISHEK V. BHASKAR, ALI FATIH DEMIR, IAN M. JOYCE, SWATANTRA KAFLE, HABIB KAYA, YI LI, AND MUSTAFA ÖZMEN for their participation in challenging and demanding experiments for various activity detection and classification purposes. Last but not least, I owe a debt of gratitude to my friends AHMET DÜNDAR SEZER AND MEHMET EMIN BAŞBUĞ for supporting me with motivation and collaboration during my research endeavors.

I am grateful for National Science Foundation and CASE: The Center for Advanced Systems and Engineering grants for supporting our research group continuously during my Ph.D. studies. We were able receive required equipments such as smart phones, smart watches, tablets, and embedded platforms, which helped us to perform experiments and acquire results for our publications. Moreover, we were able to attend many international conferences to present our research as well as to extend our knowledge and vision for future projects.

Finally, I would like give my sincerest gratification to my parents SULTAN ÖZCAN and ZEKI ÖZCAN along with my dear sister GÖZDE ÖZCAN. They provided the greatest emotional support and motivation to progress with my career and improve myself everyday. In a similar vein, my sister is studying Electrical and Electronics Engineering at Bilkent University these days. Indeed, I would be very happy to see her follow a similar path towards in her career. ■

TABLE OF CONTENTS

Acknowledgements vi

List of Tables **xii**

List of Figures **xiii**

1 Introduction **1**

1.1 Research Impact and Publications 4

1.1.1 Book Chapter 6

1.1.2 Peer-Reviewed Journal Publications 6

1.1.3 Peer-Reviewed Conference Publications 7

1.2 Organization of Thesis 8

2 Related Work on Activity Classification using Wearable Sensors **10**

2.1 Recent work using only wearable cameras 11

2.1.1 Global Body Motion Activity Classification 13

2.1.2 Local Interaction Activity Classification 14

2.2 Recent work using hybrid sensor modalities 17

2.2.1 Works using hybrid sensor modalities with cameras 18

3 Automatic Fall Detection and Activity Classification by a Wearable Embedded Smart Camera **21**

3.1 Introduction 21

3.2 State of the Art 24

3.2.1 Systems using non-vision sensors 25
3.2.2 Systems using vision-based sensors 26
3.2.3 Differences of the proposed method 27
3.3 Proposed Method 28
3.3.1 Event Detection 28
3.3.2 Dissimilarity Distance 31
3.3.3 Adaptive Number of Cells 32
3.3.4 Event Classification 34
3.3.5 Fall detection using modified HOG 34
3.3.6 Event classification using Optical Flow 35
3.4 Experimental Results 37
3.4.1 Sensitivity and Specificity Comparison 37
3.4.2 Experiments with free fall of a broomstick 38
3.4.3 Experiments with people 39
3.4.4 Fall Experiments 40
3.4.5 Sitting and Lying down Classification Experiments 41
3.4.6 Outdoor experiments 43
3.4.7 Embedded camera experiments with CITRIC motes 43
3.5 Conclusion 45

4 Wearable Camera- and Accelerometer-based Fall Detection on Portable Devices 48
4.1 Introduction 48
4.2 Proposed Method 49
4.2.1 Summary of HOG and Modified HOG 49
4.2.2 Gradient Local Binary Pattern Features 50
4.2.3 Camera-based Detection 51
4.2.4 Detecting a fall 52

4.2.5 Accelerometer-based Detection 52
4.2.6 Camera Data Fused with Accelerometer Data 53
4.3 Experimental Results 54
4.3.1 Camera-based Detection on Recorded Videos 54
4.3.2 Detection with Fusion on Actual Smart Phones 55
4.3.3 Battery Consumption 57
4.4 Conclusion 58

5 Autonomous Fall Detection with Wearable Cameras by using Relative Entropy Distance Measure 59

5.1 Introduction 59
5.2 Problem Definition and Challenges 60
5.3 Proposed Fall Detection Algorithm with Autonomous Threshold Computation 61
5.3.1 Fall Detection 62
5.3.2 Autonomous Threshold Computation for Fall Detection 66
5.4 Experimental Results 71
5.4.1 Indoor Experiments 73
5.4.2 Outdoor Experiments 75
5.5 Conclusion 76

6 Robust and Reliable Step Counting by Mobile Phone Cameras 82

6.1 Introduction 82
6.2 Proposed Method 85
6.3 Experimental Results 88
6.4 Proposed Method with Kalman Tracking 90
6.5 Experimental Results with Kalman Tracking 95
6.6 Conclusion 97

7 Doorway Detection for Autonomous Indoor Navigation of Unmanned Vehicles 99

7.1 Introduction 99

7.2 Proposed Method 100

7.2.1 Door Detection on RGB Images for Verification 101

7.3 Experimental Results 102

7.4 Conclusion 103

8 Real-time Traffic Sign Detection for low quality mobile cameras with improved Aggregate Channel Features 105

8.1 Introduction 105

8.2 Proposed Method 107

8.2.1 Training Stage 109

8.2.2 Testing Stage 109

8.2.3 Model Training employing Fast-RCNN 111

8.2.4 Object Localization with Selective Search 111

8.3 Experimental Results 114

8.4 Conclusion 118

9 Conclusion and Future Work 119

9.1 Future Work 120

References 121

LIST OF TABLES

2.1 Papers classified by activity type (Global Body Motion or Local Interaction) using Camera 12
2.2 Papers classified by activity type(Global Body Motion or Local Interaction) Using A Hybrid Set of Sensors 17
2.3 Papers classified according to activity and sensor type and processing location 20

3.1 Falls from standing position : Sensitivity and Specificity using different methods . 39
3.2 Falls from Sitting Position: Sensitivity and Specificity 41
3.3 Classification rate on 105 event trials 41
3.4 Sensitivity and False Positive with sitting and lying down on CITRIC platform . . . 45

4.1 Sensitivity and Specificity Comparison 55
4.2 Sensitivity and Specificity values for fall detection 57
4.3 False Positives generated for 30-min. videos 58

5.1 Mean and Standard Deviation of Sensitivity-Specificity 75

6.1 Table caption text 90
6.2 Comparison of minimum, maximum, and standard deviation of error rates 90
6.3 Table caption text 97
6.4 Comparison of minimum, maximum, and standard deviation of error rates 97

7.1 ACF detector performance 103

8.1 Comparison of computation efficiency on CPU 116

LIST OF FIGURES

3.1 Example frames captured by the camera during a fall. 29

3.2 Edge strength values corresponding to frames in (a) Fig. 2.1(a), and (b) Fig. 2.2(b). 29

3.3 (a) False 'fall' alarms are generated during lying down and sitting events when using (a) the original HOG; (b) proposed approach with fixed number of cells; (c) proposed approach with adaptive number of cells. 30

3.4 False 'fall' alarm when using the original HOG. 32

3.5 Cells before and after a fall event. 32

3.6 Histogram of Edge Orientations using a fixed number of cells (a) before falling and (b) after falling. Employing adaptive number of cells (c) before falling and (d) after falling. 33

3.7 Plots of different distances for a typical falling down. 34

3.8 (a) Relative object motion at a distance, (b) relative object motion towards the camera, (c) relative object rotation at a distance, (d) relative object rotation about its' axis. 35

3.9 Horizontal and vertical mean values for a sitting down event. 36

3.10 Horizontal and vertical mean values for a lying down event. 37

3.11 Free fall of a broomstick is detected. 38

3.12 Example frames captured during a free fall. 39

3.13 Example frames captured during a sitting down event. 42

3.14 Example frames captured during a lying down event. 42

3.15 Example frames captured during a fall from sitting. 42

3.16 Example frames captured during a fall from standing. 43

3.17 Example frames captured during a fall in outdoor scene. 43
3.18 Outdoor scene dissimilarity distances of the proposed method and original HOG. . 44
3.19 (a) CITRIC platform, (b) which is connected around the waist. 45

4.1 GLBP histogram generation steps. 50
4.2 Plots of different distance measures for a typical fall. 53
4.3 Example frames captured during a fall from standing up position. 55
4.4 An Android™ smart phone attached to the waist. 56

5.1 The steps showing the generation of 56-dimensional GLBP feature for a cell. . . . 63
5.2 Advantages of local binary pattern (LBP) methods; (a) HOG, (b) GLBP. 63
5.3 Different distance measure plots obtained from a video that contains walking followed by a fall. 65
5.4 Dissimilarity distance histograms when using (a) 20 and (b) 8 bins. 68
5.5 (a) Microsoft® LifeCam™, (b) which is attached to a belt around the waist. 72
5.6 Example images captured from the body-worn camera during a fall from standing up position in an indoor environment. 73
5.7 (a) and (b) Example images, captured by the body-worn camera, while a fall occurs ; (c) The dissimilarity distances obtained for this video. 77
5.8 Receiver Operating Characteristic (ROC) curves, for varying threshold values, obtained from the indoor dataset containing falls and non-fall activities. 78
5.9 Sensitivity-Specificity curves, for varying threshold values, obtained from the indoor experiment dataset containing falls and non-fall activities. 78
5.10 Sensitivity values for falls from standing up position with varying thresholds. . . . 79
5.11 Sensitivity values for falls from sitting down position with varying thresholds. . . . 79
5.12 Example images captured from the body-worn camera during a fall from standing up position in an outdoor environment. 80
5.13 Receiver Operating Characteristic (ROC) curve, for varying threshold values, obtained from the outdoor dataset. 80

5.14 Sensitivity-Specificity curves, for varying threshold values, obtained from the outdoor dataset. 81

6.1 (a) A top view of a user holding the smart phone, (b) Experimental setup showing a user simultaneously carrying five devices: A smart phone is held at one hand, a smart watch is worn on the other wrist, while three other smart phones are put in the front pocket and back pocket of trousers and inside a backpack. 85

6.2 Example frames captured during a walk. 86

6.3 Flow chart of the proposed camera-based algorithm. 86

6.4 Images showing the stages of processing: (a) Original input image, (b) edge detection output, (c) result after lines are removed, (d) detected FAST features, (e) kept FAST features and the cluster center. 91

6.5 Movement of the cluster center in the horizontal direction over time: (a) before smoothing, (b) after Savitzky-Golay smoothing. The detected valleys are shown in (c). 91

6.6 Example frames captured during a walk. 92

6.7 Images showing the stages of processing: (a) Original input image, (b) edge detection output, (c) detected FAST features, (d) kept FAST features and the cluster center, (e) actual cluster center, tracked cluster center and bounding box for the foot region. 93

6.8 Flow chart of the proposed camera-based algorithm. 94

6.9 Movement of the cluster center tracked with Kalman filter in the horizontal direction over time: (a) before smoothing, (b) after Savitzky-Golay smoothing. The detected valleys are also shown in (b). 95

7.1 Two doorway detection examples. Blue, red and green boxes represent the candidate region, its' padded version and the detected door, respectively. 102

7.2 Candidate regions that are correctly declared as not being doors (no green box). . . 102

8.1 First two rows: Examples images used for training; third row: Four traffic sign shapes used as templates to match the chain code histograms. 108

8.2 ROC curves comparing the proposed detector (Shape-ACF) with two other detectors. 115

8.3 Example detection results of the ACF-based detector and the proposed detector (Shape-ACF). 117

8.4 (a,b,c) Example detections results of the detector trained with Fast-RCNN, (d,e,f) and false positives. 118

9.1 Example stop sign that is detected and classified with its' type. 120

CHAPTER 1

INTRODUCTION

Mobile sensors, including cameras, have found widespread use in recent years due to their ever-decreasing cost, ease of deployment and use, and ability to provide continuous monitoring, across larger areas, as opposed to static sensors installed at fixed locations. Since many smart phones and tablets are now equipped with a variety of sensors, including accelerometer, gyroscope, magnetometer, microphone and camera, it has become more feasible to develop algorithms for different applications by using data from one or more of these sensors. In addition, since these mobile devices are also equipped with processors, many of these algorithms can be implemented locally onboard. In this thesis, we focus on multiple mobile camera applications, and present lightweight algorithms suitable for embedded mobile platforms. The mobile camera scenarios presented in this thesis are: (i) activity detection and step counting from wearable cameras; (ii) door detection for indoor navigation of unmanned vehicles, and (iii) traffic sign detection from vehicle-mounted cameras for autonomous driving and driver assistance.

Wearable sensors are widely being used to monitor daily human activities and vital signs. Robust detection of events and activities, such as falling, sitting and lying down, is a key to a reliable activity monitoring system. While fast and precise detection of falls is critical in providing immediate medical attention, other activities like sitting and lying down can provide valuable information for early diagnosis of potential health problems. In this thesis, we first present a fall detection and activity classification system using wearable mobile cameras. Since the camera is worn by the subject, monitoring is not limited to confined areas, and extends to wherever the subject may travel including indoors and outdoors. We propose an autonomous fall detection system by taking a completely different view compared to existing vision-based activity monitoring systems and applying a reverse approach. In our system, in contrast to static sensors installed at fixed locations, the camera is worn by the subject, and thus, monitoring is not limited only to areas where the sensors are located, and extends to wherever the subject may travel. Furthermore, since the captured images are not of the subject, privacy concerns are alleviated. Proposed fall detection algorithm employs histograms of edge orientations and strengths, and proposes an optical flow-based method for activity classification.

As mentioned above, thanks to the recent advances in wearable device technology, it has become feasible to employ them as standalone platforms and perform different tasks. Mobile devices provide high processing capability and various sensor information such as camera, accelerometer, gyroscope and magnetometer. Therefore, we also present a fall detection system using wearable devices, e.g., smart phones and tablets, equipped with cameras and accelerometers. A camera provides abundance of information, and the results presented here show that fusing camera and accelerometer data not only increases the detection rate, but also decreases the number of false alarms compared to only accelerometer-based or only camera-based systems. We employ histograms of edge orientations together with the accelerometer data for smart phone optimized fall detection system. The proposed algorithm can run in real-time on an actual smart phone and has been used for experimental evaluation with different subjects.

In order to increase the accuracy for camera-based detection even further, we then present another fall detection algorithm, which incorporates different features, and computes the threshold autonomously. We employ a modified version of the histograms of oriented gradients (HOG) together with the gradient local binary patterns (GLBP). It has been shown that, with the same training set, the GLBP feature is more descriptive and discriminative than HOG, histograms of template, and semantic local binary patterns. Moreover, we autonomously compute a threshold, for the detection of fall events, from the training data based on relative entropy, which is a member of Ali-Silvey distance measures. Overall performance improvement regarding detection with lower false positives has been presented for both indoor and outdoor experiments as wells as various types of falls including falls from standing up position and falls from sitting down position.

As another application of wearable cameras, we present an algorithm for autonomous footstep counting by using camera data from a mobile device. Accelerometer-based step counters are commonly available, especially after being integrated into smart phones and smart watches. Accelerometer data is also used to measure traveled distance for indoor positioning systems. Yet, accelerometer-based algorithms are prone to over-counting, since they also count other routine movements, including movements of the phone, as steps. In addition, when users walk really slowly, or when they stop and start walking again, the accelerometer-based counting becomes unreliable. Since accurate step detection is very important for indoor positioning systems, more precise alternatives are needed for step detection and counting. Therefore, we present a robust and reliable method for counting footsteps using videos captured with a Samsung Galaxy S™ 4 smart phone.

As a second mobile camera scenario, we study autonomous indoor navigation of unmanned vehicles by using vehicle-mounted cameras. Fully autonomous navigation of unmanned aerial vehicles, without relying on pre-installed tags or markers, still remains a challenge for GPS-denied areas and complex indoor environments. Doors are important for navigation as the entry and exit points. A novel approach is proposed to autonomously detect doorways by using the Google Project Tango™ platform. We first detect the candidate door openings from the 3D point cloud, and then use a pre-trained detector on corresponding RGB image regions to verify if these openings are indeed doors. We employ Aggregate Channel Features (ACF) for detection, which are computationally efficient for real-time applications. We obtain the doorway detections from depth sensor and then check the corresponding regions in RGB images to verify that they are actual door openings. Since door detection is only performed on candidate regions, the system is more robust against false positives.

Mobile cameras are also being employed in intelligent transportation systems as wehicle-mounted cameras. For instance, accurate traffic sign detection, from vehicle-mounted cameras, is an important task for driving assistance, autonomous driving, and warning purposes. It is a challenging task especially when the videos acquired from mobile cameras on portable devices are low-quality. In this thesis, we focus on naturalistic videos captured from vehicle-mounted cameras. Recently, it has been shown that Region-based Convolutional Neural Networks (R-CNN) [1] provide high accuracy rates in object detection tasks. Yet, R-CNN-based methods are computationally expensive, and often require a Graphics Processing Unit (GPU) for faster training and processing. Thus, we present a new method, incorporating Aggregate Channel Features (ACF) [2] and Chain Code Histograms (CCH) [3], with the goal of providing much faster training and testing, and comparable or better performance with respect to deep neural network approaches, without requiring specialized processors.

Proposed computer vision algorithms provide promising results for various useful applications despite the limited energy and processing capabilities of mobile devices.

1.1 Research Impact and Publications

We present our methods and algorithms with possible applications aimed for mobile cameras. In certain applications, we also incorporate other sensor modalities, more specifically accelerometers and infrared depth cameras, for improved detection and classification purposes. The mobile camera applications we focus on include fall detection, activity classification and footstep counting from wearable cameras, and doorway detection for autonomous navigation of unmanned vehicles, and traffic sign detection from lower-resolution and noisy videos using cameras of mobile devices. These algorithms have been designed to be implemented on mobile platforms, despite limited energy and processing capabilities of mobile devices. Mobile cameras provide continuous monitoring and data across larger areas, as opposed to static cameras that can only monitor certain parts of an environment.

Previous fall detection works in the literature concentrate on gyroscope/accelerometer-based systems, stationary acoustic/vibration based approaches or static camera-based systems. Accelerometers, gyroscopes, or magnetometers have their own limitations such as lacking generalization across different people and test data, being prone to false positives, and requiring higher number of sensors for accurate classification and detection. On the other hand, static camera and acoustic/vibration-based sensors are limited to the environment, where they are installed. The developed system is among the first camera-based systems that use a mobile wearable camera, as opposed to static cameras installed in certain rooms. Thus, monitoring is not limited to conned areas, and extends to wherever the subject may travel including indoors and outdoors. Furthermore, since the captured images are not of the subject, privacy concerns of the subjects are alleviated. We also implemented a real-time fall detection system using a Samsung Galaxy S4 smartphone and its' front-facing camera camera and accelerometer. The average detection rates obtained when using only the accelerometer, only the camera and when fusing accelerometer and camera modalities were 65.66%, 74.33% and 91%, respectively. Moreover, on 30 min-videos from 10 subjects, performing different daily activities such as running and using the stairs, it was shown that fusing accelerometer and camera data decreased the number of false positives. More specifically, the average number of false positives were 3.4 and 6.3 when using fusion of camera and accelerometer data, and only accelerometer data, respectively. Experimental results and trials with actual Samsung Galaxy phones show that the proposed method, combining two different sensor modalities, provides much higher sensitivity, and a significant decrease in the number of false positives during daily activities, compared to accelerometer-only and camera-only methods.

In our improved fall detection work, incorporating gradient local binary patterns (GLBP) increases the overall accuracy on our fall detection dataset, while relative entropy-based automatic thresholding provides optimal operating point for fall detection. For fall detection problem, we propose an optimal selection criteria based on information theoretic based relative entropy approach. We created a database of videos recorded with a wearable camera. We recorded and labeled around 300 activities performed by 10 different people. Activities include walking, sitting down, lying down, walking and falling down for activity detection and classification purposes. Experimental results show that, with the autonomously computed threshold, the proposed method provides 93.78% and 89.8% accuracy for detecting falls with indoor and outdoor experiments, respectively.

For autonomous footstep counting work, we constructed another database of videos recording the feet movement of 10 people while they are walking. This is the first method to use mobile camera for tracking and counting footsteps without relying on template matching for footsteps. The method provided significantly less error rate compared to the accelerometer-based step counting apps running on a smart watch, and smart phones held in hand, and carried in front pocket, back pocket, or inside a backpack. The experimental results show that camera-based step counting has the lowest average error rate for different users, and is more reliable compared to accelerometer-based counters. In addition, the results show the high sensitivity of the accelerometer-based step counters to the location of the device and high variance in their performance across different users.

We also presented a new method for accurate traffic sign detection from lower-resolution and noisy videos captured by vehicle-mounted cameras. In this method, we incorporate Aggregate Channel Features (ACF) and Chain Code Histograms (CCH) with the goal of providing much faster training and testing, and obtaining comparable or better performance with respect to deep neural network approaches without requiring specialized processors. We used a custom dataset of lower resolution and noisy images and videos for both training and testing of the proposed detector. We tested different detector performances on 39 videos samples from various weather and daylight conditions. The proposed method provided the highest true positive rate for lower false positive values while performing much faster than Fast-RCNN on CPU.

The research presented in this thesis resulted in several publications including a book chapter and respected IEEE journals and international conference proceedings.

1.1.1 Book Chapter

- K. Ozcan, A. Mahabalagiri, and S. Velipasalar, "Automatic Fall Detection and Activity Classification by a Wearable Camera," Distributed Embedded Smart Cameras, pp. 151–172, Springer New York, New York, NY, 2014.

1.1.2 Peer-Reviewed Journal Publications

- K. Ozcan, S. Velipasalar, and P. K. Varshney, "Autonomous Fall Detection with Wearable Cameras by Using Relative Entropy Distance Measure," IEEE Transactions on Human-Machine Systems, vol. 47, no. 1, pp. 31-39, Feb. 2017.

- M. Cornacchia, K. Ozcan, Y. Zheng and S. Velipasalar, "A Survey on Activity Detection and Classification Using Wearable Sensors," IEEE Sensors Journal, vol. 17, no. 2, pp. 386-403, Jan.15, 2017.

- K. Ozcan and S. Velipasalar, "Wearable Camera- and Accelerometer-Based Fall Detection on Portable Devices," IEEE Embedded Systems Letters, vol. 8, no. 1, pp. 6-9, March 2016.

- K. Ozcan, A. K. Mahabalagiri, M. Casares and S. Velipasalar, "Automatic Fall Detection and Activity Classification by a Wearable Embedded Smart Camera," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 3, no. 2, pp. 125-136, June 2013.

1.1.3 Peer-Reviewed Conference Publications

- B. Kakillioglu, K. Ozcan and S. Velipasalar, "Doorway detection for autonomous indoor navigation of unmanned vehicles," 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 2016, pp. 3837-3841.
- K. Ozcan, A. Mahabalagiri and S. Velipasalar, "Autonomous tracking and counting of footsteps by mobile phone cameras," 2015 49th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, 2015, pp. 1408-1412.
- K. Ozcan and S. Velipasalar. 2015. Robust and reliable step counting by mobile phone cameras. In Proceedings of the 9th International Conference on Distributed Smart Cameras (ICDSC '15). ACM, New York, NY, USA, 164-169.
- Y. Zheng, K. Ozcan, S. Velipasalar, Hao Shen, and Qinru Qiu. 2014. Energy Efficient Tracking by Dynamic Voltage and Frequency Scaling on Android Smart Phones. In Proceedings of the International Conference on Distributed Smart Cameras (ICDSC '14). ACM, New York, NY, USA.
- A. Mahabalagiri, K. Ozcan, and S. Velipasalar, “Camera motion detection for mobile smart cameras using segmented edge-based optical flow," Advanced Video and Signal Based Surveillance (AVSS), 2014 11th IEEE International Conference on, vol., no., pp.271,276, 26-29 Aug. 2014.
- A. Mahabalagiri, K. Ozcan, and S. Velipasalar, “A robust edge-based optical flow method for elderly activity classification with wearable smart cameras," Distributed Smart Cameras (ICDSC), 2013 Seventh International Conference on, vol., no., pp.1,6, Oct. 29 2013-Nov. 1 2013.

- K. Ozcan, A. Mahabalagiri, and S. Velipasalar, “Fall detection and activity classification using a wearable smart camera," Multimedia and Expo (ICME), 2013 IEEE International Conference on, vol., no., pp.1,6, 15-19 July 2013.
- M. Casares, K. Ozcan, A. Mahabalagiri, and S. Velipasalar, “Automatic fall detection by a wearable embedded smart camera," Distributed Smart Cameras (ICDSC), 2012 Sixth International Conference on, vol., no., pp.1,6, Oct. 30 2012-Nov. 2 2012.

1.2 Organization of Thesis

The remainder of this thesis is organized as follows:

In Chapter 2, we provide a summary of recent publications within the area of wearable cameras and their applications for healthcare, activity detection and classification. In addition to the type of sensors and type of activities classified, we provide details on learning algorithm type, and extent of experimental setup. We further discuss where the processing is performed, i.e., local versus remote processing, for different systems.

In Chapter 3, we present a fall detection and activity classification system developed for embedded smart camera platform CITRIC [4]. Experimental results show the success of the proposed method.

Next, in Chapter 4, we present a fall detection algorithm, that incorporates accelerometer and camera data, and is implemented on an actual smart phone.

Then, in Chapter 5, we present another camera-based fall detection algorithm, which incorporates different image features and computes the threshold autonomously. The threshold for fall detection is computed from the training data based on relative entropy, which is a member of Ali-Silvey distance measures.

The autonomous footstep counting algorithm is presented in Chapter 6.

In Chapter 7, we present the autonomous doorway detection and verification algorithm for autonomous indoor navigation unmanned vehicles.

In Chapter 8, we present our proposed autonomous traffic sign detection algorithm for lower resolution and noisy videos. We compare the performance of the proposed method with other detectors, namely a pure ACF-based detector, and Fast R-CNN-based detector, both in terms of accuracy, through ROC curves, and processing times on lower-resolution videos. The conclusions and future work are presented in Chapter 9.

CHAPTER 2

RELATED WORK ON ACTIVITY CLASSIFICATION USING WEARABLE SENSORS

In this chapter, we provide an extensive summary of related works in human activity classification using wearable sensors. Human activity detection and classification are important for a wide range of applications including monitoring activities of elderly people for assistive living, robotics, human computer interaction, and surveillance. With developing technology and decreasing cost of devices with various sensors, life-logging has become more popular. Many of these applications still require a person to manually record activities. There is a growing demand for autonomous activity monitoring and classification systems. Therefore, we present the state-of-the-art in activity monitoring systems, specifically for the ones employing wearable cameras.

Existing activity monitoring systems can be broadly classified into two categories based on how the sensors are deployed: (1) fixed sensor setting, where information is gathered from static sensors mounted at fixed locations, and (2) mobile sensor setting, where the sensors are wearable and thus mobile. The alternative to fixed-sensor settings for activity monitoring is mobile sensing, where the sensors are wearable and activity is monitored in a more first-person perspective. Before the last decade and a half, wearable technologies were often large and cumbersome devices more often found in specialized healthcare facilities. However, for the last decade or so we observe mobile sensors become more widespread. The change was spurred by a shift of cultural acceptance of mobile devices, the advancement in hardware size and power, and the marketing of smartphones. The acceptance of mobile hardware can be seen through the statistic that in 1998 the U.S. census recorded that only 31% of households had cellphones compared to the 71% of households in 2005 [5]. Now, there are smartphones, wrist worn and sensor-equipped watches, and even glasses equipped with onboard computing.

With wearable and mobile technologies already a part of our daily lives, human activity detection and classification approaches have become both feasible and more culturally acceptable. This type of mobile sensing also allows monitoring of subjects wherever they may travel. With the availability of affordable hardware for the purposes of mobile sensing, and the widespread use of mobile devices, the focus of this chapter is therefore on activity classification using wearable cameras.

We group activities into two main categories: (1) activities involving global body motion and (2) activities involving local interaction. Global motion type activities will include ambulation, transportation, and exercise/fitness activities. We use the term global motion activities to describe motions that involve displacement of the entire body, such as walking, running, and climbing stairs. Local interaction activities are those that generally involve the interaction with an object and would be the activities which are categorized in [6] as phone usage and daily activities. Specifically, local interaction type activities do not involve motion of the entire body, but rather only involve motion of the extremities. These local interactions often require additional context in terms of identification of an object with which a subject is interacting. For example, the local interaction activity of brushing teeth involves the use of a tooth brush. Although we use these two categories to group papers, not all works classify tasks only from either global or local activities, works such as [7, 8, 9, 10, 11] classify activities from both categories.

2.1 Recent work using only wearable cameras

Over the years, the flexibility of wearable camera systems has improved due to decreasing camera and processor sizes and increasing resolution as well as processing power. Table 2.1 groups

Table 2.1: Papers classified by activity type (Global Body Motion or Local Interaction) using Camera

Global Body Motion Activities	
Activity Type	References
Sitting	[20, 21]
Standing	[22, 21]
Walking	[23, 20, 24, 21, 13, 25]
Running	[23, 24, 25]
Jumping	-
Lying	[21]
Stairs	[13, 25]
Other	[20, 22, 21, 26, 13, 14, 27, 28]
Local Interaction Activities	
Activity Type	References
Hygeine Activity	[29, 30, 17, 31]
House Cleaning	[29, 30, 17, 31]
Eating	[29, 30, 17, 31]
Construction Activities	-
Food Preparation	[32, 16, 15, 33, 29, 30, 17, 31, 34]
Office Activities	[35, 29, 30, 17, 31, 34, 26]
Other	[36, 37, 38]

camera-based works into two classes (based on whether they focus on global body motion or local interaction with objects), and lists different activity types that these works address. Initially, many of the efforts involved backpack-based camera systems such as those in [12]. Camera technologies have now been developed to the point where image capture can be accomplished by less intrusive setups. Numerous demonstration systems, especially with the introduction of Google Glass®, have prototyped algorithms for activity classification based on images captured from an eye-glass worn camera [9][13]. Others have adopted head-[14, 15, 16] or chest-mounted cameras [17, 18]. Bambach [19] summarizes the advances in computer vision algorithms for egocentric video.

While these wearable camera-based activity classification systems can be distinguished from one another in terms of the placement of the camera on the body, the majority of these camera based setups choose either a head or torso-mounted configuration. Additionally, all works include only a single camera and often do not specify the number of subjects or have fewer than 10 subjects. The high-level distinguishing factor between these camera-based systems is the type of features extracted from the scene. These features may be more general, high-level scene features that quantify the motion of the scene or more fine-grained details that try to capture interactions of the subject with objects in the scene. We therefore group the wearable camera-based activity classification work into two categories: Global body motion and local interaction activity classification described in Sections 2.1.1 and 2.1.2, respectively.

2.1.1 Global Body Motion Activity Classification

While there are some works in the area of global motion activity classification based on wearable cameras, one will notice that this section has comparatively less number of works than the global body motion activity detection using accelerometer, magnetometer, or gyroscope. Nonetheless, there are some works that have explored global motion using camera-based techniques. Global body motion is often inferred from images based on an estimate of the camera motion. The methods for estimating this camera motion often fall into two main groups, (i) techniques that leverage methods based on point-based features [39, 40, 41, 22], and (ii) techniques that employ optical flow like features [13, 25, 26, 14, 28].

While some works employing point-based features used standard detection algorithms such as Speeded Up Robust Features (SURF) [22], others focused on developing novel point-based features. Zhang et al. [40, 41] use a novel point feature detection based on defining interest points from the covariance matrix of intensities at each pixel. They then calculate motion histograms based on the point features to capture the global motion distribution in the video.

The optical flow based techniques appear to differ based on the learning algorithm employed. Kitani et al. [26] investigate the use of motion-based histograms from optical flow vectors and unsupervised Dirichlet learning on classification of 11 sports-based ego-actions. Zhan et al. [13] extract optical flow features for each frame, and pool the optical flow vectors over numerous frames to provide temporal context. They compare three classification approaches, namely K-Nearest Neighbor (KNN), SVM, and LogitBoost. The approach also includes a comparison with and without an HMM. Yin et al. [25] also use an optical flow technique, but employ an SVM for classification.

There are still other works that do not exactly fall into describing global motion based on local point features or optical flow based features. Song and Chen [21] use a histogram of oriented gradients approach to detect humans from their camera-based system. However, in the case of

[21], their system is using a mobile camera on a robot watching the human. Similarly, Watanabe and Hatanaka [24] also use a different setup of the camera, wherein a single hip or waist-mounted camera is employed to describe the gait of individual. Unlike other works, the camera is facing downward and motion is relative to the waist position. A walking state at each frame is composed of the position of the waist, and relative position of joints, as well as the angular speed of the camera. It is assumed that the intrinsic parameters of the camera are known and the world coordinates of the observed points are known.

2.1.2 Local Interaction Activity Classification

The works, described in Section 2.1.1, focusing on higher level, global body motion activities, while comparable to and competitive with the accelerometer-based approaches, do not necessarily take advantage of the visual details captured by cameras. Thus, other researchers have considered the ability to classify more detailed activities [32, 16, 36, 35, 33]. To capture this variety of classification, the works in this section view activities as involving a subject and object interaction or the interaction of multiple subjects.

The majority of the works, discussed in this section, will be on classifying activities that involve object and subject interactions, however we do not want to ignore works such as [38, 32] that classify interactions between multiple subjects. Aghazadeh et al. [38] propose novelty detection in egocentric video by measuring appearance and geometric similarity of individual frames and exploiting the invariant temporal order of the activity to determine if a subject runs into a friend, subject gives directions, or the subject goes to an ice cream shop. Fathi et al. [32] similarly classify the type of interaction multiple subjects are involved in, such as a dialogue, monologue, or discussion.

To begin to understand these interactions, there have been supporting works that contributed to individual aspects of being able to tackle the larger problem of activity classification. These supporting works have addressed problems such as detecting a hand [37], better recognizing an object from an egoview [42], methods for gaze/attention focus detection and 3D estimation of hand-held objects [43], or linking an object to a location [44]. Another area of work has been in form of video summarization, which contributes to activity classification by developing techniques for temporally segmenting a video into distinct actions. The ability to properly segment regions of activity has been further improved by several works of Grauman et al. [45, 46, 47, 48, 49] that proposed algorithms for video event summarization for ego-centric videos including predicting important objects and people [46, 47], extracting frames that might be a snapshot from ego-centric video [45], and video summarization of daily activity ego-centric videos based on detected objects and their relations and co-occurrences [48]. In addition to temporal segmentation, Lee and Grauman [49] also explored within frame segmentation, developing region cues indicative of high-level saliency.

Attempts at activity classification, began with works such as [35] by Mayol and Murray, who use a shoulder-mounted camera to recognize manipulations of objects by an individual's hands. This work, however, is constrained to a static workspace view and manipulations are described as actions carried out by the subject's hands. In fact, to first determine the center of focus, they use the center of mass of a detected skin region, where these skin regions are assumed to be either one or both hands. More recent works began to use less constrained experimental setups, where the camera is in fact mobile. Sundaram et al. [33] use a shoulder-mounted camera on a moving subject. They however, like their predecessors, also attempt to detect the hands and represent activities as manipulations by hands. They divide their system into two parts; the first, where the vision is relied on to recognize the manipulation motions and the second, where a Dynamic Bayesian Network (DBN) is used to infer objects and activities from the motions of the first part. An accuracy of 60.99% is obtained for the manipulation motions that include making a cup of coffee, making a cup of tea, washing a dish etc.

Fathi et al. [16] also use manipulations by the hand to classify fine-grained activities and similarly focus on food preparation tasks such as making a hotdog, a sandwich, or coffee. They propose a framework for describing an activity as a combination of numerous actions. For example, actions of the coffee making activity include opening coffee jar, pouring water in coffee maker, and so on. They attempt to assign action labels to each segmented image interval and an object, hand, or background label to each super-pixel in each frame. The action models are learned using Adaboost [50] and a set of features that include information such as object frequency, object optical flow, hand optical flow, and so on. The action verbs are estimated first, then the object classes are inferred with a probabilistic model. The final steps involve refining decisions from previous stages based on the final decisions of activities, using a conditional random field. This work is evaluated on the

GeorgiaTech egocentric activity dataset, which includes video from a head-mounted camera. Frame based action recognition accuracy is 45%, whereas activity classification accuracy is 32.4%. They further demonstrate that object recognition is in fact improved based on action classification results. In [15], Fathi et al. demonstrate improved results on the same GeorgiaTech dataset by modeling actions through the state changes that are caused on an object or materials. This is in contrast to their initial work that uses information from all frames. To detect changed regions the authors sample frames from the beginning and end of an action sequence. Change is measured in terms of the color difference of matched pixels. An SVM is trained to detect regions that correspond to specific actions. Using state change detection, 39.7% accuracy is achieved over 61 classes on the GeorgiaTech egocentric activity dataset. The activity segmentation results in 42% accuracy.

As accuracies in object detection works began to improve, others began to use these more specific object detection models for activity classification. Pirsiavash and Ramanan [30] use fully supervised learning with dense labels to train deformable parts-based object detectors. With the additional label information, this enables more complex understanding of multiple objects interacting in a single scene. They also recognize that changes in the objects occur throughout an action sequence and develop a notion of active versus passive objects and create separate detectors for objects in each state. To evaluate their system, they created and annotated a 1 million-frame dataset, and obtain results of around 30% frame classification accuracy. However, with an assumed ideal object detector, they demonstrate that this method would obtain a 60% frame classification accuracy. The created dataset is from a chest-mounted Go-Pro camera and contains label information about activities such as brushing teeth, combing hair, making coffee, and so on. Numerous works leverage this dataset to evaluate their approaches including [29, 17, 31]. Mccandless and Grauman [29] propose to learn the spatio-temporal discriminative partitions for egocentric videos and apply object detection to obtain a strong classifier for recognition of 18 activities, achieving an overall 38.7% F score.

Matsuo et al. [17] further extend the work in [30], through visual attention and saliency, accounting for cases where a hand may not be involved in the interaction. The groundwork for using visual attention and saliency can be linked to other works, including [51], [34] and [52]. Matsuo et al. [17] propose a method for quantifying visual saliency, not only through the static saliency of an

Table 2.2: Papers classified by activity type(Global Body Motion or Local Interaction) Using A Hybrid Set of Sensors

Global Body Motion Activities	
Activity Type	References
Sitting	[53, 54, 18, 55, 56, 57, 58, 59, 60, 61, 23, 62, 63, 64, 65, 66, 67, 68]
Standing	[54, 18, 55, 57, 58, 69, 59, 60, 61, 23, 62, 63, 70, 64, 65, 66, 67, 71]
Walking	[72, 53, 54, 18, 55, 57, 58, 69, 59, 73, 60, 74, 61, 62, 63, 70, 64, 75, 65, 67, 76, 77, 68, 71]
Running	[54, 18, 55, 57, 58, 59, 74, 61, 23, 63, 64, 75, 68, 71, 78]
Jumping	[54, 55, 79, 75]
Lying	[53, 55, 56, 57, 58, 73, 60, 61, 23, 63, 66, 80, 76, 68]
Stairs	[53, 18, 57, 58, 69, 61, 79, 63, 70, 64, 75, 67, 77, 71]
Other	[57, 58, 69, 59, 73, 60, 74, 79, 70, 64, 75, 65, 76, 68, 71, 81]

Local Interaction Activities	
Activity Type	References
Hygeine Activity	[72, 53, 54, 55, 82, 59, 74, 67, 80, 71, 78]
House Cleaning	[72, 53, 54, 55, 82, 58, 59, 74, 79, 63, 75, 65, 77, 78]
Eating	[82, 59, 63, 67, 76, 68]
Construction Activities	[72, 12]
Food Preparation	[53, 63]
Office Activities	[82, 58, 67, 76, 77, 68, 71, 78]
Other	[72, 54, 55, 75, 83, 78, 84]

image, but through the ego-motions of the first person viewer. The results are based on the same dataset used in [30]. The results improve the average recognition accuracy, over all activities, from 36.9% to 43.3% and decrease variance in accuracy over numerous subjects from 9.8% to 7.1%.

2.2 Recent work using hybrid sensor modalities

With the advances in single sensors, and the size of the sensors becoming smaller, cheaper and more commercialized, there has been many research efforts that now look at not only a single modality sensors but rather a hybrid combination of multiple types of sensors. Table 2.2 presents the activity types of hybrid se

These custom applications are not as easily separated as the classification of global body and local interaction type activities, and a single work in this area can possibly address activities in both categories. It is observed that hybrid sensing systems, usually employ supervised learning methods. In the next subsection, we will discuss some different combinations of sensor modalities with camera.

2.2.1 Works using hybrid sensor modalities with cameras

There are also multiple hybrid systems that include vision-based sensors in addition to other sensor types [72], [53], [18]. While Nam et al. [18] simply fuse information from a camera and accelerometer to increase the accuracy of classification for ambulatory activities, Doherty et al. [72] and Hsien et al. [53] use the context provided by cameras to identify the specific class of activity once an accelerometer has identified the level of activity being undertaken.

Numerous of these works favor a camera and orientation-based sensor combination. Spriggs et al. [85] focus on temporal segmentation of an activity in order to recognize different temporal parts. The authors explore the usage of GMMs, HMMs, and K-Nearest Neighbors for segmenting and classifying various actions involved in the cooking of different meals. The results demonstrate that using both IMU and camera data improves results over single modality sensing. Additionally, the best results were obtained using a K-nearest neighbor approach, whose success was largely attributed to the fact that the feature vectors being used had high dimensionality. Li et al. [86] utilize gyroscope to obtain candidate boundary between different daily activities, and extract visual features from images for better video segmentation.

More recently, a common proposal, for these camera and orientation-based sensors has been an eyeglass-mounted system [9], [87], [66], [67]. Windau and Itti [9] use the inertial sensing from a gyroscope and accelerometer to normalize the coordinate system for different head orientations. For the inertial sensors, energy consumption and movement intensity, and the mean and variance of the sensor readings are extracted. As for the camera-based features, a single GIST vector per frame is calculated. The video data is also used to classify an indoor versus outdoor environment using the GIST vectors. The activities classified include lying down, walking, jogging, biking, washing dishes, brushing teeth, etc. The results show an 81.5% classification accuracy on 20 different activities. Zhan et al. [67] also design an eye glasses-like system with a first person view camera and accelerometer. This work shows that while accelerometer-only classification has proven effective for dynamic activities, the camera is more suitable for static activities. The extracted accelerometer features include both frequency and time domain information. The extracted video features include optical flow vectors. The authors compare classification results for different sensors using LogitBoost and SVM with numerous different kernels. Combining the classifiers through structured prediction using a Conditional Random Field (CRF) with Tree Re-Weighted Belief Propagation, they obtain an overall accuracy of 84.45% on 12 activities. Hernandez et al. [66] propose a real-time human activity recognition for sitting, standing and supine with glasses-based system with onboard processing.

Ishimaru et al. [87] additionally make use of an Infrared (IR) proximity sensor on a glasses-based system. The IR proximity sensor is used to measure the distance between the eyes and the eyewear in order to perform blink detection. Moreover, the average variance of 3D-accelerometer is calculated to construct a head motion model. They try to distinguish between the activities of watching, reading, solving, sawing and talking. An overall accuracy of 82% is achieved on an 8-person dataset for these five activities. Other works have used IR, orientation, and camera systems. Fleury et al. [80] use a webcam, a 3D-accelerometer, a 3D-magnetometer, an IR sensor, and a microphone. They apply Principal Component Analysis (PCA) on extracted features to get ten dominant features that are fed through a multi-SVM for 35 activities, reaching 86% accuracy. Punangpakisiri et al. [76] deploy senseCam with a 3D accelerometer, a light sensor, and a passive IR sensor to classify ten activities.

With advances in gaming systems, such as the Kinect, Red, Green, Blue, and Depth (RGB-D) sensors have also become popular the last few years. Bahle et al. [77], Damen et al. [12], and Moghimi et al. [69] explore the use of RGB-D sensors that provide both a regular imaging and depth information. Bahle et al. [77] reach 92% overall accuracy by using camera and depth-based information with a Dynamic Time Warping technique when classifying walking, writing, using stairs, and using the dishwasher. A helmet-mounted RGB-D camera is employed in [69]. The features used include GIST and a skin segmentation algorithm. To classify the activities, learning based methods such as bag of SIFT words, Convolutional Neural Networks (CNNs), and SVMs were explored. They conclude that CNN-based features provide the best representation for finer-grained activity recognition or tasks that involve manipulations of objects by hands. These works however are still relatively new and have explored limited activity sets compared to other hybrid approaches with around three to six activities classified in each of these works.

We have covered a wide range of wearable sensor approaches by also discussing the progression of how methods using different sensor modalities approach various different problems. Initial

Table 2.3: Papers classified according to activity and sensor type and processing location

Activity Type	Sensor Type	Processing Location	Reference
Global Body	Camera	Onboard	[27, 37]
		Remote	[13, 39, 25, 40, 22, 41, 24, 21, 14, 26]
	Hybrid	Onboard	[81, 88, 65, 89, 90]
		Remote	[68, 9, 67, 91]
Local Interaction	Camera	Onboard	[14, 12]
		Remote	[13, 41, 32, 16, 36, 42, 44, 46, 45, 48, 49, 92, 47, 29, 38, 31, 43, 35, 33, 30, 17, 51, 34, 52]
	Hybrid	Onboard	[93, 94, 54, 55, 87, 79]
		Remote	[85, 9, 67, 72]

A:Accelerometer G:Gyroscope M:Magenetometer

orientation-based sensing approaches started with stable sensors at the waist and classified a limited set of activities. As orientation-based sensors and processing power advanced, other works explored orientation-based sensors on the extremities. With this advancement to the extremities, the number of classes that an approach was able to distinguish increased. These approaches additionally saw improvements and can be distinguished from one another in terms of the increasing feature sets used. Moving these orientation sensors to the extremities also enabled the ability to approach new activity types that involve more localized motions such as brushing teeth, vacuuming, and so on. However, orientation-based sensors are limited by the information they provide, and with improvements in the camera technology, others have been able to show that cameras can provide details that these other orientation sensors cannot. That is, images and videos can provide more detailed information and context for a specific action sequence, and not only be able to distinguish between making food, but also making specific types of food. Yet, despite the additional activities these camera only systems can tackle, the accuracies of these systems, often below 60%, are far lower than other works discussed. Hence, more recent hybrid works have provided the ability to classify both global and local activities, while maintaining higher accuracies of above 80%. However, it can be argued that these single-sensor based approaches have established the groundwork for more advanced hybrid approaches. These hybrid approaches have also tended to be more extensive in their experimentation, often using more realistic experimentation approaches with more subjects. With community datasets sparse in all but the camera-based only works, the experimentation is often specific to each work. Hence, a potential contribution to activity classification using wearable sensors would be the creation of a standard dataset. Table 2.3 reflects that there is still progress to be accomplished until the systems perform processing fully onboard and real-time compared to majority of works where analysis is remotely processed.

CHAPTER 3

AUTOMATIC FALL DETECTION AND ACTIVITY CLASSIFICATION BY A WEARABLE EMBEDDED SMART CAMERA

3.1 Introduction

A nation's progress is determined by the quality of life of its' citizens. Elderly healthcare plays an integral part in this progress. According to the U.S. Census Bureau [95], the old-age dependency ratio (the number of people 65 and over relative to those between 15 and 64) is projected to increase from a current value of 22% in 2010 to 37% by 2050. This is also the projected trend worldwide. This percentage increase would have an exponential effect on the costs involved in social security and healthcare. Therefore, with technological advancement, increasing research effort is expended in the field of elderly healthcare.

In recent years, one of the key aspects of the elderly care has been intensive activity monitoring, and it is imperative that any such activity monitoring be also autonomous. Activity monitoring can focus on short-term, event-based activities, such as falling, sitting and lying down. It can also cover longer-term posture and motion analysis in behavioral assessment for chronic diseases such as arthritis, cardiovascular or neurodegenerative diseases. According to Kang et al. [96], activities of elderly can be classified into four states: (1) resting state such as sitting, lying and standing; (2) locomotion state such as walking and running; (3) emergency state such as fall, and (4) transition state such as stand to sit, stand to lying down and so on. Thus, an ideal autonomous activity monitoring system should be able to classify activities into critical events, such as falling, and non-critical events, such as sitting and lying down. Furthermore, the system should be able to smartly expend its' resources for providing quick and accurate real-time response to critical events versus performing computationally intensive operations for non-critical events. The objective of this report is to contribute towards the development of such an autonomous system. While the general focus is to provide an innovative solution to address the critical fall event, we show means of extending this methodology to non-critical events as well.

Activity monitoring systems have been introduced as part of elderly care in recent years, especially for elderly people living independently. Fall detection is a crucial part of elderly activity monitoring systems, since falls are considered to be the eighth leading cause of death in the U.S. [97], and falls lead to severe complications [98, 99]. It has been reported that 10% of the falls cause fractures and 20% of injuries due to falls need rapid medical attention [100]. In the literature, it has been reported that timely treatment to fall related injuries reduces the morbidity-mortality rate quite substantially [101, 102]. Even though several *user-activated* commercial devices are available in the market (e.g. that require pressing of a button), they provide limited benefits since they assume that the user is conscious after the fall. Autonomous fall detection systems have been introduced in response to growing needs. They can reduce the impact and recovery times, due to injuries caused by falls, by informing others quickly and reducing the time people remain on the floor. However, to find acceptance and widespread use, these systems should be robust and precise, and provide real-time fall detection with tolerable, ideally zero, false positive rate.

Many fall detection algorithms have been proposed relying only on accelerometer data [103, 104, 105]. Koshmak et al. [103] test their method on actual falls of ice-skaters. Yet, since every fall has different acceleration characteristics and the magnitude of acceleration has high variation among various body types, it is challenging to detect different types of falls for different people. Cao et al. [104] employ adaptive thresholds for motion sensors. Accelerometers have also been used in the classification of sports activities [105] to capture training statistics. As discussed in [105], since the placement of the phone differs from person to person, using just the accelerometer might not be sufficient for activity classification. In such cases, use of camera sensors in tandem with the accelerometer can help resolving such issues. Wu et al. [106] also discuss the limitations of using just the accelerometer. Accelerometer-based systems, although simple and cost-effective, can still create false positive alarms even with multiple sensors, especially in environments such as high speed trains and elevators, where people are exposed to acceleration.

Even though several user-activated commercial devices are available, they have limited benefits, especially in situations where the user loses consciousness. In response to growing needs, researchers have been working on autonomous fall detection via dedicated signal processing devices. Noury et al. [107] and Mubashir et al. [108] have provided well-organized surveys on the principles and approaches involved in fall detection. Classification of falls is quite important as well. Falls can be either from standing, sitting or lying down positions. They can also be forward, backward or lateral falls. According to Noury et al. [107], though there are some common characteristics among these falls, different scenarios must be considered for different kinds of falls. For example, falls from a standing position are much easier to detect as compared to falls from sitting or lying down positions. An autonomous system should provide a real-time detection of falls free from false positives.

Vision-based systems have been introduced as an alternative to approaches using non-vision sensors. In most of the existing vision-based systems, one or more cameras are installed at fixed locations to monitor a subject. Similar to the approaches using acoustic and vibrational sensors, using cameras installed at fixed static locations confines the monitoring environment only to those regions. In many systems, videos captured by the cameras are transferred to a central location for processing, which requires extensive communication. In addition, subjects, continuously being monitored by these cameras, often raise privacy concerns [109].

Classification of other non-critical activities such as walking, sitting and lying down are useful in the study of chronic diseases and functional ability monitoring [96, 110]. These activities, though not time-critical, are more complex and computationally more expensive to classify. Thus, developing a system that can handle both critical and non-critical activities is challenging, since the system needs to address both real-time and computationally expensive problems. The issue is even more elevated when we want to implement this on an embedded and wearable platform.

Compared to accelerometers and other non-vision sensors, cameras provide a much richer set of data including contextual information about the environment, which allows the analysis of a variety of activities including falls. In this chapter, we present an autonomous fall detection algorithm employing images from a wearable camera. In the proposed system, we take a completely different view compared to existing vision-based activity monitoring systems, by applying a reverse approach. In our system, in contrast to static sensors installed at fixed locations, the camera is body-worn, and thus, monitoring is not limited only to areas where the sensors are located, and extends to wherever the subject may travel including indoors and outdoors. Moreover, since the captured images are the images of the surrounding environment and not of the subject, the privacy concerns are alleviated, if not eliminated, compared to the static cameras capturing the videos of the subject. Furthermore, the images are not saved or transmitted to a central processor, but processed onboard locally. In a recent study, no concerns have been reported by bystanders about somebody carrying a wearable camera [111]. Based on the current trends, *wearable cameras* are expected to find increasing use to understand lifestyle behaviors for health purposes [72].

3.2 State of the Art

There are detailed surveys available on various approaches used for fall detection [112, 113]. In general, the fall detection and activity monitoring systems can be classified into two main categories: a) systems using non-vision sensors, and b) systems using vision-based sensors. Acoustic, vibrational and other ambience sensor-based methods use the characteristic vibration patterns to detect different events. However, with these systems, the monitoring is limited to only those areas, where the sensors are installed. Moreover, it is usually assumed that there is only one subject performing the activities. Accelerometer-based fall detection systems [103, 104, 114, 115] employ wearable devices containing an accelerometer, and are simple and cost effective. Yet, even with multiple sensors, these systems are prone to creating false positives, especially when people are exposed to acceleration due to e.g. being in an elevator or high-speed car or train. The limitations of using just the accelerometer are also discussed by Wu et al. [116]. Thus, due to shortcomings of systems that only rely on acceleration and gyroscope data, more robust methods are needed to differentiate between falls and other regular daily activities.

There has been a lot of research work in the area of autonomous activity monitoring for the elderly in the last two decades. There are multiple methods being employed differing primarily based on the type of sensor being employed. In the literature, these methods are being classified into three broad categories: (1) Accelerometer-based approaches, (2) vision (static camera)-based approaches, (3) acoustic, vibrational and other ambience sensor-based approaches [117][108]. Below, we provide an overview of the systems using vision-based and non-vision sensors.

3.2.1 Systems using non-vision sensors

Accelerometer-based systems use wearable devices containing an accelerometer, the output of which is used to detect a fall [118]. Using accelerometers has been one of the most popular approaches. Initial prototypes were designed for detecting falls in the elderly and were based on autonomous belt devices which detected impact of shock on the ground along with mercury tilt switches to detect a person lying on the floor [119][120]. Since then, a lot of work has been done in the area of accelerometry [107][121][122],tri-axial accelerometry [123][124][110], posture based [125] and many other techniques based on the fusion of the above systems [126]. A number of approaches have been proposed for minimizing the false-alarm rate, including watching for no-movement [127, 128] and statistical models [129].

Accelerometer-based systems are simple and cost effective. However, robustness and accuracy of such a system demand multiple sensors being placed at strategic positions on the body which can be inconvenient. Even with multiple sensors, these methods can still create false positive alarms, especially in environments such as high speed trains and elevators. Also, they may not always perform activity classification.

Acoustic, vibrational and other ambience sensor-based methods record major events such as walking or falling based on their characteristic vibration patterns. Alwan et al. [130] designed a floor-vibration based fall detector. Zhuang et al. [131] proposed a method using Gaussian mixture models on audio signals from a microphone. One of the limitations of these kinds of sensors is that the monitoring environment is confined to where the sensors are installed. In addition, these methods assume that the subject of interest is the only one performing these events.

3.2.2 Systems using vision-based sensors

Recent advances in camera technology together with efficient image processing algorithms have enabled researchers to consider vision-based systems as a viable option in activity monitoring. Vision-based methods involve processing images from one or more cameras monitoring a subject [132]. Most approaches use raw video data, while others address the concerns of privacy [133] by using infrared or contrast-detection cameras [134, 135]. Stereoscopic vision and 3D scene reconstruction are other variations that aim to increase system accuracy [136, 137].

Wu [124] showed that during a fall, vertical and horizontal speeds were three times higher than any other activity. Along the same lines, Rougier et al. [138] tracked head movements and change in human shape and applied appropriate thresholds to detect falls. Yu et al. [132] achieved 97.08% detection rate with their method where they detect foreground human body by background subtraction and classify different projection histogram-based postures using Support Vector Machines (SVM). Belbachir et al. [139][140] presented an event-driven, stereo vision-based system for ambient monitoring. Other works include systems based on spatiotemporal feature analysis [141], shape change analysis [142], posture analysis [125] and 3D head position analysis [143].

There have also been implementations of static camera-based algorithms on embedded platforms. Belbachir et al. [144] recently presented a Dynamic Visual Sensor (DVS)-based system consisting of two optical sensors with 304×240 event-driven pixels and an FPGA for the processing. Fleck et al. [145] presented a distribute camera network for assisted living using FPGA and PowerPC based smart cameras.

In all the aforementioned methods, cameras are static at fixed locations, thus introducing the issue of confining the monitoring environment to the region where the cameras are installed. The images acquired from the cameras are usually offloaded to a dedicated central processor. Also, 3D model-based techniques require initializations and are not always robust. Another major practical issue is that the subjects being monitored often raise privacy concerns [133] as they feel they are being watched all the time.

3.2.3 Differences of the proposed method

In this chapter, we present a novel and efficient method to detect falls, and classify events of sitting and lying down by using a wearable smart camera, which is a new approach. With this system, wherein the camera is worn by the subject, we address the issues discussed above. First, since the camera is on the subject, contrary to other static sensor-based approaches, the monitoring is not limited to a confined area, and can cover wherever the subject may travel. Second, since the images captured will not be of the subject, as opposed to static cameras watching the subject, the privacy issue is alleviated. Moreover, the frames are not transmitted to anywhere, but processed onboard by the microprocessor. *Only* when a fall occurs, an appropriate message can be sent wirelessly to the emergency response personnel, optionally including an image from the subject's camera. This image of the surroundings can aid in locating the subject. Third, the captured images carry a lot of information about the surroundings that the other type of sensors cannot provide.

The proposed approach is based on the oriented image gradients, and there are major differences from the Histogram of Oriented Gradients (HOG) introduced by Dalal and Triggs [146]. First, we build separate histograms for gradient orientations and gradient strengths, and then find the correlation between them. Another difference is that we do not use a constant number of cells in a block. We adaptively determine the cells that do not contribute to overall edge information, and remove them autonomously. As will be shown by experimental results, the proposed method is more robust compared to using fixed number of cells in detecting falls. We implemented this algorithm on a wearable embedded smart camera, which is a small, stand-alone, battery-operated unit. In addition to detecting falls, the proposed algorithm provides the ability to classify events of sitting and lying down using optical flow.

The initial steps and an earlier version of this work was presented in a conference paper [147], wherein the focus is only on fall detection, and non-fall activities are not classified. In this chapter, we extend this work significantly by (i) incorporating activity classification by optical flow, (ii) diagnosing the problematic case of camera occlusions and avoiding creating false positives, (iii) improving the fall detection algorithm, and (iv) conducting more extensive experiments with more subjects for different scenarios.

The rest of this chapter is organized as follows: The proposed method is described in detail in Section 3.3. The experimental results are presented in Section 5.4, and the chapter is concluded in Section 3.5.

3.3 Proposed Method

The proposed method is composed of two stages. The first stage involves detection of an *event*. In our case, an event can be one of the following: falling, sitting or lying down. Once an event is detected, the next stage is the classification of this event.

Histograms of oriented gradients (HOG) [146] are used as image feature descriptors in the proposed algorithm. According to the HOG algorithm [146], the image is divided into blocks and then each block is divided into n cells. The magnitude and orientation of the gradient for each pixel are calculated for generating histograms of strength and orientation. Different from the original HOG algorithm, for every cell, two separate m-bin histograms are built for the edge strength and orientation. The combination of n histograms forms the HOG descriptor, with the size of $m \times n$ entries. As will be described in Section 3.3.1, these descriptors are used to detect the occurrence of an *event*.

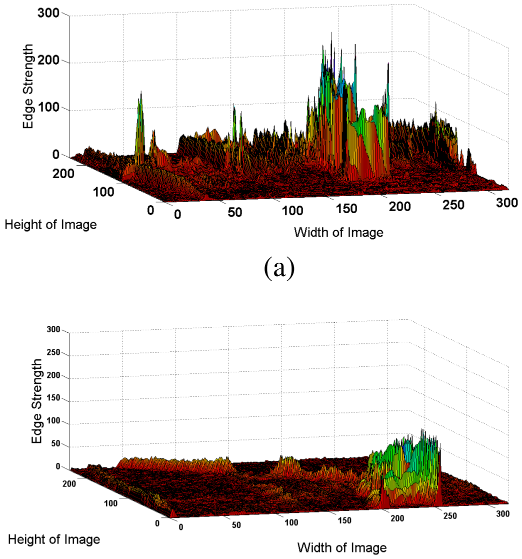
Once an event is detected, it is checked whether it is a *fall event*. If it is determined that it is not a fall event, an optical flow-based approach is used to classify this event as sitting or lying down. The complete algorithm is presented in Algorithm 1, and the details of event detection and classification are described in detail in Sections 3.3.1 and 3.3.4, respectively.

3.3.1 Event Detection

As stated above, and shown in Algorithm 1, the first step in our algorithm is to detect the occurrence of an *event*. To detect an event, *separate* histograms of edge orientation (EO) and edge strength (ES) are used. During a fall, edge orientations change significantly, which is reflected in the gradient orientation histograms. Also, since falls are relatively fast events, the edges in images get blurred as seen in Fig. 3.1(a) versus (b). This is captured by the change in the gradient strength histograms. The edge strength values corresponding to the frames given in Fig. 3.1(a) and 3.1(b) are shown in Fig. 3.2(a) and Fig. 3.2(b), respectively. As can be seen, during the fall the edge strength values decrease significantly.

(a) (b)

Fig. 3.1: Example frames captured by the camera during a fall.



(a)

(b)

Fig. 3.2: Edge strength values corresponding to frames in
(a) Fig. 2.1(a), and (b) Fig. 2.2(b).

We have seen in our experiments that using original HOG can create false positives while walking. In addition, we do not use a fixed number of cells in each block. Rather, we adaptively determine the cells that do not contribute to overall edge information, and remove them autonomously.

We have determined that for the detection of abrupt changes, a reduced number of blocks is sufficient. In order to lighten the processing load of the embedded camera, our implementation only uses one block that is initially divided into 16 cells, as including larger number of blocks would unnecessarily compromise the efficiency. We also adaptively change the number of cells in the block so that the cells that do not contribute to overall edge information are autonomously removed. The details will be described in Section 3.3.3.

To build the histograms, horizontal (dx) and vertical (dy) gradients are computed first for every pixel within a cell. Then, these values are used to calculate the gradient orientation ($tan^{-1}(dy/dx)$) and the gradient strength ($\sqrt{dx^2 + dy^2}$) at each pixel.

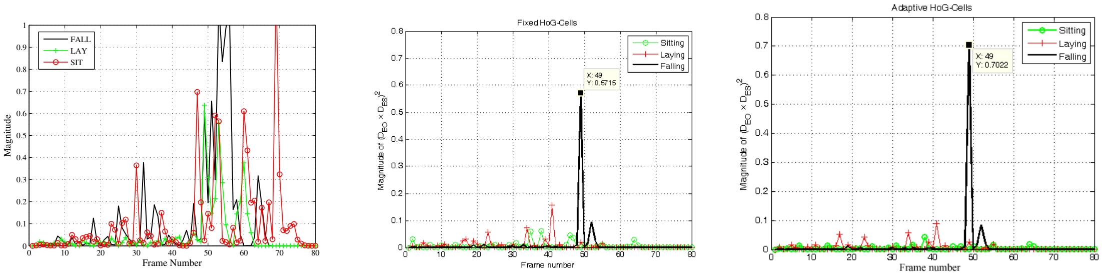


Fig. 3.3: (a) False ‘fall’ alarms are generated during lying down and sitting events when using (a) the original HOG; (b) proposed approach with fixed number of cells; (c) proposed approach with adaptive number of cells.

In the original HOG algorithm, the orientation values are placed in a 9-bin histogram (with range $0°$ to $180°$) using a voting mechanism based on the gradient strength. This causes false alarms in some cases. An example is shown in Fig. 3.3(a), where ‘lying down’ and ‘sitting’ were classified as a fall with the original HOG. Another example is seen in Fig. 3.4, where the fall occurs between frames 50 and 60, yet walking triggers a false ‘fall’ alarm a little after frame 30.

In the proposed method, we also use 9 bins for our EO histogram as in [146]. On the other hand, we use 18 bins for the ES histograms, which makes them more descriptive. This is important especially when we consider the fact that a fall event involves changes in edge strengths. The histograms from all the kept cells are concatenated to form a multi-dimensional vector. Thus, the descriptor for a frame consists of two vectors: one containing the concatenated histograms for edge orientations (EO) and another containing the concatenated histograms for edge strengths (ES).

3.3.2 Dissimilarity Distance

Once the extracted feature histograms EO and ES are normalized, the dissimilarity distance between two histograms (r and s) is calculated using:

$$D = 1 - \frac{\sum_{i=0}^{N-1} (r_i - \bar{r})(s_i - \bar{s})}{\sqrt{\left[\sum_{i=0}^{N-1} (r_i - \bar{r})^2 \sum_{i=0}^{N-1} (s_i - \bar{s})^2\right]}}, \tag{3.1}$$

$$\bar{r} = \frac{1}{N} \sum_{i=0}^{N-1} (r_i) \quad , \quad \bar{s} = \frac{1}{N} \sum_{i=0}^{N-1} (s_i)$$

Dissimilarity distance values for ES (D_{ES}) and EO (D_{EO}) are cross-correlated, which attenuates the noise in the signal and emphasizes the peaks. To increase the robustness, the attenuated signal is autocorrelated ($d = (D_{ES}D_{EO})^2$). The result of this operation is shown in Fig. 3.3 and Fig. 3.4.

Once D_{ES} and D_{EO} are cross-correlated, followed by autocorrelation of the resulting signal, the gradual motion of the subject (i.e. walking, lying, sitting) is significantly attenuated, which provides a clear peak corresponding to actual events.

To detect an event, we compute these distance values, and buffer them in an array of $B_\triangle$ for the last $\triangle$ frames such that $B_\triangle$ holds $\triangle$-many $d_{t-\triangle}^{t}$ values, where $\triangle$ is an integer, and $d_{t-\triangle}^{t}$ is computed between the current frame t and the frame $t - \triangle$. If the maximum computed distance in the buffer $B_\triangle$ is larger than the empirically determined threshold ρ, this indicates the occurrence of an *event*. Searching over a one-second window first and using $\triangle$, rather than directly comparing current and previous frames, eliminates potential false positives that could be caused by sudden changes between consecutive frames as a result of sudden illumination changes, camera occlusion etc. Hence, an event is detected first, before declaring a *fall*.

3.3.3 Adaptive Number of Cells

We also propose a mechanism that adaptively controls the number of cells to be used for the feature descriptor according to their content. The motivation is that cells containing no edges or edges with low strength do not contribute to the scene information, and increase the similarity score between concatenated histograms. Figure 3.5 illustrates a scenario for a fall event, wherein the camera points to a table. As can be seen, cells $1, 2, 5, 6, 9, 10, 13$, and 14 add minimal or no useful information to detect the fall or differentiate it from walking. Including the histograms for these cells in the feature descriptor would result in lower dissimilarity scores.

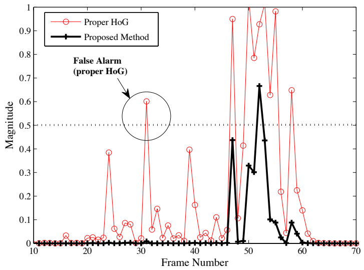


Fig. 3.4: False ‘fall’ alarm when using the original HOG.

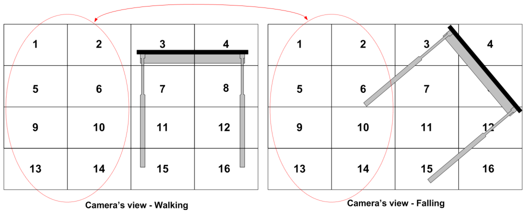


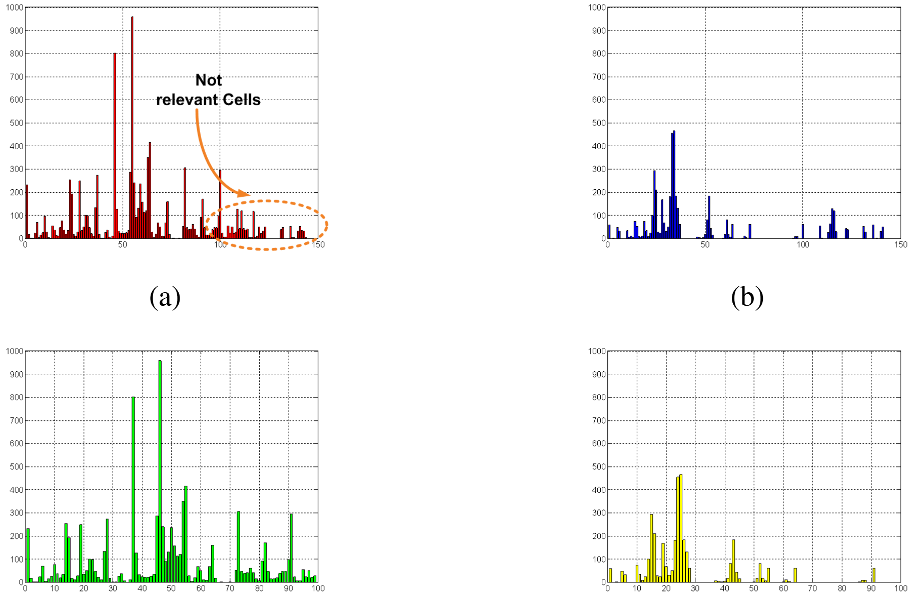
Fig. 3.5: Cells before and after a fall event.

Figures 3.6a and 3.6b are the histograms of edge orientations before and after a fall, respectively, obtained by using a fixed number of cells. The adjusted histograms obtained by removing the least contributing cells with the proposed method are shown in Fig. 3.6c and 3.6d. The dissimilarity distance between the histograms in Fig. 3.6a and 3.6b is 0.866. On the other hand, if we remove the histograms of cells with the least contribution (circled in Fig 3.6a) from the feature vector, the dissimilarity distance increases to 1.024.

Another supporting example can be seen by comparing Fig. 3.3b and 3.3c. The amplitude of the peak for dissimilarity in a falling event is higher when using an adaptive number of cells

(Fig. 3.3c). Having a higher dissimilarity distance between falling and previous states contributes to the robustness of the system to reliably detect fall events. Consequently, the system is less prone to false negatives, i.e. missing ‘fall’ events. More results comparing adaptive number of cells with fixed number of cells are presented in Sec. 5.4.

To determine which cells to remove, the maximum amplitude among the bins within a cell is found first. Then, we calculate the mean value and the standard deviation of the vector of maximums from the n cells in a frame. Finally, the algorithm removes the cells whose maximum value are α standard deviation away from the computed mean. Thus, not only the number of removed cells is adaptive, but also the threshold is adapted according to the cell content within current frame at time t. To avoid possible introduction of false positives by discarding too many cells, the algorithm is designed to remove a maximum of 8 cells (half of the total number of cells).



(a)

(b)

(c)

(d)

Fig. 3.6: Histogram of Edge Orientations using a fixed number of cells (a) before falling and (b) after falling. Employing adaptive number of cells (c) before falling and (d) after falling.

3.3.4 Event Classification

As described above and seen in Algorithm 1, for every frame, it is checked whether there is an event or not. When there is an event occurring, the algorithm first checks whether the fall condition is satisfied. If the event is not a fall, it performs optical flow calculations to classify the event as sitting or lying down. Fall detection part is more critical and has higher priority when compared to classification of sitting and lying down.

3.3.5 Fall detection using modified HOG

To detect a fall, the dissimilarity $d_{t-1}^{t}$is calculated between current and previous frames. If it is greater than the threshold τ, a *fall event* is detected, and fall alarm is created. Figure 3.7 shows plots of different distances for a typical falling event. Looking at the maximum of dissimilarity distances over $\triangle$ frames (solid red plot) helps detecting events when they are in progress. After the occurrence of an event is detected, the 'fall' is recognized by using the dissimilarity distance between the current and the previous frames (solid blue plot). Whenever the distance is greater than the threshold τ that is shown in the graph, it is declared as a fall.

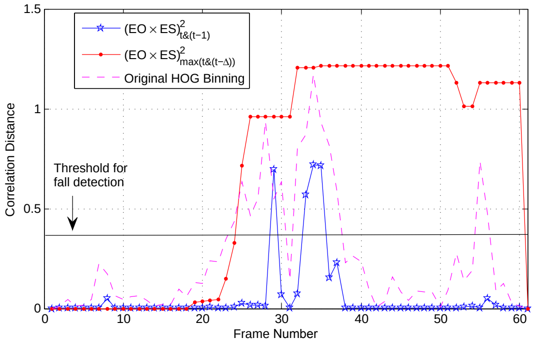


Fig. 3.7: Plots of different distances for a typical falling down.

As described above, we build two separate histograms, and thus build descriptors in a different way compared to the original HOG algorithm [146]. The advantage of this approach over the original HOG can also be seen in Fig. 3.7, where the dissimilarity distances, between the current and previous frames, obtained with the original HOG (dashed plot) and the separate EO and ES histograms (solid blue plot) are plotted. As can be seen, the original HOG creates a false positive (declaring a fall event when there is not an actual fall) between frames 50 and 60.

3.3.6 Event classification using Optical Flow

According to Horn and Schunck [148], optical flow is the distribution of apparent velocities of movement of brightness patterns in an image, and it can arise from relative motion of objects and the camera. Four general scenarios determine the relative motion between an object and a camera [149]: (a) relative object at a distance, (b) relative object motion towards the camera, (c) relative object rotation at a distance, (d) relative object rotation about its' axis. All other relative motions can be derived from these. Figure 3.8 gives a pictorial description of the four different motions. Since the motion is relative, the same principle applies if the objects in the background are still and the camera moves in and around the scene. Our general methodology for event classification in this chapter is based on this concept. Any horizontal, vertical or rotatory motion of the camera would generate significant respective velocity vector components. By splitting these vector components into horizontal and vertical velocities and by studying their characteristic behavior, certain decisions can be made on the event that the subject is performing with the camera attached to her/his waist.

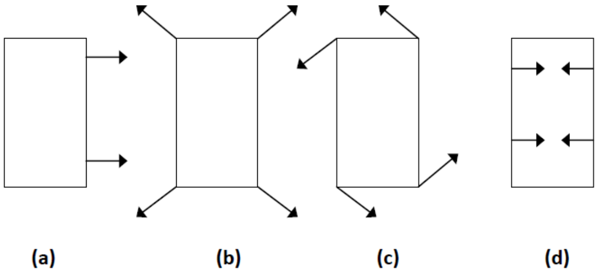


Fig. 3.8: (a) Relative object motion at a distance, (b) relative object motion towards the camera, (c) relative object rotation at a distance, (d) relative object rotation about its' axis.

When the detected event is not a fall event, the algorithm computes the average optical flow over γ consecutive frames, for vertical and horizontal directions, using the method introduced in [150]. Optical flow vectors help us differentiate the events of sitting and lying down. As seen in Fig. 3.9 and 3.10, taking the average of vertical and horizontal velocities over γ frames gives us distinctive features of events. For a typical sitting event, the event starts and continues with vertical direction. As it can be observed in Fig. 3.9, sitting down event is detected when the vertical mean of optical vectors is greater than the horizontal mean. Similarly, if the horizontal mean of optical vectors is greater than the vertical mean, it will be detected as a lying down event. When the vertical mean is greater, the difference between the vertical and horizontal values is added to the variable $vrbl_Sit$. Similarly, when the horizontal mean is greater, the difference is added to the variable $vrbl_Lay$. Since a typical lying down event usually starts with sitting down, it is expected the horizontal average vector to become greater than the vertical one during the course of an event. This cross-over of the velocities is described with the variable $crossOver$ when the horizontal flow becomes greater than the vertical flow and vice versa. Therefore, if $crossOver$ condition is satisfied, and the horizontal mean is significantly bigger than the vertical one for a predetermined duration of κ frames, and $vrbl_Lay$ is greater than φ, the event is classified as a lying down as seen in Fig. 3.10. If the $crossOver$ condition is not satisfied, the algorithm compares the variables $vrbl_Sit$ and $vrbl_Lay$ to distinguish sitting and lying down events. The algorithm decides that it is a sitting or lying down event according to $vrbl_Sit$ and $vrbl_Lay$, whichever is larger respectively.

Furthermore, although it is not included in the algorithm, instead of taking the absolute values, if we use the direction of the vertical flow vectors, we can also decide whether it is a transition from sitting to standing or vice versa.

For optical flow computation, the entire image is used instead of using the cells that contain significant edge information. The reason is that it is more in alignment with the algorithm proposed in [150] in terms of computational performance.

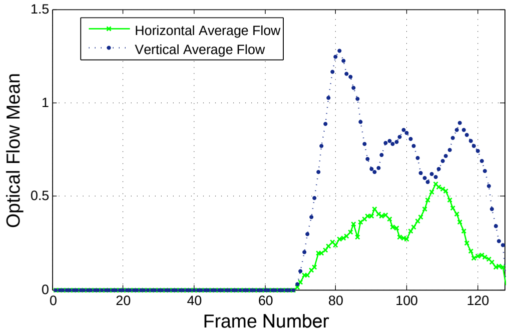


Fig. 3.9: Horizontal and vertical mean values for a sitting down event.

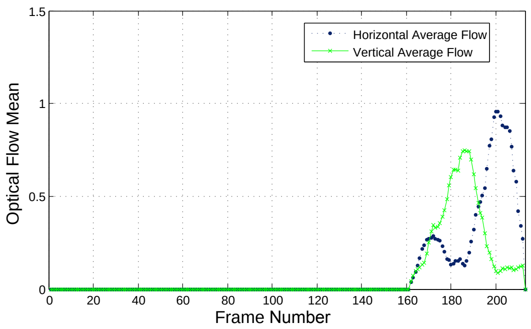


Fig. 3.10: Horizontal and vertical mean values for a lying down event.

3.4 Experimental Results

In order to verify the robustness of the algorithm, various experiments have been performed, which include: (i) trials wherein camera is mounted on a broom stick to imitate an actual free fall, (ii) experiments wherein videos are captured with a camera attached at the waist of different subjects, and then later processed on a PC, (iii) embedded smart camera experiments wherein images are captured and then locally processed in real-time on the microprocessor of the CITRIC [4] camera board.

3.4.1 Sensitivity and Specificity Comparison

For the evaluation of the experiments, sensitivity and specificity measures described by Noury et al. [107] are employed. Sensitivity is the portion of the falls that are detected as falls with the proposed algorithm. Specificity is the portion of the non-fall events (sitting and lying down) that does not create any false alarms according to the algorithm. *Sensitivity* and *Specificity* are given by:

$$\begin{aligned} Sensitivity &= \frac{TP}{TP+FN} \\ Specificity &= \frac{TN}{TN+FP}, \end{aligned} \tag{3.2}$$

where True Positive (TP) is the system detecting a fall when it occurred, False Positive (FP) is the system detecting a fall when it did not occur, True Negative (TN) is system not detecting a fall when fall does not occur, and False Negative (FN) is system not detecting a fall when it occurs.

3.4.2 Experiments with free fall of a broomstick

It is very difficult to recreate free falls with stunt actors since cautiousness and fear of subjects interfere with the falls. Therefore, to test the proposed algorithm with actual free falls, the camera was attached to a broomstick. It was held vertically, and then released to hit the ground in a free fall. In order to simulate the free fall a human, the broomstick was chosen so that its' length is similar to the average human height.

15 free fall trials were performed in total, and the fall detection rate is 100%, which proves the robustness of the developed method to detect free falls. As shown in Fig. 3.11, free fall of a broomstick creates a fall alarm between frames 35 and 45. A free fall event has more acceleration when compared to falls acted by subjects. For this reason, maximum dissimilarity distance over δ frames is higher with a free fall when compared with the distance obtained during acted falls (Fig. 3.7). Example captured frames obtained during free falls are shown in Fig. 3.12. Due to the acceleration gained during a free fall, a significant change in edge strengths is observed, caused by the blurriness of the images captured during movement.

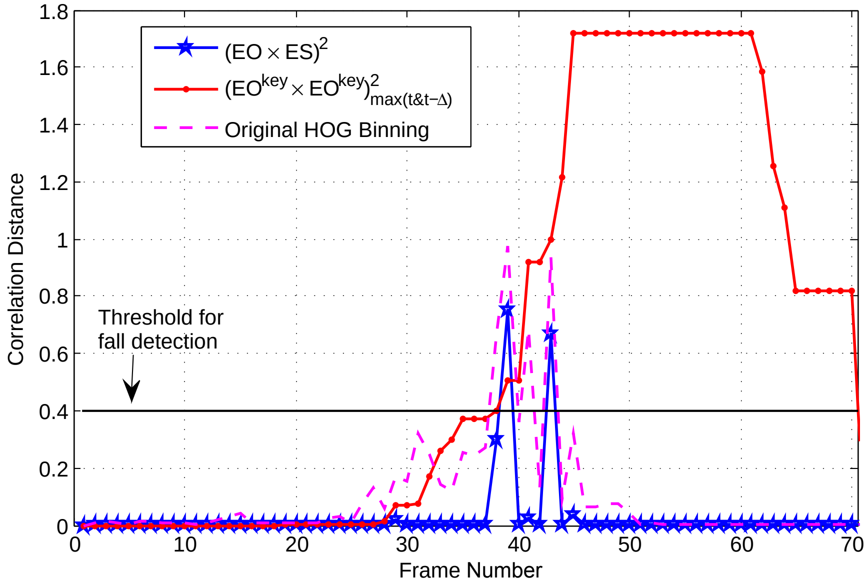
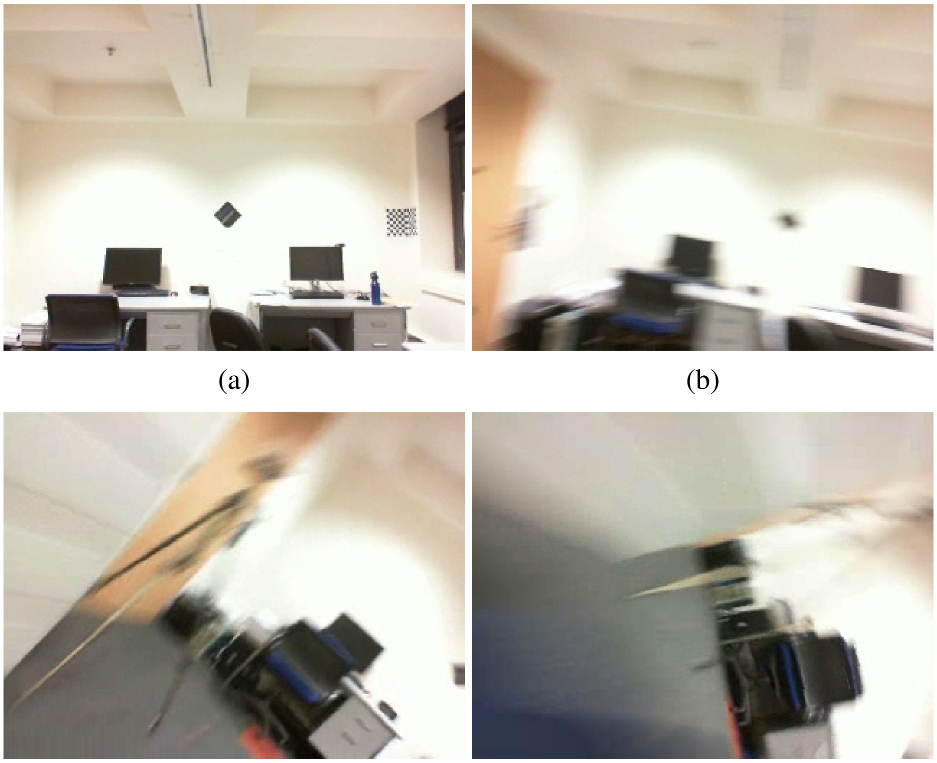


Fig. 3.11: Free fall of a broomstick is detected.



(a) (b)

(c) (d)

Fig. 3.12: Example frames captured during a free fall.

3.4.3 Experiments with people

Table 3.1: Falls from standing position : Sensitivity and Specificity using different methods

	Proposed Method		Fixed-cell Modified Histogram		Original HOG [146]	
	Sensitivity	**Specificity**	**Sensitivity**	**Specificity**	**Sensitivity**	**Specificity**
Subject 1	100 %	95.24 %	100 %	95.24 %	100%	61.84 %
Subject 2	100 %	90.48 %	90.91 %	100 %	100%	66.66%
Subject 3	100 %	76.19 %	100 %	76.19 %	100 %	38.1 %
Subject 4	90.91 %	80.95 %	72.73 %	90.48 %	100%	50%
Subject 5	90 %	100 %	70 %	100 %	100%	40%
Subject 6	81.82 %	85 %	63.64 %	90 %	100%	5%
Subject 7	80 %	100 %	70 %	100 %	100%	72.73 %
Subject 8	60 %	85 %	50 %	95 %	100%	50%

All the experiments were performed with the camera attached to the belt around the waist. The cameras were positioned to be at the center of the waist facing front. It should be noted that, in these experiments, the subjects are imitating or acting the events. Thus, it is very difficult to recreate a free fall. The cautiousness and fear of subjects can sometimes interfere with imitating a fall event. Even with soft cushions and other safety precautions in place, we found that sometimes subjects are too afraid to 'actually fall'. In addition, since the experiments were repetitive at times, the performance and attention of the subjects could degrade over time.

The first set of experiments was performed with pre-recorded videos captured with eight different subjects. Video sequences were captured with a Microsoft® LifeCam™ camera that has a $CMOS$ sensor, and captures image frames at 30 fps. The captured image size is 320×240 pixels. To decrease processing time, we only processed the even-numbered frames. In order not to increase the computation load unnecessarily, we have used only one block and 16 cells in our implementation. The parameters of the algorithm are selected to be $\triangle = 17$, $\kappa = 2$, $\rho = 0.2$, $\tau = 0.37$, $\varphi = 0.2$, and $\xi = 0.9$. The same values have been used in all the experiments. $\triangle$ value is selected to be 17 to cover information from approximately last one second of the movement. κ is selected to be 2 to have reliable classification without being effected from sudden changes of the observed parameters. φ and ξ are determined experimentally to effectively detect lying and sitting down, respectively. ρ and τ are empirically determined correlation distance thresholds.

3.4.4 Fall Experiments

8 different subjects performed 10 fall events from standing up position and 10 fall events from sitting position. Thus, a total of 80 falls from standing position and 80 falls from sitting positions were performed. Table 3.1 summarizes the sensitivity and specificity values for falls from standing position with (i) the proposed method with adaptive number of cells, (ii) when using fixed number of cells, (iii) when using original HOG [146]. As can be seen, the proposed method with the adaptive number of cells provides the best sensitivity-specificity combinations among all methods.

Table 3.2 shows the Sensitivity and Specificity values for falls from sitting position. Falling from sitting position is more challenging when compared to falling from standing position. Amount of motion and distance traveled to hit to the floor is less when falling from sitting position. Hence, for some subjects falling from sitting position does not create dissimilarity distance that is high enough to be classified as falling down. Therefore, the detection rate occurred to be less than the sensitivity of falling from standing position. In table 3.2, specificity rates are calculated with sitting and lying down experiments.

Table 3.2: Falls from Sitting Position: Sensitivity and Specificity

	Falling From Sitting	
	Sensitivity	**Specificity**
Subject 1	100 %	95.24 %
Subject 2	100 %	90.48 %
Subject 3	80 %	76.19 %
Subject 4	70 %	80.95 %
Subject 5	44.44 %	100 %
Subject 6	50 %	85 %
Subject 7	27.27 %	100 %

3.4.5 Sitting and Lying down Classification Experiments

Classification experiments were performed on 5 subjects with 10 trials of sitting and 10 trials of lying down. Classification results for lying down and sitting events are given in Table 3.3. As it can be seen from the table, we achieved classification rates of 82.7% and 86.8% for lying down and sitting, respectively.

Table 3.3: Classification rate on 105 event trials

	Correct Classification	**Wrong Classification**	**True Positive**
Lying Down	43	9	82.7 %
Sitting Down	46	7	86.8 %

The average processing time for the fall detection part of the algorithm is less than 0.1 sec on a computer with Intel® Core™ i7-3630QM at 2.4 GHz and 6 GB of RAM. When there is an event detected, and it is not a fall event, it requires 0.7 sec on the average to process a frame including optical flow computation. As will be presented below, we also implemented the fall detection part of the algorithm on the microprocessor of an embedded smart camera, specifically a CITRIC mote.

Example of captured frames for a sitting down event is given in Fig. 3.13. The given frames are not consecutive, but they summarize the movement by showing some key frames of the whole event. As it can be observed, sitting is an activity consisting of vertical movement of the camera with respect to the scene.

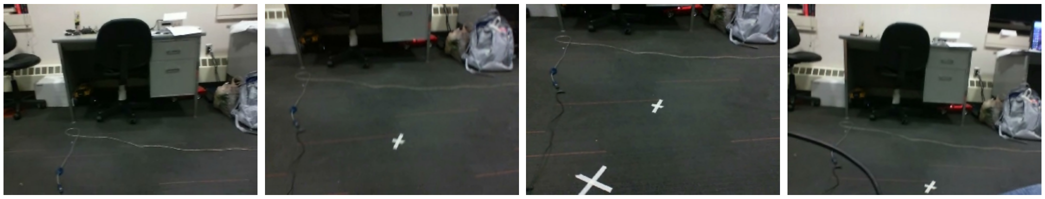

(a) (b) (c) (d)

Fig. 3.13: Example frames captured during a sitting down event.

Some of the key frames, which are not successive, corresponding to a lying down event are given in Fig. 3.14. As it can be observed, the lying down event starts with sitting down first. It should be noted that the algorithm is also capable of detecting a lying down event for the person who is already in a sitting position. We can detect sitting to lying transition that can also be observed from the given frames.

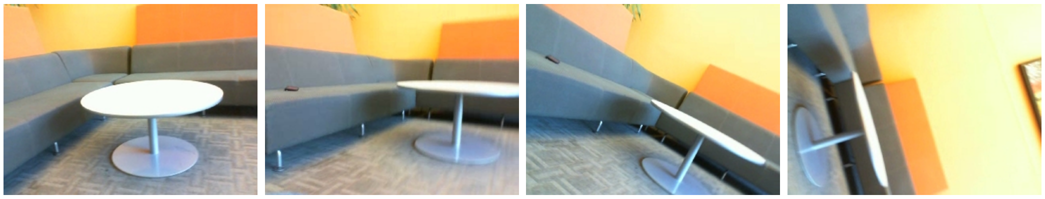

(a) (b) (c) (d)

Fig. 3.14: Example frames captured during a lying down event.

Example frames captured during fall events are given in Fig. 3.15 and 3.16. As it can be seen, falling causes a sudden change in the view, which may introduce blurriness for the captured frames. As a result of the sudden change in the edge orientation and strengths, we are able to achieve very high fall detection rates.

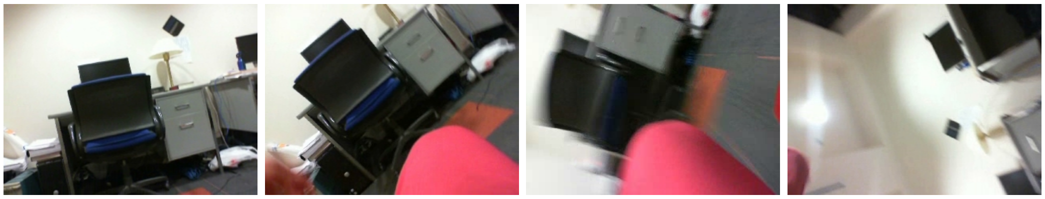
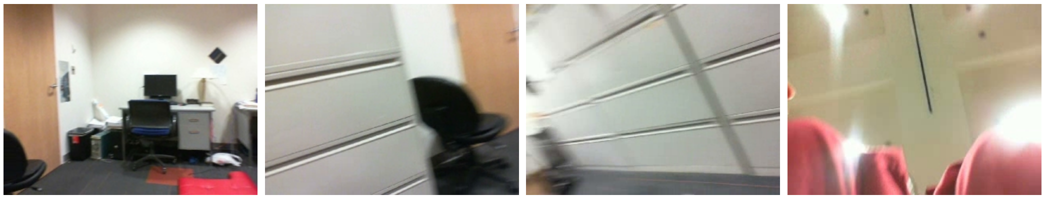

(a) (b) (c) (d)

Fig. 3.15: Example frames captured during a fall from sitting.

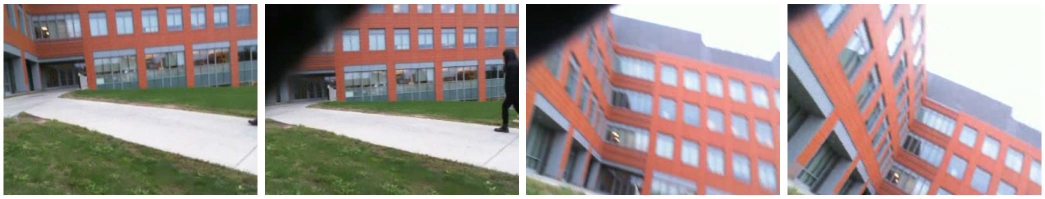


(a) (b) (c) (d)

Fig. 3.16: Example frames captured during a fall from standing.

3.4.6 Outdoor experiments

Proposed system was tested outdoors as well. As it can be seen in Fig. 3.17 (showing frames in time order), when a fall occurs, the scene observed by the camera changes significantly making it possible for the algorithm to detect falls. If the subject is lying on his back towards the end of the fall, the dissimilarity distance becomes very high as compared to a typical fall in indoor environments. One of the main reasons is that the camera is facing towards the bright sky at the end, which creates considerably higher dissimilarity distance. Dissimilarity distances during a fall event are shown in Fig. 3.18. As it can be seen, the correlation distance between the current frame and the previous one (blue plot) reaches 1, which does not usually happen for a typical fall in indoor environments.



(a) (b) (c) (d)

Fig. 3.17: Example frames captured during a fall in outdoor scene.

3.4.7 Embedded camera experiments with CITRIC motes

Fall detection part of the algorithm was implemented on CITRIC embedded camera platform [4], which features a 624-MHz fixed-point microprocessor, 64 MB SDRAM, and 16 MB NOR FLASH. The wireless transmission of data is performed by a Crossbow TelosB mote. The images are processed locally onboard, and then dropped, thus are not transferred anywhere. When a fall is detected as a result of the processing on CITRIC, only the corresponding fall alarm and some of the captured

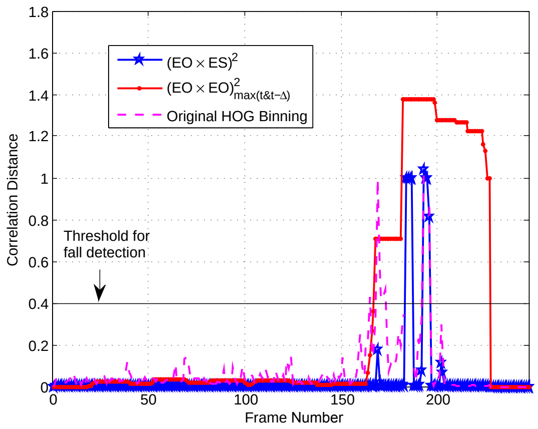


Fig. 3.18: Outdoor scene dissimilarity distances of the proposed method and original HOG.

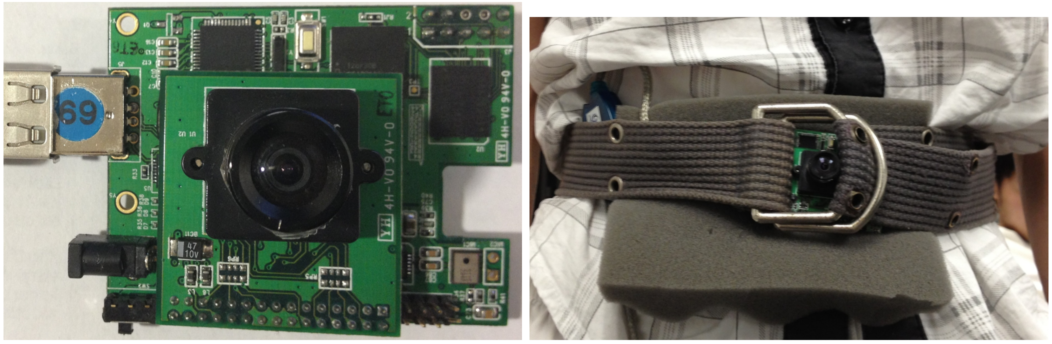
frames during the fall event may be sent wirelessly to emergency response personnel to locate the subject more easily according to images of the environment.

Figure 3.19(a) and (b) show the CITRIC camera and the camera attached at the belt, respectively. All of the subjects wore the camera at the waist level. With the camera facing forward at the waist, captured images provide effective information about the environment. Also, the location of the camera does not interfere with the movements of the subjects.

The results of the experiments are presented in Table 3.4. Experimental trials have been performed with three different subjects, and include 50 falling down, 15 sitting down along with getting up and 15 lying down events. The detection rate of fall trials was 84-86%. False positives (wrongly created 'fall' alarms) caused by sitting down and standing up, and lying down are shown in the corresponding columns. Number of false positives caused by lying down events is higher when compared to sitting down. Since sitting consists of straight motion of going down and up, it is less prone to create significant distance between frames when compared to lying down. False positive rates due to lying down did not exceed the rate of 2/15 in the experiments. With the implementation on the CITRIC embedded platform, the proposed algorithm gives promising results.

The program running on the embedded platform includes event detection, fall detection and camera occlusion detection. Since optical flow calculations are computationally more expensive, event classification part is not implemented on the embedded platform yet.

Due to the design of CITRIC camera, exposure adjustment is done only once before the program



(a) (b)

Fig. 3.19: (a) CITRIC platform, (b) which is connected around the waist.

Table 3.4: Sensitivity and False Positive with sitting and lying down on CITRIC platform

	Falling Down		Sitting Down Standing Up	Lying Down
	Sensitivity	**False Positive**	**False Positive**	**False Positive**
Subject 1	43/50	1/50	2/30	2/15
Subject 2	45/50	5/50	1/30	1/15
Subject 3	42/50	0/50	0/30	2/15

starts. Since exposure adjustment is not performed periodically, the algorithm performs well when the lighting intensity is similar during the trial. However, when the person, who is wearing the camera, changes room or opens a door, it may create a false alarm due to a sudden change in lighting.

3.5 Conclusion

We have presented a novel algorithm to detect fall events and classify significant activities like sitting and lying down by a wearable camera. Fall detection employs histograms of edge orientations and strengths, and an optical flow-based method is used for activity classification.

Since the camera is worn by the subject, the monitoring can continue wherever the subject may travel including outdoors. Wearable camera alleviates the privacy concerns, since the captured images are not of the subject, and these images are neither stored nor transmitted. Only if a fall occurs, an alarm signal may be sent to the emergency response personnel with the option to send the image frames during and after a fall. The images of surroundings may help the emergency personnel to locate the subject.

We have also implemented the fall detection part of the algorithm on an actual smart camera. Over 320 trials, performed with eight different subjects, demonstrate the effectiveness of the algorithm in detecting falls and classifying activities of sitting and lying down. Additionally, 50 falls and 30 non-fall trials were performed with subjects wearing actual embedded smart cameras, and 15 trials were performed to imitate free falls.

For the falls starting from a standing up position, an average detection rate of 87.84% has been achieved with pre-recorded videos. With the embedded camera implementation, the fall detection rate is 86.66%. Moreover, the correct classification rates for the events of sitting and lying down are 86.8% and 82.7%, respectively.

The idea of applying optical flow can also be extended to classify other types of human activities.

Algorithm 1 Fall detection and Activity Classification

```
for For all frames do
    if It is the first frame then
        Initialize histogram and cell vectors
    else
        if average intensity $\leq 30$ then
            Camera occlusion detected.
        else
            if $max(B_{\triangle}) \geq \rho$ then
                Event Detected
                if $d^{t}_{t-1} \geq \tau$ then
                    Fall Detected
                else
                    Over $10$ frames:
                    $\rightarrow$ Find average vertical flow$= \alpha$
                    $\rightarrow$ Find average horizontal flow$= \beta$
                    if $crossOver \geq \kappa$ then
                        if $vrbl\_Lay > \varphi$ then
                            Lying down detected
                        else if $vrbl\_Sit > \xi$ then
                            Sitting down detected
                        end if
                    else
                        if $vrbl\_Sit \geq vrbl\_Lay$ then
                            Sitting down detected
                        else if $vrbl\_Lay \geq vrbl\_Sit$ then
                            Lying down detected
                        end if
                    end if
                end if
            end if
        end if
        if $\alpha > \beta$ then
            $vrbl\_Sit+ = (\alpha - \beta)$
        else
            $vrbl\_Lay+ = (\beta - \alpha)$
        end if
    end if
end for
```

CHAPTER 4

WEARABLE CAMERA- AND ACCELEROMETER-BASED FALL DETECTION ON PORTABLE DEVICES

4.1 Introduction

Images captured by a camera sensor provide abundance of data including contextual information about the surroundings. In this chapter, we propose a system that employs a novel approach of using both accelerometer and camera modalities to detect falls by differentiating them from other daily activities including walking, sitting, lying down and going up and down stairs. This is one of the first works that uses data from a *wearable camera* to overcome shortcomings of accelerometer-only systems.

Wearable cameras alleviate, if not eliminate, privacy concerns of users since the captured images are not of the subjects but the surroundings. Also, with smart phone implementation, images are processed locally *on the device*, and they are not saved or transmitted anywhere. Also, a study about privacy behaviors of lifeloggers using wearable cameras, discusses privacy of bystanders and ways to mitigate concerns [111]. It is also expected that *wearable cameras* will be employed more to understand lifestyle behaviors for health purposes [72].

For the camera-based part, the proposed method employs histograms of edge orientations together with the gradient local binary patterns (GLBP), which use image features that have more descriptive power [151]. GLBPs have been used for human detection applications, and were derived from an operator named local binary pattern [152]. We show that the novel camera-based fall detection method proposed in this chapter is more robust, and outperforms our previous work that uses only histograms of edge strengths and edge orientations [27]. Moreover, we present an accelerometer-based fall detection algorithm and a fusion approach to combine results from these two sensors. We first present the significant improvement provided by the proposed camera-based algorithm on recorded videos. We also show results on actual smart phones with a simpler version of the camera-based algorithm for real-time performance. The fusion of camera-based results with accelerometer data provides significant decrease in the number of false positives compared to using only accelerometer data.

4.2 Proposed Method

4.2.1 Summary of HOG and Modified HOG

In the original HOG-based algorithm, proposed for human detection, an image is divided into blocks and then each block is divided into n cells. For each cell, an m-bin histogram is built, wherein each bin corresponds to a gradient orientation span. The concatenation of n histograms forms the HOG descriptor for a block, with $m \times n$ bins. For each pixel in a cell, the intensity gradient magnitude and orientations are calculated. Each gradient has a vote in its' bin, which is its' magnitude. Then, block-based normalization is applied.

In [27], we proposed a modified HOG algorithm for fall detection, wherein, different from original HOG, separate histograms are constructed for edge strength (ES) and edge orientations (EO). The edge orientation range is between 0 and 180 degrees and it is equally divided into nine bins. The edge strength histogram contains 18 bins. Moreover, the cells that do not contain significant edge information are excluded from the descriptor in this modified HOG algorithm.

4.2.2 Gradient Local Binary Pattern Features

The computation flow of GLBP is illustrated in Fig. 5.1. For each center pixel, its' eight neighboring pixels are checked. A value of '1' or '0' is assigned to a neighboring pixel if its' intensity value is greater or less than the center pixel, respectively. This results in an 8-bit binary number. Only the sequences that have a maximum of 2 transitions (from 0 to 1 or 1 to 0) are kept, and the others, including all 0 and all 1 sequences, are considered as noise. In accordance with [152], we employ 58 uniform patterns out of 256 possible patterns. Then, we analyze the 8-bit sequence to find the length of longest consecutive sequence of 1s and the angle of the edge, as illustrated in Fig. 5.1. Then, these values provide the index of the entry of the 7×8 matrix, which is incremented by the edge strength value. This matrix is filled by visiting each pixel in a cell, and then normalized. This results in a 56-dimensional GLBP feature that is used in our algorithm. We use one block divided into 16 cells, therefore our concatenated GLBP vector for one frame is of length 16×56. After calculating the GLBP feature for each cell, L2 normalization is applied before concatenation.

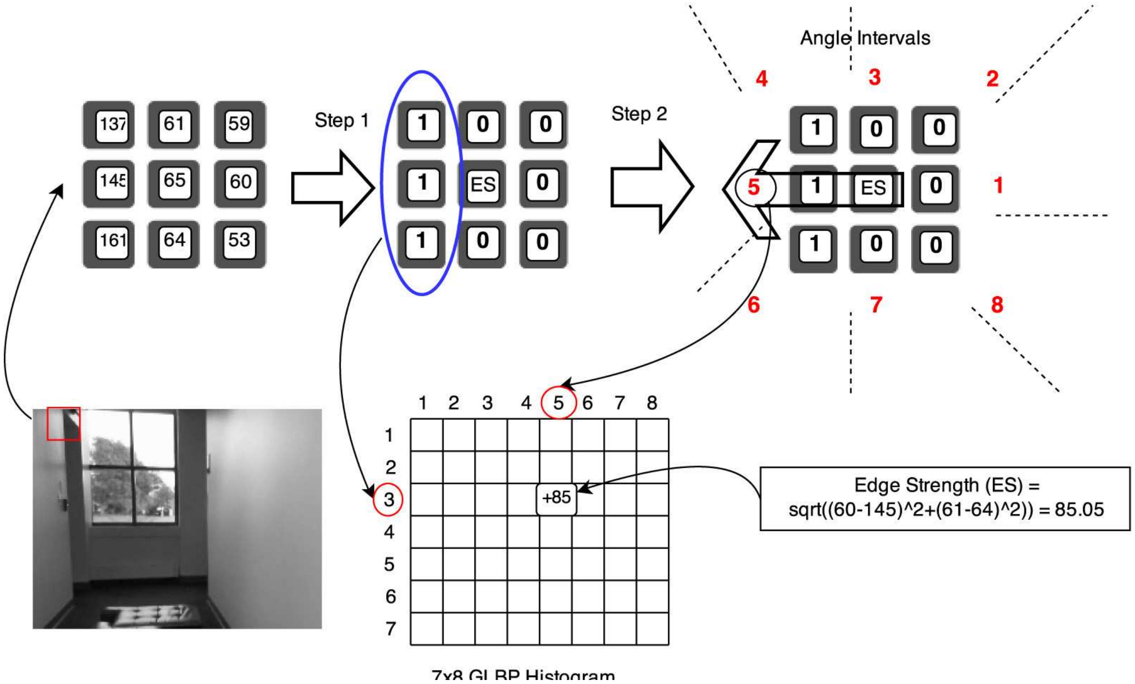


Fig. 4.1: GLBP histogram generation steps.

4.2.3 Camera-based Detection

The proposed camera-based fall detection algorithm consists of two stages. The first stage detects a significant *event*, which could be caused by sitting down, lying down or falling down etc. Once an event is detected, the second stage of the algorithm is employed to detect whether it is a fall or not.

We employ a combination of edge orientation (EO) histograms and GLBP features. To compute EO histograms and GLBP features, we use one block divided into 16 cells. This not only decreases the computational load, but also is sufficient to detect abrupt changes. Moreover, the cells that do not contain significant edge information are removed autonomously and adaptively. To determine which cells to remove, the maximum amplitude among the bins within a cell is found first. Then, we calculate the mean and the standard deviation of the vector of maximums from the n cells in a frame. The algorithm removes the cells whose maximum value are α standard deviation away from the computed mean.

To build the EO histogram, horizontal (dx) and vertical (dy) intensity gradients are computed for every pixel within a cell. Then, these values are used to compute the gradient strength ($\sqrt{dx^2 + dy^2}$) and orientation ($tan^{-1}(dy/dx)$). We use 9 bins for each of the 16 cells, and then concatenate 16-many 9-bin histograms to obtain a 9×16-dimensional vector. We obtain the GLBP descriptor as described in Section 4.2.2. After the feature descriptors (EO and GLBP) are normalized, the dissimilarity distance is used to compute D_{EO} and D_{GLBP} between two frames. The dissimilarity distance between two N-dimensional vectors (r and s) is calculated using:

$$D = 1 - \frac{\sum_{i=0}^{N-1} (r_i - \bar{r})(s_i - \bar{s})}{\sqrt{\left[\sum_{i=0}^{N-1} (r_i - \bar{r})^2 \sum_{i=0}^{N-1} (s_i - \bar{s})^2\right]}}, \tag{4.1}$$

$$\bar{r} = \frac{1}{N} \sum_{i=0}^{N-1} (r_i) \quad , \quad \bar{s} = \frac{1}{N} \sum_{i=0}^{N-1} (s_i)$$

Dissimilarity distance values for EO (D_{EO}) and GLBP (D_{GLBP}) are multiplied ($d = D_{EO} \times D_{GLBP}$) in order to attenuate the noise in the signal while emphasizing the peaks for the detection.

Detecting an event

We store the distance values $d^t_{t-\triangle}$ (values of $D_{EO} \times D_{GLBP}$ from time $t - \triangle$ to time t) in an array of $B_\triangle$ for the last $\triangle$ frames. Therefore, the $B_\triangle$ saves $\triangle$-many $d^t_{t-\triangle}$ values, which is computed between the current frame t and the frame $t - \triangle$, such that $\triangle$ is an integer value. If the maximum distance value in the buffer array $B_\triangle$ is larger than a threshold ρ, which has been chosen empirically, it implies the occurrence of an *event*.

4.2.4 Detecting a fall

Once an event is detected, the second stage of the algorithm is employed to detect whether it is a fall or not. The dissimilarity distances D_{EO} and D_{GLBP} are computed between the current and previous frames. If the multiplied value ($D_{EO} \times D_{GLBP}$) is greater than the threshold τ_c, a *fall* is detected based on camera sensor information. Dissimilarity distance values for a typical fall event are plotted in Fig. 5.3 for different features, namely when using original HOG, $(D_{EO} \times D_{ES})^2$ and the proposed $D_{EO} \times D_{GLBP}$ (solid red plot). In this video, the fall is taking place between frames 40 and 55. As can be seen, the proposed method (employing $D_{EO} \times D_{GLBP}$) gives a higher dissimilarity distance value during the fall, compared to using $(D_{EO} \times D_{ES})^2$ [27], and thus better discriminates the fall from the rest, and has better detection capability. Moreover, the proposed method results in less false positives compared to the original HOG. As seen in Fig. 5.3, the original HOG approach (light blue plot) has high values before and after the fall. The detailed rates are provided in Tables 4.1, showing the increase in sensitivity and specificity.

4.2.5 Accelerometer-based Detection

We observe the magnitude of linear acceleration with the gravity component extracted from the corresponding direction. Whenever the magnitude of 3-axis vector is greater than γ (an

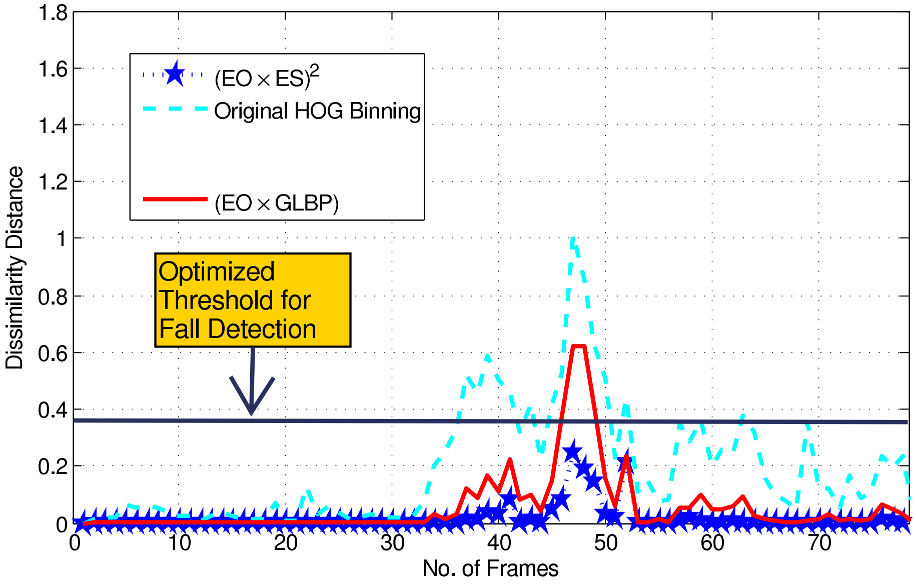


Fig. 4.2: Plots of different distance measures for a typical fall.

empirically-determined threshold), it is declared as a fall by the accelerometer-based part.

4.2.6 Camera Data Fused with Accelerometer Data

We normalize the accelerometer data by the maximum value of the accelerometer sensor of the smart phone. The fusion method is inspired by the sum rule of two normalized classifiers, since it gives the least detection error rate [153]. The only requirement is that classifiers need to be conditionally independent, which is assumed to be true since camera and accelerometer are independent sensors. Other compared methods include product, minimum, maximum and median of different classifiers. Whenever the sum value of the $D_{EO} \times D_{ES}$ and the accelerometer data is greater than $\tau_f = 0.7$, $fallDetected$ alarm is triggered, and the result is displayed on the screen of the phone. The fusion of different sensor modalities also helps to eliminate false positives caused by using only camera or only accelerometer as fall detection sensor.

4.3 Experimental Results

4.3.1 Camera-based Detection on Recorded Videos

We first compared the proposed camera-based detection part of the algorithm (incorporating GLBP features) with original and modified HOG by using 10 different subjects in the experiments. Each subject performed 10 falls from standing up and 10 falls from sitting down positions as well as 10 sitting and 10 lying down activities. Fall experiments were used to compute the sensitivity of the algorithm while the rest were used for specificity. Sensitivity and specificity are defined as:

$$\text{Sensitivity} = \frac{TP}{TP + FN}, \text{Specificity} = \frac{TN}{TN + FP} \tag{4.2}$$

where TP, FP, TN and FN denote the true positive, false positive, true negative and false negative, respectively.

For convenience of subjects, and for repeatability purposes (so that different approaches can be compared on the same videos), the experiments were performed on pre-recorded videos from ten different subjects. Videos were captured with a Microsoft® LifeCam™ camera with image size of 320×240 pixels. All the experiments were performed with the camera attached to the belt around the waist facing front.

An example set of captured frames for falling from standing up position is presented in Fig. 5.6. The parameters of the camera-based algorithm are $\triangle = 10$, $\rho = 0.2$, and $\tau_c = 0.375$. The same values have been used in all the experiments. $\triangle$ value is selected to be 10 to cover information from approximately last one second of the movement. ρ and τ_c are empirically-determined dissimilarity distance thresholds.

Average sensitivity and specificity values for 10 different subjects are presented in Table 4.1 for all the falls from standing up position. The proposed method outperforms using modified HOG with separate EO and ES histograms [27]. More specifically, the sensitiv-

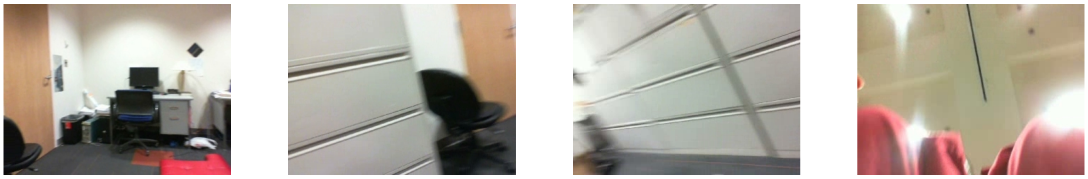


Fig. 4.3: Example frames captured during a fall from standing up position.

ity is increased from 87.84% to 96.36%, and the specificity is increased from 89.11% to 92.45%. Moreover, when compared to using original HOG, the proposed method provides a very high specificity rate. The specificity rate of using original HOG is 48.04%, which is unacceptably low, since it is highly prone to creating many false positives.

Average sensitivity and specificity rates for falling from sitting down position are also presented in Table 4.1. This is a more challenging scenario compared to falls from standing up positions. The sensitivity rate has been increased from 67.39% to 90.91%, and the specificity is increased from 89.69% to 92.45% compared to using modified HOG [27].

Table 4.1: Sensitivity and Specificity Comparison

	Falls from Standing		Falls from Sitting	
	Sensitivity	**Specificity**	**Sensitivity**	**Specificity**
Proposed Meth.	96.36 %	92.45 %	90.91 %	92.45%
Mod. HOG [27]	87.84 %	89.11 %	67.39 %	89.69 %
Org. HOG [146]	100 %	48.04 %	97.98%	66.04%

4.3.2 Detection with Fusion on Actual Smart Phones

We have also performed experiments with people carrying a Samsung Galaxy S®4 phone with Android™ OS. The experimental setup can be seen in Fig. 4.4. The subjects, 1 female and 9 male, are between the ages of 24 and 30. Their heights and weights range from 165 cm to 183 cm and from 50 kg to 103 kg, respectively. They tried to act as if they were actually falling down. It should be noted that, due to fear or cautiousness of subjects, it is difficult to recreate a free fall or collapse. It was observed that sometimes the falls were almost like in ‘slow motion’. Thus, the performed set of experiments proved to be even more challenging than an actual free fall.

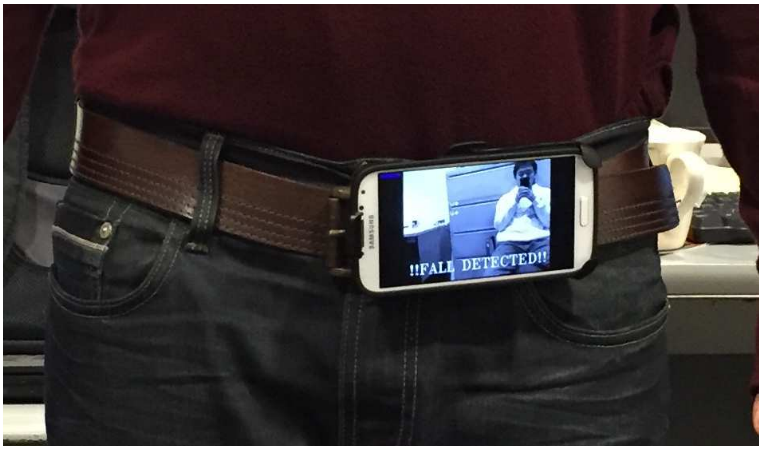


Fig. 4.4: An Android™ smart phone attached to the waist.

For real-time computation *on the phone*, without loss of generality, we have used a simplified version, with EO and ES histograms, for the camera-based detection on the Samsung Galaxy S®4 phone. The algorithm runs at 15 fps on the smart phone. In the first set of experiments, we compared sensitivity values and the number of false positives for three different cases: i) Accelerometer only, ii) Camera only, iii) Camera fused with accelerometer. The results are summarized in Table 4.2. The performed non-fall activities include 15 of sitting down and then standing up, walking and 15 of lying down and then standing up. Some of the lying down experiments include lying on the floor, which is a very complicated scenario to differentiate from actual falls. As can be seen, the proposed method provides the highest detection rate. When GLBP features are incorporated into the phone implementation, sensitivity and specificity values are expected to increase even more as demonstrated in Section 5.4.1.

It should be noted that, for the above set of experiments, the non-fall activities (walking, sitting and lying down) are not very fast and complicated in nature to cause false positives. Thus, in the third set of experiments, the goal was to demonstrate the effectiveness of fusing camera data with accelerometer data in decreasing the number of false positives created during a variety of daily activities, when only a single modality is used. For a duration of

Table 4.2: Sensitivity and Specificity values for fall detection

	Accelerometer only		Camera only		Camera+ Accelerometer	
	Sens.	FP	Sens.	FP	Sens.	FP
S1	24/30	0/30	19/30	1/30	27/30	0/30
S2	20/30	0/30	20/30	5/30	26/30	1/30
S3	10/30	0/30	28/30	1/30	28/30	0/30
S4	24/30	0/30	21/30	4/30	30/30	0/30
S5	16/30	0/30	16/30	3/30	22/30	0/30
S6	28/30	0/30	22/30	14/30	30/30	1/30
S7	21/30	0/30	25/30	10/30	27/30	1/30
S8	19/30	0/30	24/30	4/30	29/30	0/30
S9	22/30	0/30	25/30	6/30	27/30	3/30
S10	13/30	0/30	23/30	11/30	27/30	2/30
Perc.	65.66%	0%	74.33%	0.16%	91%	0.026%

about 30 min., ten subjects performed various activities including going up and down the stairs, running, jumping, changing rooms, opening doors and changing directions. For the proposed fusion-based algorithm, $fallDetected$ alarm is triggered when the summation of features from different modalities is greater than $\tau_f = 0.7$. When this happens, an alert is displayed on the phone's screen. The number of false positives, when we use (i) accelerometer only, (ii) camera only, and (iii) camera fused with accelerometer (proposed method) are summarized in Table 4.3. As seen, using camera-based features and fusing the results with the accelerometer data decreases the number of false positives.

4.3.3 Battery Consumption

The current consumption of the running algorithm is measured with a Monsoon Power Analyzer. With 2600 mAh battery capacity, the smart phone is running on 80 mA with estimated battery life of 32 hours. With the proposed algorithm continuously running on the device, it draws 542 mA of current with an estimated battery life of 4.76 hours.

Table 4.3: False Positives generated for 30-min. videos

	Acc.	Camera	Camera + Acc.
Subject 1	1	11	0
Subject 2	11	6	5
Subject 3	5	90	2
Subject 4	17	21	9
Subject 5	8	11	3
Subject 6	4	10	4
Subject 7	8	10	5
Subject 8	0	26	0
Subject 9	4	9	2
Subject 10	5	11	4

4.4 Conclusion

First, a robust and reliable algorithm for fall detection with a wearable camera has been proposed. By combining GLBP features with edge orientation histograms, this camera-based method provides higher sensitivity and specificity rates compared to using original HOG and its' modified version. A simplified version of this camera-based algorithm has been implemented on a Samsung Galaxy S®4 smart phone. The features computed from camera modality have been fused with accelerometer data. In addition, longer-duration experiments have been performed to analyze the false alarms. It has been shown that it is neither reliable nor robust to rely only on the accelerometer or only on the camera by itself, and fusing these two modalities provides much higher sensitivity, and a significant decrease in the number of false positives.

CHAPTER 5

AUTONOMOUS FALL DETECTION WITH WEARABLE CAMERAS BY USING RELATIVE ENTROPY DISTANCE MEASURE

5.1 Introduction

In this chapter, we present a fall detection algorithm that is different and improved compared to our previous work in Chapters 3 and 4 in multiple ways. First, we employ EO histograms together with gradient local binary patterns (GLBP), which are more descriptive and discriminative than Histograms of Oriented Gradients (HOG), Histograms of Template (HOT) and Semantic Local Binary Patterns (S-LBP) [151]. Secondly, we autonomously compute an optimal threshold for fall detection, from the training data, by employing the relative entropy approach from the class of Ali-Silvey distance measures. In previous Chapters 3 and 4, we had used an empirically-determined threshold. Thirdly, we compare the performance of the proposed method with three other approaches by presenting the ROC curves. Fourthly, experiments have been performed in both indoor and outdoor environments with 10 different subjects, and the results show that the proposed method is more robust and outperforms our previous work. More specifically, the method proposed in this chapter provides 93.78% and 89.8% detection rates for falling in indoor and outdoor environments, respectively.

5.2 Problem Definition and Challenges

A fall detection system should be robust and highly reliable for real-life applications. Therefore, it should have high sensitivity and specificity. In many previously proposed systems, certain performance levels for fall detection can be reached during controlled experiments. However, when applied to a real-world scenario, the detection rate significantly decreases [112]. Also, in most cases, falls and other daily activities of younger people are often used due to a lack of standardized procedures or real databases. There are very limited studies that incorporate data from older people [154, 155, 156, 157, 158], and their participation is limited to a set of simulated daily activities for a few minutes to hours. Elderly people need to wear the devices for longer periods so that a more complete dataset could be used for validation of proposed systems, and there are some studies in this direction [154, 159]. However, proposed systems create significant number of false positives, among other concerns.

Comparative evaluation of different fall detection algorithms is challenging since different researchers collect data in different ways and employ different datasets. The position of the sensor, sampling frequency, extracted features, types of activities simulated along with falls are all different depending on the employed algorithm or device. Therefore, it is hard to compare performances of different algorithms due to the lack of common datasets. In addition, it is quite difficult to build a large dataset of actual falls of elderly people. Most of the existing work involves experiments with younger people performing daily activities and imitating falls. Even though it is desirable to collect and conduct performance evaluation based on elderly data, our study is also based on data collected using younger subjects.

Furthermore, there is no published wearable camera dataset for fall events and, therefore, we have prepared our own dataset for testing purposes. It should also be noted that, due to cautiousness of subjects, it is very hard to actually recreate a free fall by imitating. Even with safety precautions in place, we observed that sometimes subjects are too afraid to fall, and the captured event looks like a slow-motion fall. This actually makes the videos used in our experiments more challenging than most of the real free fall scenarios.

In our proposed system, the wearable camera is attached around the waist facing forward. It has been verified that the waist is the optimal camera location for fall detection [109]. The camera at the waist is close to the body's center of gravity, providing trustworthy information regarding the movement of the body [160]. The captured images are not of the subject but of her/his surroundings. In addition, these images are not saved or transmitted anywhere.

5.3 Proposed Fall Detection Algorithm with Autonomous Threshold Computation

Similar to our previous work [81], the camera-based method proposed in this chapter employs EO histograms together with the GLBP as the image descriptive features. The proposed algorithm consists of two main stages: (i) detecting an event, such as sitting down, lying down or falling down, and (ii) deciding whether the detected event is a fall event or not. Different from our previous work, we also introduce a relative entropy-based method to autonomously compute a threshold, for the detection of fall events, from the training data.

In Section 5.4, we will compare our proposed method with the original HOG [146], the modified HOG, which we introduced in [27], and the GLBP-based approach. In modified

HOG method, different from the original HOG [146], ES and EO histograms are built separately. Moreover, the cells that do not contain significant edges are discarded when building the histograms, as described in more detail in [27].

Next, in Section 5.3.1, we describe the proposed method incorporating edge orientations and GLBP features. We then introduce the relative-entropy based threshold selection in Section 5.3.2.

5.3.1 Fall Detection

It has been shown that, with the same training set, the GLBP features are more descriptive and discriminative than HOG, histograms of template (HOT) and Semantic Local Binary Patterns (S-LBP) [151].

An overview of the GLBP feature generation is provided in Fig. 5.1. Since angle values are divided into eight equal intervals and the width of the LBP is characterized by seven different lengths, one GLBP feature for a cell is of length 56. In our method we use 16 cells, which results in a concatenated GLBP vector of length 16×56 for one frame. We apply L2 normalization before concatenation [81].

As seen in Fig. 5.1, a neighboring pixel is assigned the value of '1' or '0', if its' intensity value is greater or less than the center pixel, respectively. In the resulting 8-bit binary sequence, the length of the longest consecutive sequence of 1s is determined, which happens to be 4 in Fig. 5.1. Also, the edge angle is found, which is 4 in our example. These numbers $(4, 4)$ are used as the entry index of an 7×8 matrix, and this entry is incremented by the value of the edge strength. Filling and normalizing this matrix, results in an 56-dimensional GLBP feature. More details can be found in [152, 81].

The advantages of using GLBP features compared to only using HOG features are threefold: (i) with GLBP all eight neighbors are used to acquire a pattern, whereas HOG relies only on the four neighboring pixels to calculate the edge orientation and edge strength values (Fig. 5.2); (ii) the GLBP approach uses only uniform patterns [152]. Consequently,

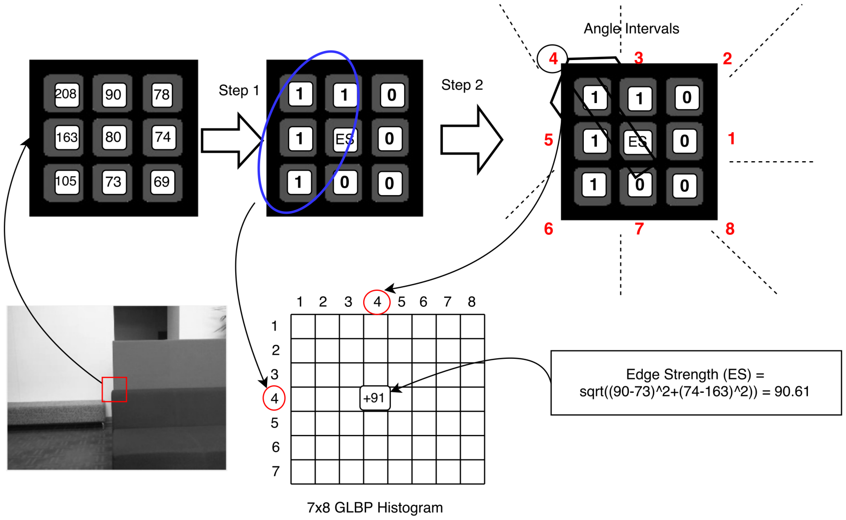


Fig. 5.1: The steps showing the generation of 56-dimensional GLBP feature for a cell.

non-uniform patterns are considered as noise and they are not included in the overall LBP descriptor. On the other hand, every pixel is included in HOG, and noisy pixels can affect the histogram; (iii) with GLBP, the gradient strength has significant effect on the pattern that is assigned to a pixel.

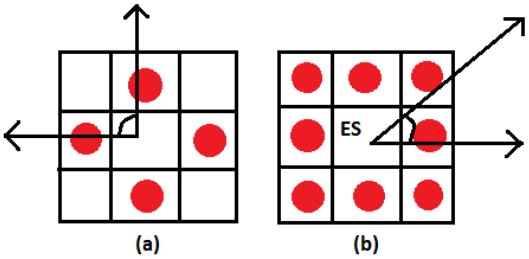


Fig. 5.2: Advantages of local binary pattern (LBP) methods; (a) HOG, (b) GLBP.

Our proposed camera-based fall detection algorithm has two main stages: (i) detecting an event, such as sitting down, lying down or falling down, and (ii) deciding whether the detected event is a fall event or not.

In this algorithm, EO histograms are employed together with the GLBP features. The EO histogram is constructed by computing the gradient orientation for every pixel within a cell. To decrease the computational load, one block is divided into 16 cells. Furthermore, the cells that do not contain significant edges are discarded autonomously, and not included in the EO histograms as described in [27].

The GLBP descriptor is obtained as described above. After the EO and GLBP feature descriptors are normalized, the dissimilarity distance values D_{EO} (for EO) and D_{GLBP} (for GLBP) are computed between two frames, by using Eq. (5.1), wherein r and s are N-dimensional feature vectors [81]. In our case, the vector for EO is of size $9 \times k$, where k is the number of cells that are kept. The size of the GLBP feature vector is 16×56.

$$D = 1 - \frac{\sum_{i=0}^{N-1}(r_i - \bar{r})(s_i - \bar{s})}{\sqrt{\left[\sum_{i=0}^{N-1}(r_i - \bar{r})^2 \sum_{i=0}^{N-1}(s_i - \bar{s})^2\right]}}, \tag{5.1}$$

$$\bar{r} = \frac{1}{N}\sum_{i=0}^{N-1}(r_i) \quad , \quad \bar{s} = \frac{1}{N}\sum_{i=0}^{N-1}(s_i).$$

D_{EO} and D_{GLBP} are multiplied ($d = D_{EO} \times D_{GLBP}$) to attenuate the noise in the signal, and emphasize the peaks for the detection.

Detecting an event

To detect the occurrence of an event, the distance values $d^t_{t-\triangle}$ (value of $D_{EO} \times D_{GLBP}$ computed between frame t and $t - \triangle$) are stored in an array $A_\triangle$ for the last $\triangle$ frames. In other words, $A_\triangle$ contains $\triangle$-many $d^t_{t-\triangle}$ values. If the maximum value in $A_\triangle$ is greater than an empirical threshold ρ, an *event* is detected, and the next part of the proposed algorithm is used to decide whether or not this event is a fall.

Detecting a fall

To detect a fall event, in the second stage of the algorithm, the dissimilarity distances D_{EO} and D_{GLBP} are computed between consecutive frames. If the product ($D_{EO} \times D_{GLBP}$) is greater than the threshold τ_c, a *fall* is detected based on camera sensor information (as will be described in Section 5.3.2, this threshold value is determined autonomously by using the class of Ali-Silvey distance measures).

As an example, plots of different dissimilarity distance measures for a fall event are displayed in Fig. 5.3 when using original HOG, modified HOG employing $(D_{EO} \times D_{ES})^2$ and the proposed method employing $D_{EO} \times D_{GLBP}$ (solid red plot). In the video employed for this illustration, the person is walking first and the fall takes place between frames 40 and 55. The proposed method (employing $D_{EO} \times D_{GLBP}$) better distinguishes fall from the walking part compared to using $(D_{EO} \times D_{ES})^2$ [27]. Moreover, original-HOG (cyan plot) is more prone to creating false positives, and has high distance values not only during a fall, but also before and after. Modified-HOG, on the other hand, is more prone to missing a fall event based on the distance values. The detailed ROC curves are provided in Fig. 5.8. In addition, Fig. 5.9 shows the higher sensitivity and specificity values for the proposed method for varying thresholds.

The pseudo code for the proposed camera-based algorithm is presented in Algorithm 2.

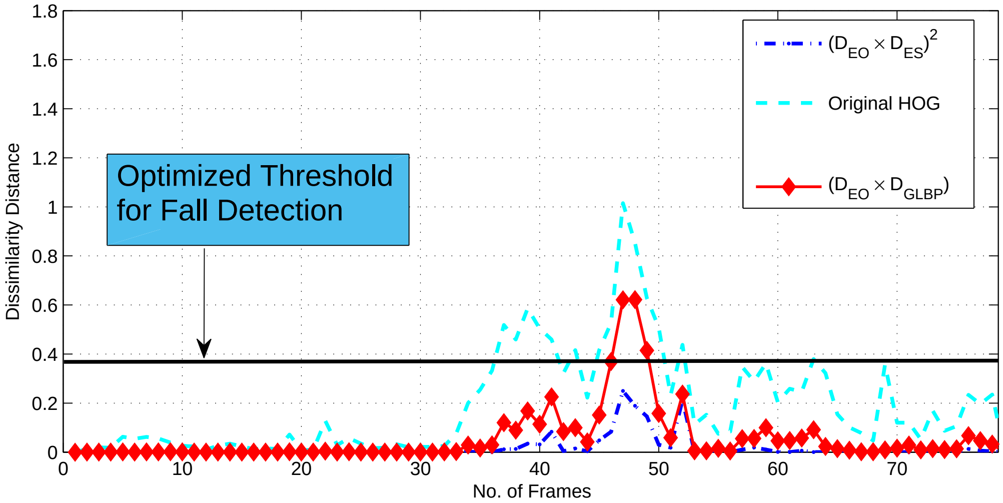


Fig. 5.3: Different distance measure plots obtained from a video that contains walking followed by a fall.

Algorithm 2 Fall Detection Algorithm

```
for $All\ Frames$ do
    if $t == 1$ (first image frame) then
        Initialize feature vectors
    else
        if Average Intensity $\leq 30$ then
            Occlusion detected.
        else
            if $max(A_{\triangle}) \geq \rho$ then
                $eventDetected = true$
                if $(D_{EO} \times D_{GLBP})(t) \geq \tau_c$ then
                    $fallDetected = true$
                end if
            end if
        end if
    end if
end for
```

5.3.2 Autonomous Threshold Computation for Fall Detection

Ali-Silvey Distance Measures

As described above, we employ a threshold τ_c to differentiate falls from other daily activities such as walking, lying down and sitting. We employ the class of Ali-Silvey distance measures [161] between probability distributions to select the optimal threshold for the proposed fall detection method, which employs the dissimilarity distance $D_{EO} \times D_{GLBP}$. We use Ali-Silvey distance measures for separating two classes, namely fall and non-fall activities.

A threshold τ_c separates the dissimilarity distances into two classes, f_0 and $\mathrm{f}_1 \in \mathrm{F}$, for fall and non-fall activities, respectively, such that:

$$(D_{EO} \times D_{GLBP})(t) \ \epsilon \ \begin{cases} f_0 & \text{if } (D_{EO} \times D_{GLBP})(t) \geq \tau_c \\ f_1 & \text{if } (D_{EO} \times D_{GLBP})(t) < \tau_c \end{cases}, \tag{5.2}$$

where $(\mathrm{D_{EO}} \times \mathrm{D_{GLBP}})(\mathrm{t})$ is the dissimilarity distance computed between frames t and $t-1$ as described above.

In order to build a probability density function of transitions, we need to obtain a co-occurrence matrix to represent the jumps in dissimilarity distance values between consecutive image frames. Since falls from standing up position are different from falls from sitting down position, we need to compute different optimal thresholds for different types of falls. Based on Eq. (5.1), the maximum dissimilarity distance value for falls starting from a standing up position has been observed to be 1 whereas the maximum value for falls starting from a sitting position has been observed to be 0.9005. In general, the distance traveled when someone is falling from standing up position is higher. Moreover, for outdoor scenarios the maximum value for falling from standing up positions has been observed to be 1.2272. In most outdoor scenarios, when a fall occurs, the view changes from buildings, other surrounding structures and trees etc. to wide open skies with barely any edge information. Thus, falls result in higher dissimilarity distances.

In the next step, we build histograms of the dissimilarity distances. We have experimented with different number of bins $(7, 8, 10, 15, \ldots, 40)$ for ranges starting from 0 to various maximum values 1. As expected, when the number of bins is too large (≥ 10 in turn), the population in certain bins becomes very low compared to other bins, which in turn affects the accuracy of the overall distribution. Fig. 5.4 shows two histograms, obtained from all the fall experiments, when using 20 bins versus 8 bins. When 8 bins are used, a more distinct gap is observed between the 3th and 4th bins of the histogram as seen in Fig. 5.4(b). In our algorithm, we used 8 bins, resulting in a step size of 0.125. For selecting the optimal threshold τ_c for fall detection, we employed relative entropy-based thresholding from the class of Ali-Silvey distance measures as described in the next section.

Relative Entropy-based Distance Measure

Among various Ali-Silvey distance measures, the relative entropy-based method provided the most discriminative threshold for differentiating falls from other daily activities. We

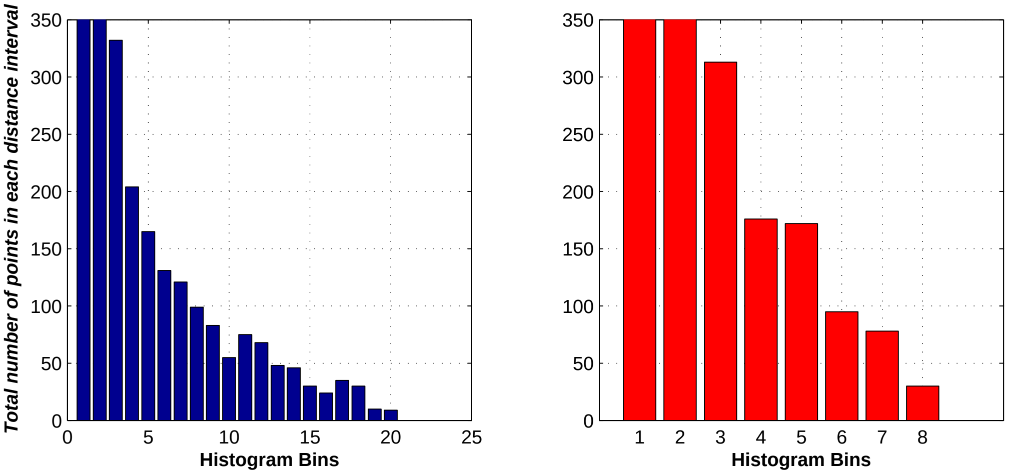


Fig. 5.4: Dissimilarity distance histograms when using (a) 20 and (b) 8 bins.

employed a co-occurrence (or transition) matrix, since we only compute a dissimilarity distance from two consecutive images. To differentiate falls from other activities, we are more interested in the change of dissimilarity distances over time and this information can be captured in a co-occurrence matrix that contains jumps between different distance levels. Therefore, the dissimilarity distance values calculated between consecutive image frames is represented as a co-occurrence matrix. Since the distance value is divided into l bins, the co-occurrence matrix is $l \times l$, wherein, p_{ij}, the entry (i, j) represents the transition probability from the ith distance level to the jth distance level as shown in (5.3) [161], where N represents the duration of a fall event, and M is the total number of fall experiment for that fall type (fall from sitting position or fall from standing position).

$$p_{ij} = \frac{t_{ij}}{\sum\limits_{i=0}^{l-1} \sum\limits_{j=0}^{l-1} (t_{ij})}, \tag{5.3}$$

where

$$t_{ij} = \sum_{m=0}^{M} \sum_{n=0}^{N} \delta_m(n), \tag{5.4}$$

and

$$\delta_m(n) = \begin{cases} 1 & \text{if } f(n) = i, f(n+1) = j, \\ 0 & \text{otherwise.} \end{cases} \tag{5.5}$$

Diagonal elements of the co-occurrence matrix include the distance points where the previous distance value falls into the same distance interval as the current distance. Since they do not provide significant information for fall detection, the diagonal elements of the matrix are assigned the value of zero in the proposed method. Another motivation and supporting argument for this is the following: for walking and other routine daily activities, there is a significantly large accumulation in the first several diagonal entries of the co-occurrence matrix that dominates and significantly affects the probability distribution calculated among the entries of the co-occurrence matrix. Relative entropy-based method to compute the threshold becomes more effective for differentiating fall and non-fall activities by setting the diagonal elements of the matrix to zero. Since the goal is to detect the abrupt changes, we are not interested in the same dissimilarity distance level occurrences over the course of an event. The co-occurrence matrix is then divided into intra-class and inter-class transitions. The threshold t, selected between 0 and $l-1$, partitions the matrix into quadrants. The quadrant probabilities are defined as [161, 162]:

$$P_A = \sum_{i=0}^{t} \sum_{j=0}^{t} p_{ij}, \;\; P_B = \sum_{i=0}^{t} \sum_{j=t+1}^{l-1} p_{ij}, \tag{5.6}$$

$$P_C = \sum_{i=t+1}^{l-1} \sum_{j=t+1}^{l-1} p_{ij}, \;\; P_D = \sum_{i=t+1}^{l-1} \sum_{j=0}^{t} p_{ij}, \tag{5.7}$$

Quadrant A and quadrant C represent the intra-class transition probabilities, whereas quadrant B and quadrant D represent inter-class transition probabilities. $\mathrm{P_A}$, $\mathrm{P_B}$, $\mathrm{P_C}$ and $\mathrm{P_D}$ sum up to 1.

After the quadrant probabilities are calculated, the probabilities assigned to the diagonal elements are also zero. Then, the transition probabilities $q_A(t)$, $q_B(t)$, $q_C(t)$, and $q_D(t)$ are calculated by Eq. (5.8) through (5.10) [161]. The transition probabilities are modified versions of the ones proposed in [162], since they provide higher values for a given threshold t and contain more information regarding the spatial relationships of the distance levels.

$$q_k(t) = r_k(t) \cdot s_{ij}, k = A, B, C, D, \tag{5.8}$$

where

$$r_k(t) = P_k(t)/n_k t, \tag{5.9}$$

with

$$n_A = \sum_{i=0}^{t}\sum_{j=0}^{t} s_{ij}, n_B = \sum_{i=0}^{t}\sum_{j=t+1}^{l-1} s_{ij},$$

$$n_C = \sum_{i=t+1}^{l-1}\sum_{j=t+1}^{l-1} s_{ij}, n_D = \sum_{i=t+1}^{l-1}\sum_{j=0}^{t} s_{ij}, \tag{5.10}$$

$$s_{ij} = \begin{cases} 1 & \text{if } p_{ij} \neq 0, \\ 0 & \text{if } p_{ij} = 0. \end{cases}$$

Then, we calculate $I'(t)$ by using

$$\begin{aligned} I'(t) = {} & P_A(t) \ln q_A(t) + P_B(t) \ln q_B(t) + \\ & P_C(t) \ln q_C(t) + P_D(t) \ln q_D(t). \end{aligned} \tag{5.11}$$

We compute the optimal threshold bin, t^*, by finding the value of t that maximizes (5.11). Thus,

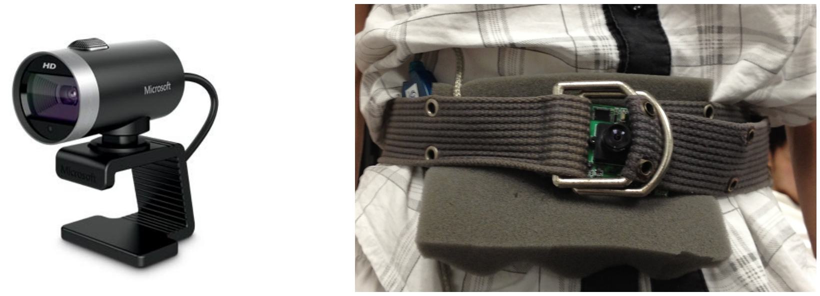
$$t^* = \arg \max_{0 \leq t \leq l-1} (I'(t)). \tag{5.12}$$

After selecting the optimal bin number via (5.12), t^* is multiplied with the bin step-size in order to obtain the threshold τ_c for the corresponding scenario (falling from standing up or falling from sitting down).

Among different Ali-Silvey distance measures, such as relative entropy, J-divergence, Bhattacharyya distance, Chernoff distance, and Matsuista distance, the relative entropy-based approach provided the best overall results for selecting the optimal threshold for fall detection. In other words, the relative entropy-based distance threshold gave the highest sensitivity and specificity rates compared to other Ali-Silvey distance measures. For indoor experiments, the step size value is selected to be 0.125 (corresponding to 8 bins) for all types of fall events.

5.4 Experimental Results

Experiments have been performed indoors and outdoors to compare the performance of the proposed camera-based fall detection algorithm with the original and modified HOG-based algorithms, and using only the GLBP features. Each of the 10 subjects participating the experiments performed a total of 20 falls (10 from a sitting positions and 10 from a standing position) in addition to 10 sitting and 10 lying down activities. The ages, heights

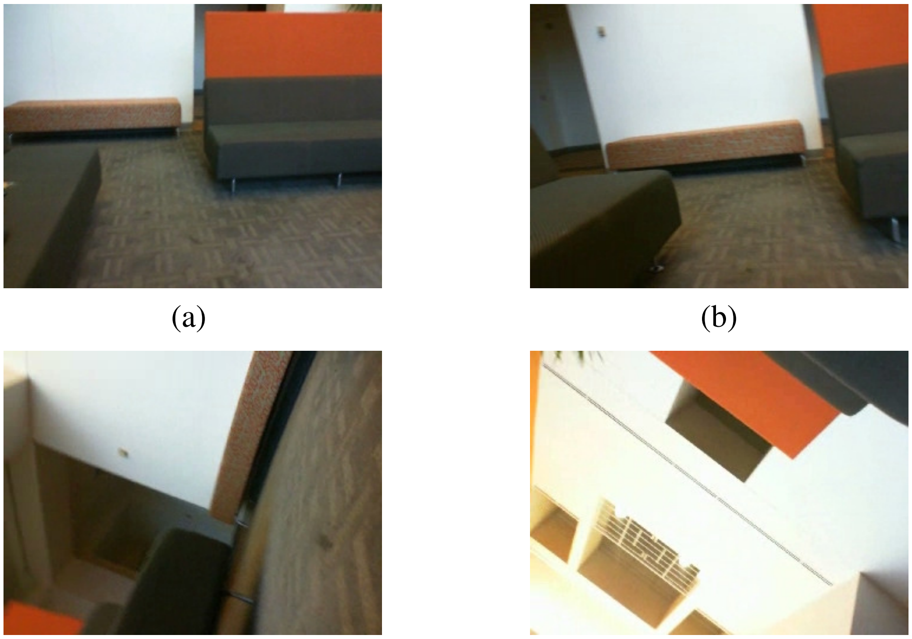


(a) (b)

Fig. 5.5: (a) Microsoft® LifeCam™, (b) which is attached to a belt around the waist.

and weights of the subjects range from 24 to 30, 165cm to 183 cm, and 50 kg to 103 kg, respectively. The subjects imitated the fall events and, as mentioned above, even with the safety precautions, sometimes subjects were too afraid to fall, resulting in slow-motion falls and creating a much more challenging dataset for us than having actual free fall events.

To be able to compare different approaches on the same videos, we first performed the experiments on recorded videos, which were captured with a Microsoft® LifeCam™ camera (Fig. 5.5(a)) attached to a subject's belt facing front. In order not to increase the computational complexity, image size is kept at 320×240. Also, we only processed even-numbered frames to acquire a sequence with a gap between consecutive images. This allowed us to show the portability of the proposed method to a mobile platform (like a smart phone), wherein the processing time and thus the distance between processed frames is expected to be larger. The parameters of the algorithm are $\triangle = 10$, $\alpha = 0.5$, and $\rho = 0.2$. $\triangle$ value is set to 10 to cover and extract information from approximately the last one second. The value of l, determining the size of the co-occurrence matrix $(l \times l)$ is selected to be (8×8). τ_c is the dissimilarity distance threshold that has been determined based on relative entropy-based threshold selection. The same values have been used in all the experiments.



(a) (b)

(c) (d)

Fig. 5.6: Example images captured from the body-worn camera during a fall from standing up position in an indoor environment.

5.4.1 Indoor Experiments

An example sequence of images captured indoors with the body-worn camera is shown in Fig. 5.6. As can be seen, the captured images get blurry during a fall, when the movement is fast.

An example of a fall event together with the plot of calculated dissimilarity distances can be seen in Fig. 5.7. A falling down event occurs between frames 35 and 50. As seen from the solid red plot in Fig. 5.7(c), the proposed method has more discriminative power than our previous work in [27]. Moreover, it is less prone to creating false positives than using the original HOG.

In order to evaluate the performances and compare different methods, we obtained the Receiver Operating Characteristic (ROC) curves for indoor and outdoor experiments for varying threshold values. As mentioned above, the datasets consist of videos of falls from sitting and standing up positions and non-fall activities (lying down and sitting).

We compared the proposed method with original HOG [146], modified HOG [27], and using only the GLBP features. To perform this comparison, we divided one block into 16 cells for all the approaches. As can be seen in Fig. 5.8, the proposed method operates closest to the upper left-hand corner, and provides a better performance compared to other approaches. We also marked the operating point, which was obtained by setting $\tau_c = 0.375$ on Fig. 5.8. This is the threshold value obtained by the autonomous threshold computation described in Section 5.3.2. The operating point corresponds to the location $(0.0755, 0.9378)$ in Fig. 5.8.

We have also obtained the sensitivity/specificity curves, for the indoor dataset and varying threshold values, which are shown in Fig. 5.9. Sensitivity is defined as the ratio of the number of True Positives (TP) to the summation of TP and False Negatives (FN). Specificity, on the other hand, is defined as the ratio of True Negatives (TN) to the summation of TN and False Positives (FP). As can be seen in Fig. 5.9, the proposed method outperforms the others, and operates closest to the upper right-hand corner. The proposed method, with the autonomously computed threshold value, operates at the point of $(0.938, 0.876)$.

In general, detecting falls from sitting down position is relatively harder than detecting falls from standing up position. In order to compare the performance of the proposed method with that of modified-HOG [27], when detecting falls from sitting and standing up positions with varying thresholds, we performed experiments by setting the value of τ_c between 0.2 and 0.5 in increments of 0.02. The sensitivity values for varying thresholds for falls from standing up and sitting positions are shown in Figures 5.10 and 5.11, respectively. As can be seen, the proposed method provides higher sensitivity than the modified HOG for different threshold values for both type of falls. In addition, although the sensitivity values for the modified-HOG drops significantly for falls from sitting down position, the proposed method still provides high sensitivity rates as seen in Fig. 5.11.

We performed another set of experiments to observe the effect of content and the size of the training data on the computation of the threshold τ_c, and thus on the detection performance. The threshold is computed from a set of video data that include a certain number of subjects performing fall and non-fall activities. We wanted to investigate how the algorithm with this threshold will perform on a different set of people. In general, it is more practical to autonomously obtain the threshold from a certain training dataset and then apply it to any test dataset without requiring per user training. Thus, for all the fall experiments, we performed a five-fold cross validation. When the derived threshold is tested on the test datasets, which is one-fifth of the entire dataset, we observed that the average sensitivity is 93.77% and the average specificity is 92.44% as seen in Table 5.1. The standard deviation of sensitivity was 3.24%.

Table 5.1: Mean and Standard Deviation of Sensitivity-Specificity

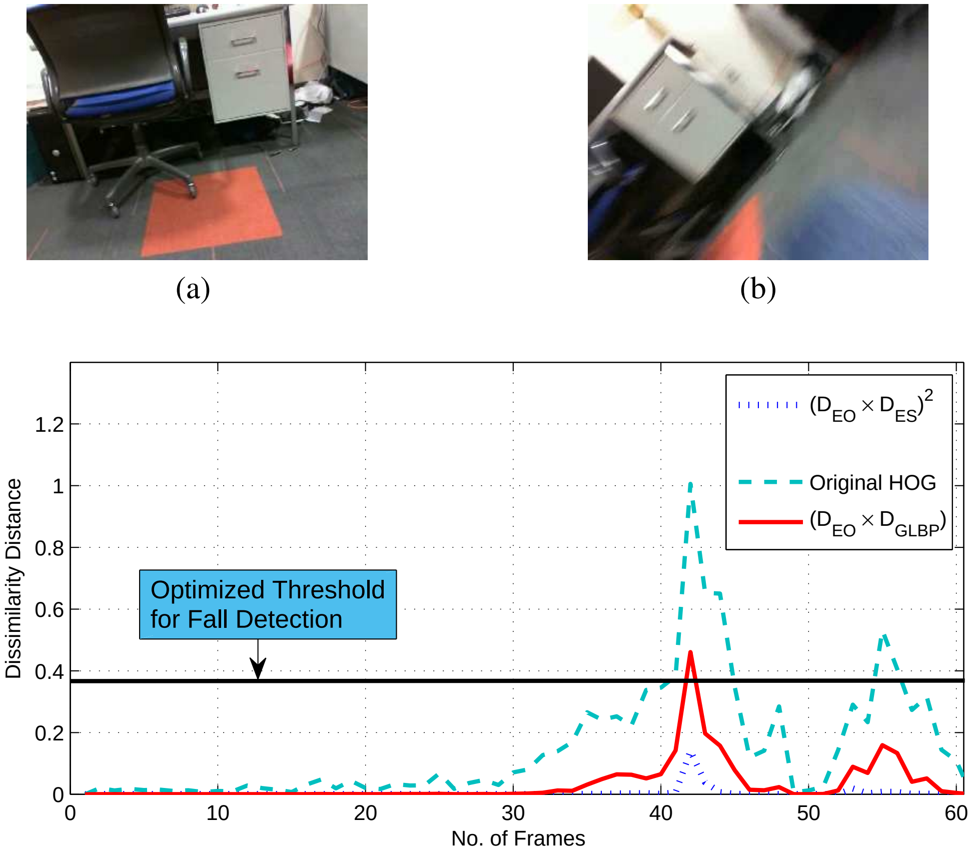
(dEOxdGLBP)	**Mean**	**STD**
Sensitivity	93.77%	3.24%
Specificity	92.44%	4.03%

5.4.2 Outdoor Experiments

As mentioned above, wearable sensors allow the monitoring of people wherever they may travel including indoors and outdoors. We also tested the proposed method on outdoor scenarios. Example frames captured by the body-worn camera during the course of a fall event can be seen in Fig. 5.12, which also shows the significant change in the scene, going from building and trees to wide open skies. The threshold calculated from the training set is $\tau_c = 0.4602$. For outdoor scenarios, the change in scenery is usually much more significant compared to indoor scenarios resulting in higher dissimilarity distances between frames. The proposed approach for autonomous threshold computation is able to capture this resulting in a higher threshold value of 0.4602. Similar to indoor experiments, we obtained the ROC curves for the proposed method as well as three other approaches as seen in Fig. 13. The operating point corresponding to the autonomously computed threshold value is (0.0755, 0.898).

5.5 Conclusion

We have presented a new approach to detect falls by employing a wearable camera. We employed a modified version of the histograms of oriented gradients (HOG) approach together with the gradient local binary patterns (GLBP). Moreover, we have proposed an approach to autonomously compute the threshold, for the detection of fall events, from the training data by using the relative entropy approach from the class of Ali-Silvey distance measures. Different sets of experiments were performed on data obtained from 10 different people to show the performance of the proposed approach under varying threshold values and with different training sets varying in terms of their size and content. The proposed method provides 93.78% and 89.8% accuracy for detecting falls in indoor and outdoor experiments, respectively, while providing low false positive rates. We have also compared the proposed method with three other approaches using ROC curves, and showed that the proposed approach outperforms the original HOG, modified HOG and GLBP-based methods.



(a) (b)

(c)

Fig. 5.7: (a) and (b) Example images, captured by the body-worn camera, while a fall occurs ; (c) The dissimilarity distances obtained for this video.

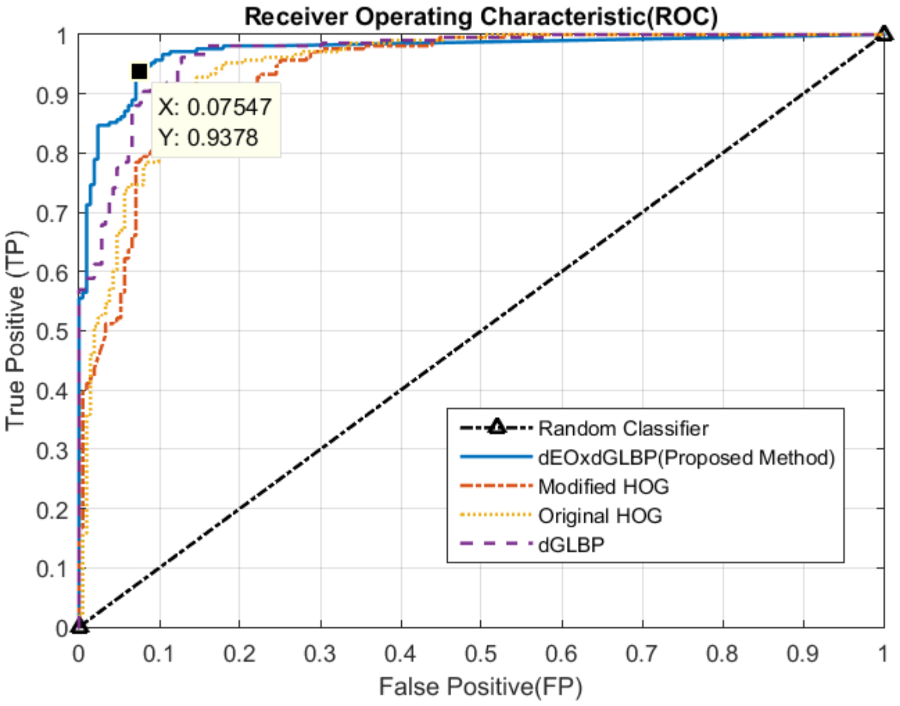


Fig. 5.8: Receiver Operating Characteristic (ROC) curves, for varying threshold values, obtained from the indoor dataset containing falls and non-fall activities.

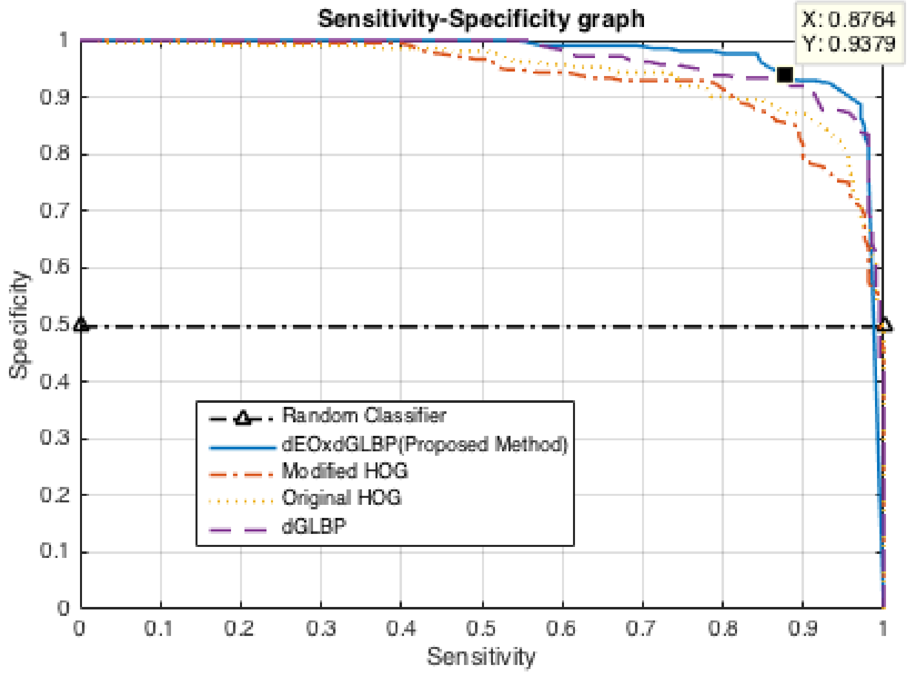


Fig. 5.9: Sensitivity-Specificity curves, for varying threshold values, obtained from the indoor experiment dataset containing falls and non-fall activities.

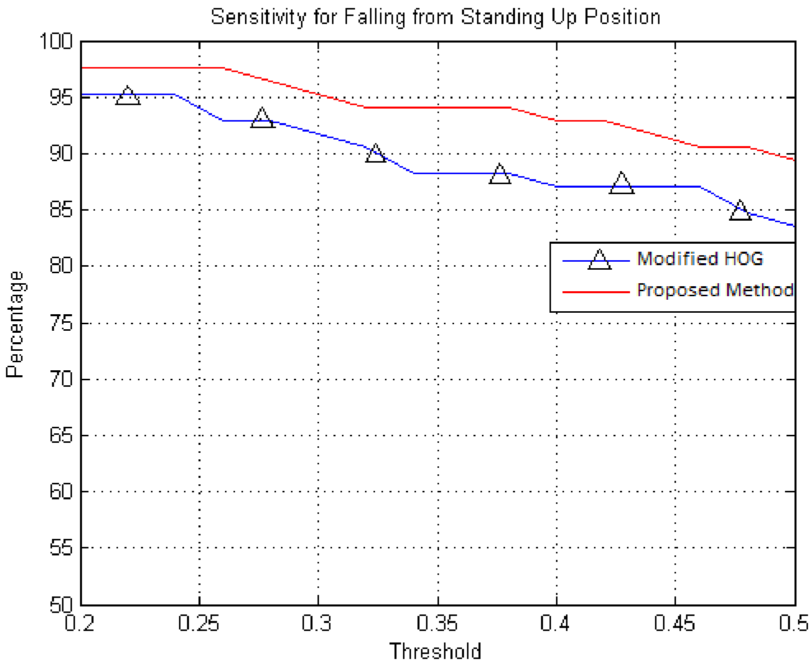


Fig. 5.10: Sensitivity values for falls from standing up position with varying thresholds.

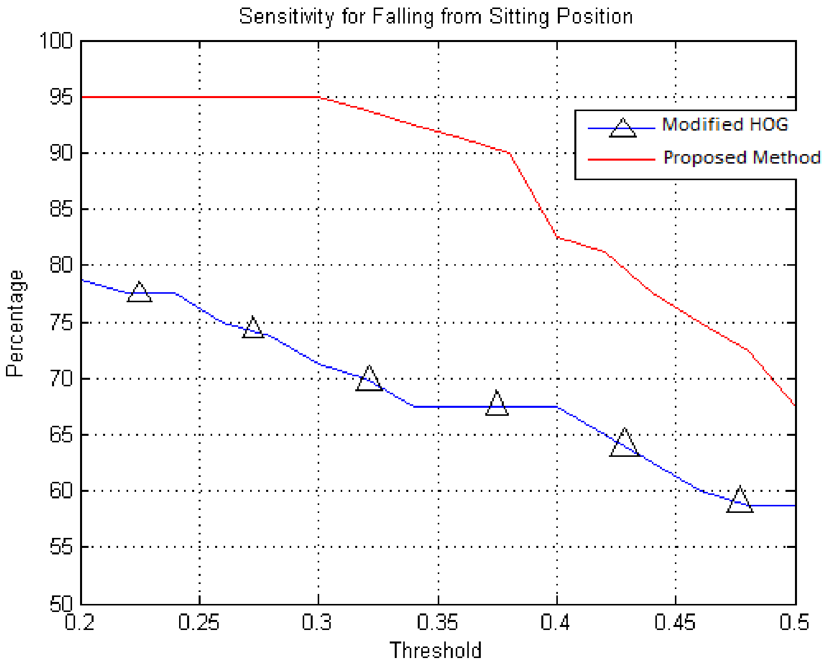


Fig. 5.11: Sensitivity values for falls from sitting down position with varying thresholds.

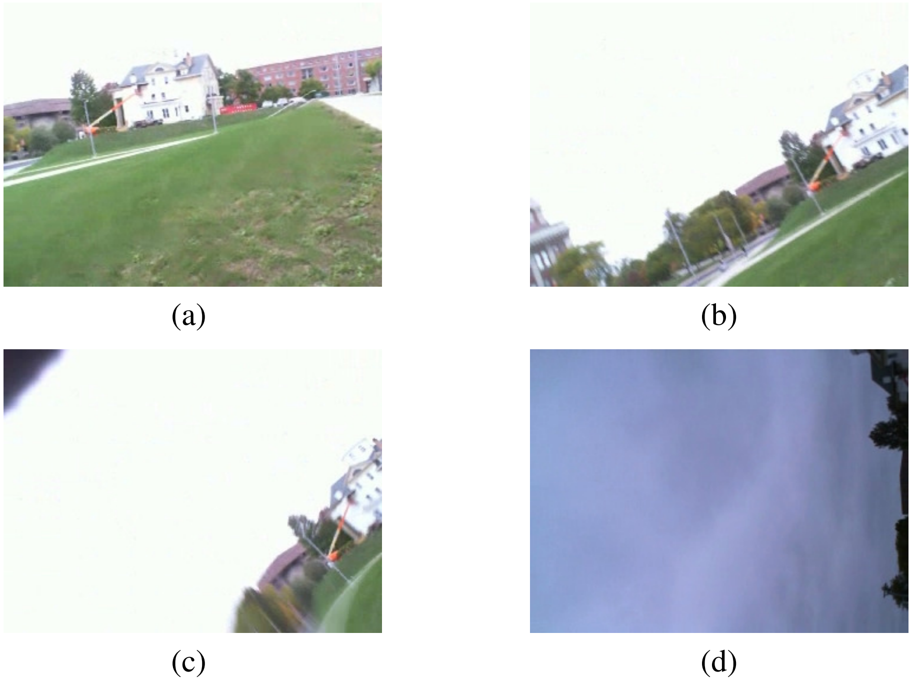


Fig. 5.12: Example images captured from the body-worn camera during a fall from standing up position in an outdoor environment.

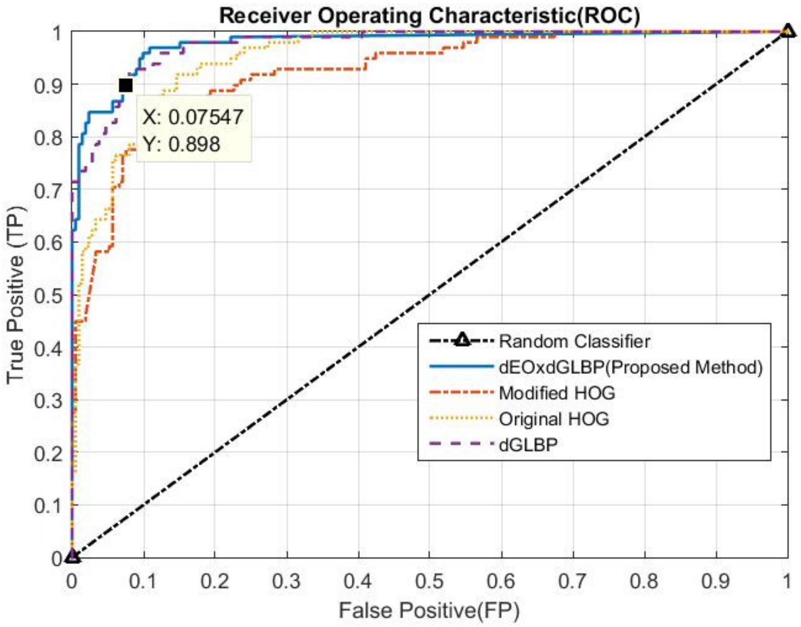


Fig. 5.13: Receiver Operating Characteristic (ROC) curve, for varying threshold values, obtained from the outdoor dataset.

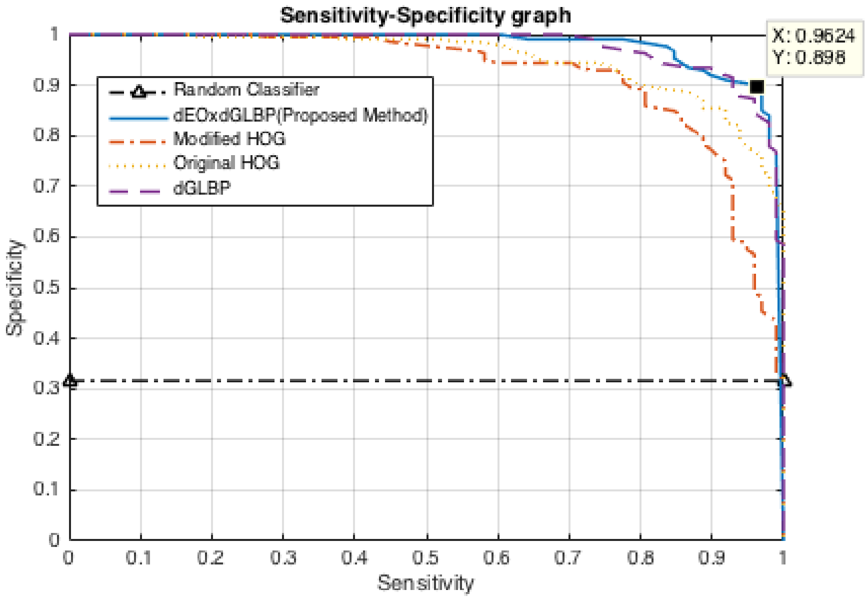


Fig. 5.14: Sensitivity-Specificity curves, for varying threshold values, obtained from the outdoor dataset.

CHAPTER 6

ROBUST AND RELIABLE STEP COUNTING BY MOBILE PHONE CAMERAS

6.1 Introduction

Wearable devices, such as smart watches, smart bands and activity trackers, are efficient gadgets to track human activity. On the other hand, smart phones equipped with accelerometers and gyroscopes also provide users with similar functionalities as wearable devices. In U.S., only 1-2% of people own a wearable device while 65% of people own a smart phone [163]. It has also been shown that wearable gadgets can have higher error rates compared to smart phone applications during step counting experiments [163]. Therefore, with their lower error rates, higher accessibility and widespread use, smart phones are feasible and preferable platforms for step counting. Moreover, current smart phones and tablets, equipped with powerful processors and a wide variety of sensors, have also become ideal platforms for activity monitoring. Zhang et al. [20] describe a hierarchical method of activity classification based on a smart phone, equipped with an embedded 3D-accelerometer, worn on the belt.

Accelerometer-based step counters are commonly available, especially after being integrated as applications into smart phones and smart watches. Moreover, accelerometer data is also used to measure step length and frequency for indoor positioning systems. There has been significant amount of research on algorithms using the accelerometer data from smart phones. Park et al. [164] presented an accelerometer-based activity tracker on smart phones to provide high accuracy in location estimation for indoor environments. Pan and Lin [165] proposed an accelerometer-based step counting algorithm for smart phone users, which does not require the user to have the smart phone attached to the body while walking. Another smart phone application uses accelerometer data to analyze walking patterns and compute distance traveled for clinical purposes when the phone is attached on top of a belt around the waist [166]. Brajdic and Harle [167] tested accelerometer-based step-counter algorithms while trying different locations for the smart phone. They have found that certain locations such as back pocket of trousers degrade the performance of the algorithms significantly.

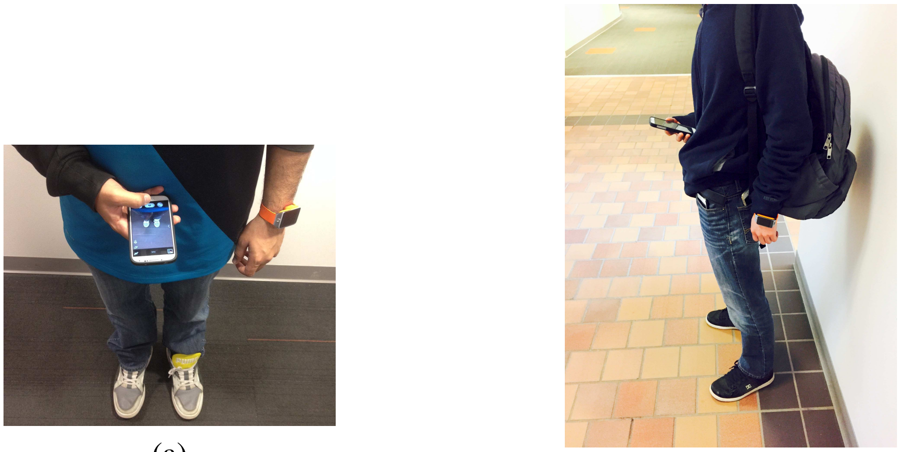
In general, accelerometer-based applications on smart watches or smart phones are prone to overcounting, since they are sensitive to even smallest motions, and count other routine movements as steps. People tend to move their smart phones quite often while using them, and these moves are counted as steps most of the time with accelerometer-based approaches. For instance, using the smart phone for browsing the web, text messaging or taking a video can easily increase the number of counted steps. Moreover, when people are exposed to acceleration, e.g. inside a vehicle or an elevator, accelerometer-based step counters usually keep counting. When we tested an accelerometer-based step counter application running on Apple™ iPhone 6, it was observed that the application increments the number of steps when the user is traveling in a car, taking the elevator, or moving the smart phone up and down in the air. Also, it has been observed that when users walk really slowly, or when they stop and start walking again, the accelerometer-based counting becomes unreliable. Since accurate step detection is very important for indoor positioning systems, reliable and more precise alternatives are needed for step detection and counting.

Different from the aforementioned work, Aubeck et al. [168] use data from a camera sensor to detect steps as an extension to an indoor positioning system. They use template matching to count steps based on the appearance and disappearance of the forward section of the feet. It is stated that the method does not perform well for fast movements since moving objects are often fuzzy and template matching and generation become problematic.

A camera sensor provides abundance of information, and it is expected that *wearable cameras* will be employed more to understand lifestyle behaviors for health purposes [72]. In this chapter, we propose a robust system using smart phone to detect and count steps during walking. This is one of the few works that use data from a *mobile smart phone camera* to overcome shortcomings of the accelerometer-based step counting systems. Actual smart phones have been used to collect the video data. As seen in Fig. 6.1, subjects hold the phone as they normally would when they use the phone to read e-mails, use different applications or check different pages on the web. Since the rear-facing camera points to the floor, captured videos contain the feet and also legs of the subjects. The proposed method does not rely on templates, and does not make any assumptions about the duration of a step or the distance between steps. The system performs reliably across different users, such as tall versus shorter people. We have performed experiments with subjects carrying five devices simultaneously, namely four smart phones and a smart watch, while they are walking so that we can compare the step-counting results. Three of the smart phones and the smart watch have accelerometer-based applications for step counting. The watch used in the experiments is a Samsung™ Gear 2 Neo smart watch. Three smart phones were carried in a backpack, in front pocket and in back pocket of trousers. The fourth smart phone was held by hand to capture video data for the proposed method. Experimental results show the high sensitivity of the accelerometer-based algorithms to the location of the device. Moreover, a large variance is observed across different users with the accelerometer-based step counting compared to our proposed method. These numbers will be discussed in detail in Section 6.5. Although not implemented in its' entirety on the smart phone yet, the proposed algorithm has been designed so that it can be ported to, and run on the smart phone.

6.2 Proposed Method

We propose a vision-based algorithm to reliably detect and count steps using mobile phone or tablet cameras. As seen in Fig. 6.1, subjects hold the phone as they normally would when they use the phone to read e-mails, use different applications or check different pages on the web. Since the rear-facing camera points to the floor, captured videos contain the feet and also legs of the subjects. Example images captured during a walk are shown in Fig. 6.6 as the user is walking across different surfaces.



(a)

(b)

Fig. 6.1: (a) A top view of a user holding the smart phone, (b) Experimental setup showing a user simultaneously carrying five devices: A smart phone is held at one hand, a smart watch is worn on the other wrist, while three other smart phones are put in the front pocket and back pocket of trousers and inside a backpack.

The flow diagram of the proposed method is provided in Fig. 6.8. First, Canny edge detection [169] algorithm is applied to detect the edges in the image. Then, the lines in the image are detected and removed by using the Hough transform [170]. The resulting images after the edge detection and line removal can be seen in Fig. 6.7(b) and 6.7(c), re-

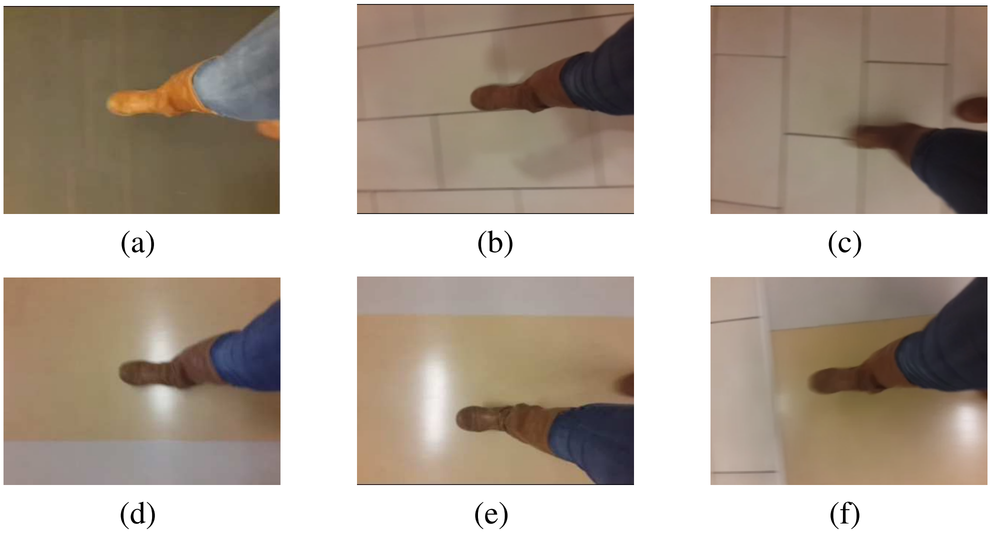


Fig. 6.2: Example frames captured during a walk.

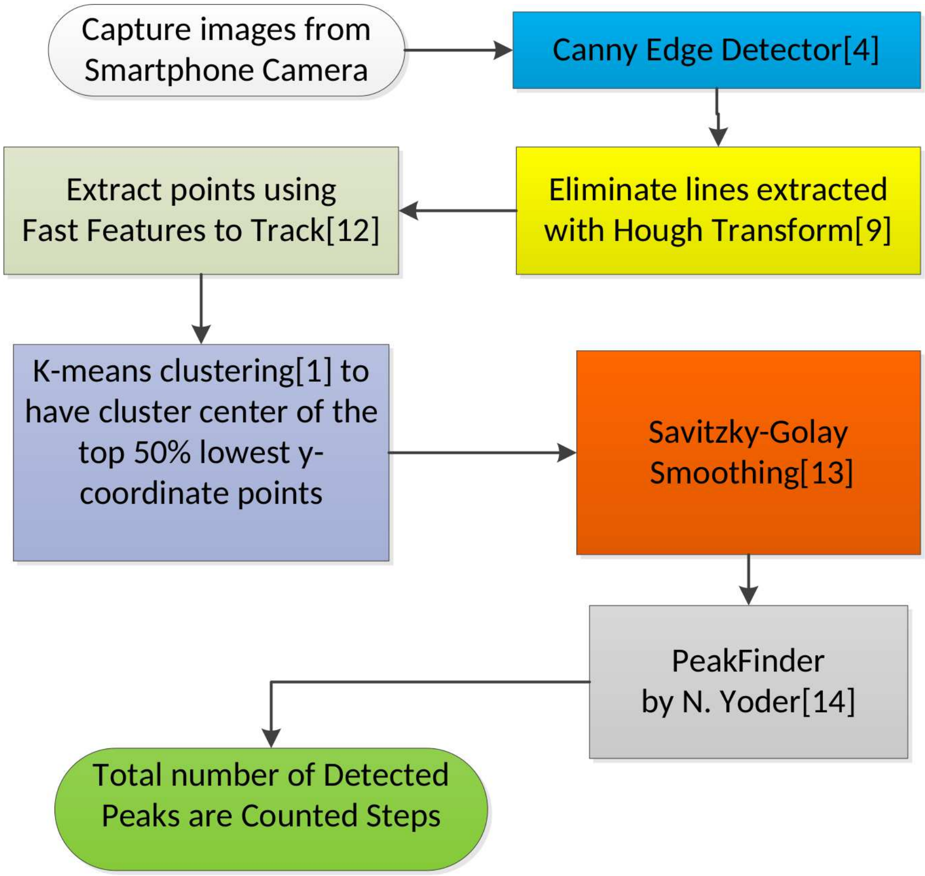

Capture images from Smartphone Camera
Canny Edge Detector[4]
Eliminate lines extracted with Hough Transform[9]
Extract points using Fast Features to Track[12]
K-means clustering[1] to have cluster center of the top 50% lowest y-coordinate points
Savitzky-Golay Smoothing[13]
PeakFinder by N. Yoder[14]
Total number of Detected Peaks are Counted Steps

Fig. 6.3: Flow chart of the proposed camera-based algorithm.

spectively. Then, the FAST features [171] are detected in the image. Since the proposed algorithm has been designed to be ported to actual smart phones, FAST feature point extraction has been chosen due to its' computational efficiency. The detected feature points are marked as red points in Fig. 6.7(d). In order to avoid potential leg points corrupting the cluster center for foot points, 50% of the feature points with the highest x-coordinates are removed (images come from the camera in the orientation seen in Fig. 6.7, and the highest x-coordinate corresponds to the right edge of the image). Then, the cluster center of the remaining detected feature points is calculated with K-means clustering as described in [172]. The cluster center is shown as a green point in Fig. 6.7(e). Removing the points with the highest x-coordinates also allows the algorithm to handle situations when both feet are visible without increasing computational complexity. Since we are in the process of implementing the proposed method in its' entirety on the smart phone, processing efficiency is very important.

The change in the x-coordinate of the cluster center over time is used for step counting. A plot of the x-coordinate values over time is shown in Fig. 6.9(a). In order to reduce the noise in the signal, we apply a low-pass Savitzky-Golay smoothing filter described in [173]. The resulting signal obtained after applying the filter is shown in Fig. 6.9(b). As it can be observed, the filtering removes most of the high frequency noise while preserving the peaks and valleys in the signal. We have compared Savitzky-Golay smoothing filter with other filters as well. It has been observed that Savitzky-Golay smoothing filter preserved more peaks and valleys compared to moving average or local regression-based smoothing filters.

After filtering, we employ a peak detection algorithm by N. Yoder [174]. We have chosen this algorithm, instead of more complex peak detection algorithms, for computational efficiency. This peak detection algorithm [174] requires a percentage of the difference between the maximum and minimum values of the input data as a parameter. In all our experiments, we set this parameter to 20% to differentiate peaks (valleys) from their surrounding data points. The detected valleys can be seen in Fig. 6.9(c). A flow diagram of the proposed method, and a pseudo code for the algorithm are provided in Fig. 6.8 and Alg.3, respectively.

Algorithm 3 Step-Counter Algorithm

```
$stepCounter = 0$
for $all\ frames$ do
    Extract edges with Canny Edge detector
    Detect and eliminate lines with Hough Transform
    Compute features with Fast Features to Track
    Calculate the cluster center of feature points with K-means
    Save $x$-coordinate of the cluster center
end for
Apply smoothing with Savitzky-Golay low pass filter
$stepCounter =$ total number of local minimums (valley)
```

6.3 Experimental Results

We have performed experiments with 10 different subjects walking in an indoor environment with various floor types including carpet and tiled surfaces with light and dark colors. Each subject walked for five minutes while carrying four smart phones and a smart watch simultaneously. Three of the smart phones and the smart watch have accelerometer-based applications for step counting. The watch used in the experiments is a Samsung™ Gear 2 Neo smart watch seen on the wrist of the users in Fig. 6.1(a) and 6.1(b). Three smart phones were carried in a backpack, in front pocket and in back pocket of trousers. The fourth smart phone was held by hand, as seen in Fig. 6.1(b), to capture video data for the proposed method. The captured image size is 320×240 pixels. QVGA image size has been selected in order not to increase the computation requirements. It has also proven to be enough for detecting and counting the steps. Subjects counted the steps they were taking to provide the ground truth values.

These experiments with five devices allowed us to compare the performance of the proposed method with the accelerometer-based counters. In addition, results show the high sensitivity of the accelerometer-based algorithms to the location of the device. Moreover, a large variance is observed across different users with the accelerometer-based step counting compared to our proposed method.

The results of these experiments are presented in Table 6.1 and Table 3.2. As seen from the tables, the proposed method provides the lowest average error (3.064%) for the ten different subjects, and the standard deviation of the error across the subjects is lowest (2.98%) for the proposed method. The standard deviation of the error for the accelerometer-based counters are much higher, and range between 5.45% and 112%. Thus, there is a significant variation in the results of accelerometer-based counters across different users.

Moreover, the results show the high sensitivity of the accelerometer-based algorithms to the location of the device. As seen from Table 6.3, for the same subject, carrying the phone in the front pocket versus back pocket of the trousers makes a significant difference, and this is the case for nine out of the 10 subjects. For instance, for Subject 3, the error rates when carrying the phone in the front pocket and back pocket are 0.7% and 22.5%, respectively. For Subject 8, the rates are 1.15% and 8.3%.

Another point that needs to be emphasized is the following: In all these experiments, the users themselves were not exposed to any acceleration (e.g. being on an elevator, or transportaion vehicle). In those situations, accelerometer-based counters tend to significantly overcount. Even without those scenarios, the proposed method provides the lowest average error, and thus is more robust across different users.

Table 6.1: Comparison of step counting results and their error rates for different sensors and device locations

Subjects	Ground Truth	Proposed Method		Smart phone app Backpack		Smart phone app Front pocket		Smart phone app Back pocket		Smart watch app on wrist	
	Steps	Counted	Error	Counted	Error	Counted	Error	Counted	Error	Counted	Error
Subject 1	400	397	0.75%	1925	381%	372	7%	447	11.75%	438	9.5%
Subject 2	383	402	1.05%	356	7%	439	14.6%	356	7%	441	15.1%
Subject 3	408	416	1.96%	553	35.5%	405	0.7%	500	22.5%	394	3.4%
Subject 4	380	376	1.05%	392	3.2%	354	6.8%	392	3.2%	348	8.4%
Subject 5	529	484	8.51%	526	0.6%	317	40.1%	635	20%	620	17.2%
Subject 6	474	464	2.11%	509	7.38%	458	3.38%	476	0.42%	503	6.12%
Subject 7	527	478	9.3%	557	5.7%	315	39.9%	550	4.4%	524	0.57%
Subject 8	520	511	1.73%	504	3.1%	514	1.15%	563	8.3%	502	3.5%
Subject 9	478	465	2.72%	484	1.3%	459	4%	499	4.4%	465	2.7%
Subject 10	417	436	1.46%	462	10.8%	378	9.4%	471	13%	422	1.2%
			Avg. Err: **3.064%**		Avg. Err: 45.558%		Avg. Err: 12.703%		Avg. Err: 9.497%		Avg. Err: 6.769%

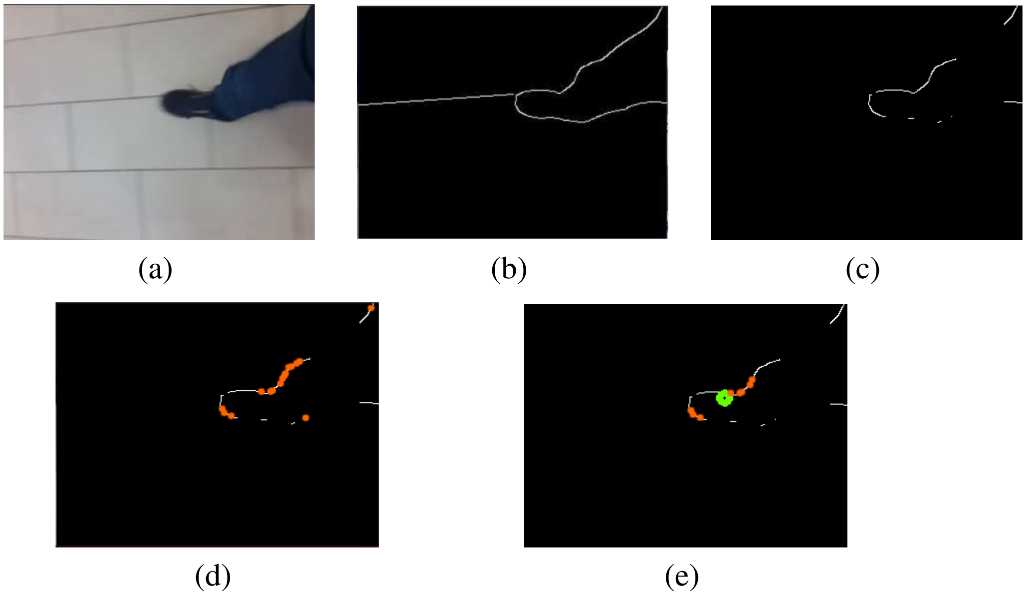
Table 6.2: Comparison of minimum, maximum, and standard deviation of error rates

Methods	Min. Error	Max. Error	Std. Dev.
Proposed Method	0.75%	9.3%	2.9768%
Smart phone App. Backpack	0.6%	381%	112.2232%
Smart phone App. Front Pocket	0.7%	40.1%	14.1857%
Smart phone App. Back Pocket	0.42%	22.5%	6.9167%
Smart Watch App. on Wrist	0.57%	17.2%	5.4478%

6.4 Proposed Method with Kalman Tracking

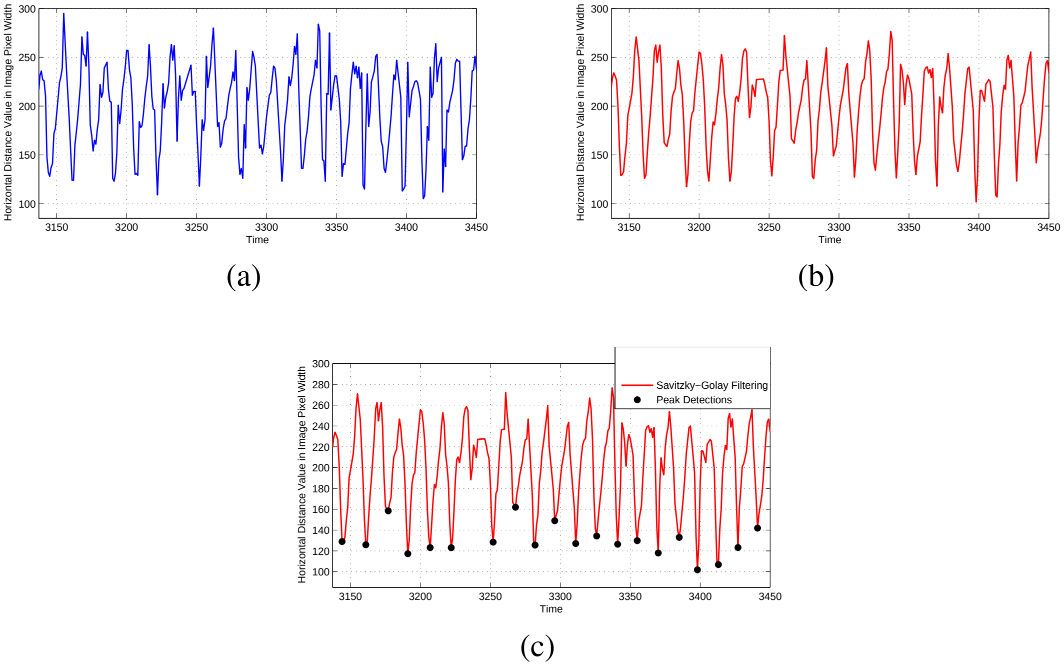
We propose an autonomous method to track and count footsteps by using the camera data from mobile phones or tablets. Users hold the phone as they normally would when they use their phone to check different web pages, read e-mails or run different applications. An image of a subject using the phone can be seen in Fig. 6.1. Since the rear-facing camera points to the floor, captured videos contain the feet and also legs of the subjects. Example images captured during a walk are shown in Figures 6.6 and 6.7 as the user is walking across different surfaces.

The flow diagram of the proposed method is provided in Fig. 6.8. The edges in the image are detected first by using the Canny edge detection [169] algorithm. Then, the FAST


(a) (b) (c)

(d) (e)

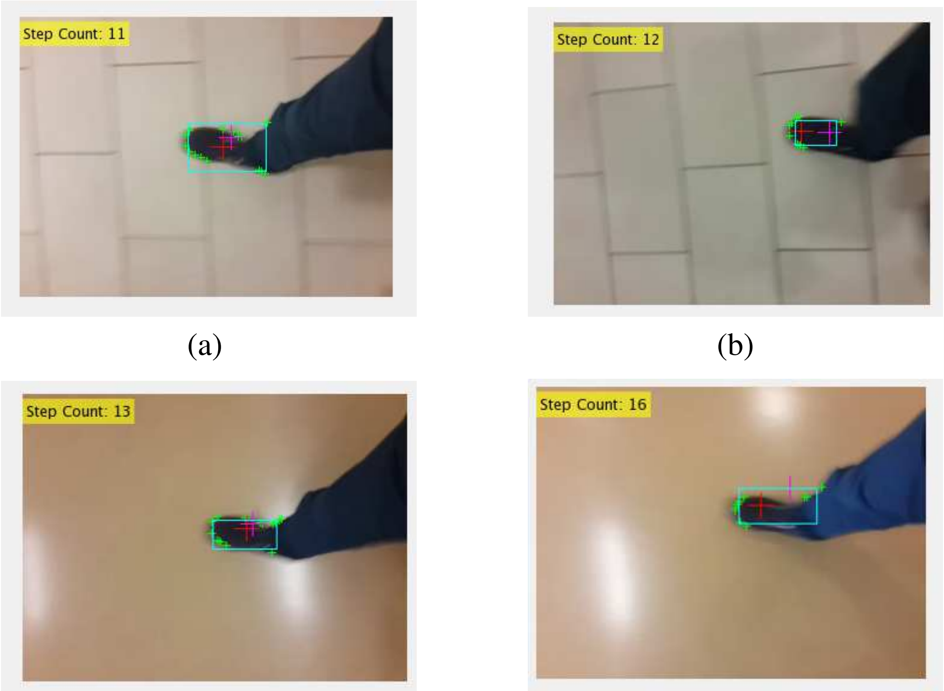
Fig. 6.4: Images showing the stages of processing: (a) Original input image, (b) edge detection output, (c) result after lines are removed, (d) detected FAST features, (e) kept FAST features and the cluster center.



(a) (b)

(c)

Fig. 6.5: Movement of the cluster center in the horizontal direction over time: (a) before smoothing, (b) after Savitzky-Golay smoothing. The detected valleys are shown in (c).



(a) (b)

(c) (d)

Fig. 6.6: Example frames captured during a walk.

features [171] are detected on the resulting image. The images showing the detected edges and the FAST features can be seen in Fig. 6.7(b) and (c). The detected FAST feature points are shown as green '+' marks in Fig. 6.7(c). In order to avoid potential leg points corrupting the cluster center for foot points, 50% of the feature points with the highest x-coordinates are removed (images come from the camera in the orientation seen in Fig. 6.7(d), and the highest x-coordinate corresponds to the right edge of the image). Removing the points with the highest x-coordinates also allows the algorithm to handle situations when both feet are visible without increasing computational complexity. Then, the cluster center of the remaining detected feature points is calculated with K-means clustering [172]. However, it is not very robust to only rely on the cluster center of these feature points. Since the algorithm is employing the FAST features, textured surfaces and the points detected on them can degrade the performance by causing the cluster center to drift away. Thus, in order to increase robustness and accuracy, the cluster center is tracked with Kalman filter tracking [175]. The color histogram of the bounding box, around the kept FAST features, is computed. The color histograms from the two consecutive frames are compared by using the Bhattacharyya distance [176]. If the histograms are similar enough (Bhattacharyya distance is greater than the empirically set threshold), the current cluster center is set as the new observation point. Otherwise, the tracker is not corrupted, and it depends on the previous observation point to predict the current location of the cluster center. The detected cluster center is shown as a red plus sign, and the tracked center location is marked with a magenta plus sign in Fig 6.7(e). As will be discussed more in Section 6.5, the tracking provides improved results and better accuracy compared to our previous work [177].

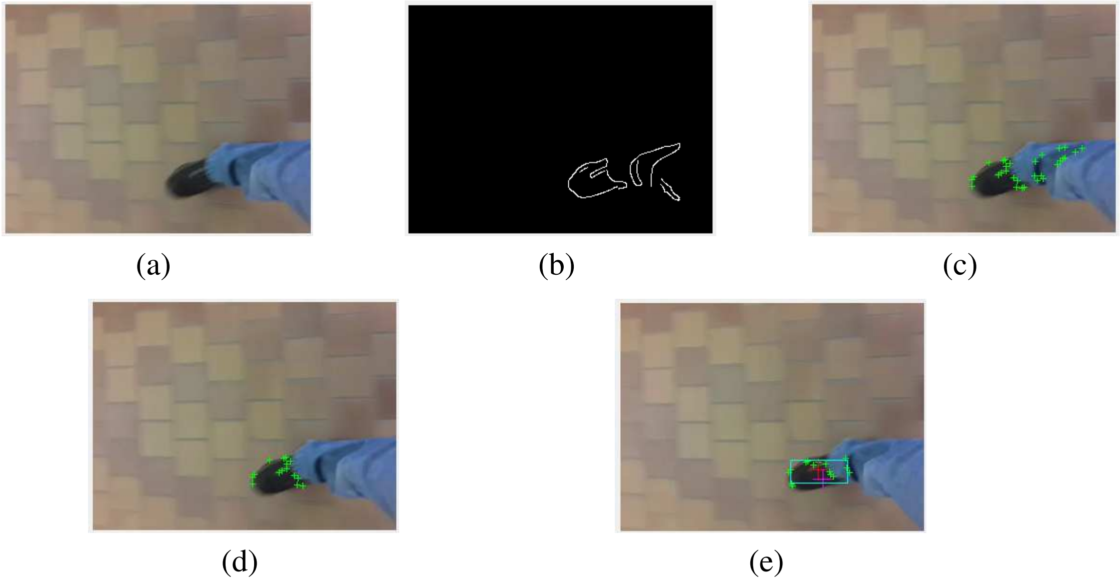

(a) (b) (c)
(d) (e)

Fig. 6.7: Images showing the stages of processing: (a) Original input image, (b) edge detection output, (c) detected FAST features, (d) kept FAST features and the cluster center, (e) actual cluster center, tracked cluster center and bounding box for the foot region.

In order to count the footsteps, the change in the x-coordinate of the tracked cluster center over time is used. A plot of the x-coordinate values over time is shown in Fig. 6.9(a). In order to reduce the noise in the signal, we apply a low-pass Savitzky-Golay smoothing filter described in [173]. The resulting signal obtained after applying the filter is shown in Fig. 6.9(b). We compared Savitzky-Golay smoothing filter with other filters, and observed that Savitzky-Golay smoothing filter preserved more peaks and valleys compared

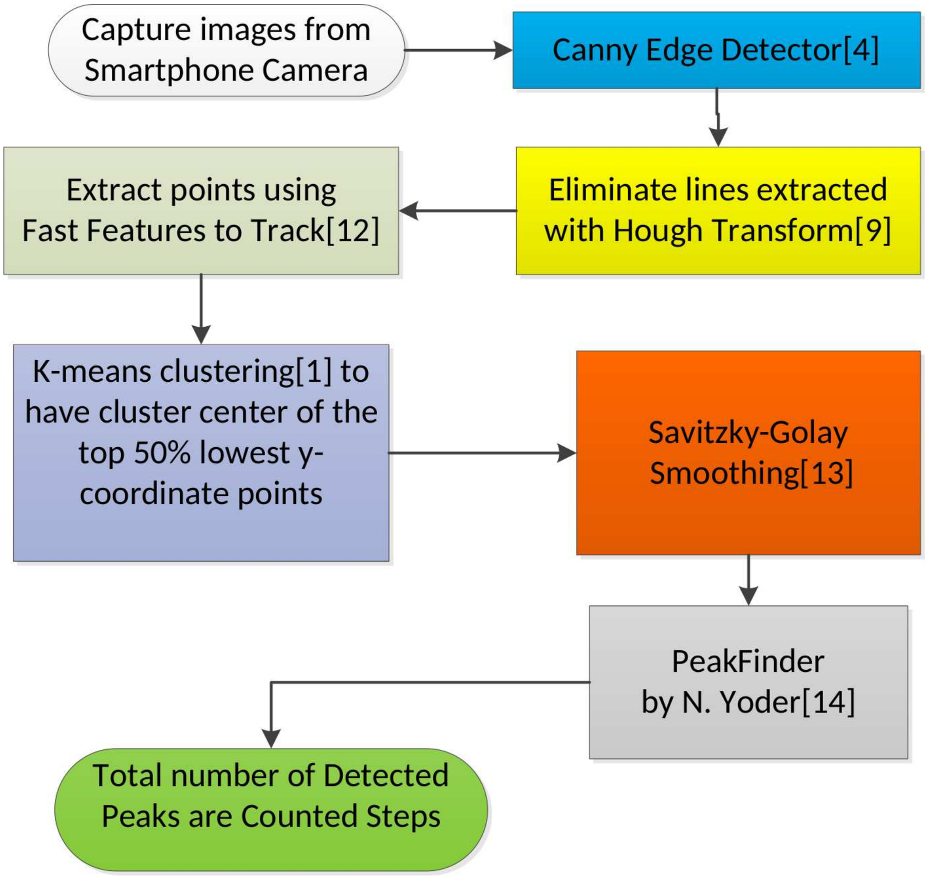
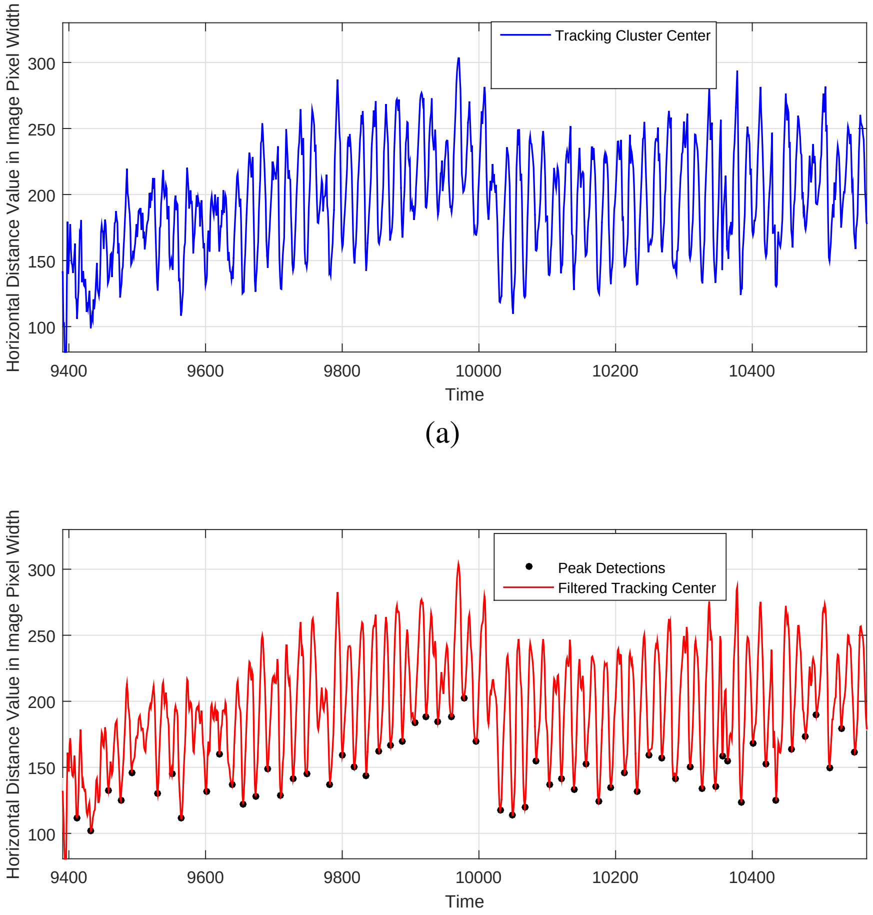

Capture images from Smartphone Camera
Canny Edge Detector[4]
Eliminate lines extracted with Hough Transform[9]
Extract points using Fast Features to Track[12]
K-means clustering[1] to have cluster center of the top 50% lowest y-coordinate points
Savitzky-Golay Smoothing[13]
PeakFinder by N. Yoder[14]
Total number of Detected Peaks are Counted Steps

Fig. 6.8: Flow chart of the proposed camera-based algorithm.

to moving average or local regression-based smoothing filters. After filtering, we employ a peak detection algorithm by N. Yoder [174]. We have chosen this algorithm, instead of more complex peak detection algorithms, for computational efficiency. The detected valleys can be seen in Fig. 6.9(b). The number of steps taken is the total number of valleys in x-coordinate movement of the tracked center.

Example frames captured during the course of experiment are given in Fig. 6.6 and 6.7. The FAST feature points are marked with green plus signs. Actual cluster center of these points is marked with red plus sign while the estimated point location based on the previous observations is marked with the magenta sign. Also, the bounding box is drawn around the kept feature points, which are the lower 50% of all the detected points based on their x-coordinate value.



(a)

(b)

Fig. 6.9: Movement of the cluster center tracked with Kalman filter in the horizontal direction over time: (a) before smoothing, (b) after Savitzky-Golay smoothing. The detected valleys are also shown in (b).

6.5 Experimental Results with Kalman Tracking

We have performed experiments with 10 different subjects walking in an indoor environment with various floor types including carpet and tiled surfaces with light and dark colors. Each subject walked for five minutes while carrying four smart phones and a smart watch simultaneously. Three of the smart phones and the smart watch have accelerometer-based applications for step counting. The watch used in the experiments is a Samsung™ Gear 2 Neo smart watch seen on the wrist of the users in Fig. 6.1(a) and 6.1(b). Three smart phones were carried in a backpack, in front pocket and in back pocket of trousers. The fourth smart phone was held by hand, as seen in Fig. 6.1(b), to capture video data for the proposed method. The captured image size is 320×240 pixels. Subjects counted the steps while they were capturing video to provide the ground truth values. Also, the ground truth values have been verified by watching the captured videos to count the actual number of steps taken.

These experiments with five devices allowed us to compare the performance of the proposed method with the accelerometer-based counters. In addition, results show the high sensitivity of the accelerometer-based algorithms to the location of the device. Moreover, a large variance is observed across different users with the accelerometer-based step counting compared to our proposed method.

In addition, these videos allowed us to compare the step tracking and counting method we propose here with our previous work [177]. The results of these experiments are summarized in Tables 6.3 and 3.4. The proposed method provides the lowest average error (2.68$\%$) for the ten different subjects, and the standard deviation of the error across the subjects is lowest (2.39$\%$) for the proposed method. Incorporating the tracking of the cluster center in the proposed method reduced the average error from 3.064% to 2.68% and the standard deviation of the error from 2.98% to 2.39% compared to our previous work [177]. The standard deviation of the error for the accelerometer-based counters are much higher, and range between 5.45% and 112%. Thus, there is a significant variation in the results of accelerometer-based counters across different users.

Moreover, the results show the high sensitivity of the accelerometer-based algorithms to the location of the device. As seen from Table 6.3, for the same subject, carrying the phone in the front pocket versus back pocket of the trousers makes a significant difference, and this is the case for nine out of the 10 subjects. For instance, for Subject 3, the error rates when carrying the phone in the front pocket and back pocket are 0.7% and 22.5%, respectively. For Subject 8, the rates are 1.15% and 8.3%.

It should be noted that, in all these experiments, the users themselves were not exposed

Table 6.3: Comparison of step counting results and their error rates for different sensors and device locations

Subjects	Ground Truth	Proposed Method		Previous Work [177]		Smart phone app Backpack		Smart phone app Front pocket		Smart phone app Back pocket		Smart watch app on wrist	
	Steps	Counted	Error	Counted	Error	Counted	Error	Counted	Error	Counted	Error	Counted	Error
Subject 1	400	401	0.25%	397	0.75%	1925	381%	372	7%	447	11.75%	438	9.5%
Subject 2	383	382	0.26%	402	1.05%	356	7%	439	14.6%	356	7%	441	15.1%
Subject 3	408	407	0.25%	416	1.96%	553	35.5%	405	0.7%	500	22.5%	394	3.4%
Subject 4	380	381	0.26%	376	1.05%	392	3.2%	354	6.8%	392	3.2%	348	8.4%
Subject 5	529	508	3.97%	484	8.51%	526	0.6%	317	40.1%	635	20%	620	17.2%
Subject 6	474	442	6.75%	464	2.11%	509	7.38%	458	3.38%	476	0.42%	503	6.12%
Subject 7	527	516	2.09%	478	9.3%	557	5.7%	315	39.9%	550	4.4%	524	0.57%
Subject 8	520	487	6.35%	511	1.73%	504	3.1%	514	1.15%	563	8.3%	502	3.5%
Subject 9	478	459	3.97%	465	2.72%	484	1.3%	459	4%	499	4.4%	465	2.7%
Subject 10	417	406	2.64%	436	1.46%	462	10.8%	378	9.4%	471	13%	422	1.2%
			Avg. Err: **2.68%**		Avg. Err: 3.064%		Avg. Err: 45.558%		Avg. Err: 12.703%		Avg. Err: 9.497%		Avg. Err: 6.769%

to any acceleration (e.g. being on an elevator, or being inside a traveling vehicle). In those situations, accelerometer-based counters tend to overcount. Even without those scenarios, the proposed method provides the lowest average error, and thus is more robust across different users.

Table 6.4: Comparison of minimum, maximum, and standard deviation of error rates

Methods	Min. Error	Max. Error	Std. Dev.
Proposed Method	0.25%	6.75%	2.39%
Prev. Work [177]	0.75%	9.3%	2.9768%
Smart phone App. Backpack	0.6%	381%	112.2232%
Smart phone App. Front Pocket	0.7%	40.1%	14.1857%
Smart phone App. Back Pocket	0.42%	22.5%	6.9167%
Smart Watch App. on Wrist	0.57%	17.2%	5.4478%

6.6 Conclusion

An autonomous method has been proposed to track and count footsteps by using the camera data from mobile phones or tablets. The proposed method incorporates Kalman filter tracking, which provides a more robust step counting mechanism even for challenging floor and ground surfaces with detailed textures. This method does not rely on templates, and does not make any assumptions about the duration of a step or the distance between steps. The system performs reliably across different users, such as tall versus shorter people. The performance of the proposed method was compared with our previous work as well as accelerometer-based step counting applications running on smart phones and smart watches. It was shown that the proposed method provides the lowest average error rate (2.68%) for the 10 subjects, and the standard deviation of the error across the subjects is lowest (2.39%) for the proposed method. It has been observed that there is a significant variation in the results of accelerometer-based counters across different users. The standard deviation of the error for the accelerometer-based counters range between 5.45% and 112%.

CHAPTER 7

DOORWAY DETECTION FOR AUTONOMOUS INDOOR NAVIGATION OF UNMANNED VEHICLES

7.1 Introduction

Unmanned ground vehicles (UGVs) and unmanned aerial vehicles (UAVs) have important and wide-ranging application areas including military tasks, search and rescue missions, robotics, surveillance, journalism, inspection of buildings and bridges, and monitoring of wildlife habitats. With their ever decreasing costs, UAVs are becoming increasingly available, and are already being used by many hobbyists.

Camera or vision-based solutions have also been proposed to address the indoor navigation problem. It was shown in the literature that a UAV with an onboard camera can navigate by itself in simple structures like corridors or stairs using different techniques including optical flow [178], image perspective cues [179]. In [178], the focus is on avoidance of collision with walls by using depth map and optical flow. The UAV is navigated through corridors by measuring the distance to the walls on each side and to ground. In

[178] and [179], they do not focus on leaving or entering a room/corridor through door(s).

In this chapter, we present a more general approach for navigation and environment understanding for reliable detection of open doors. Without loss of generality, we focus on doorway or open door detection. However, the approach can be applied to recognizing windows, some architectural structures and obstacle types. We employ a door detector, trained with Aggregate Channel Features (ACF) [2], on the candidate region(s) of RGB images acquired at the same time with the point cloud data. This trained detector is computationally efficient for real-time applications while achieving comparable accuracy in object detection tasks as it was shown for the pedestrian detection problem in [2]. In addition, since the detection is only performed on the candidate regions obtained from the 3D point cloud data, the computational efficiency is even higher and the issue of false positives is alleviated if not eliminated. This step is used for verification and to determine whether it is safe to approach and go through the doorway. Thus, different from [180], we also employ image data and a trained detector to verify whether the detected gaps are indeed doors or not. We do not require the use of predetermined door sizes.

7.2 Proposed Method

The proposed method for doorway detection has two main parts: (i) detecting candidate door openings from the 3D point cloud data, and (ii) using a pre-trained detector on corresponding RGB image regions to verify if these candidate openings are indeed doors.

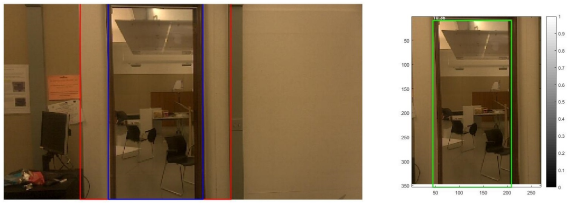
First, dominant planes are detected in the scene by using a RANSAC-based plane estimation algorithm. Then, we search for empty region(s) on the detected plane(s) as the candidate door openings. This step allows us to find openings on dominant planes or detect depth differences, but does not guarantee that the detected opening is a door. Windows, some architectural structures and mirrors can also be detected as openings on dominant planes. Therefore, a verification step is needed to decide whether these are actual open doors. To accomplish this goal, we employ a detector trained with ACF [2]. It runs with a sliding window approach on the RGB image regions corresponding to detected candidate openings. These RGB images are acquired simultaneously with the point clouds by the Google Project Tango™ platform. This trained detector is computationally efficient for real-time applications while achieving comparable accuracy in object detection tasks as it was shown for the pedestrian detection problem in [2]. If a door is detected on this RGB image region, then the door is marked both in the point cloud and the RGB image.

7.2.1 Door Detection on RGB Images for Verification

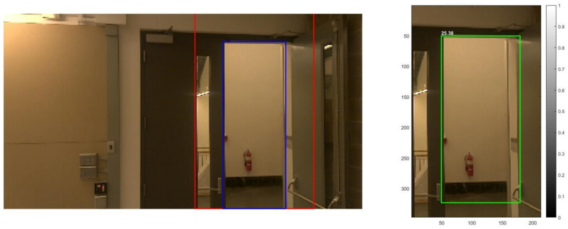
A candidate opening detected from the point cloud does not necessarily correspond to an open door. In fact, scenarios in which no IR reflection is measured can also cause these openings. Examples include windows, mirrors, very shiny wall surfaces under bright sunlight and closely spaced large building columns. Therefore, in addition to the 3D point cloud, we use the RGB images for verification. We trained our ACF-based door detector with sampled door images from databases, such as the one in [181], and search engines results for door images.

After the candidate region is detected, its' corresponding bounding box is obtained in the RGB image, and is padded around all four sides to guarantee that the door frames are visible. The candidate region and the enlarged bounding box are shown in blue and red, respectively, in Figures 7.1 and 7.2. The door detection is performed in the red bounding box only. If a door is detected in this region of the RGB image, then it is verified that the candidate opening is a doorway.

As seen in Fig. 7.1, the ACF-based detector detects the green boxes as doors with high detection scores. Fig. 7.2 shows two scenarios in which the detected openings from the 3D point cloud do not correspond to actual doorways. Our trained door detector applied on the RGB images does not detect a door in these candidate regions. Hence, thanks to this verification step, our ultimate decision is more reliable compared to using only the depth data.

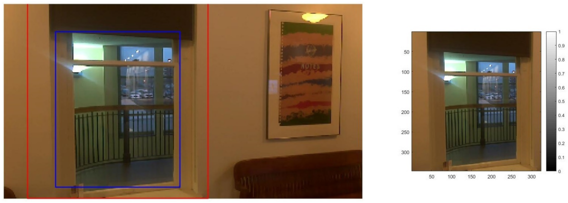


(a) ACF detection score: 10.36

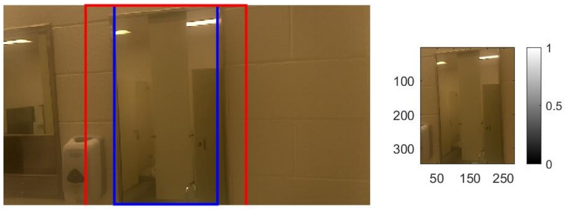


(b) ACF detection score: 25.38

Fig. 7.1: Two doorway detection examples. Blue, red and green boxes represent the candidate region, its' padded version and the detected door, respectively.



(a) Window scene



(b) Mirror scene

Fig. 7.2: Candidate regions that are correctly declared as not being doors (no green box).

7.3 Experimental Results

In our experiments, we analyzed false positive ratio of ACF detector when we do not give any prior candidate region to it. We run ACF detector on a non-door image set. Over 242 images ACF detector gave 1075 false positives. This means that using only ACF results in 4.44 false positive per image on the average. In addition, we analyzed the false positive rate of the ACF detector when it is run on entire RGB images, i.e. when depth data is not used to present the ACF detector with candidate regions. Over 242 non-door images, it detected 1075 false door regions. Table 7.1 summarizes the performance of the ACF-based door detection, more specifically it lists the number of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN), after it is presented with candidate regions. As can be seen, the false positive rate is decreased significantly, which shows the importance of incorporating the depth data and getting candidate regions.

Table 7.1: ACF detector performance

Type	Count	ACF (door detected)	ACF (no door)
Open Doors	50	43 (TP)	7 (FN)
Other (mirror, window etc.)	40	3 (FP)	37 (TN)

We also compared the processing times of applying the door detection across entire RGB images versus just applying it in the candidate regions. Over 457 images, the average processing times per frame are 0.0258s and 0.0133s, respectively. Thus, detecting the candidate door openings from 3D point clouds and using these regions provide 48% savings in the processing time of ACF-based door detection.

7.4 Conclusion

We have proposed a novel approach to autonomously detect doorways/open doors for indoor navigation of unmanned vehicles. We employ a Project Tango™ tablet by Google™ as our onboard system to be installed on unmanned vehicles. We first detect the candidate door regions or openings from the 3D point cloud, and then use a pre-trained detector, based on ACF, on corresponding RGB image regions to verify whether the detected gaps/openings are indeed doors or not. ACF-based detection is computationally efficient for real-time applications. In addition, since the detection is only performed on the candidate regions obtained from the point cloud, the issue of false positives is alleviated if not eliminated.

CHAPTER 8

REAL-TIME TRAFFIC SIGN DETECTION FOR LOW QUALITY MOBILE CAMERAS WITH IMPROVED AGGREGATE CHANNEL FEATURES

8.1 Introduction

Accurate traffic sign detection is a challenging task, especially with lower quality videos captured by vehicle-mounted, mobile cameras. With the vast availability and ever-decreasing cost of mobile cameras and smart phones, it has now become less expensive and more feasible to mount them on vehicles for applications including collusion prevention systems, traffic sign detection for driver assistance and warning, autonomous driving etc. Most vision-based algorithms developed so far are not optimized for naturalistic real-world settings. They suffer from varying lighting conditions, noisy images as well as varying shadows in naturalistic scenarios. Therefore, in this chapter, we focus on naturalistic videos captured from vehicle-mounted mobile cameras.

Aggressive and distractive driving along with sudden driver maneuvers are the most probable causes of traffic collisions. Therefore, an analysis of driver behaviors under naturalistic real-world scenarios is needed for evaluating dangerous traffic incidents. With such motivation, Strategic Highway Research Program (SHRP-2) has been established. SHRP-2 has created a Naturalistic Driver Study (NDS) database that includes very deep level of detail on driver performance, behavior, and environment related with critical incidents and almost-accidents for 2360 drivers during a period of one year [182].

Mathias et al. [183] investigated solutions to traffic sign recognition for datasets captured in Belgium [184] and Germany [185]. They used four different datasets for evaluation, and obtained accuracy rates between 95% and 99%. However, the type of traffic signs used in the U.S. are different from the ones in Europe. Previously evaluated datasets include signs varying in shape, orientation, and type. However, they do not include examples and data covering varying weather conditions, and daytime as well as nighttime scenarios. They also employ higher resolution images in general, having very limited number of blurry traffic signs. In this chapter, different from previous works, our concentration is on lower quality videos captured under naturalistic scenarios covering various lighting and weather conditions.

Convolutional Neural Networks (CNNs) have received a lot of attention recently, especially after achieving a very good performance in the ImageNet challenge [186]. Later, Girshick et al. [1] combined region proposals with CNNs, and introduced Region-based Convolutional Neural Networks (R-CNN), regions with CNN features, for object detection. Then, Fast R-CNN [187] was proposed, which is one of the most recent works that provides state-of-the-art performance in object detection. These methods are computationally more expensive, and often require a GPU for faster training and processing times.

Dollar et al. [2] introduced the Aggregate Channel Features (ACF). ACF-based detectors can be trained and tested much faster than R-CNN based detectors. However, as will be seen in Section 8.3, for true positive values greater than 0.9, they tend to result in higher false positive rates compared to R-CNN detectors.

In this chapter, we present a new method, which incorporates ACF-based detection and chain code histograms (CCH), with the goals of (i) providing much faster training and testing, and comparable or better performance, when compared to R-CNN based detectors, and (ii) not requiring specialized processors. We compare the performance of the proposed method with two other detectors, namely a pure ACF-based detector and an R-CNN-based detector, both in terms of accuracy, through receiver operating characteristic (ROC) curves, and processing time. As mentioned previously, we focus on naturalistic videos. We have trained the detectors on the same datasets composed of images from the publicly available LISA dataset [188] as well as images from lower resolution videos, in the SHRP-2 dataset [182], captured from moving vehicles. Training dataset was selected according to the need of having naturalistic traffic signs captured in real-life settings and were annotated for experiments. The detectors were then tested on 37 test videos covering a range of different scenarios including daytime and nighttime videos, and varying weather conditions such as cloudy and sunny days.

The experimental results show the promise of the proposed method, which provided the highest true positive rate for lower false positive values, and a faster performance compared to the R-CNN-based detector.

8.2 Proposed Method

For the purpose of accurately detecting traffic signs from lower quality videos, captured by mobile cameras, we propose a new method that combines an ACF-based detector with the chain code histograms (CCHs) as the shape descriptor. CCHs, proposed by Iivarinen and Visa [3], provide a scale and translation invariant shape descriptor for the contours of objects in binary images. It has been shown that printed letters can be differentiated and grouped based on their CCHs.

Aggregate channel features include channels of gradient, HOG and LUV colors [2]. We first train an ACF-based detector, and build the CCHs for four different shapes corresponding to triangle, diamond, rectangle and octagon as seen in the last row of Fig. 8.1. These histograms are built using eight directions, and thus are composed of eight bins. They are normalized to make the shape descriptor scale invariant.

During the testing stage, the ACF-based detector provides the candidate regions with corresponding detection scores. In order to increase the robustness of the traffic sign detector, and decrease the number of false positives, a shape descriptor is used to compare the shape of these candidate regions against the four traffic sign shapes mentioned above. The CCHs are used as the shape descriptors and the CCH obtained from the candidate region is compared to the four CCHs, corresponding to triangle, diamond, rectangle and octagon shapes.

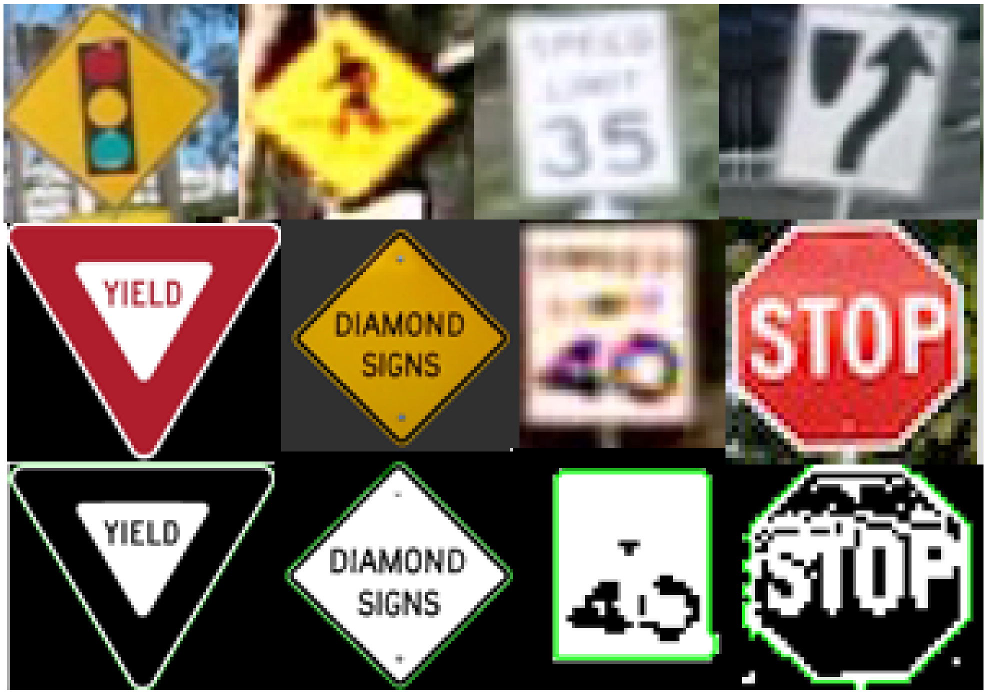


Fig. 8.1: First two rows: Examples images used for training; third row: Four traffic sign shapes used as templates to match the chain code histograms.

8.2.1 Training Stage

In this work, we use a decision tree of depth 3. Total number of training stages are 5 and the final stage has 4096 trees. The number of training images, used for training the detectors, range from 30 to 130 depending on the traffic sign. While the annotated traffic sign regions are used as the positive training samples, the rest of the image is sampled to obtain the negative windows. During training, 500 negative samples were extracted from each image, and the total number of negative samples was limited to 40K. Also, the maximum accumulated negative samples across stages of training has been set to 80K. Since a high number of negative samples is used for better classification, the depth of the decision tree is selected to be 3 for better detection performance. Some example images, used for training the detectors, are displayed in the first two rows of Fig.8.1.

8.2.2 Testing Stage

Steps of the proposed algorithm, during testing stage, are provided in Algorithm 4. First, the ACF-based detector is applied resulting in candidate regions on an image with corresponding detection scores. In order to increase the robustness of the traffic sign detector, and decrease the number of false positives, a shape descriptor is used to compare the shape of these candidate regions against the four traffic sign shapes mentioned above. Every proposed candidate region is converted into a binary image using Otsu's thresholding method [189]. Then, the longest boundary is extracted by using the method described in [190]. Extracted boundaries are shown in green color in Fig. 8.1. Based on the pixel locations of the object boundary, chain codes are computed using 8 different directions starting from the bottom left corner. When a continuous chain code is calculated, we build an 8-bin histogram of the chain codes based on the frequency of occurrence of each direction. The CCH is used as the shape descriptor of the corresponding object proposal.

There are four different traffic sign shapes in our dataset, namely triangle, diamond, rectangle and octagon as seen in the last row of Fig. 8.1. The extracted boundary of each

Algorithm 4 Traffic Sign Detection Algorithm

```
for $All\ Frames\ in\ Videos$ do
    Detect candidate regions using ACF-based detector
    Convert detected regions into grayscale
    Apply Otsu's thresholding to obtain binary image [189]
    Extract the longest boundary [190]
    Calculate the CCH [3]
    if Diss. dist. to any of the four template CCHs $\leq 0.1$ then
        Save detected bounding box and its' score
    else
        Assign detection score as zero.
    end if
end for
```

shape, marked as a green contour, and its' corresponding CCH are saved as templates for comparison with the detected candidate regions. In the final stage of Algorithm 4, we compare the CCH of the detected candidate region with all of the four template CCHs. For comparing the similarity of the CCHs, we employ the dissimilarity distance in Eq. 8.1, wherein r and s are N-dimensional histogram vectors. As can be seen 0 corresponds to identical histograms, whereas 1 represents high dissimilarity. If the dissimilarity score to any of the four templates is less than 0.1, we keep the candidate region together with its' ACF-based detection score. However, when the dissimilarity score to all four templates is higher than 0.1, the detection score is set to be 0 so that false positive regions with high detection scores are eliminated. For instance, with the ACF-based detector, cars might be detected as traffic signs as it can be observed in the first column of Fig. 8.3. Comparing the CCHs with template shape descriptors allows us to successfully suppress possible false positives, that might have relatively high detection scores, as seen in the second column of Fig. 8.3.

$$D = 1 - \frac{\sum_{i=0}^{N-1} (r_i - \bar{r})(s_i - \bar{s})}{\sqrt{\left[\sum_{i=0}^{N-1} (r_i - \bar{r})^2 \sum_{i=0}^{N-1} (s_i - \bar{s})^2\right]}}, \bar{r} = \frac{1}{N}\sum_{i=0}^{N-1}(r_i),\ \bar{s} = \frac{1}{N}\sum_{i=0}^{N-1}(s_i). \tag{8.1}$$

To evaluate the traffic sign detection performance on videos, we used the intersection over union criteria similar to the Pascal Visual Object Classes (VOC) [191] challenge. Whenever the intersection over union (IoU) is greater than 0.5, we count these detections as true positives. When IoU criteria is smaller than 0.5, it is considered as a false positive. This evaluation criteria is given in Eq. (8.2), where B_d and B_{gt} represent the bounding boxes of the detected region and the ground truth, for the objects, respectively. To test the videos for detection performance, we used intersection over union criteria for evaluation similar to the Pascal Visual Object Classes (VOC) [191] challenge as given in Eq. 8.2. B_p represents the bounding box of the proposed detection region while B_{gt} represents the bounding box of the ground truth for the objects. when the proposed detection B_d is not overlapping with the ground truth bounding box of B_{gt},

$$IoU = \frac{area(B_d \cap B_{gt})}{area(B_d \cup B_{gt})} \tag{8.2}$$

8.2.3 Model Training employing Fast-RCNN

In this section, we provide a summary of the Fast-RCNN based detector, with which we compare our proposed method. The training method for RCNN involves three main tasks: object localization, feature extraction and classification. It begins with localization by generating class-independent region proposals with an algorithm called Selective Search [192] and it is preferred over Region Proposal Network(RPN) [193] due to its' performance on low resolution images. Then, it extracts Deep Convolutional Neural Network (DCNN) feature descriptors on the proposed regions after warping them to a fixed square size (256 x 256). Finally, each proposed region is counted as a detection with a bounding box and corresponding score.

8.2.4 Object Localization with Selective Search

There are two main traditional approaches for object localization in images: segmentation and exhaustive search. Segmentation tries to break a single partitioning of an image into its' unique objects before any recognition. This is sometimes extremely hard if there are disparate hierarchy of information in the image. Other researchers resort to localize objects through recognition by performing an exhaustive search within the image by using mostly sliding window approaches. Exhaustive search however fails to detect objects with low-level cues.

Uijlings et al. [192] developed the Selective Search, an approach which combines the best of both worlds: segmentation and exhaustive search. It exploits the hierarchical structure of the image (segmentation) with the aim of generating all possible object locations (exhaustive search). The algorithm uses hierarchical grouping to deal with all possible object scales. Then, the color space of the image is used to deal with different invariance properties. Finally, region-based similarity functions are used to deal with the diverse nature of objects. The final algorithm is fast and accurate, more specifically within 4 seconds it can generate 2,134 boxes with an Average Best Pascal Overlap score of 0.804. The reader is referred to [192] for a more detailed description of selective search.

Feature Extraction and Classification

After obtaining candidate region proposals with selective search, each region is fed through a DCNN for classification. The key algorithms of DCNN can be traced back to the late 1980s [194]. DCNNs saw heavy use in the 1990s. Interest in DCNNs was rekindled again in 2012 by when Krizhevsky ey al. [195] showed that substantially higher image classification accuracy could be achieved in the ImageNet dataset with DCNNs. Since then, profound improvements in the accuracy of object detection in complex scenes have been achieved. In this work, a DCNN classifier is built to support key algorithms for traffic scene understanding in SHRP2 video data.

After obtaining candidate region proposals with selective search, each region is fed through a DCNN for classification. DCNN models are computationally expensive, which could be a problem for practical applications. The recent interest in DCNNs could be attributed to the rise of efficient GPU implementations such as cuda-convnet [195], Torch [196] and Caffe [197]. In this work, a GeForce GTX Titan X GPU is used for model training and processing of SHRP2 videos. Model training involves two main steps: Supervised pre-training and Domain-specific fine-tuning.

(a) Supervised pre-training: We adopt a similar approach by pre-training our CNN model on a large auxiliary dataset (ILSVRC2012) using image-level annotations. The resulting output is a rich feature detector which will later be fine-tuned to suit our purposes. Open source Caffe CNN library was used for the pre-training model on 100 classes at a learning rate of 0.01.

DCNN models usually consists of thousands of parameters and millions of learned weights. This means that a very large training dataset (more than a million) will be required to avoid over-fitting the model. However, it has been proved [198] that when labeled data is scarce, supervised pre-training for an auxiliary task with large training data followed by domain-specific fine-tuning could yield significant boost in performance.

(b) Domain-specific fine-tuning: To adapt the pre-trained model to the proposed task (detecting the traffic signs in SHPR2 data), the CNN model parameters are fine-tuned. First, the 100-way classification layer of the pre-trained model is replaced with 11 classes. We start Stochastic Gradient Descent (SGD) at a learning rate of 0.001, which allows fine-tuning to make progress while not clobbering the initialization. In each SGD iteration, we uniformly sample 20 positive windows for all classes and 70 background windows to construct a mini-batch of size 90. DCNN is used to extract a 4096 dimensional feature vector using Caffe implementation of CNN by Krizhesky [197]. Each mean subtracted candidate region proposal is forward propagated through a network with five convolutional layers and two fully connected layers. The modeling architecture is explained as follows: (1) Each class-independent region proposal from the previous step is warped to a 256 x 256 image; (2) the input warped image is filtered with 96 kernels of size 11X11, with a stride of 4 pixels. This is followed by max pooling in 3x3 grid; (3) two subsequent convolutions with 384 kernels are carried out without pooling; (4) output of fourth layer is convolved with 256 kernel, then spatial max pooling is applied in 3x3 pixel grid; (5) last 2 layers: Fully connected layer of 4096 dimensions from the last layer.

1. Each class-independent region proposal from the previous step is warped to a 256 x 256 image.
2. The input warped image is filtered with 96 kernels of size 11X11, with a stride of 4 pixels. This is followed by max pooling in 3x3 grid.
3. Two subsequent convolutions with 384 kernels are carried out without pooling
4. Convolve output of fourth layer with 256 kernel, then apply spatial max pooling in 3x3 pixel grid.
5. Last 2 Layers: Fully connected layer of 4096 dimensions from the last layer.

8.3 Experimental Results

We compared the performance of the proposed method with two other detectors both in terms of detection accuracy as well as processing speed. Henceforth, we will refer to our proposed method, incorporating a shape descriptor, as Shape-ACF. The other two detectors will be referred to as ACF and Fast-RCNN based detectors. A total of 37 videos have been used for testing and comparing the performances. Test videos cover a range of scenarios, including weather conditions varying from sunny to cloudy, and different times of the day such as daytime and nighttime. Some example images with detections and false positives from the test videos can be seen in Figures 8.3 and 8.4.

We have obtained the ROC curves for all detectors to provide a comparison of the true positive and false positive rates. ROC curves are very commonly used to compare the performance of different detectors. A graph of true positive vs. false positive(FP) rates provides a comprehensive comparison on a single plot. To obtain the ROC curves, we have used all of the 37 test videos. The ROC curves for the Shape-ACF-based, ACF-based and Fast RCNN-based detectors are presented in Fig. 8.2. The ideal operating point here is the upper left-hand corner corresponding to a true positive rate of 1 and false positive rate of 0. As can be seen, overall, the proposed method (shown in solid red plot) operates closer to the upper left-hand corner. For FPs less than 0.15, the proposed method (Shape-ACF) provides the highest true positive rate among the three detectors. For FPs smaller than 0.13, both Shape-ACF, and ACF provide higher true positive rates than Fast-RCNN.

As mentioned above, true positives corresponds to detections that satisfy the intersection over union criteria, provided in Eq. (8.2) with respect to the annotated ground truths. The detections that do not satisfy Eq. (8.2) are counted as false positives.

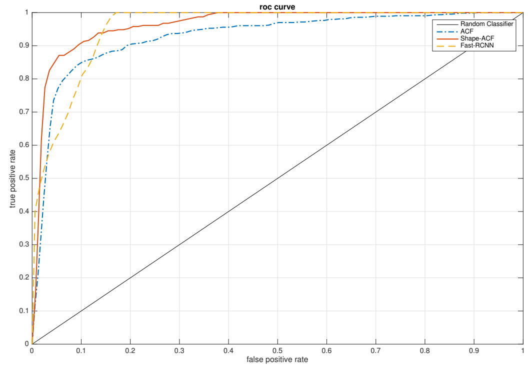


Fig. 8.2: ROC curves comparing the proposed detector (Shape-ACF) with two other detectors.

The processing times on a CPU for the Shape-ACF, ACF and Fast-RCNN-based detectors are 0.15s, 0.09s, and 12s, respectively, as presented in Table 8.1. These are the times that it requires for each detector to process a single image of size 458x356. As can be seen, the proposed method (Shape-ACF) performs much faster than Fast RCNN on a CPU providing 80× speed up. This makes the proposed method more suitable for CPU and/or embedded platform implementations. Compared to ACF, the processing time does not increase significantly. The improvement in the performance provided by the proposed method (shown in Fig. 8.2) justifies the slight increase in the processing time compared to

ACF.

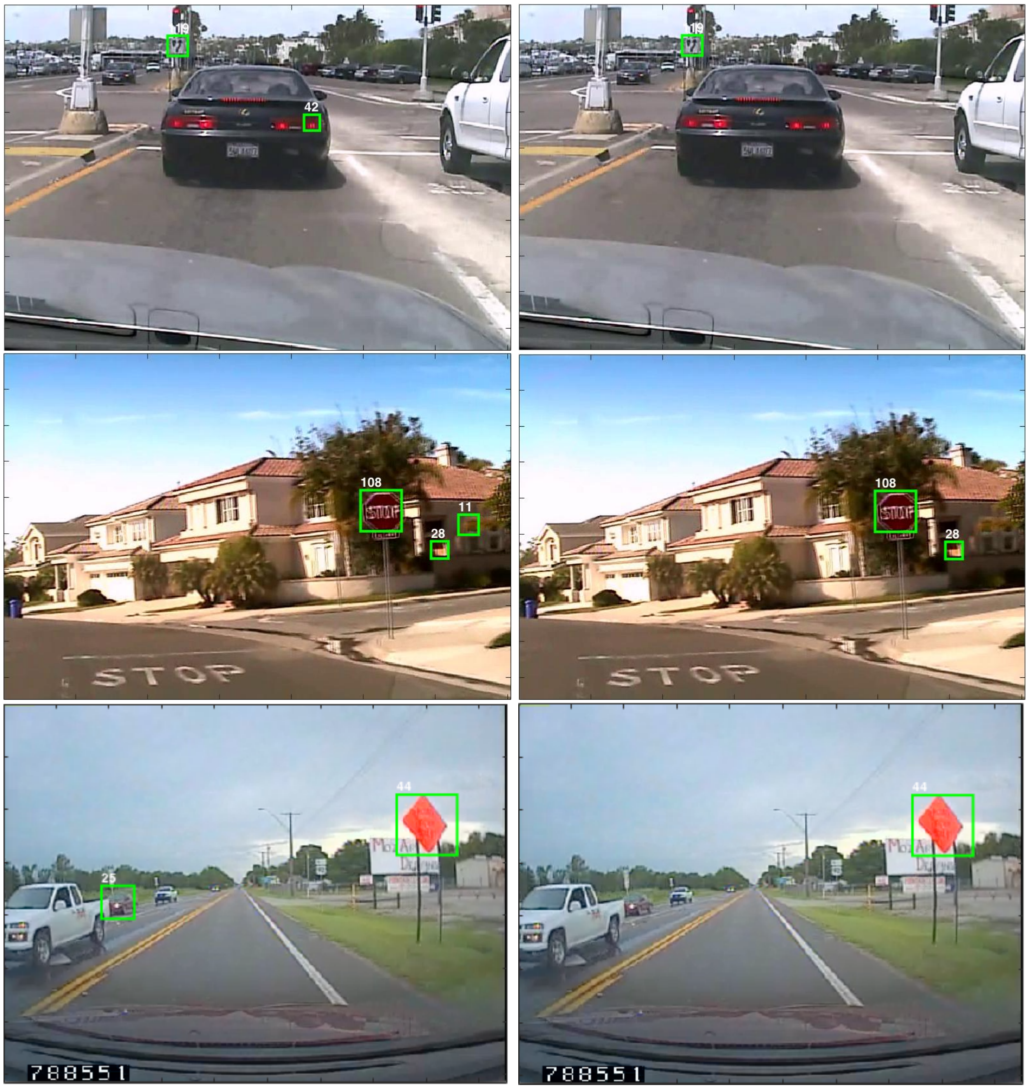
Table 8.1: Comparison of computation efficiency on CPU

Method	Processing Time on CPU
Shape ACF	0.15s
ACF	0.09s
Fast-RCNN	12 s

We used performance tables reporting true positives and false positives for evaluating the performance on the test videos. Also, corresponding processing times on CPU are reported for both Fast-RCNN and ACF results in Table 8.1. The detector using ACF has advantage for processing on CPU compared to the detector trained with Fast-RCNN [187]. On the other hand, the detector with Fast-RCNN is optimized for processing on GPU and can reach up to 25-30 fps. The results on Table 8.1 presents the corresponding time it requires for each detector to process a single a image of resolution 458×356. For processing on CPU, the detector trained ACF, which is based on sliding window approach on multiple scales to search for suitable candidate regions, is much more suitable for CPU implementations.

We present detection performances of two detectors for our traffic sign detection for low quality videos captured. In the experimental results section, the receiver operating characteristics gives a convenient comparison between true positives and false positives for proposed detectors. Also, example images showing the detections of proposed detectors are also presented for qualitative evaluation.

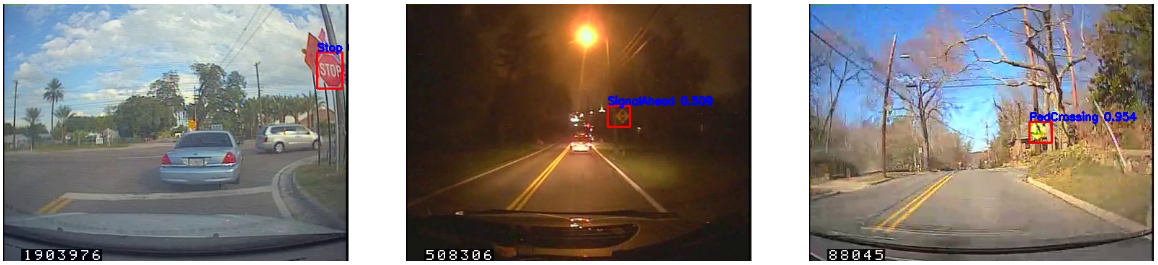
For consecutive occurrence of ground truth bounding boxes, at least a single correct detection of the bounding box is good enough for classifying the corresponding traffic sign as detected. In other words, we are not counting the bounding boxes as false positives even if they do not satisfy the intersection over union criteria Eq. 8.2 but they are overlapping with the correct bounding box. For the detectors proposed, we are presenting the overall detector performances in Fig. 8.2. Traffic sign detector trained with Fast-RCNN[187] reached higher accuracy. On the other hand, the detector trained with ACF [2] provides higher true positive rate for false positives rates up to 0.15.



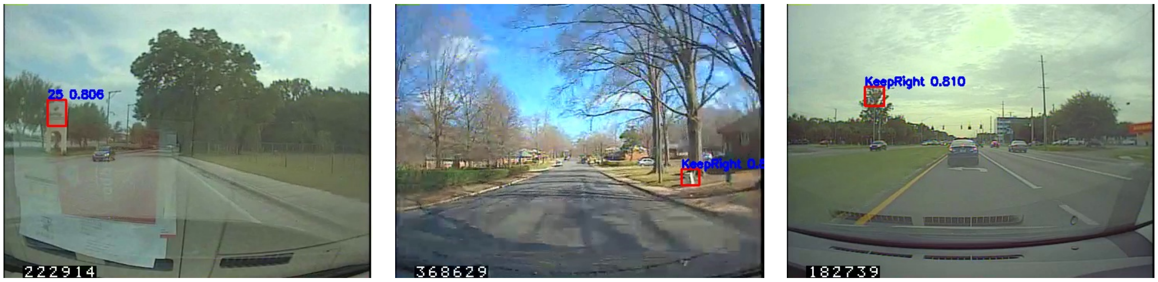
(a) Outp. of ACF-detector (b) Proposed Detector

Fig. 8.3: Example detection results of the ACF-based detector and the proposed detector (Shape-ACF).

Figures 8.3 and 8.4 show example detection results for different detectors. As seen in the second column of Fig. 8.3, the proposed method (Shape-ACF) eliminates the FPs created by the ACF-based detector (first column of the figure) by incorporating a shape descriptor based on CCHs.



(a) Stop Sign (b) Warning (Night) (c) Ped. Crossing



(d) (e) (f)

Fig. 8.4: (a,b,c) Example detections results of the detector trained with Fast-RCNN, (d,e,f) and false positives.

8.4 Conclusion

In this chapter, we have focused on naturalistic videos of U.S. traffic signs, proposed a new, robust and efficient detector, incorporating ACF-based detection with a shape descriptor, and compared its' performance with two state-of-the-art detectors, namely Fast-RCNN and ACF-based detectors, on lower resolution videos. The videos used for testing include a range of different scenarios including daytime/nighttime videos, and varying weather conditions such as cloudy, sunny, and bright days. We have provided ROC curves for all the detectors as well as example visual detections on test videos. The proposed method provided the highest true positive rate for lower FP values while performing much faster than Fast-RCNN on CPU. We have built a brand new dataset for training the detectors, which contains traffic signs in lower resolution videos captured with vehicle-mounted, mobile cameras. As a future direction, we are in the process of implementing Shape-ACF detector to be employed on smart phones that are simply attached dashboards.

CHAPTER 9

CONCLUSION AND FUTURE WORK

Mobile cameras provide wide-ranging and beneficial applications for the society. In this dissertation, we have reported on our algorithms and methodologies on different mobile camera applications. We covered various applications of mobile cameras with possible addition of motion sensors, such as accelerometers, and infrared depth cameras for improved detection and classification capabilities. Proposed applications include fall detection, activity classification and footstep counting using cameras of mobile devices, doorway verification for UAVs, and traffic sign detection from lower-resolution videos. The proposed algorithms and solutions presented in this dissertation can be utilized in healthcare, activity monitoring, driving assistance, and autonomous driving of unmanned aerial and ground vehicles.

The proposed algorithms were designed to be implemented on mobile platforms, and the fall detection algorithm described in Chapter 4, incorporating camera and accelerometer data, was implemented in its' entirety on an actual smartphone. Other applications presented in the remaining chapters can also be efficiently implemented to run on mobile devices. Some applications as presented in Chapters 5 and 7 require higher processing capabilities for real-time implementation purposes for now. As computation capabilities of mobile platforms increase, presented algorithms along with many other computer vision algorithms can be implemented to run on mobile devices.

9.1 Future Work

Activity classification work can be extended to detect and classify other types of daily activities of people. Unsupervised machine learning algorithms can be employed to classify types of activities such as walking, running, sitting or laying down, using the stairs, taking the elevator etc. Also, mobile camera and depth sensor may be combined to guide people with disabilities in their daily living. For instance, camera may provide real-time contextual information regarding surroundings of a person, and a depth sensor can provide distance towards the objects and walking direction estimation for blind people to find their way within their living environments.

Traffic sign detection work may be extended to classify the type of the traffic sign based on the content of the detection. An example classification result is provided below in Fig. 9.1. Moreover, we want to be able to detect and classify every type of object that is visible within the view of mobile camera installed in a vehicle. These cameras can also provide real-time information about the road conditions. Especially the effect of weather conditions on the road might be useful for state authorities to concentrate on the road regions that need the most urgent snow removal, salting etc.

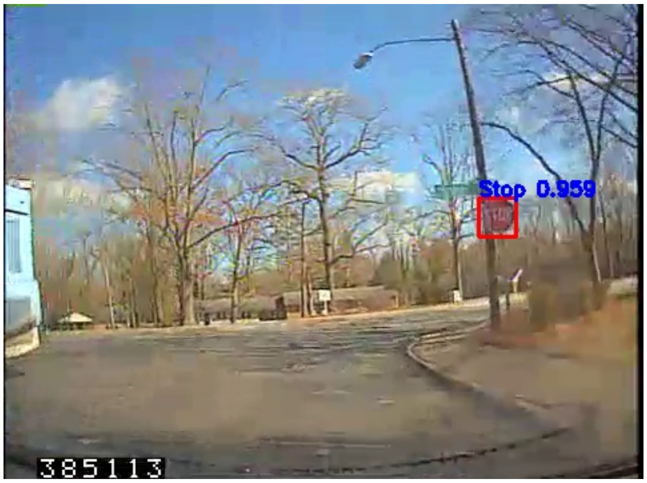


Fig. 9.1: Example stop sign that is detected and classified with its' type.

REFERENCES

[1] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Computer Vision and Pattern Recognition*, 2014.

[2] P. Dollar, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 8, pp. 1532–1545, Aug 2014.

[3] Jukka Iivarinen and Ari Visa, "Shape recognition of irregular objects," in *Intelligent Robots and Computer Vision XV: Algorithms, Techniques, Active Vision, and Materials Handling, Proc. SPIE 2904*, 1996, pp. 25–32.

[4] P. Chen et al., "Citric: A low-bandwidth wireless camera network platform," in *Proc. of the ACM/IEEE International Conference on Distributed Smart Cameras*, 2008, pp. 1–10.

[5] "www.census.gov," .

[6] Oscar D. Lara and Miguel a. Labrador, "A Survey on Human Activity Recognition using Wearable Sensors," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1192–1209, 2013.

[7] Barbara Bruno, Fulvio Mastrogiovanni, Antonio Sgorbissa, Tullio Vernazza, and Renato Zaccaria, "Analysis of human behavior recognition algorithms based on

acceleration data," *IEEE Int'l Conf. on Robotics and Automation*, pp. 1602–1607, 2013.

[8] Saisakul Chernbumroong, Shuang Cang, Anthony Atkins, and Hongnian Yu, "Elderly activities recognition and classification for applications in assisted living," *Expert Systems with Applications*, vol. 40, no. 5, pp. 1662–1674, 2013.

[9] Jens Windau and Laurent Itti, "Situation awareness via sensor-equipped eyeglasses," *IEEE Int'l Conf. on Intelligent Robots and Systems*, pp. 5674–5679, 2013.

[10] Jhun Ying Yang, Jeen Shing Wang, and Yen Ping Chen, "Using acceleration measurements for activity recognition: An effective learning algorithm for constructing neural classifiers," *Pattern Recognition Letters*, vol. 29, no. 16, pp. 2213–2220, 2008.

[11] Stefan Dernbach, Barnan Das, Narayanan C. Krishnan, Brian L. Thomas, and Diane J. Cook, "Simple and Complex Activity Recognition through Smart Phones," *8th Int'l Conf. on Intelligent Environments*, pp. 214–221, 2012.

[12] D. Damen, A. Gee, W. Mayol-Cuevas, and A. Calway, "Egocentric real-time workspace monitoring using an rgb-d camera," in *IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2012, pp. 1029–1036.

[13] Kai Zhan, Fabio Ramos, and Steven Faux, "Activity recognition from a wearable camera," *Int'l Conf. on Control, Automation, Robotics and Vision, ICARCV*, vol. 2012, no. December, pp. 365–370, 2012.

[14] Michael S. Ryoo and Larry Matthies, "First-person activity recognition: What are they doing to me?," *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 2730–2737, 2013.

[15] Alireza Fathi and James M. Rehg, "Modeling actions through state changes," *Proc. of the IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, pp. 2579–2586, 2013.

[16] Alireza Fathi, Ali Farhadi, and James M. Rehg, "Understanding egocentric activities," *Proc. of the IEEE Int'l Conf. on Computer Vision*, pp. 407–414, 2011.

[17] K. Matsuo, K. Yamada, S. Ueno, and S. Naito, "An attention-based activity recognition for egocentric video," in *IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, June 2014, pp. 565–570.

[18] Yunyoung Nam, Seungmin Rho, and Chulung Lee, "Physical activity recognition using multiple sensors embedded in a wearable device," *ACM Trans. on Embedded Computing Systems*, vol. 12, no. 2, pp. 1–14, 2013.

[19] S. Bambach, "A Survey on Recent Advances of Computer Vision Algorithms for Egocentric Video," *ArXiv e-prints*, Jan. 2015.

[20] Shumei Zhang, P. McCullagh, C. Nugent, and Huiru Zheng, "Activity monitoring using a smart phone's accelerometer with hierarchical classification," in *Intelligent Environments (IE), 2010 Sixth International Conference on*, July 2010, pp. 158–163.

[21] K.T. Song and W.J. Chen, "Human activity recognition using a mobile camera," *Ubiquitous Robots and Ambient Intel., 8th Int'l Conf. on*, pp. 3–8, 2011.

[22] Lu Li, Hong Zhang, Wenyan Jia, Zhi Hong Mao, Yuhu You, and Mingui Sun, "Indirect activity recognition using a target-mounted camera," in *Image and Signal Proc., 4th Int'l Congress on*, Oct 2011, vol. 1, pp. 487–491.

[23] Ming Li, Viktor Rozgić, Gautam Thatte, Sangwon Lee, Adar Emken, Murali Annavaram, Urbashi Mitra, Donna Spruijt-Metz, and Shrikanth Narayanan, "Multimodal physical activity recognition by fusing temporal and cepstral information,"

IEEE Trans. on Neural Systems and Rehabilitation Eng., vol. 18, no. 4, pp. 369–380, 2010.

[24] Y. Watanabe, T. Hatanaka, T. Komuro, and M. Ishikawa, "Human gait estimation using a wearable camera," in *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, Jan 2011, pp. 276–281.

[25] Bo Yin, Wenjuan Qi, Zhiqiang Wei, and Jie Nie, "Indirect human activity recognition based on optical flow method," *Int'l Congr. on Image and Signal Processing*, , no. Cisp, pp. 99–103, 2012.

[26] Kris M. Kitani, Takahiro Okabe, Yoichi Sato, and Akihiro Sugimoto, "Fast unsupervised ego-action learning for first-person sports videos," *Proc. of the IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, pp. 3241–3248, 2011.

[27] K. Ozcan, A.K. Mahabalagiri, M. Casares, and S. Velipasalar, "Automatic fall detection and activity classification by a wearable embedded smart camera," *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, vol. 3, no. 2, pp. 125–136, 2013.

[28] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, pp. 2432–2439.

[29] Tomas McCandless and Kristen Grauman, "Object-Centric Spatio-Temporal Pyramids for Egocentric Activity Recognition," *Proc. of the British Machine Vision Conf.*, pp. 30.1–30.11, 2013.

[30] H. Pirsiavash and D. Ramanan, "Detecting activities of daily living in first-person camera views," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conf. on*, June 2012, pp. 2847–2854.

[31] Yan Yan, E. Ricci, Gaowen Liu, and N. Sebe, "Egocentric daily activity recognition via multitask clustering," *Image Processing, IEEE Trans. on*, vol. 24, no. 10, pp. 2984–2995, Oct 2015.

[32] Alireza Fathi, Xiaofeng Ren, and James M. Rehg, "Learning to recognize objects in egocentric activities," *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 3281–3288, 2011.

[33] S. Sundaram and W.W.M. Cuevas, "High level activity recognition using low resolution wearable vision," *IEEE Conf.e on Computer Vision and Pattern Recognition Workshops*, pp. 25–32, 2009.

[34] Yin Li, Alireza Fathi, and James M. Rehg, "Learning to predict gaze in egocentric video," *Proc. of the IEEE ICCV*, , no. 1, pp. 3216–3223, 2013.

[35] W. W. Mayol and D. W. Murray, "Wearable hand activity recognition for event summarization," *Proc. Int'l Symp. on Wearable Computers, ISWC*, vol. 2005, pp. 122–129, 2005.

[36] Alircza Fathi, Jessica K. Hodgins, and James M. Rehg, "Social interactions: A first-person perspective," *Proc. of the IEEE Comput. Soc. Conf. on Computer Vision and Pattern Recognition*, pp. 1226–1233, 2012.

[37] T. Starner, J. Weaver, and a. Pentland, "Real-time American sign language recognition using desk and wearable computer based video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 12, pp. 1371–1375, 1998.

[38] Omid Aghazadeh, Josephine Sullivan, and Stefan Carlsson, "Novelty detection from an ego-centric perspective," *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 3297–3304, 2011.

[39] Lu Li, Hong Zhang, Wenyan Jia, Jie Nie, Weidong Zhang, and Mingui Sun, "Automatic video analysis and motion estimation for physical activity classification," *Proc. IEEE 36th Northeast Bioeng. Conf.*, , no. 4, pp. 1–2, 2010.

[40] Hong Zhang, Lu Li, Wenyan Jia, John D. Fernstrom, Robert J. Sclabassi, and Mingui Sun, "Recognizing physical activity from Ego-motion of a camera," *Annu. Int'l Conf. of the IEEE Eng. in Medicine and Biology Soc., EMBS*, pp. 5569–5572, 2010.

[41] Hong Zhang, Lu Li, Wenyan Jia, John D. Fernstrom, Robert J. Sclabassi, Zhi Hong Mao, and Mingui Sun, "Physical activity recognition based on motion in images acquired by a wearable camera," *Neurocomputing*, vol. 74, no. 12-13, pp. 2184–2192, 2011.

[42] Xiaofeng Ren and Chunhui Gu, "Figure-ground segmentation improves handled object recognition in egocentric video," *Proc. of the IEEE Comput. Soc. Conf. on Computer Vision and Pattern Recognition*, , no. 1, pp. 3137–3144, 2010.

[43] Kazuhiko Sumi, Masato Toda, Akihiro Sugimoto, Takashi Matsuyama, and Sotaro Tsukizawa, "Active wearable vision sensor: Recognition of human activities and environments," *Proc. Int'l Conf. on Informatics Research for Development of Knowledge Society Infrastructure*, pp. 15–22, 2004.

[44] R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray, "Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras," *Proc. IEEE Int'l Conf. on Robotics and Automation*, pp. 4102–4107, 2007.

[45] Bo Xiong and Kristen Grauman, "Detecting snap points in egocentric video with a web photo prior," in *Computer Vision, ECCV*, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, Eds., vol. 8693 of *Lecture Notes in Computer Science*, pp. 282–298. Springer Int'l Publishing, 2014.

[46] YongJae Lee and Kristen Grauman, "Predicting important objects for egocentric video summarization," *Int'l J. of Comput. Vision*, vol. 114, no. 1, pp. 38–55, 2015.

[47] J. Ghosh and K. Grauman, "Discovering important people and objects for egocentric video summarization," in *2012 IEEE Conf. on Computer Vision and Pattern Recognition*. June 2012, pp. 1346–1353, IEEE.

[48] Zheng Lu and Kristen Grauman, "Story-Driven Summarization for Egocentric Video," in *IEEE Conf. on Computer Vision and Pattern Recognition*. June 2013, pp. 2714–2721, IEEE.

[49] Yong Jae Lee and Kristen Grauman, "Predicting Important Objects for Egocentric Video Summarization," *Int'l J. of Computer Vision*, vol. 114, no. 1, pp. 38–55, Jan. 2015.

[50] Robert E. Schapire, *Explaining AdaBoost*, pp. 37–52, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[51] Kentaro Yamada, Yusuke Sugano, Takahiro Okabe, Yoichi Sato, Akihiro Sugimoto, and Kazuo Hiraki, "Attention prediction in egocentric video using motion and visual saliency," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intel. and Lecture Notes in Bioinformatics)*, vol. 7087 LNCS, no. PART1, pp. 277–288, 2011.

[52] Keisuke Ogaki, Kris M. Kitani, Yusuke Sugano, and Yoichi Sato, "Coupling eye-motion and ego-motion features for first-person activity recognition," *IEEE Comput. Soc. Conf. on Computer Vision and Pattern Recognition Workshops*, pp. 1–7, 2012.

[53] H. H. Wu, E. D. Lemaire, and N. Baddour, "Change-of-state determination to recognize mobility activities using a blackberry smartphone," in *Ann. Int'l Conf. of the IEEE Eng. in Medicine and Biology Soc.,EMBS*, Aug 2011, pp. 5252–5255.

[54] Yu-Jin Hong, Ig-Jae Kim, Sang Chul Ahn, and Hyoung-Gon Kim, “Mobile health monitoring system based on activity recognition using accelerometer,” *Sim. Modelling Practice and Theory*, vol. 18, no. 4, pp. 446–455, 2010.

[55] Yu-Jin Hong Yu-Jin Hong, Ig-Jae Kim Ig-Jae Kim, Sang Chul Ahn Sang Chul Ahn, and Hyoung-Gon Kim Hyoung-Gon Kim, “Activity Recognition Using Wearable Sensors for Elder Care,” *2nd Int’l Conf. on Future Generation Communication and Networking*, vol. 2, pp. 1–4, 2008.

[56] Roberto L. Shinmoto Torres, Damith C. Ranasinghe, Qinfeng Shi, and Alanson P. Sample, “Sensor enabled wearable RFID technology for mitigating the risk of falls near beds,” *IEEE Intl. Conf. on RFID*, pp. 191–198, 2013.

[57] E. M. Tapia, S. S. Intille, W. Haskell, J. Larson, K. Wright, a. King, and R. Friedman, “Real-Time Recognition of Physical Activities and their Intensitiies Using Wireless Accelerometers and a Heart Monitor,” *Int’l Symp. on Wearable Computers*, pp. 37–40, 2007.

[58] Che Wei Lin, Ya Ting C Yang, Jeen Shing Wang, and Yi Ching Yang, “A wearable sensor module with a neural-network-based activity classification algorithm for daily energy expenditure estimation,” *IEEE Trans. on Information Techn. in Biomedicine*, vol. 16, no. 5, pp. 991–998, 2012.

[59] Debasmit Banerjee, Subir Biswas, Courtney Daigle, and Janice M. Siegford, “Remote activity classification of hens using wireless body mounted sensors,” *Proc. of Int’l Workshop on Wearable and Implantable Body Sensor Networks*, pp. 107–112, 2012.

[60] J. Surana, C. S. Hemalatha, V. Vaidehi, S. A. Palavesam, and M. J. A. Khan, “Adaptive learning based human activity and fall detection using fuzzy frequent pattern

mining," in *Recent Trends in Information Technology (ICRTIT),Int'l Conf. on*, July 2013, pp. 744–749.

[61] Ruiting Jia and Bin Liu, "Human daily activity recognition by fusing accelerometer and multi-lead ECG data," *IEEE Int'l Conf. on Signal Proc., Comm. and Computing, ICSPCC*, 2013.

[62] C M El Achkar, F Masse, a Arami, and K Aminian, "Physical activity recognition via minimal in-shoes force sensor configuration," *7th Int'l Conf. on Pervasive Computing Technologies for Healthcare and Workshops, PervasiveHealth*, pp. 256–259, 2013.

[63] Debraj De, Pratool Bharti, Sajal K. Das, and Sriram Chellappan, "Multimodal wearable sensing for fine-grained activity recognition in healthcare," *Internet Computing, IEEE*, vol. 19, no. 5, pp. 26–35, Sept 2015.

[64] Wenlong Tang and Edward S. Sazonov, "Highly accurate recognition of human postures and activities through classification with rejection," *IEEE Journ. Biomed. and Health Inform.*, vol. 18, no. 1, pp. 309–315, 2014.

[65] Edward Sazonov, Nagaraj Hegde, Raymond Browning, Edward Melanson, and Nadezhda Sazonova, "Posture and Activity Recognition and Energy Expenditure Prediction in a Wearable Platform," *IEEE Journal of Biomedical and Health Informatics*, vol. 2194, no. c, pp. 1–1, 2015.

[66] J. Hernandez, Y. Li, J. M. Rehg, and R. W. Picard, "Bioglass: Physiological parameter estimation using a head-mounted wearable device," in *Wireless Mobile Communication and Healthcare (Mobihealth), EAI 4th Int'l Conf. on*, Nov 2014, pp. 55–58.

[67] K. Zhan, S. Faux, and F. Ramos, “Multi-scale conditional random fields for first-person activity recognition,” in *Pervasive Computing and Communications (PerCom), IEEE Int’l Conf. on*, 2014, pp. 51–59.

[68] Juha Pärkkä, Miikka Ermes, Panu Korpipää, Jani Mäntyjärvi, Johannes Peltola, and Ilkka Korhonen, “Activity classification using realistic data from wearable sensors.,” *IEEE Trans. on Inf. Technol. in Biomedicine*, vol. 10, no. 1, pp. 119–128, 2006.

[69] M. Moghimi, P. Azagra, L. Montesano, A. C. Murillo, and S. Belongie, “Experiments on an rgb-d wearable vision system for egocentric activity recognition,” in *IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, June 2014, pp. 611–617.

[70] Edward S. Sazonov, George Fulk, James Hill, Yves Schutz, and Raymond Browning, “Monitoring of posture allocations and activities by a shoe-based wearable sensor,” *IEEE Trans. Biomed. Eng.*, vol. 58, no. 4, pp. 983–990, 2011.

[71] D. Riboni and C. Bettini, “COSAR: Hybrid reasoning for context-Aware activity recognition,” *Personal and Ubiquitous Computing*, vol. 15, no. 3, pp. 271–289, 2011.

[72] Aiden R. Doherty, Steve E. Hodges, Abby C. King, Alan F. Smeaton, Emma Berry, Chris J.A. Moulin, SiÃćn Lindley, Paul Kelly, and Charlie Foster, “Wearable cameras in health: The state of the art and future possibilities,” *American Journal of Preventive Medicine*, vol. 44, no. 3, pp. 320 – 323, 2013.

[73] T. Fujimoto, H. Nakajima, N. Tsuchiya, H. Marukawa, K. Kuramoto, S. Kobashi, and Y. Hata, “Wearable human activity recognition by electrocardiograph and accelerometer,” in *Multiple-Valued Logic (ISMVL), IEEE 43rd Int’l Symp. on*, May 2013, pp. 12–17.

[74] Kazushige Ouchi, "Smartphone-based Monitoring System for Activities of Daily Living for Elderly People and Their Relatives Etc .," *UbiComp'13*, pp. 103–106, 2013.

[75] Adil Mehmood Khan, Ali Tufail, Asad Masood Khattak, and Teemu H. Laine, "Activity recognition on smartphones via sensor-fusion and KDA-based SVMs," *Int'l J. of Distributed Sensor Networks*, 2014.

[76] Waythit Puangpakisiri, Toshihiko Yamasaki, and Kiyoharu Aizawa, "High level activity annotation of daily experiences by a combination of a wearable device and Wi-Fi based positioning system," *IEEE Intl. Conf. on Multimedia and Expo, ICME Proc.*, vol. 9, pp. 1421–1424, 2008.

[77] Gernot Bahle, Paul Lukowicz, Kai Kunze, and Koichi Kise, "I see you: How to improve wearable activity recognition by leveraging information from environmental cameras," *IEEE Int'l Conf. on Pervasive Computing and Commun. Workshops*, , no. March, pp. 409–412, 2013.

[78] David Tacconi, Oscar Mayora, Paul Lukowicz, Bert Arnrich, Cornelia Setz, Gerhard Troster, and Christian Haring, "Activity and emotion recognition to support early diagnosis of psychiatric diseases," *2nd Int'l Conf. on Pervasive Computing Technologies for Healthcare*, pp. 100–102, 2008.

[79] S. Ryan Edgar, George D. Fulk, and Edward S. Sazonov, "Recognition of household and athletic activities using smartshoe," *Proc. of the Annu. Int'l Conf. of the IEEE Eng. in Medicine and Biology Soc.*, pp. 6382–6385, 2012.

[80] Anthony Fleury, Michel Vacher, and Norbert Noury, "SVM-based multimodal classification of activities of daily living in health smart homes: Sensors, algorithms, and first experimental results," *IEEE Trans. on Information Technology in Biomedicine*, vol. 14, no. 2, pp. 274–283, 2010.

[81] K. Ozcan and S. Velipasalar, "Wearable camera- and accelerometer-based fall detection on portable devices," *IEEE Embedded Systems Letters*, vol. 8, no. 1, pp. 6–9, March 2016.

[82] Eri Shimokawara, Tetsuya Kaneko, Toru Yamaguchi, Makoto Mizukawa, and Nobuto Matsuhira, "Estimation of Basic Activities of Daily Living Using ZigBee 3D Accelerometer Sensor Network," *Int'l Conf. on Biometrics and Kansei Eng.*, pp. 251–256, 2013.

[83] Abhishek Basak, Seetharam Narasimhan, and Swarup Bhunia, "KiMS: Kids' health monitoring system at day-care centers using wearable sensors and vocabulary-based acoustic signal processing," *IEEE 13th Int'l Conf. on e-Health Networking, Applications and Services, HEALTHCOM*, pp. 1–8, 2011.

[84] T Tagniguchi, Keita Hamahata, and N Iwahashi, "Unsupervised Segmentation of Human Motion Data Using Sticky HDP-HMM and MDL-based Chunking Method for Imitation Learning," *Adv. Robotics*, 2011.

[85] Ekaterina H. Spriggs, Fernando De La Torre, and Martial Hebert, "Temporal segmentation and activity classification from first-person sensing," *2009 IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 17–24, 2009.

[86] Zhen Li, Zhiqiang Wei, Wenyan Jia, and Mingui Sun, "Daily life event segmentation for lifestyle evaluation based on multi-sensor data recorded by a wearable device," *Proc. of the Int'l Conf. of the IEEE Engineering in Medicine and Biology Society*, pp. 2858–2861, 2013.

[87] S Ishimaru, K Kunze, and K Kise, "In the Blink of an Eye âĂŞ Combining Head Motion and Eye Blink Frequency for Activity Recognition with Google Glass," *Proc.,5th Augmented Human Intl. Conf.*, 2014.

[88] Andrea Mannini and Angelo Maria Sabatini, "On-line classification of human activity and estimation of walk-run speed from acceleration data using support vector machines," *Proc. of the Int'l Conf. of the IEEE Engineering in Medicine and Biology Society*, pp. 3302–3305, 2011.

[89] Richard Ribon Fletcher, Sharon Tam, Olufemi Omojola, Richard Redemske, and Joyce Kwan, "Wearable sensor platform and mobile application for use in cognitive behavioral therapy for drug addiction and PTSD," *Annual Int'l Conf. of the IEEE Eng. in Medicine and Biology Soc.*, pp. 1802–1805, 2011.

[90] S. Keskar, R. Banerjee, and R. Reddy, "A dual-psoc based reconfigurable wearable computing framework for ecg monitoring," in *Computing in Cardiology*, Sept 2012, pp. 85–88.

[91] Michael B Del Rosario, Kejia Wang, Jingjing Wang, Ying Liu, Matthew Brodie, Kim Delbaere, Nigel H Lovell, Stephen R Lord, and Stephen J Redmond, "A comparison of activity classification in younger and older cohorts using a smartphone," *Physiological Measurement*, vol. 35, no. 11, pp. 2269, 2014.

[92] Bo Xiong and Kristen Grauman, "Detecting Snap Points in Egocentric Video with a Web Photo Prior," *Computer Vision, ECCV*, 2014.

[93] Davide Curone, Gian Mario Bertolotti, Andrea Cristiani, Emanuele Lindo Secco, and Giovanni Magenes, "A real-time and self-calibrating algorithm based on triaxial accelerometer signals for the detection of human posture and activity," *IEEE Trans. on Information Technology in Biomedicine*, vol. 14, no. 4, pp. 1098–1105, 2010.

[94] David Naranjo-Hernández, Laura M. Roa, Javier Reina-Tosina, and Miguel Ángel Estudillo-Valderrama, "SoM: A smart sensor for human activity monitoring and assisted healthy ageing," *IEEE Trans. on Biomed. Eng.*, vol. 59, no. 12 PART2, pp. 3177–3184, 2012.

[95] Grayson K. Vincent and Victoria A. Velkoff, “The next four decades, the older population in the united states: 2010 to 2050,” 2010.

[96] Dong Won Kang, Jin Seung Choi, Jeong Whan Lee, Soon Cheol Chung, Soo Jun Park, and Gye Rae Tack, “Real-time elderly activity monitoring system based on a tri-axial accelerometer,” 2010, pp. 247–253, PMID: 20302417.

[97] Melonie Heron, “Deaths: Leading causes 2007,” August 2011, number 8, pp. 17, 21–22.

[98] Janet Shelfer, David Zapala, and Larry Lundy, “Fall risk, vestibular schwannoma, and anticoagulation therapy,” United States, McLean, 2008, pp. 237–45, American Academy of Audiology.

[99] R. C. O. Voshaar, S. Banerjee, M. Horan, R. Baldwin, N. Pendleton, R. Proctor, N. Tarrier, Y. Woodward, and A. Burns, “Predictors of incident depression after hip fracture surgery,” *The American Journal of Geriatric Psychiatry*, vol. 15, no. 9, pp. 807 – 814, 2007.

[100] LD Gillespie, WJ Gillespie, MC Robertson, SE Lamb, RG Cumming, and BH Rowe, “Interventions for preventing falls in elderly people,” 2003, pp. 692 – 693.

[101] S. Cagnoni, G. Matrella, M. Mordonini, F. Sassi, and L. Ascari, “Sensor fusion-oriented fall detection for assistive technologies applications,” in *Intelligent Systems Design and Applications, 2009. ISDA ’09. Ninth International Conference on*, Nov 2009, pp. 673–678.

[102] Miao Yu, A. Rhuma, S.M. Naqvi, Liang Wang, and J. Chambers, “A posture recognition-based fall detection system for monitoring an elderly person in a smart home environment,” *Information Technology in Biomedicine, IEEE Transactions on*, vol. 16, no. 6, pp. 1274–1286, Nov 2012.

[103] G.A. Koshmak, M. Linden, and A. Loutfi, “Evaluation of the android-based fall detection system with physiological data monitoring,” in *Engineering in Medicine and Biology Society, 35th Annual Int’l Conf. of the IEEE*, July 2013, pp. 1164–1168.

[104] Yabo Cao, Yujiu Yang, and Wenhuang Liu, “E-falld: A fall detection system using android-based smartphone,” in *Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th Int’l Conf. on*, May 2012, pp. 1509–1513.

[105] Ken Taylor, Umran A. Abdulla, Richard J.N. Helmer, Jungoo Lee, and Ian Blanchonette, “Activity classification with smart phones for sports activities,” *Procedia Engineering*, vol. 13, pp. 428 – 433, 2011.

[106] Wanmin Wu, Sanjoy Dasgupta, E. Ernesto Ramirez, Carlyn Peterson, and J. Gregory Norman, “Classification accuracies of physical activities using smartphone motion sensors,” *J Med Internet Res*, vol. 14, no. 5, pp. e130, Oct 2012.

[107] N. Noury, T. Herve, V. Rialle, G. Virone, E. Mercier, G. Morey, A. Moro, and T. Porcheron, “Monitoring behavior in home using a smart fall sensor and position sensors,” in *Microtechnologies in Medicine and Biology, 1st Annual International, Conference On. 2000*, 2000, pp. 607 –610.

[108] Muhammad Mubashir, Ling Shao, and Luke Seed, “A survey on fall detection: Principles and approaches,” 2013, pp. 144 – 152, <ce:title>Special issue: Behaviours in video</ce:title>.

[109] R. Igual, C. Medrano, and I. Plaza, “Challenges, issues and trends in fall detection systems,” *BioMedical Engineering OnLine*, vol. 12, no. 1, pp. 1–24, 2013.

[110] D.M. Karantonis, M.R. Narayanan, M. Mathie, N.H. Lovell, and B.G. Celler, “Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring,” jan. 2006, pp. 156 –167.

[111] Roberto Hoyle, Robert Templeman, Steven Armes, Denise Anthony, David Crandall, and Apu Kapadia, "Privacy behaviors of lifeloggers using wearable cameras," in *Proc. of the 2014 ACM Int'l Joint Conf. on Pervasive and Ubiquitous Computing*, 2014, pp. 571–582.

[112] N. Noury, A Fleury, P. Rumeau, AK. Bourke, G.O. Laighin, V. Rialle, and J. E. Lundy, "Fall detection - principles and methods," in *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, Aug 2007, pp. 1663–1666.

[113] Muhammad Mubashir, Ling Shao, and Luke Seed, "A survey on fall detection: Principles and approaches," *Neurocomputing*, vol. 100, no. 0, pp. 144 – 152, 2013, Special issue: Behaviours in video.

[114] Shih-Hau Fang, Yi-Chung Liang, and Kuan-Ming Chiu, "Developing a mobile phone-based fall detection system on android platform," in *Computing, Communications and Applications Conf. (ComComAp)*, Jan 2012, pp. 143–146.

[115] Jiangpeng Dai, Xiaole Bai, Zhimin Yang, Zhaohui Shen, and Dong Xuan, "Perfalld: A pervasive fall detection system using mobile phones," in *Pervasive Computing and Communications Workshops, IEEE Int'l Conf. on*, March 2010, pp. 292–297.

[116] Wanmin Wu, Sanjoy Dasgupta, E. Ernesto Ramirez, Carlyn Peterson, and J. Gregory Norman, "Classification accuracies of physical activities using smartphone motion sensors," *J Med Internet Res*, vol. 14, no. 5, pp. e130, Oct 2012.

[117] F. Hijaz, N. Afzal, T. Ahmad, and O. Hasan, "Survey of fall detection and daily activity monitoring techniques," in *Information and Emerging Technologies (ICIET), 2010 International Conference on*, june 2010, pp. 1 –6.

[118] T. Tamura, "Wearable accelerometer in clinical use," in *Proc. 27th Annual Int. Conf. of the Engineering in Medicine and Biology Society IEEE-EMBS 2005*, 2005, pp. 7165–7166.

[119] C.J. Lord and D.P. Colvin, "Falls in the elderly: Detection and assessment," in *Engineering in Medicine and Biology Society, 1991. Vol.13: 1991., Proceedings of the Annual International Conference of the IEEE*, oct-3 nov 1991, pp. 1938 –1939.

[120] G. Williams, K. Doughty, K. Cameron, and D.A. Bradley, "A smart fall and activity monitor for telecare applications," in *Engineering in Medicine and Biology Society, 1998. Proceedings of the 20th Annual International Conference of the IEEE*, oct-1 nov 1998, pp. 1151 –1154 vol.3.

[121] J. Chen, Karric Kwong, D. Chang, J. Luk, and R. Bajcsy, "Wearable sensors for reliable fall detection," in *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, jan. 2005, pp. 3551 –3554.

[122] Chia-Chi Wang, Chin-Yen Chiang, Po-Yen Lin, Yi-Chieh Chou, I-Ting Kuo, Chih-Ning Huang, and Chia-Tai Chan, "Development of a fall detecting system for the elderly residents," in *Bioinformatics and Biomedical Engineering, 2008. ICBBE 2008. The 2nd International Conference on*, may 2008, pp. 1359 –1362.

[123] M.R. Narayanan, S.R. Lord, M.M. Budge, B.G. Celler, and N.H. Lovell, "Falls management: Detection and prevention, using a waist-mounted triaxial accelerometer," in *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, aug. 2007, pp. 4037 –4040.

[124] G. Wu, "Distinguishing fall activities from normal activities by velocity characteristics," 2000, pp. 1497 – 1500.

[125] Miao Yu, A. Rhuma, S.M. Naqvi, Liang Wang, and J. Chambers, "A posture recognition-based fall detection system for monitoring an elderly person in a smart home environment," nov. 2012, pp. 1274 –1286.

[126] A.K. Bourke, J.V. O Brien, and G.M . Lyons, "Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm," 2007, pp. 194 – 199.

[127] Jiewen Zheng, Guang Zhang, and Taihu Wu, "Design of automatic fall detector for elderly based on triaxial accelerometer," in *Proc. 3rd Int. Conf. Bioinformatics and Biomedical Engineering ICBBE 2009*, 2009, pp. 1–4.

[128] T. Degen, H. Jaeckel, M. Rufer, and S. Wyss, "SPEEDY: a fall detector in a wrist watch," in *Proc. Seventh IEEE Int Wearable Computers Symp*, 2003, pp. 184–187.

[129] Xiuxin Yang, Anh Dinh, and Li Chen, "A wearable real-time fall detector based on naive bayes classifier," in *Proc. 23rd Canadian Conf. Electrical and Computer Engineering (CCECE)*, 2010, pp. 1–4.

[130] M. Alwan, P.J. Rajendran, S. Kell, D. Mack, S. Dalal, M. Wolfe, and R. Felder, "A smart and passive floor-vibration based fall detector for elderly," in *Information and Communication Technologies, 2006. ICTTA '06. 2nd*, 0-0 2006, pp. 1003 –1007.

[131] Xiaodan Zhuang, Jing Huang, G. Potamianos, and M. Hasegawa-Johnson, "Acoustic fall detection using gaussian mixture models and gmm supervectors," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, april 2009, pp. 69 –72.

[132] M. Yu, A. Rhuma, S. M. Naqvi, L. Wang, and J. Chambers, "A posture recognition-based fall detection system for monitoring an elderly person in a smart home environment," nov. 2012, pp. 1274 –1286.

[133] N. Noury, A. Galay, J. Pasquier, and M. Ballussaud, "Preliminary investigation into the use of autonomous fall detectors," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, aug. 2008, pp. 2828 –2831.

[134] Zhengming Fu, T. Delbruck, P. Lichtsteiner, and E. Culurciello, "An address-event fall detector for assisted living applications," 2008, pp. 88–96.

[135] A. Sixsmith and N. Johnson, "A smart sensor to detect the falls of the elderly," 2004, pp. 42–47.

[136] S. Zambanini, J. Machajdik, and M. Kampel, "Detecting falls at homes using a network of low-resolution cameras," in *Proc. 10th IEEE Int Information Technology and Applications in Biomedicine (ITAB) Conf*, 2010, pp. 1–4.

[137] P. Siciliano, A. Leone, G. Diraco, C. Distante, M. Malfatti, L. Gonzo, M. Grassi, A. Lombardi, G. Rescio, and P. Malcovati, "A networked multisensor system for ambient assisted living application," in *Proc. 3rd Int. Workshop Advances in sensors and Interfaces IWASI 2009*, 2009, pp. 139–143.

[138] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Fall detection from human shape and motion history using video surveillance," in *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, may 2007, pp. 875 –880.

[139] A.N. Belbachir, S. Schraml, and A. Nowakowska, "Event-driven stereo vision for fall detection," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, june 2011, pp. 78 –83.

[140] A.N. Belbachir, A. Nowakowska, S. Schraml, G. Wiesmann, and R. Sablatnig, "Event-driven feature analysis in a 4d spatiotemporal representation for ambient

assisted living," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, nov. 2011, pp. 1570 –1577.

[141] H. Foroughi, A. Naseri, A. Saberi, and H.S. Yazdi, "An eigenspace-based approach for human fall detection using integrated time motion image and neural network," in *Signal Processing, 2008. ICSP 2008. 9th International Conference on*, oct. 2008, pp. 1499 –1503.

[142] Shaou-Gang Miaou, Pei-Hsu Sung, and Chia-Yuan Huang, "A customized human fall detection system using omni-camera images and personal information," in *Distributed Diagnosis and Home Healthcare, 2006. D2H2. 1st Transdisciplinary Conference on*, april 2006, pp. 39 –42.

[143] Bart Jansen and Rudi Deklerck, "Context aware inactivity recognition for visual fall detection," in *Pervasive Health Conference and Workshops, 2006*, 29 2006-dec. 1 2006, pp. 1 –4.

[144] A.N. Belbachir, M. Litzenberger, S. Schraml, M. Hofstatter, D. Bauer, P. Schon, M. Humenberger, C. Sulzbachner, T. Lunden, and M. Merne, "Care: A dynamic stereo vision sensor system for fall detection," in *Circuits and Systems (ISCAS), 2012 IEEE International Symposium on*, may 2012, pp. 731 –734.

[145] S. Fleck, R. Loy, C. Vollrath, F. Walter, and W. Strasser, "Smartclassysurv - a smart camera network for distributed tracking and activity recognition and its application to assisted living," in *Distributed Smart Cameras, 2007. ICDSC '07. First ACM/IEEE International Conference on*, sept. 2007, pp. 211 –218.

[146] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, june 2005, pp. 886 –893 vol. 1.

[147] Mauricio Casares, Koray Ozcan, Akhan Almagambetov, and Senem Velipasalar, "Automatic fall detection by a wearable embedded smart camera," in *Distributed Smart Cameras (ICDSC), 2012 Sixth International Conference on*, 2012, pp. 1–6.

[148] Berthold K.P. Horn and Brian G. Schunck, "Determining optical flow," 1981, pp. 185 – 203.

[149] AA Shafie, F. Hafiz, and MH Ali, "Motion detection techniques using optical flow," 2009, pp. 559–561, Citeseer.

[150] Deqing Sun, S. Roth, and M.J. Black, "Secrets of optical flow estimation and their principles," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, june 2010, pp. 2432 –2439.

[151] N. Jiang, J. Xu, W. Yu, and S. Goto, "Gradient local binary patterns for human detection," in *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*, 2013, pp. 978–981.

[152] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 971–987, 2002.

[153] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas, "On combining classifiers," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 3, pp. 226–239, Mar 1998.

[154] Simon Brownsell and Mark S. Hawley, "Automatic fall detectors and the fear of falling," *Journal of Telemedicine and Telecare*, vol. 10, no. 5, pp. 262–266, 2004.

[155] U. Lindemann, A. Hock, M. Stuber, W. Keck, and C. Becker, "Evaluation of a fall detector based on accelerometers: A pilot study," *Medical and Biological Engineering and Computing*, vol. 43, no. 5, pp. 548–551, 2005.

[156] AK. Bourke, P. van de Ven, M. Gamble, R. O'Connor, K. Murphy, E. Bogan, E. McQuade, P. Finucane, G. OLaighin, and J. Nelson, "Assessment of waist-worn tri-axial accelerometer based fall-detection algorithms using continuous unsupervised activities," in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, Aug 2010, pp. 2782–2785.

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

[158] Maarit Kangas, Irene Vikman, Jimmie Wiklander, Per Lindgren, Lars Nyberg, and Timo JÃď'msÃď', "Sensitivity and specificity of fall detection in people aged 40 years and over," *Gait & Posture*, vol. 29, no. 4, pp. 571 – 574, 2009.

[159] Fabio Bagala, Clemens Becker, Angelo Cappello, Lorenzo Chiari, Kamiar Aminian, Jeffrey M. Hausdorff, Wiebren Zijlstra, and Jochen Klenk, "Evaluation of accelerometer-based fall detection algorithms on real-world falls," *PLoS ONE*, vol. 7, no. 5, pp. e37062, 05 2012.

[160] Maarit Kangas, Antti Konttila, Per Lindgren, Ilkka Winblad, and Timo JÃď'msÃď', "Comparison of low-complexity fall detection algorithms for body attached accelerometers," *Gait and Posture*, vol. 28, no. 2, pp. 285 – 291, 2008.

[161] Liane C. Ramac and Pramod K. Varshney, "Image thresholding based on ali-silvey distance measures," *Pattern Recognition*, vol. 30, no. 7, pp. 1161 – 1174, 1997.

[162] Chein-I Chang, Kebo Chen, Jianwei Wang, and Mark L.G. Althouse, "A relative entropy-based approach to image thresholding," *Pattern Recognition*, vol. 27, no. 9, pp. 1275 – 1289, 1994.

[163] Case MA, Burwick HA, Volpp KG, and Patel MS, "Accuracy of smartphone applications and wearable devices for tracking physical activity data," *JAMA*, vol. 313, no. 6, pp. 625–626, 2015.

[164] Kwanghyo Park, Hyojeong Shin, and Hojung Cha, "Smartphone-based pedestrian tracking in indoor corridor environments," *Personal and Ubiquitous Computing*, vol. 17, no. 2, pp. 359–370, 2013.

[165] Meng-Shiuan Pan and Hsueh-Wei Lin, "A step counting algorithm for smartphone users: Design and implementation," *Sensors Journal, IEEE*, vol. 15, no. 4, pp. 2296–2305, April 2015.

[166] N.A. Capela, E.D. Lemaire, and N.C. Baddour, "A smartphone approach for the 2 and 6-minute walk test," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*, Aug 2014, pp. 958–961.

[167] Agata Brajdic and Robert Harle, "Walk detection and step counting on unconstrained smartphones," in *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, New York, NY, USA, 2013, UbiComp '13, pp. 225–234, ACM.

[168] F. Aubeck, C. Isert, and D. Gusenbauer, "Camera based step detection on mobile phones," in *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, Sept 2011, pp. 1–7.

[169] John Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-8, no. 6, pp. 679–698, Nov 1986.

[170] J. Matas, C. Galambos, and J. Kittler, "Robust detection of lines using the progressive probabilistic hough transform," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 119 – 137, 2000.

[171] Edward Rosten and Tom Drummond, "Machine learning for high-speed corner detection," in *Computer Vision — ECCV 2006*, Ales Leonardis, Horst Bischof, and Axel Pinz, Eds., vol. 3951 of *Lecture Notes in Computer Science*, pp. 430–443. Springer Berlin Heidelberg, 2006.

[172] David Arthur and Sergei Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, Philadelphia, PA, USA, 2007, SODA '07, pp. 1027–1035, Society for Industrial and Applied Mathematics.

[173] A. Savitzky and M. J. E. Golay, "Soothing and differentiation of data by simplified least squares procedures," in *Anal. Chem.*, 1964, vol. 36, pp. 1627–1639.

[174] N. Yoder, "Peakfinder: Quickly finds local maxima (peaks) or minima (valleys) in a noisy signal.," http://www.mathworks.com/matlabcentral/fileexchange/25500-peakfinder, 2014.

[175] E. Cuevas, D. Zaldivar, and R. Rojas, "Kalman filter for vision tracking," *Technical Report B 05-12*, 2005.

[176] T. Kailath, "The divergence and bhattacharyya distance measures in signal selection," *Communication Technology, IEEE Transactions on*, vol. 15, no. 1, pp. 52–60, February 1967.

[177] Koray Ozcan and Senem Velipasalar, "Robust and reliable step counting by mobile phone cameras," in *Proceedings of the 9th International Conference on Distributed Smart Cameras*, New York, NY, USA, 2015, ICDSC '15, pp. 164–169, ACM.

[178] Simon Zingg, Davide Scaramuzza, Stephan Weiss, and Roland Siegwart, "Mav navigation through indoor corridors using optical flow," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 3361–3368.

[179] Cooper Bills, Joyce Chen, and Ashutosh Saxena, “Autonomous mav flight in indoor environments using single image perspective cues,” in *Robotics and automation (ICRA), 2011 IEEE international conference on*. IEEE, 2011, pp. 5776–5783.

[180] Matthew Derry and Brenna Argall, “Automated doorway detection for assistive shared-control wheelchairs,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1254–1259.

[181] Zhichao Chen and Stanley T Birchfield, “Visual detection of lintel-occluded doors from a single image,” in *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW’08. IEEE Computer Society Conference on*. IEEE, 2008, pp. 1–8.

[182] Kenneth L Campbell, “The shrp 2 naturalistic driving study: Addressing driver performance and behavior in traffic safety,” *TR News*, , no. 282, 2012.

[183] Mayeul Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool, “Traffic sign recognition- how far are we from the solution?,” in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–8.

[184] Radu Timofte, Karel Zimmermann, and Luc Van Gool, “Multi-view traffic sign detection, recognition, and 3d localisation,” *Machine Vision and Applications*, vol. 25, no. 3, pp. 633–647, 2014.

[185] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel, “Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition,” *Neural networks*, vol. 32, pp. 323–332, 2012.

[186] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., pp. 1097–1105. Curran Associates, Inc., 2012.

[187] Ross Girshick, “Fast r-cnn,” in *International Conference on Computer Vision (ICCV)*, 2015.

[188] A. Mogelmose, M. M. Trivedi, and T. B. Moeslund, “Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1484–1497, Dec 2012.

[189] Nobuyuki Otsu, “A Threshold Selection Method from Gray-level Histograms,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, Jan. 1979.

[190] Rafael C. Gonzalez, Richard E. Woods, and Steven L. Eddins, *Digital Image Processing Using MATLAB*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2003.

[191] Mark Everingham, S. M. Ali Eslami, Luc Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman, “The pascal visual object classes challenge: A retrospective,” *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2014.

[192] Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers, and Arnold WM Smeulders, “Selective search for object recognition,” *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[193] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[194] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel, “Backpropagation applied to handwritten zip code recognition,” *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[195] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, “classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[196] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet, “Torch7: A matlab-like environment for machine learning,” in *BigLearn, NIPS Workshop*, 2011, number EPFL-CONF-192376.

[197] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, “Caffe: Convolutional architecture for fast feature embedding,” in *Proceedings of the ACM International Conference on Multimedia*. ACM, 2014, pp. 675–678.

[198] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

VITA

NAME OF AUTHOR: Koray Özcan

PLACE OF BIRTH: Ankara, Turkey

DATE OF BIRTH: March 22, 1989

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

SYRACUSE UNIVERSITY, Syracuse, New York; August 2011—January 2017

BILKENT UNIVERSITY, Ankara, Turkey; 2007—2011

DEGREES AWARDED:

B.Sc. Electrical and Electronics Engineering (2011), BILKENT UNIVERSITY

PROFESSIONAL EXPERIENCE:

June—December 2015

Research Intern, IBM T. J. WATSON RESEARCH CENTER, Yorktown Heights, NY.