AUTOMATED CROWD BEHAVIOR ANALYSIS FOR VIDEO SURVEILLANCE APPLICATIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

PÜREN GÜLER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

SEPTEMBER 2012

AUTOMATED CROWD BEHAVIOR ANALYSIS FOR VIDEO SURVEILLANCE APPLICATIONS

Submitted by **Püren Güler** in partial fulfillment of the requirements for the degree of **Master of Science in Information Systems, Middle East Technical University** by,

Prof. Dr. Nazife Baykal _____
Director, **Informatics Institute**

Prof. Dr. Yasemin Yardımcı Çetin _____
Head of Department, **Information Systems**

Assist.Prof. Dr. Alptekin Temizel _____
Supervisor, **Work Based Learning Studies, METU**

Assist.Prof. Dr. Tuğba Taşkaya Temizel _____
Co-Supervisor, **Information Systems, METU**

Examining Committee Members:

Assist.Prof. Dr. Erhan Eren _____
IS, METU

Assist.Prof. Dr. Alptekin Temizel _____
WBL, METU

Assist.Prof. Dr. Tuğba Taşkaya Temizel _____
IS, METU

Assist.Prof. Dr. Aybar C. Acar _____
MIN, METU

Assist.Prof. Dr. Sinan Kalkan _____
CENG, METU

Date: 14.09.2012

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Püren Güler

Signature : _____

ABSTRACT

AUTOMATED CROWD BEHAVIOR ANALYSIS FOR VIDEO SURVEILLANCE APPLICATIONS

GÜLER, PÜREN

M.S., Department of Information Systems

Supervisor: Assist. Prof. Dr. Alptekin Temizel

Co-supervisor: Assist. Prof. Dr. Tuğba Taşkaya Temizel

September 2012, 73 pages

Automated analysis of a crowd behavior using surveillance videos is an important issue for public security, as it allows detection of dangerous crowds and where they are headed. Computer vision based crowd analysis algorithms can be divided into three groups; people counting, people tracking and crowd behavior analysis. In this thesis, the behavior understanding will be used for crowd behavior analysis. In the literature, there are two types of approaches for behavior understanding problem: analyzing behaviors of individuals in a crowd (object based) and using this knowledge to make deductions regarding the crowd behavior and analyzing the crowd as a whole (holistic based). In this work, a holistic approach is used to develop a real-time abnormality detection in crowds using scale invariant feature transform (SIFT) based features and unsupervised machine learning techniques.

Keywords: Crowd Behavior, Abnormality Detection, Holistic, SIFT

ÖZ

VİDEO GÖZETLEME UYGULAMALARI İÇİN OTOMATİK KALABALIK DAVRANIŞI ANALİZİ

GÜLER, Püren
Yüksek Lisans, Bilişim Sistemleri Bölümü
Tez Yöneticisi: Yrd. Doç. Dr. Alptekin Temizel
Eş Tez Yöneticisi: Yrd. Doç. Dr. Tuğba Taşkaya Temizel

Eylül 2012, 73 sayfa

Gözetleme videolarını kullanarak otomatik olarak kalabalık davranışı analizi yapmak, tehlikeli kalabalıkların tespitini ve bir kalabalığın nereye gittiğinin tespitini sağladığından, toplum güvenliği açısından önemli bir konudur. Bilgisayarla görme tabanlı kalabalık analizi üç gruba ayrılabilir; kişi sayımı, kişi takibi ve kalabalık davranış analizi. Bu araştırmada, kalabalık davranış analizi konusu üzerine çalışılacaktır. Literatürde,kalabalık davranış analizi konusunda iki çeşit yaklaşım bulunmaktadır: birinci yaklaşım kalabalıktaki bireylerin davranışlarını analiz etme (obje tabanlı) ve bu bilgiyi kullanarak kalabalığın davranışı hakkında çıkarım yapma, ikinci yaklaşım ise kalabalığı bir bütün olarak analiz etme (bütüncül) olarak tanımlanabilir. Bu çalışmada, her iki yaklaşım da irdelenecek ve en uygun yaklaşım üzerine daha ileri araştırmalar yapılacaktır. Bu çalışmada, gerçek zamanlı kalabalık davranış anomalilerini tespit etmek için ölçekleme değişmez özellik dönüştürme

(SIFT) yöntemi ve makine öğrenimi yöntemleri kullanılarak bütüncül bir metod uygulanmıştır.

Anahtar Kelimeler: Kalabalık Davranış Analizi, Anomali Tespiti, Bütüncül, SIFT

ACKNOWLEDGMENTS

I wish to express my gratitude to Assist. Prof.Dr. Alptekin Temizel and Assist. Prof.Dr. Tuğba Taşkaya Temizel for their supervisions and encouragements. I am very grateful that they were always eager to help me and teach me. Thanks to their guidance, I have a chance to observe how to be good academician and develop myself academically.

This research was funded by Ministry of Science, Industry and Technology SAN-TEZ program grant number 00542.STZ.2010-1.

I am also thankful to Mustafa Teke for his help in the computational time measurement and for returning me kindly whenever I ask him something.

I would like to thank to my dearest friend Deniz Emeksiz who is always there for me whenever I need her and always encourages me. I only know her for two years but feels like a life time.

I am also grateful to my dearest friends whom I see as my second family, Seda Dumlu, Gülden Olgun, Anıl Akın, Nilüfer Aksoy, Sanem Uzeler, Suna Büyükkılıç, Emel Pehlevan, Ayça Zeybek, and Mina Nabi for their moral supports.

Lastly, I would like to express my gratefulness to my family (Metin Güler, Serap Güler, Nehir Güler). They are always there for me and patient for my needs. I always feel their love and support for me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

SIFT: Scale Invariant Feature Transform

pdf: Probability Density Function

ROC: Receiver Operating Characteristic

AUC: Area Under Curve

UMN: University of Minnesota

GMM: Gaussian Mixture Model

DOG: Difference of Gaussians

GPU: Graphical Processing Unit

AIC: Akaike Information Criterion

DC: Direction Count

FPC: Feature Point Count

V: Velocity

KPSS: Kwiatkowski Phillips Schmidt Shin

KS: Kolmogrov-Simirnov

csf: Cumulative-Step Function

cdf: Cumulative Distribution Function

cpf: Cumulative population function

EM: Expectation-Maximization

TP: True Positive

TN: True Negative

FP: False Positive

FN: False Negative

FPR: False Positive Rate

TPR: True Positive Rate

Vx.y: xth scene’s yth video

CHAPTER 1

INTRODUCTION

Automated analysis of crowd behavior has been gaining importance due to the security implications. A human operator may miss out some hints about the beginning of a dangerous situation in a crowd while watching high number of cameras. In recent years, intelligent systems that detect potential dangerous situations in crowded scenes in real time started to have a significant role in ensuring the security in public environments. As an example for dangerous situations, the disaster that happened in 2010 Love Parade in Duisburg, Germany can be given [1]. If an autonomous system detecting abnormalities were in place, many deaths and injuries could have been prevented. The purpose of this study is to analyze the crowd behavior in real time in order to detect abnormalities that could lead to dangerous situations using computer vision and machine learning techniques.

While detecting the abnormalities in a crowd, understanding the crowd behavior is a crucial issue. The meaning of the behavior in this context can be defined as velocity, direction and density of the crowd and the abnormalities can be defined as the behaviors that do not happen often. The direction of the crowd can give information about the crowd behavior such as the area where the crowd is gathering or heading. Moreover through the velocity information, it can be deduced if the crowd is speeding up or slowing down suddenly which can be an indication of an abnormality in the scene.

Nevertheless, there are some challenges in understanding crowd behavior using computer vision techniques. First of all, tracking every human being and analyzing their behavior in a crowd is a challenging task. If the density of the crowd increases, tracking might fail after a while due to occlusions. Another approach is handling the

crowd as a single entity instead of tracking every human in the crowd individually. One way of doing this is detecting the feature points in the crowd and extracting general information about crowd velocity and direction by tracking these feature points. If the crowd is considered a single entity, detection and tracking individual human beings are not issues; hence occlusion is no longer a problem. This approach is called holistic or pixel-based approach in the literature [2]. In this thesis, holistic approach is adopted due to the advantage of not dealing with the problem happening due to extreme congestion in crowd.

In the literature, there are different techniques for obtaining the features for analysis of crowd behavior such as using optical flow. In this technique, every pixel's direction and velocities are calculated by comparing consecutive frames. However, as stated by Mehran et al. [3], optical flow may result in noisy features. Hence, in this work, scale-invariant feature transform (SIFT), which is a more robust method to extract feature points of an object, is used. Therefore, direction and velocity information are extracted from the video sequence by tracking feature points that are detected by SIFT method.

In this thesis, the effectiveness of SIFT features in extracting information about crowd behavior is investigated. The results are compared with other approaches that use holistic techniques [3, 4, 5]. In most of the state-of-the-art techniques, spatio-temporal events are detected which means that the time and the location of the abnormality in video sequence are identified. However, in this thesis only the global abnormality is dealt with. Global abnormality is defined as focusing on overall change of dynamics changes throughout the video [4]. In other words, instead of modeling change of behavior in a particular area in the scene, the change of behavior in overall crowd is analyzed. In addition, in some works, the events that are aimed to be detected are pre-defined such as detection of running or merging events of people in a crowd using preset rules [6]. In this thesis, there is no pre-definition of the abnormal events. The aim is to learn the scene in normal situations and detecting the abnormalities after this learning phase.

In this study, a pixel-based approach is used for crowd behavior analysis where the crowd is handled as a single entity. The first step is foreground estimation to use as a mask to discard non-moving areas in the video. Then SIFT feature points (key points) of the crowd are extracted and tracked among consecutive frames. SIFT gives the coordinates of matching feature points in the crowd. Using these coordinates, some behavioral properties of the crowd are obtained such as its velocity and direction.

Finally, the video frames are classified as normal and abnormal using probability density function (pdf) of these properties. The abnormality starts when an unusual event occurs. Unusual event is an event that has not observed before in the scene. Figure 1 demonstrates a normal and abnormal activity where the people are walking around in a normal activity, and then they start to run to different directions. In this video, abnormality starts when people start running.

Figure 1: Normal crowd activity (left), abnormal crowd activity (right).

The contributions of this work:

- Real-time detection of global abnormalities
- The processes are justifiable and repeatable. Preset rules are not used.
- In most of the methods in the literature, the local behaviors are modeled by dividing the video frames into patches. This gives a more precise behavior model of the crowd. In this work, local information is not used and it is

shown that with a less complex method, similar performances can be obtained.

The remaining of the thesis is organized as follows. In Chapter 2, literature review on crowd behavior analysis is provided. After literature review section, methods that are applied while analyzing the data are described. First, the feature extraction methodology is described and the data is explained. Then, the statistical analysis procedure applied on the dataset is explained. In statistical analysis phase, since a stationary data is required for training, dataset is tested for stationarity. Later, the distributions in the data are identified and tested to understand the distribution differences. Lastly, training procedure is described. The third section consists of the result and discussion of these results. In the result part, the results of test phase are given. In this part, Receiver Operating Characteristic (ROC) curves, area under curve (AUC) and accuracy values are shown and these results are compared with the state-of-art methods. ROC curve is the graph of true positive rate vs. false positive rate. The ideal case is obtaining high true positive rate and low false positive rate, thus the AUC should be as close as to 1. In the last section, the conclusion and future works are mentioned.

CHAPTER 2

LITERATURE REVIEW

2.1. Overview

In the literature, there are various different approaches proposed for crowd analysis. Crowd analysis can be analyzed in three categories: (1) people counting, (2) people tracking and (3) behavior understanding [2]. In people counting methods, the crowd density is estimated which can be helpful to detect dangerous situations like crowd collapse in an area. In people tracking methods, individuals in the crowd are tracked and individuals' trajectories are used in order to detect main flows or abnormalities in the scene. Behavior understanding techniques are divided into two which are object-level approaches and holistic approaches. In object-level approaches, the individuals in the crowd are detected and the behaviors of these individuals are analyzed. In holistic approach, instead of analyzing crowd through individuals' behaviors, the crowd is treated as a single entity.

In this chapter, studies that address the crowd behavior analysis issue from different viewpoints are discussed.

2.2. Holistic approaches

The methods using holistic approaches do not aim to identify individuals in the scene. The behavior features like velocity and direction are extracted through treating the crowd as a single entity.

In the literature, for analyzing the crowd as a whole, different techniques are used. In [7], texture information is used for event detection. Image segments are divided into

regions and then flows are detected using similarity comparison of dynamic texture (texture with motion) descriptors on regions. Descriptor for an event is constructed using local binary patterns which are created by partitioning consecutive image frames into multiple regions. Hence, a flow is formed through connecting several temporal volume regions. For unusual event detection, log-likelihood is used. In this work, unusual objects in a crowd are detected like a car that goes through a crowd. Since the main goal of this thesis is the detection of abnormal behaviors in a crowd, their methodology which is detecting abnormal object like car in a crowd may not be suitable for this work. There are also other studies that use texture information in the literature [8-14]. In [8], dynamic texture is used to model activities in a crowd. They divide the video volumes into multiple regions and apply association and streamline editing to get rid of noisy links. They combine multiple streamline using Karcher mean to characterize the activities. Martin distance is used as a distance metric between two activities. In this work, it is reported that, using the dataset of a subway train station, they get unsatisfactory results due to movement in the directions of up and left. Study [9] focuses on motion segmentation. They partition the image into spatio-temporal patches. For each patch, a dynamic texture model is created. Then, Martin distance is calculated between corresponding dynamic texture models of the patches to find similar ones, hence motion segments are obtained in a video sequence. In [10], authors perform both crowd counting and event estimation. They use area, perimeter, internal edge features and texture features for crowd counting algorithm and motion fields are characterized as dynamic texture for event classification. For classification, they perform nearest-neighbor classifier in which they apply Kullback-Leibler (KL) divergence or Martin distance. Also, they apply SVM using KL kernel on the data. In the event detection, events like evacuation, dispersion, merging, walking and running are identified correctly. In [11], they apply both crowd counting and event estimation. They use area, perimeter, internal edge features and texture features for crowd counting algorithm and motion fields characterized as dynamic texture for event classification. For classification, they perform nearest-neighbor classifier in which they apply KL divergence or Martin distance. Also, they apply SVM using KL kernel on some of the data. In the event

detection, events like evacuation, dispersion, merging, walking and running are identified correctly. In [12], frames are divided into grids. Features are extracted based on motion, size and texture from the cells of the grids. Optical flow is applied to detect motion flow. They perform two classifiers which are likelihood of the magnitude of motion of foreground objects for speed control and likelihood of size of foreground objects for size and texture control of the crowd. They detect abnormal events like a biker appearing in the crowd. They state that they outperform many methods, however they detect texture based abnormalities such as detecting a skateboarders or bikers in the crowd and this is not the issue of this thesis. There should be done more tests to see if it can detect other kind of abnormalities also. Study [13] creates dynamic texture models using the Local Binary Patterns from Three Orthogonal Planes (LBP-TOP) descriptors. Then, they apply hierarchical Bayesian models in order to detect regions that contain unusual events. Then advantage of their approach is that they do perform neither tracking nor background subtraction. However, their method gives false alarm if there is a perspective distortion in the frame. In [14], uses texture features for crowd abnormality detection. Texture features are obtained using Grey Level Co-occurrence Matrix (GLCM) which is created based on textural statistics measuring like homogeneity, contrast etc. A Gaussian mixture model which is learned using EM is characterized for normal behavior. If there is a statistical outlier according to that model, it is accepted as abnormality.

In some research, energy measurements are applied to understand crowd behaviors. In [15], they construct a probabilistic model of the scene to create entropy and expectancy map for detecting interest points that represent characteristic behaviors of a crowd and abnormal events. Their methodology is better at detecting micro-events in smaller resolutions and better at detecting macro-events in higher resolutions. They use crowd simulation as testing dataset. In [16], the authors formulate two energy methodologies to detect crowd abnormal behavior. In the first energy methodology, they use the change rate of each pixel. In the second methodology, they track corners in video frames using Lucas-Kanade optical flow and use velocity of the found motion features to evaluate the energy. In [17], the authors use Markov

Random Field (MRF) to calculate the energy which crowd behavior creates in the image sequence. In [18], they use kinetic energy measurement and the motion direction is estimated through optical flow as features. Their methodology is based on feature tracking. They build a direction histogram using the tracklets of motion in the scene. Their technique identifies an abnormality in the scene from the estimated crowd motion models. They note to use more crowd characteristics and machine learning techniques in their work and also to implement their technique using GPU as future works.

In the literature, in most of the studies, feature point tracking is performed to extract the features of the crowd like velocity and direction. In [19], they detect abnormal events online. In their methodology, a non-parametric likelihood representation is constructed by learning the crowd behavior through optical flow orientation within image blocks. Then, sparse vector machine based model is constructed by using only relevant samples according to this non-parametric likelihood. Due to the possible motions that have not been learnt, they obtain some wrong positive detection. The other reason of obtaining wrong positive detections is that optical flow results in some noisy features. Method [20] tracks low level features using optical flow. Then, they cluster these feature trajectories to find dominant motion. Another study that uses optical flow is [21]. In [21], they detect the motion flows using optical flow and then cluster these flow vectors to detect typical motion patterns. Study [22] uses background subtraction and optical flow for preprocessing. They apply PCA to extract feature prototypes from extracted flow fields during preprocessing. Then, through spectral clustering, the number of Hidden Markov Models (HMM) is calculated to represent the flow sequences. Each cluster of the data is used to train HMM models. In [23], authors use optical flow to detect crowd flows. Then, adjacency-matrix is used to cluster crowds into different action patterns. Motion features which they use are orientation, position and crowd size. Method [4] applies optical flow to detect particle advection's trajectories. Then, they cluster these particle trajectories to find the directions of crowd flow. They model the scene through chaotic dynamics using clustered trajectories. Maximum likelihood estimation criterion is performed to detect video event that is normal or abnormal.

Then they apply an abnormality localization algorithm to identify the position and size of abnormal event. In [24], they characterize crowd motion patterns from local spatio-temporal regions using optical flow. Smoothed optical flow vectors are fit into a Gaussian distribution. Then these models are given as inputs into Self-Organizing Map (SOM). In test phase, it is seen that optical flow estimation causes some errors due to detection of some false flows. Also, the selection of volume location causes false positive detections. In [25], they divide the frames into grids and apply optical flow to each grid to represent motion. For motion representation, they create an 8-bin histogram of optical flow. Then, the learning phase starts. In this phase, the trajectories are learned as spatio-temporal Laglacian Eigen maps. For similarity measures of trajectories, Hausdorff is performed. In [26], the authors compute dense optical flow areas for each cell in the frame to create a 2D histogram of motion magnitude and direction of flow trajectories. They detect the abnormality by detecting change in these 2D optical flow histograms. In [27], the feature that they use in their method is the estimation of boundary point structure and critical points through particle motion field using optical flow in order to model global topological structure of a crowd behavior. The change in the topological structure indicates an abnormality in the crowd behavior.

In the literature, a widely used optical flow technique is KLT (Kanade Lucas Tomasi) tracker. In [28], they try to detect pre-defined crowd events like flow divergence and convergence. To find crowd velocity, they use KLT tracker. They clustered the vectors obtained through feature tracking. Then, they define the events according to the velocity and direction information. Their methodology can be adapted for different end-user scenarios. However, since the aim of this thesis is detecting the abnormalities without pre-defining the events, their methodology may not be suitable for this work. Study [29] models the behavior of the crowd using histogram of motion direction (HMD). They extract the tracklets using KLT feature tracker and then quantize this tracklets into 8 directions to form HMD. Then, according to some pre-defined thresholds specific to some events, they detect the abnormal events. As they state in the paper, their limitation is assumption of the

motion in the scene being consistent. Method [30] uses KLT corner detection for feature extraction. Then, they model motion through the optical flow of these corners. They classify the motion as abnormal or normal by calculating the deviation between these trained model and motion patterns. Their technique is sensitive to camera distance due to texture effect, i.e. small people with less corner points. In [31], they find region of interest by applying Motion Heat Map. First, they extract foreground using mixture of Gaussian and then through Harris Corner detection, points of interests are estimated. KLT tracker is applied to track detected feature points. For analysis, K-means clustering is performed on point of interests. Bhattacharya distances are used to measure the distances of clusters between video frames and a threshold is defined for this distance to detect abnormal events. For flow estimation, study [32] applies optical flow using KLT method. Adjacency matrix-based clustering is used to detect different flows. Then, force field is performed to find dominant behaviors in crowd. To detect the abnormality, they develop a methodology that perceives the change in the dominant orientation. Their technique compares that vector of quantized orientations of two sequential frames.

In the literature, some studies develop a hierarchical methodology for crowd behavior analysis. One of these methods is described in [33]. In [33], they construct different modules to overcome the problem of occlusion in detecting events in complex environment. In module A, they construct a non-uniform blob-based partition of ROI. In module B, dynamic congestion detection is applied using foreground detection and temporal differencing. In module C, they detect motionlessness. In module D, based on the overall congestion, general scene is analyzed. Although they have good matching rates in congestion detection, their methodology is specific to congestion detection. It is not certain whether it works for a more general abnormal detection purpose. In [34], authors create three models to represent activities: low-level visual features, simple “atomic” and multi-model activities. “Atomic” activities are represented as distributions over low-level visual features and multi-model are represented as distributions over atomic activities. These models are learned in an unsupervised manner using clustering. Then three

hierarchical Bayesian models are created to detect abnormal activities through the learned model. Their methodology is successful in summarizing of typical atomic activities and interactions in a scene, clustering long video frames into different actions and detecting different activities in a motion segment.

There are also methodologies that don't apply any tracking or training like [35]. They apply Particle Swarn Optimization for characterizing normal and most common crowd behaviors. They obtain interaction forces using Social Force Model (SFM). In this paper, they try to detect global abnormal events. They model the crowd behavior using PSO and SFM instead of using rectangular grid of particles and velocity calculation which is the most commonly used method. Using a fitness function that minimizes the interaction force, they simulate a normal event in a crowd. They don't use any tracking or training in their method.

Another paper that uses Social Force is [3]. The authors of [3] use Social Force Model to understand activities in the scene. Social Force Model is used to capture "the effect of neighboring pedestrians and environment of movement of individual in crowd" [3]. They compute social force between the moving particles in order to get interaction forces in the crowd. Then they use extracted force models to model normal events in a "bag of words" approach. "Bag of words" approach is used to find the likelihood of a force flow. Latent Dirichlet Allocation (LDA) is used for training. Through LDA, likelihood of a flow is computed. They get better results than pure optical flow in dataset of University of Minnesota (UMN). This work is one of the most compared methods in the literature.

In addition, histograms are used to analyze crowd behavior as in [36], [18], [37], [25], [26], [38] and [39]. Study [36] uses historical information of the motion to model the behavior in the scene. They split the frames into grids and characterize motion in each block using histogram of motion vectors. Motion vectors are estimated using optical flow. Then, similar behavior patterns are clustered. In order to obtain self-history and neighboring history, they model a 3D grid structure for

spatio-temporal relations between blocks. Optical flow utilization results some false motion definitions. In addition, they obtain some false detection in poorly illuminated scenes. In [37], they use MPEG-7 descriptors as features which are crowd kinetic energy, motion directions histogram, spatial distribution parameter and spatial localization. For training, they perform SVM. Their method is successful in detecting crowd; however they should test their algorithm in the scenes that contain more congested crowd. In [38], histogram of spatio-temporal orientated energy is calculated to model spatio-temporal behavior by applying some energy filters on video frames. Then, the histograms are compared to detect abnormalities. Study [39] develops a hierarchical method. For point abnormality detection, they extract feature vectors of the scene and compute histogram of all feature vectors. Then, they decide a threshold to identify bins with lower probability. For sequential abnormality detection, sequence of atomic events associated with the trajectory of an object is obtained. For co-occurrence abnormality detection, co-occurrence of multiple objects is calculated. Abnormalities are detected by first finding normal patterns of co-occurrences and then detecting the abnormalities. Frequent itemset mining algorithm is applied on co-occurrence events for finding abnormality. An HMM is created whose hidden states are normal co-occurrence events in frequent itemset mining. In [6], the authors use FAST algorithm to detect feature points and track these feature points using HOG tracker. Then, they determine some thresholds for pre-defined events and detect these events through these thresholds. Thresholds setup may result in some error.

Moreover, in the literature, there are studies that use Gaussian Mixture Models (GMM) in modeling the crowd behavior. Study [5] applies Spatial-temporal Co-occurrence Gaussian Mixture Models (STCOG). They split the video frames into non-overlapping local areas. Then, phase correlation is performed for calculating motion vectors between two consecutive frames for all local regions. After that, STCOG models the most common events and identify the abnormality. Since they apply phase correlation, their methodology has lower computational costs than most of the other techniques. Method [40] models the block clips as 2D distribution of

mixtures of Gaussian. They determine the mixture model using k-means clustering. Through conditional random field models that are estimated for each block clip, events are classified as normal or abnormal. Because of some prominent motion field, i.e. leg of a person, they get some false detection.

Some methods in the literature use heat map or energy map for crowd behavior analyzing as in [41]. In [41], they construct a motion heat map in order to detect POIs. They try to estimate sudden changes in those POIs by using optical flow. Study [42] models spatio-temporal behaviors in an image sequence. First, they extract foreground objects using energy map. Through the shape of this energy map, they obtain time-space interactions of one person or more than one person in the scene. Then, some features like invariant moment, entropy etc. are extracted from these interactions. They use hierarchical clustering for discriminating different energy maps and fuzz C-means clustering in each hierarchical cluster. For training, minimum distance between each cluster is calculated for determining threshold in order to decide if a new scene is abnormal or not by comparing its feature vector distances to cluster centroids are bigger than or smaller than the threshold.

In the literature, segmenting the crowd behavior is a popular approach as it is seen in some papers that are mentioned above. One of the example researches that use that methodology can be seen in [43]. In method [43], they use the flow segmentation technique using Lagragian coherent structures. In their methodology, new flows are detected clustering flow segments. Flow segments are defined using Lagragian coherent structures which are particles having the same behavior in the same region of a crowd.

Another used technique is dividing the video sequence into spatio-temporal patches and analyzing the crowd behavior through the model that is obtained through these patches. This method enables to localize the abnormality in the video frame. The authors of [44] model rich spatio-temporal motion patterns in local areas in order to detect dense activities in a crowded environment. They define relationships between

local spatio-temporal motion patterns using their statistical model. They construct a distribution-based HMM to define motion transactions between local video regions. They obtain some false positive detection because of some slightly irregular motion patterns that do not present in training dataset. In [45], they apply wavelet transform to estimate high-frequent and spatio-temporal features. They divide the video sequences into cuboids parallel to time direction. They perform "bag of word" technique to model global events and HMM for local events. In [46], they model the motion by first dividing the frame into blocks and then computing spatio-temporal descriptors. For abnormality detection, they search for motion patterns that deviate from the estimated model. Their spatio-temporal descriptors are based on [47]'s method which estimates "the linear dependency between spatial gradients and the temporal gradient". Method [48] extracts spatial-temporal features from 3D blocks of a video sequence. For motion estimation, they calculate a motion vector matrix by applying adaptive rood pattern search, block-matching and distance normalization. Also they create a saliency map by performing a center-surround difference operator for each block. Using motion vectors and saliency map, an attractive motion disorder descriptor is created. Although, it is a completely unsupervised method, they state that they get better performance than other state of art methods. However, they note that they should add other reliable features like location distribution prior in order to get better results. In [49], their method learns statistics about co-occurring events in a spatio-temporal volume in order to build normal behavior model. Co-occurrence matrix is input to Markov random field framework to describe video streams using the probability of observing new volumes of activity. They detect the abnormality by calculating likelihood ratio of an observation's co-occurrence matrix which is learned in training phase. In [50], their technique extracts spatio-temporal volumes from the video sequence. Representation of each volume is based on three-dimensional gradients computed using luminance values of pixels. The volume is divided into 18 cells. They run nearest-neighbor (NN) algorithm in training set for abnormal event detection. The pattern that is obtained through NN algorithm, in which if there is not any neighbor in the training set, the data is accepted as abnormal.

There can be found other different techniques that are applied in crowd behavior analysis in the literature. In [51], the authors represents crowd flow using velocity and a disturbance potential through frame sequence by predefining some obstacles and destination regions in the frame. They use simulated data. Since there is some pre-definition for an event, it is not very suitable for the approach of this thesis. In [52], they apply a method that learns motion patterns off-line. Then, through these learned motion patterns, matched crowd videos are found in a large crowd database. In the test phase, the input video patches are matched with the subset of videos. For extracting the motion pattern, Kalman filter is used. In [53], they use social entropy to overcome the problem of uncertainties in flow data. They model the crowd behavior using flow features. They divide each frame into blocks and characterize each block behaviors separately. For classification and detection of abnormal event, they perform SVM.

2.3. Object based approaches

In this approach, the mostly used technique is creating a template for the detected behavior and comparing this template with the obtained single person actions. In [54], they detect single human actions. They segment video into spatio-temporal regions using mean shift and then a template is used in order to match the over-segmented volumes for event understanding. In [55], they detect single human behavior in a crowd, such as bending down while most are walking. They use two methodologies for that. First one is to detect blob of each person and to extract the border of the blob. Second one is optical flow which is used to extract blobs in extreme occlusion situation. They use PCA for feature selection and SVM for classification. In [56], they try to detect a single person's suspicious behavior in a crowd. They extract body contours to locate the person. With the help of a parser, each body part is sent to its own gesture recognition model. If action matches one of the predefined motions, it is outputted.

Another approach is to analyze the crowd behavior with a graph based approach. The method proposed in [57] tracks each person in a crowd and extracts their trajectories using multiple cameras. After that, they represent trajectories as sequences of transitions between nodes in a graph. In [58], they determine some pre-defined activities that can occur in a crowd (close-to, moving-closer, next-to etc.). A graph is created for objects that indicate the predicates calculated at the same time for the same object. Then, spectral clustering is applied on the adjacency matrix of the graph to find objects that behave similarly. Although the methodology provides a generic activity representation, tracking every object is difficult when people are very close to each other and resolution is not very high. Methods that use chaotic invariants [4] outperform their method, but their algorithm runs faster.

In [59], the authors detect contextual abnormalities by representing motion features as spatio-temporal patches characterized by dynamic texture. Each of these patches represents a pedestrian. The authors of [60] use features that are obtained from each frame are group connectivity, moving direction, connectivity change and moving speed. To extract those features, they detect people in the frame. Then, they calculate a feature histogram for each frame and for every four sequential frames. Their method learns a group context word through these feature histograms. The bag of words approach is applied in order to train SVM. In [61], they perform an object detection algorithm before identifying abnormalities in a crowd. For doing this, they estimate the probability for a new object using GMM. Then, for detecting an abnormality in a sequence of images, they identify the image with the smallest event probability using Exponentially Weighted Moving Average (EWMA) chart. Occlusion in the video sequence decreases the accuracy rate of the method.

2.4. Studies using SIFT features

In this thesis, SIFT is applied for feature extraction. SIFT is a method that is used for object detection [62]. Since it finds reliable matched feature vectors among the different viewpoints of an object, it is also a reliable feature tracking methodology

[62]. The studies [63] and [64] proves the efficiency of the SIFT feature in finding the feature points of an object. In [64], the proposed methods is compared with FAST+HOG which is another feature point extraction method and the results show that SIFT based methods give better performance.

The basic algorithm of SIFT is as follows: First, in order to match an object that is seen from different scales or different directions in two different frames, scale-invariant features are detected [62]. Then, a "Gaussian scale-space pyramid" is generated by down sampling and low-pass filtering the image and the gradients of the image and difference-of-Gaussian (DOG) images, $D(x,y,\sigma)$ where x, y and σ are location and scale information, are calculated on this pyramid [62]. Potential interest points are found by detecting the local extremas in this DOG. Then, by comparing extracted interest points from different scales of the image, best matched feature points are detected. Therefore, the location and scale of the feature points are obtained. Then, orientation assignment to feature points is applied using gradient information [62]. After all these operations, descriptors for local image patches are calculated. These descriptors are very distinctive and invariant to various situations like illumination or 3D viewpoint changes. These descriptors are computed using the gradient magnitude and orientation in the area around the feature point's location, which is weighted with a Gaussian window [62]. These 16x16 regions are divided into 4x4 sub regions. In each of these sub regions, orientation histogram is created (Figure 2).

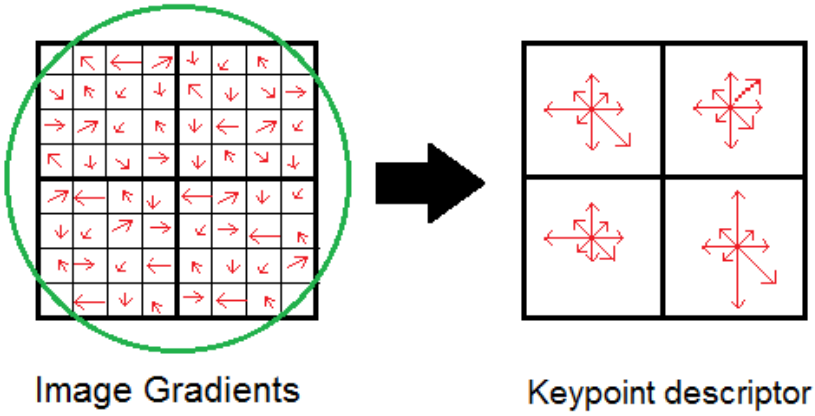

Figure 2: The figure shows the computation of feature point descriptors. It shows that how feature point descriptors are created. First, as seen in the left frame, the gradient magnitudes and its orientations are calculated. The circle around the region is the Gaussian window and is used to weight these gradients. Then in sub regions, the histograms of these gradients are computed as over 4x4 regions seen in the right frame. The figure shows an example of this process with 8x8 regions divided into 2x2 subregions [62].

Since SIFT implementations on CPU are not fast enough in real-time, in this work, SIFT-GPU [65] which allows running the SIFT algorithm in parallel on Graphics Processing Unit (GPU) is used. SIFT-GPU is based on [66] and [67]. In [66], local extremas of DOG pyramid are detected in parallel by using GPU. Another process that is done in GPU is to calculate the gradient histogram which consists of Gaussian weighted gradient vectors near the feature points. The peak detection in the histogram is done in CPU. Later, SIFT descriptors (128 elements) computation comes, which contains the calculation of orientation histograms of "16x16 image patches in invariant local coordinates determined by the associated feautre point scale, location and orientation"[66]. The calculation of gradient histogram which consists of Gaussian weighted gradient vector patches is done in GPU. The reason of partitioning SIFT descriptor process into CPU and GPU is that re-sampling of each gradient vector patch and weighting them with Gaussian window are more efficient in GPU. Weighted gradient vectors are sent back to CPU using texture specialty of GPU to improve the performance of sending the gradient information back to CPU as

stated in [66], since transferring the data as a whole back to CPU would not be efficient.

In the literature, to our knowledge, there is no other study that uses SIFT features in analyzing crowd behavior using the holistic approach. In [68], SIFT features are used for density estimation. They apply texture features extraction to detect abnormal crowd density. For classifying texture features, SVM is used. In this work, the aim of using SIFT features is to obtain more robust texture features. In [69], the authors detect a crowd in still images. While detecting a crowd in a still image, they take into consideration two main ideas: the elements in the images that look like human and some repetitive elements presenting in the crowd. The first point is for a narrow scale and the second one is for large scale. Hence, in their work, they create a pyramid of sliding window to capture these two main points in different scales of the image and to do that SIFT features are used. In [70], they detect single human actions. They use SIFT features in modeling trajectories of people in the scene.

CHAPTER 3

METHODOLOGY

3.1. Overview

In this work, our motivation is to develop a crowd abnormality detection methodology using SIFT based features. Three SIFT based features are extracted: velocity *V*, direction of the crowd *DC* and the feature point count *FPC* in the frame. These properties are obtained through the matched SIFT feature points in each frame.

The general framework of the proposed methodology can be seen in Figure 3.

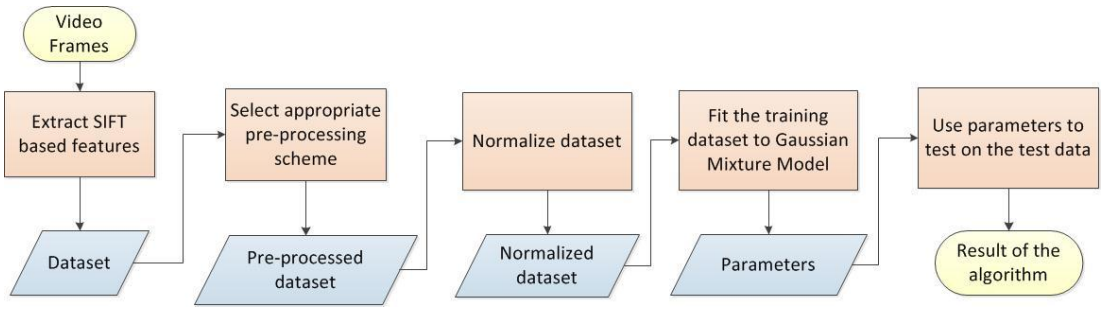

Figure 3: General framework

First, in this thesis, SIFT based features are extracted using two consecutive frames and we match them. Then, an appropriate pre-processing scheme is selected after statistical tests for each SIFT based feature. Choosing a proper pre-processing technique is important for a successful model development. In the next step, the data points are normalized.

A Gaussian mixture model is fitted on the training dataset. In order to detect abnormalities in video sequence frames, a model should be trained on training video datasets which have content that corresponds to ‘normal behavior‘. The training dataset contains events that appear frequently or are common in the scene. Abnormal

part consists of the events that happen unexpectedly in the scene. As a result, Gaussian mixture models are applied on the normalized training data points. The number of mixtures are found using Akaike Information Criterion (AIC) [71]. The thresholds to detect abnormal events are obtained automatically using the Gaussian mixtures fitted on the training data and the sample training video frames comprising abnormal behavior.

Finally, the parameters obtained from the previous step are utilized in the testing dataset for comparison.

3.2. Step 1: Feature extraction

SIFT gives information about the location of SIFT feature point in every frame. By tracking these SIFT feature points, the velocity and direction information are obtained.

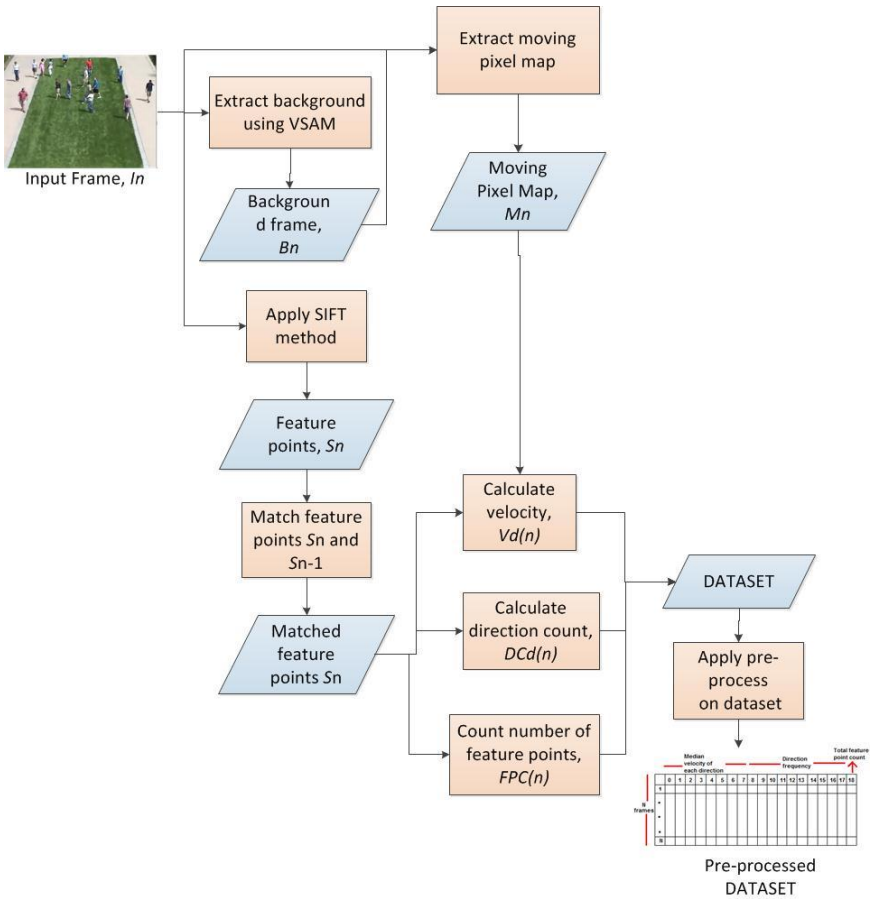

Figure 4: Figure of dataset creation from video frames

In this study, only the feature points of moving people on the foreground are needed as SIFT features obtained from non-moving parts of the video might increase the number of incorrect matches. Hence, background feature points are disregarded by detecting foreground pixels using the moving pixel map extraction processed as in [72].

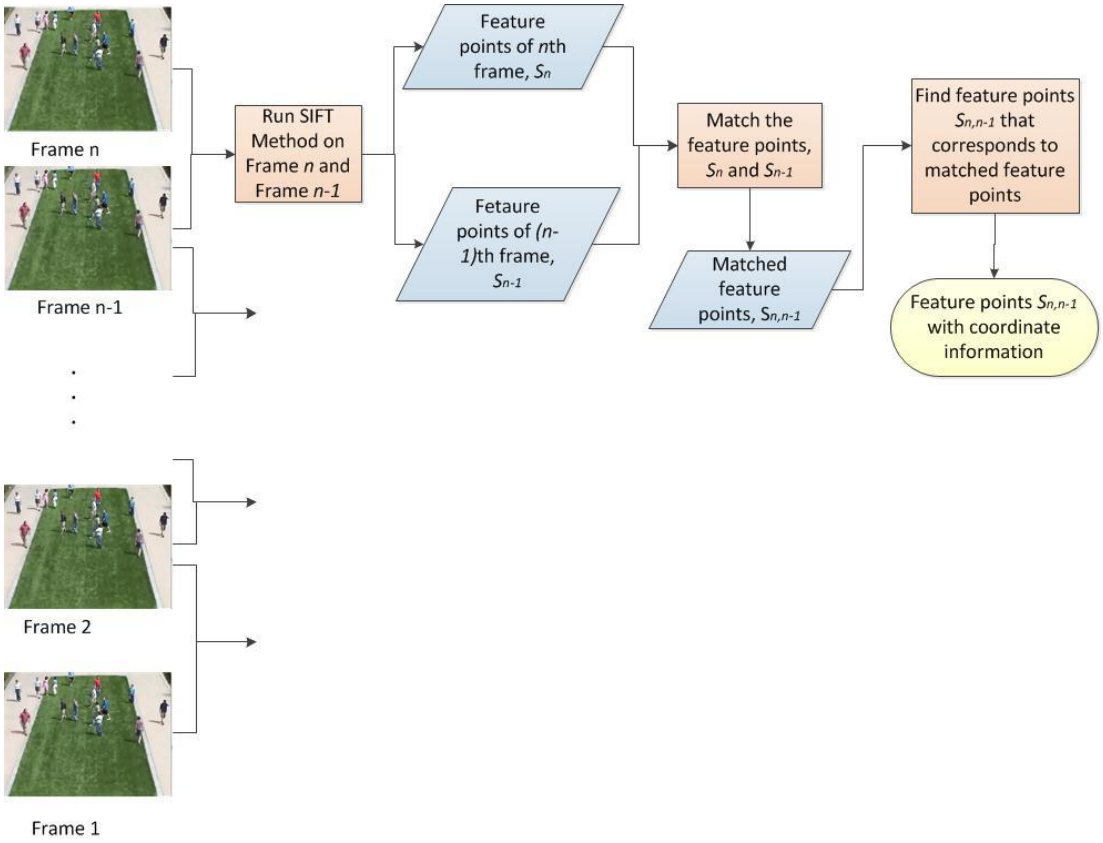

Figure 5: How the feature points are extracted

To extract moving pixel map, first background subtraction is performed using Video Surveillance and Monitoring (VSAM) algorithm [73]. In this method, the background frame is updated at each frame using equation 1.

$$B_{n+1}(x,y) = \begin{cases} \alpha B_n(x,y) + (1-\alpha) I_n(x,y) & if\ (x,y)\ \ is\ not\ moving \\ B_n(x,y) & if\ (x,y)\ \ is\ moving \end{cases} \quad (1)$$

B_n *(x,y)* and B_{n+1}*(x,y)* are the background images at time *n* and *n*+1 respectively. $I_n(x, y)$ is the *n*th frame's pixel. α which is between 0 and 1 affects the background update speed and set to 0.92. I_n*(x,y)* is the input frame at time *n*.

The first frame of the video is used to initialize the background. The moving pixels are determined by taking the difference of current and previous frames. If this difference is higher than the adaptive threshold of the corresponding pixel, then the pixel is accepted as moving (equation 2).

$$|I_n(x, y) - I_{n-1}(x, y)| > T_n(x, y) \quad (2)$$

$T_n(x, y)$ is the adaptive threshold image that holds the threshold value for each pixel in the *n*th frame. The threshold is updated as follows in equation 3:

$$T_{n+1}(x, y) = \begin{cases} \alpha T_n(x, y) + (1 - \alpha)(c|I_n(x, y) - B_n(x, y)|) & if\ (x, y) is\ not\ moving \\ T_n(x, y) & if\ (x, y)\ is\ moving \end{cases} \quad (3)$$

where *c* is the sensitivity parameter which is 5.

Moving pixel map in [72] is calculated to detect regularly changing parts of the image (equation 4).

$$M_{n+1}(x, y) = \begin{cases} M_n(x, y) + \beta|I_n(x, y) - B_n(x, y)|, & if\ (x, y) is\ moving \\ M_n(x, y) - \gamma|I_n(x, y) - B_n(x, y) + 1|, & if\ (x, y) is\ not\ moving \end{cases} \quad (4)$$

M_n*(x,y)* is the pixel of moving pixel map for *n*th frame. *B* and γ are the constant values. γ should be smaller than β in order to mark a pixel as non-moving if no motion is observed. The pixel values of M_n greater than a preset threshold are accepted as regularly changing areas in the I_n. For β, different values are tried. If it is too small, the moving pixel area could be smaller and if it is too big, than even small changes in the movement in an area could be captured such as light changes. Hence, 20 is chosen as the most suitable value. γ is set to 1.

After obtaining moving pixel maps of the frame, this map is used as a mask to obtain only the feature points of foreground objects. The mask and the feature points on the mask can be seen in Figure 6 for an example video.

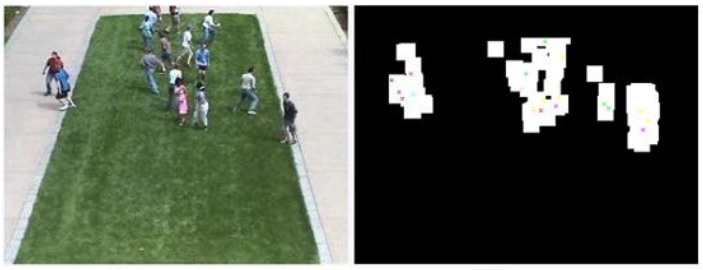

Figure 6: (a) current frame I_n, (b) M_n, blobs shows the moving areas, points are feature points.

Feature points are only calculated for these moving regions and matched between consecutive frames. Then, by taking the Euclidian distance between the coordinates of two matched feature points of two consecutive frames, the motion vector is obtained as seen in the Figure 7.

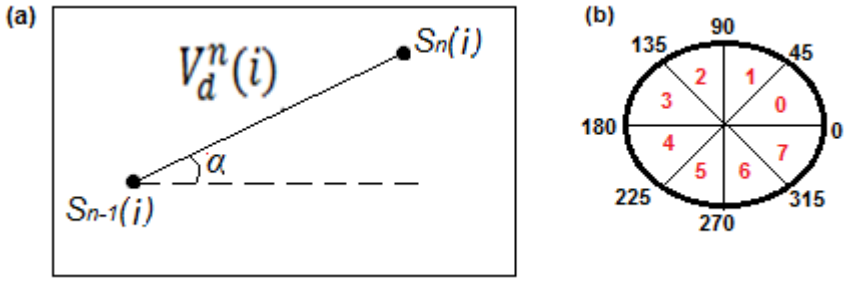

Figure 7: The computation of direction for each feature point in each frame. $S_n(k)$ is nth frame's kth feature point. $S_{n-1}(k)$ is the $(n-1)$th frame's kth feature point. $\boldsymbol{V_d^n(i)}$ is the ith velocity of dth direction in nth frame.

The direction information is calculated using angle α between the motion vector and the x-axis. The angle information is quantized between [0, 7] values (Figure 7-b) since there are 8 main directions (north, south, west, east, north-east, north-west, south-east, south-west). For instance, if the angle α is greater than 0 or less than 45 degree, then it belongs to direction 0. Based on the direction information, two features are calculated: Velocity of each direction and each direction count.

The first feature is that of velocity information. An abnormal situation can be captured using the velocity information of the overall scene. If there is a sudden increase in the velocity, it can be an indication of abnormality. To obtain that information, the velocities in each direction are placed into vector $V_d^n(i)$ where *V*, *n*, *d* and *i* denote velocity, frame number, direction and the index of the velocity value in the vector corresponding to each matched feature, respectively. Then, from this vector, $V_d(n)$ is extracted by taking the median value of $V_d^n(i)$ for all *i* as the representative velocity value of direction *d* for each frame *n*. In $V_d(n)$, *V*, *d* and *n* denote the velocity, direction and time respectively. In the Figure 8 that shows the velocity feature calculation, Euclidean distance threshold, *ds* is for getting rid of the noisy features. The detail explanation of this threshold can be found in Choosing the pre-processing technique section.

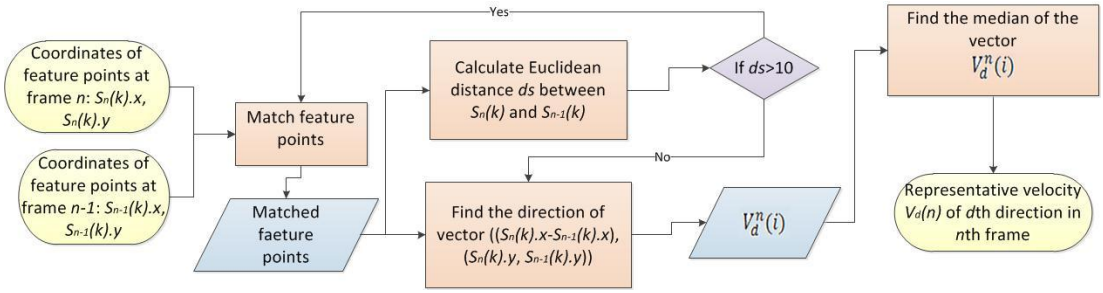

Figure 8: Velocity calculation for each matching feature point in each frame

The second feature is that of each direction's count in the frame. This information is also called direction frequency. The change in the feature count in a direction may give information about crowd behavior. For example, if the crowd suddenly goes in one direction only, this change is reflected in the feature point count in this direction. The count in each direction is calculated as follows: We find the direction of the crowd $DC_d(n)$ by counting each feature point for each direction *d* in each frame *n*.

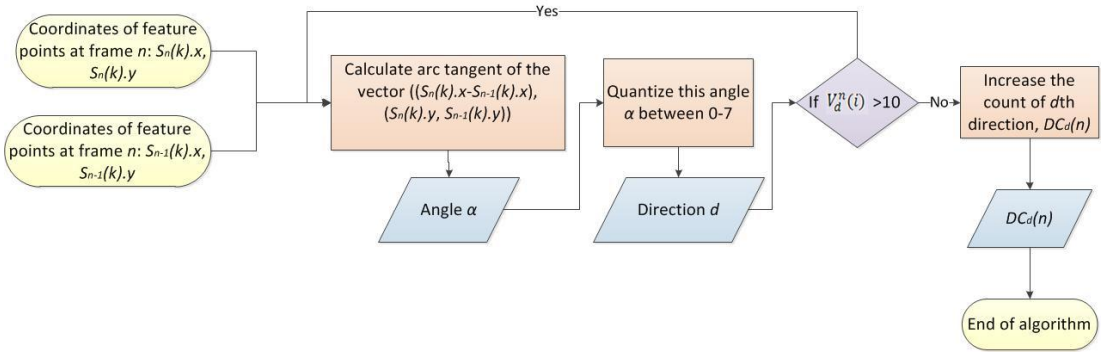

Figure 9: Direction count for each matching feature point in each frame

Another extracted feature using SIFT feature points is the total valid feature point count *FPC(n)* of each frame. Abrupt change in this count can be an indication of abnormality in the scene.

At the end of the above processes, a dataset is obtained. This dataset has *N* observations which correspondence to frames of the video and 17 dimensions. The dimensions between 0 and 7 contain the velocities of each direction in a frame. For example, direction 0's velocity of the frame *n* is in n^{th} row's first column of the dataset. In dimensions 8-17, there is direction frequency information for each frame. The total feature point count is stored in the last column.

3.2. Step 2: Pre-processing

Before training, we aim to pre-process the dataset so the training process can be carried out properly. The training data should be stationary before the training starts since a stationary data is necessary for a good fit. Hence the normalization technique that produces stationary data is selected for pre-processing.

As a result, a series of statistical tests are applied to understand the characteristics of the dataset. Figure 10 shows the statistical tests that are applied to the datasets.

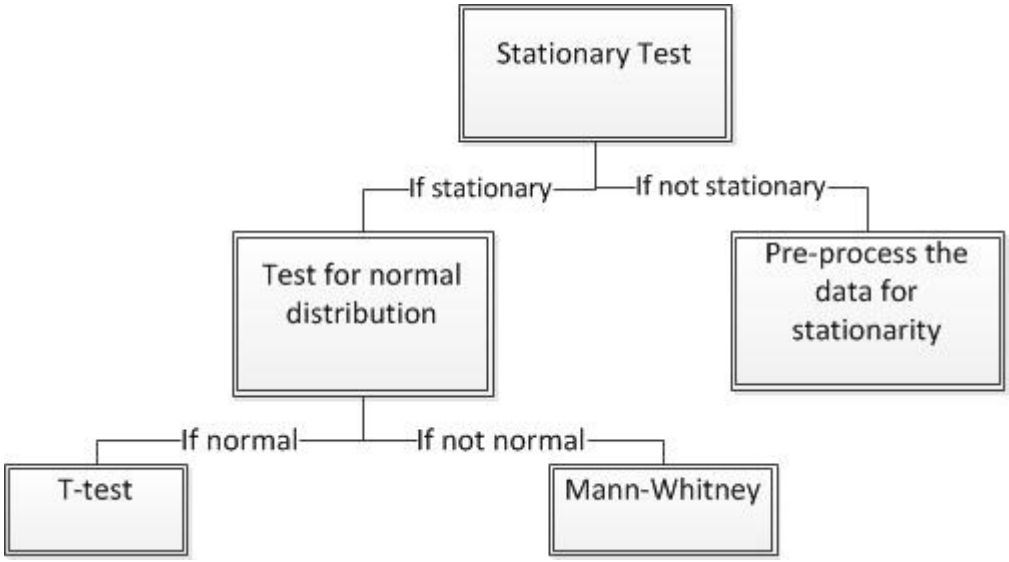

Figure 10: Statistic test phases

3.2.1. Test 1: Stationarity measure

Stationarity can be described as a time series having a constant mean, variance and covariance [74]. Another type of stationarity is trend stationarity which has a mean growing around a fixed trend. The majority of the current machine-learning techniques require that the data is stationary. That's why we tested our raw dataset against the null hypothesis that *x* (input data) is level stationary. If the dataset is not stationary, the underlying reasons are investigated and found an appropriate pre-processing technique for the raw data.

For testing stationarity of the data, KPSS (Kwiatkowski Phillips Schmidt Shin) test is used [75]. In this test, the data series are modeled as it is:

$$y_t = \xi t + r_t + \varepsilon_t \quad (7)$$

$$r_t = r_{t-1} + u_t \quad (8)$$

where y_t:series that is tested against level stationarity, $t = 1,2,\ldots, T$.

ξt: deterministic trend

r_t: random walk

ε_t: stationary error

u_t: iid(0, σ_u^2)

The hypothesis for stationarity can be stated as $\sigma_u^2 = 0$. ε_t is also stationary and iid $(0, \sigma_\varepsilon^2)$. Hence, y_t is assumed trend-stationary under the null hypothesis. For level stationarity test, ξ is set to 0. They use Lagrange Multiplier (LM) test to test the stationary hypothesis:

$$LM = \sum_{t=1}^{T} S_t^2 / \hat{e}_\varepsilon^2 \tag{9}$$

e_t: "residuals from the regression of *y* on an intercept and time trend" [75].

S_t: partial sum of residuals.

$$S_t = \sum_{i=1}^{t} e_i, \qquad t = 1,2, \dots, T \tag{10}$$

The above process is defined for null hypothesis of trend stationarity. If the null hypothesis of level stationarity is tested, e_t is modeled as the residual from the regression of *y* on an intercept which is $e_t = y_t - \overline{y}$ [75].

However, according to [75], the assumption of error ε_t being iid $N(0, \sigma_\varepsilon^2)$ is not practical since the series may be highly dependent over time. Thus, they calculate the 'long-run variance' that is a part of the calculation of the asymptotic distribution. The lag truncation in this test is an important factor that affects the iid errors. The lag values are the values come before the current value in a time series. As it is seen in Table 2 in [75], while the size of the series and the lag value are large, approximately correct sizes are obtained for tests. The choice of lag value *l* is calculated using a function of *T* which is the size of observations using this formula:

$$lx = integer\left[x(T/100)^{1/4}\right] \quad (11)$$

In [75], *l0*, *l4*, *l8* are *l12* are tried. According to the results of their tests, test statistics have correct sizes except when the *l* is large and *T* is small.

Implementation

The training data comprises video sequences having 'normal' and 'abnormal' behavior. As the characteristics of these two parts are different, we divided the training video data into two parts and tested both parts' extracted SIFT based features against level stationarity separately.

The implementation is undertaken with the Matlab function kpsstest(x) [76]. The function takes a series which are the feature vectors as input. There are also optional inputs:

Alpha: It shows the significance level of the test. If p-value of the test is less than alpha, the test always rejects the null hypothesis [76]. The lower the alpha value is, the higher the rate of false positive alarm and while it gets higher, the false negative rate increases [77]. In kpsstest(x) function, the default value is 0.05. However, we set the p-value as 0.01.

Lags: The lag value can be arranged for the rejection of the null hypothesis. Following a well-known study [76], it is considered values of the lag truncation parameters *l4*, *l8* and *l1*. Then based on these values, the statistic value of a series that exceed the critical value is accepted as rejecting the null hypothesis of being stationary.

Trend: It is set to true as default. 'trend' option assumes that the series are trend stationary. However, since we are interested in datasets to exhibit level-stationarity characteristics particularly, we set this parameter to "false".

Result

If the dataset is not stationary, we pre-process the dataset.

The test rejects the null hypothesis for *l4*, *l8* and *l12*. Hence, the chosen lag value *l* is 40 for the test since the test accepts the null hypothesis for in this lag value for the current dataset.

The below table shows the stationarity result of data, which is not applied any pre-processing. Columns show the feature vectors in the dataset. V_x, DC_x and *FPC* are the velocity, direction count for direction *x* and the feature point count respectively. V1.1, V2.1 and V3.1 are the training videos for different scenes in the used dataset (scene 1, 2 and 3, see Experimental Results and Comparisons section for details of the datasets). *Norm* and *Abn* are the normal and abnormal part's results of the tests. The values in the table are the *p* values of the statistic test. The yellow cells are the data part that rejects the null hypothesis of being stationary.

Table 1: Stationarity test results. The yellow cells are the non-stationary data.

V0	V1	V2	V3	V4	V5	V6	V7	DC1	DC2	DC3	DC4	DC5	DC6	DC7	FPC
0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.023	0.1	0.1	0.1	0.04	0.1	0.1
0.1	0.086	0.08	0.1	0.1	0.1	0.1	0.1	0.1	0.094	0.1	0.082	0.09	0.1	0.1	0.088
0.1	0.095	0.087	0.1	0.062	0.1	0.074	0.1	0.063	0.098	0.095	0.044	0.033	0.044	0.03	0.03
0.026	0.027	0.028	0.065	0.1	0.1	0.045	0.024	0.028	0.036	0.087	0.1	0.1	0.081	0.031	0.024
0.01	0.01	0.1	0.1	0.1	0.078	0.01	0.017	0.016	0.1	0.1	0.1	0.1	0.1	0.078	0.01
0.095	0.073	0.1	0.098	0.1	0.1	0.1	0.1	0.074	0.1	0.1	0.098	0.1	0.1	0.086	0.091

One of the rejected vectors of the dataset corresponding to 'normal' behavior is *FPC* of V3.1. To make this vector stationary, the rate of change (differencing) of *FPC* is calculated according the formula in [78].

$$R_i = \left(\frac{D_i - D_{i-n}}{D_{i-n}}\right) * 100 \tag{12}$$

D_i is the *i*th observation in data vector *D*. R_i is the rate of change value of D_i. *n* shows how many data before is used for calculation. D_{i-n} is the *n* data before current data D_i.

The number of previous frames to calculate the rate of change is decided as *n*=5. Now, the null hypothesis is not rejected. The *p* values of the test which is applied on *FPC* vector are displayed in the below table.

Table 2: Stationarity test results for *FPC*

	Normal Data	Abnormal Data
V1.1	0.1	0.1
V2.1	0.1	0.1
V3.1	0.07	0.1

For other rejected datasets that correspond to the 'normal' behavior of vectors *V0* and *V6*, this process is not applied since it may cause loss of data. If the rate of change calculation is applied to the velocity data, the obvious difference between normal and abnormal data may be lost that is seen clearly in the below velocity data of direction 5 of V3.1.

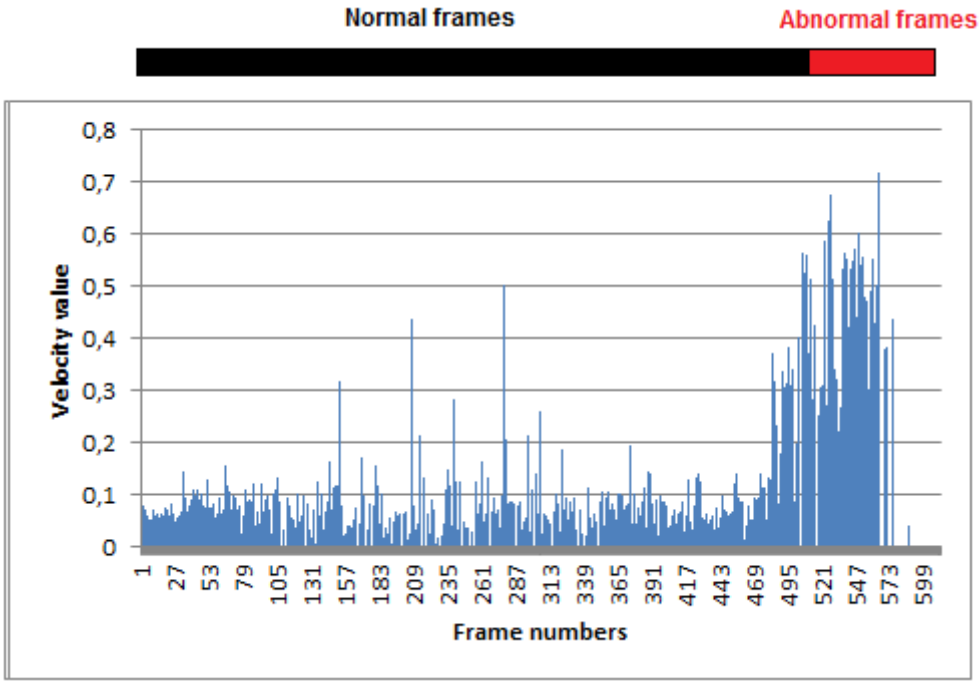

Figure 11: Example velocity data in which rate of change is not applied, y-axis: velocity value, x-axis: frame numbers

3.2.2. Test 2: Test Data for Normal Distribution

If the dataset is stationary, we tested whether the data comes from a normal distribution using Kolmogrov-Simirnov (KS) test [79].

In this test, it is assumed that a population has a cumulative distribution (cdf) function ($F_0(x)$).This cdf means that for every x, $F_0(x)$ gives the proportion of the ones whose measurements less than or equal to x in the population. The cumulative step-function (csf) of a random sample is tested against this cdf for whether the csf is close to cdf or not. If it is not, then the hypothetical distribution is not the correct distribution. Csf function is calculated as follows:

$$S_N(x) = k/N \tag{12}$$

k: number of observations less than or equal to *x* [80]

N: observation number

Hence the maximum distance between cdf and csf tells the test result.

$$d = maximum|F_0(x) - S_N(x)| \tag{13}$$

In [80], the interpretation of d value is given. In the Table 1 in [79], when level of significance is 0.20, the observation number is *N*=10 and the critical value of *d* is 0.322. This means that the maximum absolute deviation of 20 percent of random samples whose size is 10 is at least 0.322 between the sample cumulative distribution (csf) and population cumulative distribution (cpf).

In their methodology, they draw curves above and below of cdf at the distance of "$d_\alpha(N)$" (α: level of significance). The rejection of hypothesis of $F_0(x)$ being the true distribution is accepted when csf ($S_N(x)$) goes outside of this band.

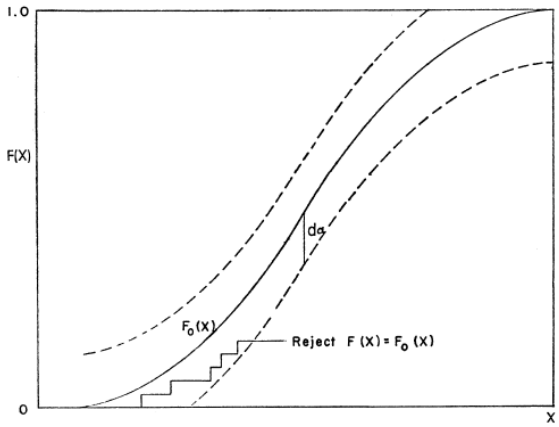

Figure 12: The illustration of how d test is applied, the F_0 (x) function and the bands. Figure is taken from [78].

Implementation

For implementation, kstest(x) function of Matlab is used [80]. The null hypothesis in the implementation is the data being normally distributed. By using CDF option, the data can also be tested for different distribution types. The optional input of the function is *type* which can be 'unequal', 'larger' and 'smaller. The *type* option determines the size difference between the population cdf and the specified CDF. For instance, if it is 'larger', the population cdf is larger than the specified CDF. In this study, the default value of *type* which is "unequal" is used. One of the outputs is *ksstat* which is the test statistic. This test statistic shows the maximum difference between the curves of cpf and csf.

Result

This test is important for us to compare the following two distributions: The distributions generated from the data points that correspond to the 'normal behavior' and 'abnormal behavior' video parts in the video dataset respectively. If they are different, it means that the model fitted on the training data using the selected features will have a greater chance to differentiate the 'normal' and 'abnormal' data. Normality test is critical for choosing the right statistical test for comparison. If the dataset is normally distributed, t-test is applied. If it is not, Mann-Whitney test is applied.

In the Matlab function, the input of ‘unequal’, ‘larger’ and ‘smaller’ are tested. Test is applied to pre-processed data as described in Stationarity Test, Result part. According to the test results, none of the feature vectors are normal. Also, the median filtered versions of the feature vectors are tested. The p value results are around 6.55e-012 and less which means the null hypothesis is rejected. Thus, Mann-Whitney test is applied on the data to compare two distributions.

3.2.3. Test 3: Test to compare two distributions

Our motivation is to choose the features which are able to differentiate the ‘normal’ behavior and ‘abnormal’ behavior in the datasets. Therefore, it is needed to compare the two distributions that correspond to ‘normal’ and ‘abnormal’ parts respectively. The ideal case would be that these two distributions will have different distribution characteristics.

The required test to understand the distribution differences in a dataset is chosen according to [81]. According to [81], if the data is continuous, there are two numbers of groups to compare and the data is not normal distribution, then Mann-Whitney should be used, otherwise t-test should be used. In this test, there are two random variables whose distributions are compared [82]. The measured statistic is ranks of these two groups for the hypothesis of these two groups having the same distributions [82].

The statistic for this test is calculated using the formula below [82]:

$$U = mn + \frac{m(m+1)}{2} - T \tag{14}$$

m: the number of observations in the first group

n: the number of observations in the second group

T: sum of ranks of the first group

If the number of observation is more than 20, then the formula is as follows [83]:

$$z = \frac{U - \frac{n_1 n_2}{2}}{\sqrt{\frac{n_1 n_2 (n_1 + n_2 + 1)}{12}}} \quad (15)$$

The corresponding probability value to *z* is found from a table that can be seen in the Table 1 in appendix of [83]. The value is extracted from 0.5 value and if this value is smaller than α, then the null hypothesis is rejected.

The corresponding probability value to z is found from a table that can be seen in the Table 1 in appendix of [83]. The value is extracted from 0.5 value and if this value is smaller than the α, then the null hypothesis is rejected.

Implementation

Matlab's Wilcoxon rank sum test (ranksum(x,y)) function is used [84]. The input is the two groups that are compared. The optional inputs are 'alpha' and 'method'. Through 'alpha', the significance level is determined. The 'alpha' is set to default value which is 0.05. The 'method' value is for determining with what kind of algorithm *p* value is calculated. The options are 'exact' and 'approximate'. "The default is exact for small samples and approximate for large samples". Hence, this option is adjusted by the function automatically according to the sample size.

Result

The distributions generated from 'normal' and 'abnormal' parts are compared to see whether the selected features are able to differentiate these parts.

Raw data's distribution difference results are below. The values show the *p* values of the test. According to the below results, all the feature vectors reject the null hypothesis of being the same distribution. The empty cells are the eliminated feature vectors from the raw data due to rejection of being stationary.

Table 3: Distribution test results. Yellow cells indicates the feature vectors with same distribution

V3	V4	V5	V6	V7	DC1	DC2	DC3	DC4	DC5	DC6	DC7	FPC
9.13E-07	1.12E-06	2.70E-01	1.09E-22	9.20E-24	9.58E-09	2.41E-55	8.63E-59	9.12E-37	1.32E-40	2.66E-18	1.58E-20	9.23E-58
8.52E-08	3.38E-08	4.11E-01	6.33E-02	1.29E-19	1.67E-05	7.71E-06	2.74E-02	1.87E-24	2.46E-16	1.03E-05	1.41E-01	1.33E-10
1.11E-30	4.10E-06	1.96E-02		9.05E-24	2.73E-16	8.40E-38	7.02E-17	4.05E-17	1.59E-27	2.00E-38	5.71E-32	

The *FPC* of V3.1 that is not stationary is pre-processed by calculating the rate of change of the data. In rate of change calculation, if n=1, stationarity test results positive, but the distribution difference test results negative which means the normal and abnormal parts' distributions are the same. Nevertheless, if n is 5, then the test finds difference between the distributions of abnormal and normal parts. In below figure it can be seen the rate of change of *FPC* of V3.1 for n=1 and n=5.

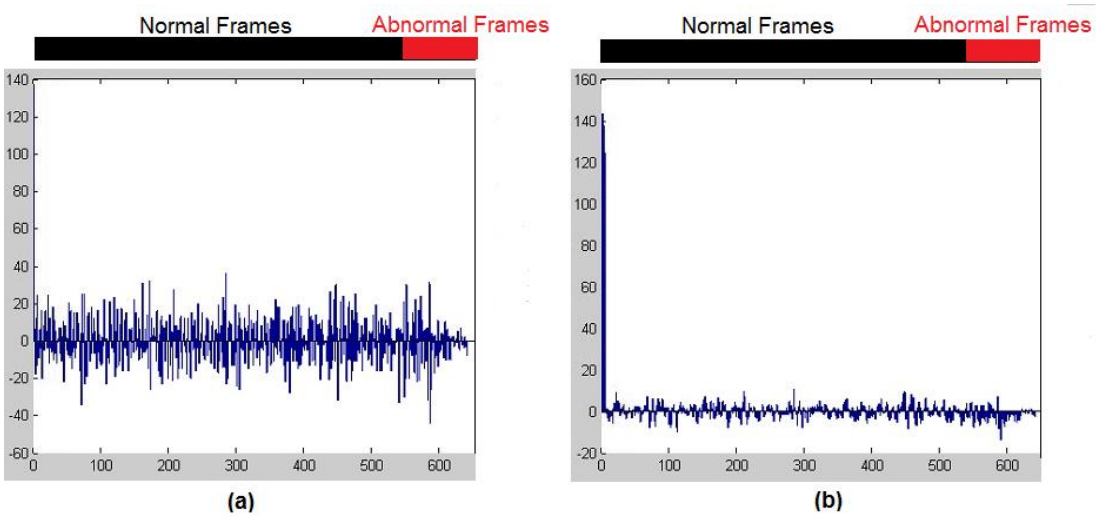

Figure 13: Rate of change of *FPC* V3.1. (a) n=1, (b) n=5

The p values of test are in Table 4:

Table 4: Distribution test results for *FPC*

V1.1	V2.1	V3.1
2.47e-002	6.77e-003	2.13e-005

According to the above results, *p* values are way lower than the alpha value. Hence, the null hypothesis of being the same distribution is rejected for all train data's *FPC*. Then, the normal and abnormal parts are different for *FPC* vectors of V1-3.1.

Now all the feature vectors for train data are ready for training except *V0* and *V6* of V3.1 which are eliminated from training.

3.2.4. Choosing the pre-processing technique

The aforementioned statistical tests are valuable for us to test the significance of the selected features and to choose an adequate pre-processing scheme.

Noise in the data may affect the distribution of the dataset negatively i.e. there may be several outliers. Trends in dataset may also affect the learning process negatively as it makes the dataset non-stationary. If these are the issues, datasets should be pre-processed accordingly. To make datasets stationary such as in *FPC* case, rate of change is calculated. This process does not apply to the velocity data of V3.1 since it may cause the loss of the data.

Additional pre-processing steps are also carried out for individual features. The results in Table 1 and Table 3 are the results of the data after applying the methods that are described below:

Velocity data:

Some peaks in the velocity data are observed even when there is no speeding up in the crowd. It is realized that these peaks occur due to the SIFT finding some incorrect matching in the frames. In order to prevent it, the following filtering process for velocity data is applied: If there is only one element in $V_d^n(x)$ and it is greater than 10, then this direction's velocity information is disregarded. If there is more than one point, and the median is greater than 10 in a direction, then we use the velocity information of the preceding median feature point data.

As a result, we obtain $V_d'(n)$which comprises the median filtered velocity data for each direction in a video sequence.

Direction of the crowd data:

In order to get the noisy feature points that cause irrelevant peaks as in the velocity data, the direction count data also is pre-processed. Each feature point's velocity data $V_d^n(x)$ is checked whether it is higher than 10 or not, if it is, then corresponding feature point is discarded from computation of $DC_d\ (n)$. The threshold for the velocity is determined ad hoc manner by observing the Euclidean distance change of matched feature points. In addition, for feature point count calculation, the same process is applied.

As a result, we obtain $DC'_d\ (n)$which comprises the filtered velocity data for each direction in a video sequence.

At the end of the above processes, a dataset is obtained. This dataset has *N-11* observations which correspond to frames of the video and 17 dimensions. The first 11 data in the dataset are discarded since they may have noisy data. The dimensions between 0 and 7 contain the median velocities of each direction in a frame. For example, direction 0's median velocity of the frame *n* is in n^{th} row's first column of the dataset. In dimensions 8-17, there is direction frequency information for each frame. The total feature point count is stored in the last column.

3.3. Step 3: Normalization

Each feature vector $(V'_d\ (n),\ DC'_d\ (n))$ is scaled separately according to min-max normalization as seen in Figure 14. The reason of normalization is that the feature vectors that are inputted into fitting function together should be in the same range for obtaining a good model fitting.

$$D'(x) = ((D(x) - min) * (newmax - newmin)/(max - min)) + newmin \quad (16)$$

D' is the normalized data. *D* is the raw data. *min* and *max* are minimum and maximum values of *D*. *newmax* and *newmin* are the minimum and maximum values of the new range in which the *D* is scaled. *newmin* and *newmax* is 0 and 1 for this study.

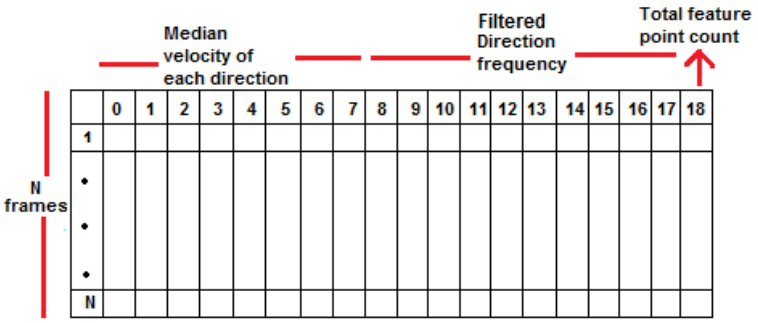

Figure 14: Dataset

3.4. Step 4: Model Fitting

The aim of this step is stochastic detection of global abnormalities. The model fitting step comprises two main sub-steps:

(i) Model fitting on the training dataset that corresponds to 'normal' behavioral part: A Gaussian Mixture Model (GMM) is fitted on this particular data in order to learn the 'normal' behavioral characteristics of the video. In GMM, the parameters of these distributions are estimated using Expectation Maximization (EM). As a result of this process, probability density function (pdf) of the normal part is obtained.

(ii) Determining the pdf threshold to differentiate between 'normal' and 'abnormal' behavior: A threshold value for each video dataset is determined automatically using the ROC curves and this threshold is utilized on testing datasets.

In this section we first explain the GMM models and EM briefly. Then, we will explain how the optimum number of Gaussians is found.

Gaussian mixtures consist of Gaussian distributions. The formula for Gaussian probability density function is as follows [85]:

$$g_{(\mu,\Sigma)}(x) = \frac{1}{\sqrt{2\pi}^{d}\sqrt{\det(\Sigma)}} e^{-\frac{1}{2}(x-\mu)^{\tau}\Sigma^{-1}(x-\mu)} \quad (16)$$

μ: mean vector

Σ: Covariance matrix

A mixture of Gaussian can be written as:

$$gm(x) = \sum_{k=1}^{K} w_k . g_{(\mu_k\ ,\ \Sigma_k)}(x) \tag{17}$$

w_k: weight whose sum is 1, k=1,2,…,K

Expectation maximization is used for completion of the data. If there is not enough data to estimate the model parameters, then through the Expectation phase, the probability distributions by completing the data and through Maximization phase, the parameters are re-estimated over these completions [86]. These steps are repeated until the parameters are converged. The main aim is to find the parameter $\hat{\theta}$ which maximizes the log probability $logP(x;\theta)$ of the dataset. In the E-step, a function g_t which is the lower bound of $logP(x;\theta)$is chosen [86]:

$$g_t\left(\hat{\theta}^{(t)}\right) = \log P(x;\hat{\theta}^{(t)}) \tag{18}$$

In the M-step, a new parameter $\hat{\theta}^{(t+1)}$ maximizing the $g_t\left(\hat{\theta}^{(t)}\right)$ is dealt with.

Implementation

In this study, the training 'normal' data is fitted into Gaussian mixture models in order to estimate the parameters for testing data and to calculate the pdf values.

For implementation, *gmdistribution.fit(X,k)* function of Matlab is used [87]. The inputs of the function are *X* matrix with *n*x*d* dimension in which *n* is the number of observations and *d* is the dimension of the dataset. *k* is the number components in the Gaussian mixture model and this function is calculated for *k* between 1 to 10. There are also optional inputs. Among these optional inputs, 'Regularize' is used and is set to a small value which is 0.00001. This option is used to get rid of the "ill-conditioned covariance matrix" warning of the function. With this option, the value that regularizes the covariance matrix is added to the diagonal of the covariance matrix. Hence, ill-conditioned matrices are discarded. In [87], some reasons are listed

for ill-conditioned covariance matrix presence. One of the reasons is some features in the data being highly correlated. The correlation between features is measured using *corrcoef* function of Matlab by giving features as pairs in this function and it is found that the features are highly correlated. Hence that is the reason of the error. The value is chosen in an ad hoc manner. The reason of choosing such a small value for regularization is that if the regularization parameter is too large, then this value may affect the feature vectors and cause to a poor fit. In [88], they show the effect of regularization parameters. If the regularization parameter is too large, they obtain a poor fit such as converging to a straight line.

The function returns an object which comprises several statistics. The most notable statistics we also used are: *mu* is the matrix that contains *d*-dimensional mean of each *k* component. *d* is the dimension of the data. *sigma* is the covariance of each component. *p* is the mixing proportions for each component, in which the default is equal proportion. The function also returns model selection criterions that are Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC). AIC is computed for each *k* between 1 and 10. Then, the GMM with the least AIC value on the training dataset is selected.

After obtaining parameters through *gmdistribution.fit(X,k)* function, using *gmdistribution(mu,sigma,p)* function of Matlab, a Gaussian mixture distribution is constructed. *mu* and *sigma* values are set using the best GMM obtained from *gmdistribution.fit(X,k)* according to AIC value. After Gaussian mixture distribution object is created, the pdf of the training data comprising 'normal' part is calculated using the created distribution. The pdf values are calculated using *pdf(name,X)* function where name is the output object generated by the *gmdistribution* function and densities are evaluated at the values in *X*.

Algorithm 1 The pseudo code of fitting the training dataset into Gaussian model and learning the normality of the train data:

Inputs: *X*, data points, *k*, number of components
Outputs: *M*, the learned model

```
M_k ← model with k components
AIC_k ← value of kth component
n_k ← number of parameters of the model with k component
L_k ← maximized likelihood of the model with k component
mu ← means of components in the learned model
sigma ← covariance matrices of components in learned model
for k ← 1, 2… 10 do
    M_k ← EM(X,k)
    AIC_k ← 2n_k − 2ln(L_k)
end for
minAIC_index ← index of minimum value of AIC_k
mu ← mean of minAIC_index
sigma ← covariance matrix of minAIC_index
M ← N(mu, sigma)
```

Finally, we aim to obtain a threshold pdf value which will be able to differentiate between 'normal' and 'abnormal' data points. When we find the probability that a given data point is less than this threshold value, abnormality alarm is given. In order to decide the threshold, *perfcurve* function of Matlab is used [89]. This function is used for computing Receiver Operating Characteristic (ROC) curve. For determining optimum threshold value for abnormal behavior detection, the training data's pdf values for each observation including both normal and abnormal data and actual label of the data are necessary. The inputs of the function are *labels, scores* and *posclass*. *labels* is the actual label of the data. The observations in the dataset are labeled as 1 or 0. The ones that are labeled with 1 are the abnormal frames and the ones that are labeled with 0 are normal frames. *scores* can be any score returned from a classifier or a fitting function and it is not necessary that the score should be scaled between 0 and 1. Hence, in this case, *scores* are the pdf values of the observation in the training dataset. *posclass* is the positive class. In this dataset, it is 0, since positive class should be the class whose pdf values are higher than the threshold value. After

that, the function returns three vectors which are X vector that holds the false positive rates for different cut off points, Y vector that holds the true positive rates for different cut off points and T vector that holds the threshold values. X, Y and T have the same size. True positive and false positive rates are calculated using threshold values in T. T is a vector that holds the pdf values between [min(*scores*), max(*scores*)]. According to [89], score threshold values are used to label ROC graph. If the pdf value of an observation of training dataset is less than or equal to threshold and is labeled as a positive class, it is accepted as false positive (FP). If the pdf value of an observation of training data is more than threshold and is labeled as a positive class, it is accepted as true positive (TP). If the pdf value of an observation of training dataset is less than or equal to the threshold and is labeled as a negative class, it is accepted as true negative (TN). If the pdf value of an observation of training dataset is more than threshold and is labeled as a negative class, it is accepted as false negative (FN). This process is applied for all threshold values in T. Then the true positive and false positive rates are calculated using above values. At the end, a matrix of true positive rates (Y) and false positive rates (X) are obtained. Using these values, a ROC curve is drawn and the optimum TP rates and FP rates are detected. The detail algorithm of this calculation can be found in [90]. Optimality can be defined as a value with a high TP rate and a low FP rate. The threshold that is used in the tests is the threshold value in T that gives the optimum TP and FP rates.

In addition, a frame count threshold *FCT* is determined for *labeling a given frame as abnormal*. If the number of the observations (frames) whose pdf values are less than the threshold pdf value, is more than this frame count threshold, than it means that abnormality is detected. This threshold is determined as 10 frames. The number is determined in an ad hoc manner by observing the training dataset. Due to the noises in the frames, it is possible to mark a frame as abnormal even though it is not true. To prevent it, *FCT* is necessary. Another frame count threshold *FCT_normal* is set for *labeling a given frame as normal*. After labeling a frame as abnormal, if the pdf values of new observations are higher than the threshold pdf value, the number of consecutive observations whose pdf value is higher than the threshold should be greater than *FCT_normal*.

Algorithm 2 Threshold decision:

Inputs: *X*, data points, *M*, learned model, *N*, negative labeled data with actual label, *P*, positive labeled data with actual label
Output: Threshold

X←train data
M←learned model
mu_k←mean of *k*th component in *M*
$sigma_k$←covariance matrix of *k*th component in *M*
k←number of components
w_k←weight of *k*th component
pdf_x←pdf value of observations (frames) in *X*
for each observation $x \in X$ **do**
    $pdf_x \leftarrow P(x|M) \leftarrow \sum_{k=1}^{K} w_k N(x; mu_k, sigma_k)$
end for
T_i←vector that holds pdf threshold values that ranges between [min(pdf_x), max(pdf_x)] to use to label ROC curve
FP←false positive rates
TP←true positive rates
N←actual negative labeled data in *X*
P←actual positive labeled data in *X*
for *Ti*= min(pdf_x) to max(pdf_x) by increment **do**
    FP←0
    TP←0
    for $x \in X$ **do**
        if $pdf_x \geq T_i$ **then**
            if *x* is a positive labeled data **then**
                TP←*TP*+1
            else
                FP←*FP*+1
            end if
        end if
    end for
    //TPR, FPR data to create ROC curve
    TPR_i←*TP*/*P*
    FPR_i←*FP*/*N*
end for
TPR_{opt}← the optimal TPR_i value from ROC curve
FPR_{opt}← the optimal FPR_i value from ROC curve
pdf_thresh← the threshold in *T* that corresponds to index of TPR_{opt},
$FPRT_{opt}$

Algorithm 3 Abnormality detection:

Inputs: X_{test}, test data points, M, learned model, *pdf_thresh*, pdf threshold for abnormality detection
Output: Abnormality alarm

$\mu_k \leftarrow$ mean of kth component in M
$\sigma_k \leftarrow$ covariance matrix of kth component in M
$k \leftarrow$ number of components
$w_k \leftarrow$ weight
pdf $\leftarrow$ pdf value of observations (frames) in X_{test}
count_abnormal $\leftarrow$ count of consecutive observations whose pdf value is less than the threshold
count_normal $\leftarrow$ count of consecutive observations whose pdf value is more than the threshold
FCT $\leftarrow$ frame count threshold for abnormal frames
FCT_normal $\leftarrow$ frame count threshold for normal frames
count_abnormal $\leftarrow 0$
count_normal $\leftarrow 0$
state_abnormal $\leftarrow 0$
for each observation $x \in X_{test}$ do
  //pdf calculation
  pdf $\leftarrow P(x|M) \leftarrow \sum_{k=1}^{K} w_k N(x; \mu_k, \sigma_k)$
  if *pdf*<*thresh_pdf*
    count_abnormal++
    count_normal $\leftarrow 0$
  else
    count_normal++
    count_abnormal $\leftarrow 0$
  end if
  if *count_normal* >=*FCT_normal*
    state_abnormal $\leftarrow 0$
  end if

  if *count_abnormal* >= *FCT*
    state_abnormal $\leftarrow 1$
  end if
end for

This threshold is set to 10. Both of these thresholds are set to the same value determined in an ad-hoc fashion by observing the performance of the training set. These thresholds might also be chosen automatically for obtaining an optimum threshold. The FPR and TPR rates could be calculated for different threshold value

and the threshold that gives the optimum FPR and TPR values could be chosen as the optimum threshold.

CHAPTER 4

EXPERIMENTAL RESULTS AND COMPARISONS

4.1. Overview

In this part, the proposed method is evaluated using benchmark datasets and the results are demonstrated.

In this section, first dataset of University of Minnesota (UMN) [91] is introduced. Then, environments for testing and implementation are presented. Finally, abnormality detection results on UMN dataset are discussed.

4.2. Dataset

In this thesis, publicly available UMN dataset is used. From UMN dataset, we used 11 videos in total in which there are 2, 6 and 3 videos for scene 1, scene 2 and scene 3 respectively. The scenarios of the selected videos are very similar to each other: Several people walk around in an environment and then they start to run to one direction or several directions. The abnormality begins when all people start running in the scene. Abnormality appears in the overall scene not in a specific region of the scene. That is why this dataset is chosen for this thesis whose aim is to detect global abnormalities in a given scene.

In each scene, the first video in each scene is used as training data. As a result, 3 training video datasets are employed. The rest are reserved as testing datasets. These videos are named as V1.1 or V2.1 in this thesis where the number to the left of the decimal point shows the scene number and the number to the right of the decimal point implies the video number in the specified scene. For instance, V2.1

indicates the first video in scene 2 that is reserved for training. V2.2 denotes the second video in the scene 2 which is used for testing. This process is carried out in line with the literature.

Table 5: The dataset details used in the experiments

	Number of frames	**The frame # where abnormality starts**
V1.1 (Training)	612	506
V1.2 (Testing)	801	675
V2.1 (Training)	508	306
V2.2 (Testing)	664	585
V2.3 (Testing)	604	524
V2.4 (Testing)	529	442
V2.5 (Testing)	892	746
V2.6 (Testing)	575	450
V3.1 (Training)	640	548
V3.2 (Testing)	665	571
V3.3 (Testing)	768	719

In the datasets, abnormality alert is given a little bit later than the moment when the abnormality actually starts. Hence, in this work the abnornormality start frame is accepted as the actual frame when the abnormal behavior starts. For instance, in Figure 15, people starts running, but there is no “Abnormal Crowd Activity” alarm on the upper left corner on the frame. Another study that applies the same approach is [3].

Figure 15: People start running, but abnormality alarm is not given in ground truth

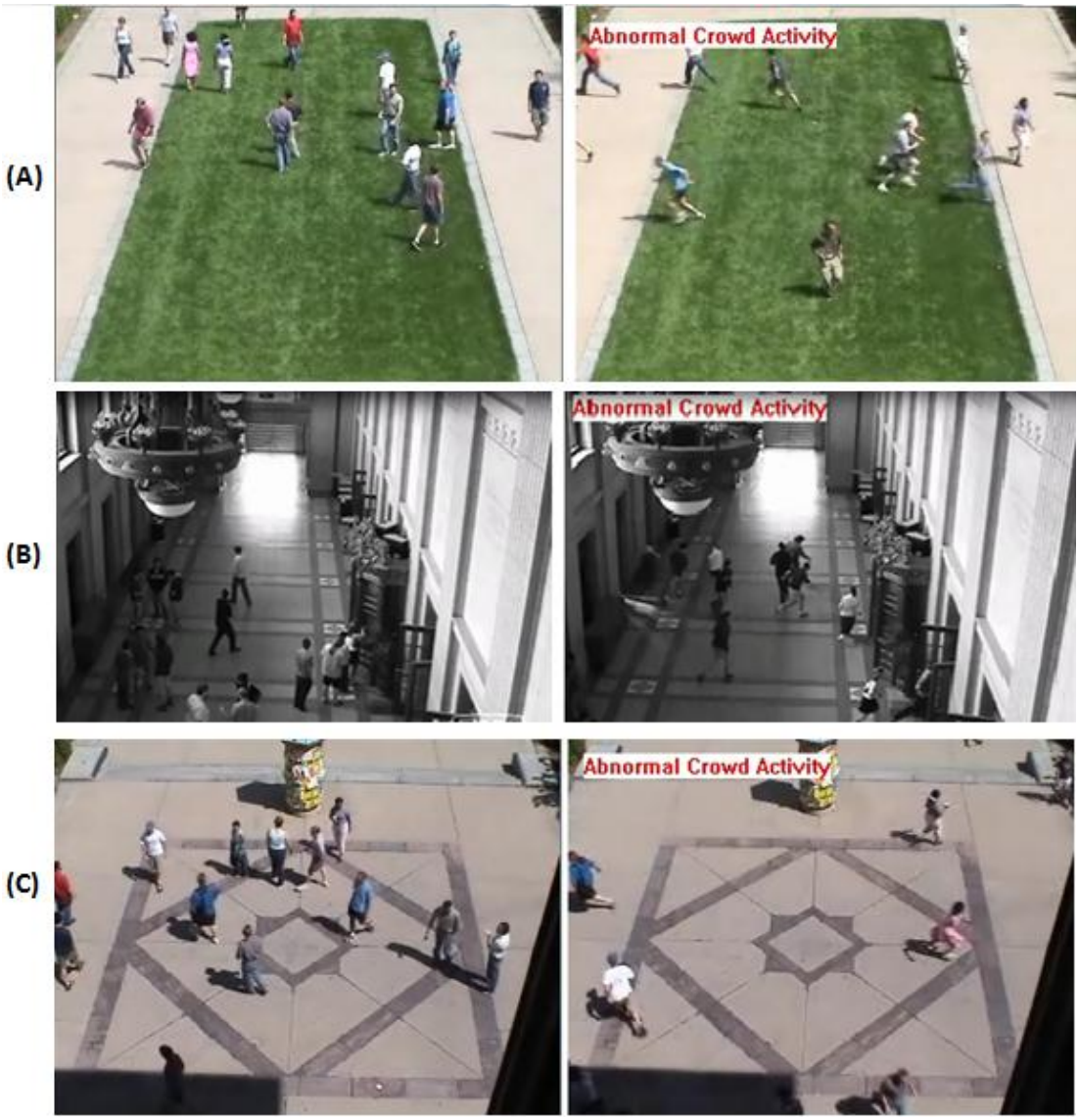

Figure 16: The training videos. (A) First scene, (B) Second scene, (C) Third scene.

4.3. Test and application environment

In the implementation of SIFT, C++ and CUDA code implemented in the Visual Studio 2008 is used [65]. There is another implementation of SIFT-GPU using OpenGL. However, CUDA implementation is faster according to the results of [65]. The proposed methods for velocity and direction count calculation introduced in chapter *3.2. Step 2:* Pre-processing are also implemented in Visual Studio 2008 C++. In the visualization such as pointing out the coordinates of SIFT feature points on the frame, OpenCV is used (Figure 17).

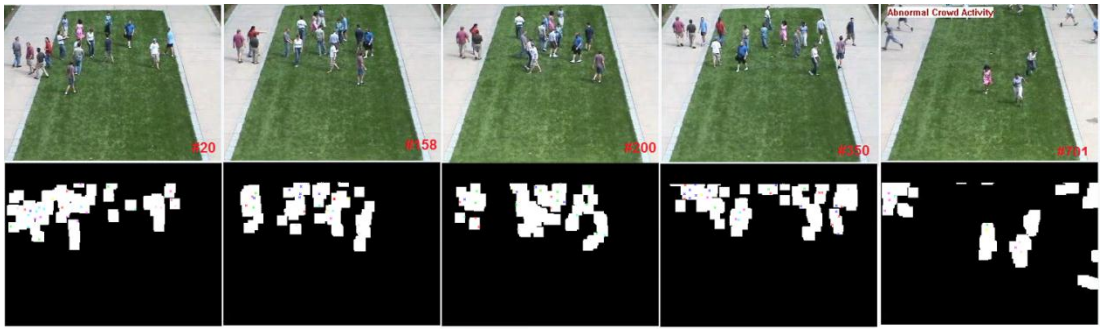

Figure 17: Scene 1 frames and moving pixel maps

The statistical tests that are introduced in 3.2. Step 2: Pre-processing are implemented using Matlab. CUDA is a parallel programming platform that is developed by NVIDIA. It benefits from the efficiency of the graphical programming unit (GPU). The GPU card that is used is NVIDIA GeForce GTX 670. CPU is Intel Core i7 2.8 GHz.

4.4. Experimental results

After the data is trained using the proposed method, the true/false positive/negative alarm numbers are computed.

The tests are repeated with five different feature vector combinations:

(i) *all-features*: all feature vectors (velocity, direction frequency and feature point count). For the videos in scene 3, V3.x, the feature vectors of velocity in direction 0, 1 and 6 and feature point count are discarded. For

V2.x and V1.X, the feature vectors of velocity in direction 5 are discarded. For V2.x, the feature vectors of direction count in direction 7 is discarded. The detailed reason of this elimination can be found in 3.2.1. Test 1: Stationarity measure section.

(ii) *velocity-features:* velocity feature vectors of the dataset. For V3.x, the feature vectors of velocity in direction 0, 1 and 6 are discarded. For V2.x and V1.X, the feature vectors of velocity in direction 5 are discarded.

(iii) *direction-count-features:* direction count feature vectors of V1.x, V2.x and V3.x. For V2.x, the feature vectors of direction count in direction 7 is discarded.

(iv) *feature-point-count-features:* feature point count feature vectors V1.x, V2.x and V3.x.

In our method, if the abnormality is detected at least for 10 consecutive frames, these frames are labeled as abnormal. In the methods that we compared with, such a threshold is not used as abnormality detection is already localized unlike our method. Hence their classification scores do not increase suddenly as our pdf results, i.e Figure 18 in [4] and our method's pdf value results, Figure 19. As seen in Figure 19, in our pdf graph, pdf value may suddenly decrease even in the normal frames, in which the pdf value should be higher than threshold. In order to prevent false positive alarm in this frames, we set a frame count threshold which is 10. If the pdf values are under the threshold for 10 consecutive frames, it means that abnormality starts. The reason of this sharp decreases of pdf in normal frames may be because of the false matched SIFT features. Although the pre-processes that are described in 3.2.4. Choosing the pre-processing technique applied, there might still exist some false feature point matching and this may cause the decrease in pdf value and as a consequence, cause the false positive alarms. However, in other method's classification score plots in Figure 18, there is no such sudden decrease. This may be because of the training and testing methodology. The authors of [4] partition the videos into clips with T frames in both testing and training and this may cause the smooth transition of pdf values between normal frames.

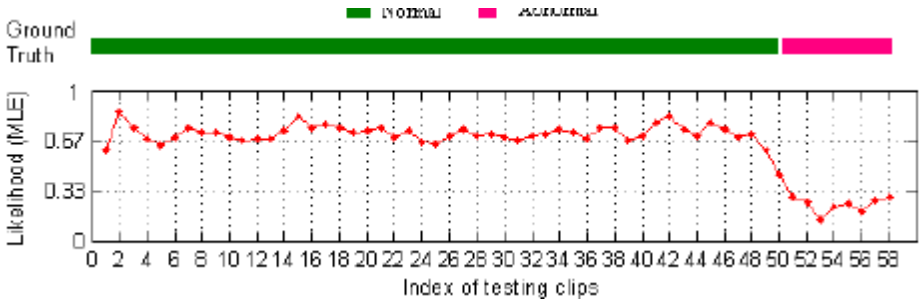

Figure 18: Likelihood results of the method [4]. The green line indicates the normal frames and the pink line indicates the abnormal frames.

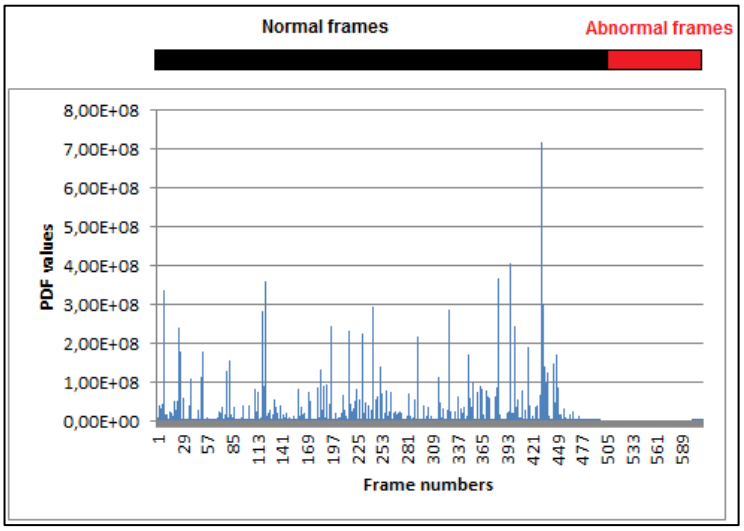

Figure 19: Example pdf of the proposed method. The y-axis is the pdf value, x-axis is the frame numbers

First, normal part of the data is trained and tests are applied according to the training parameters. For each feature vector combination, the tests are repeated. After the tests, receiver operating characteristic (ROC) curve's area under curve (AUC) is calculated. The AUC is calculated as follows: The test is repeated for five times and the TP rate and FP rate are calculated for each time. Then using these calculated rates a ROC curve is drawn and the area under this curve is calculated. Total AUC value is calculated by summing TP, FP, TN and TN values of all scenes for five run and computing TPR and FPR values using these summations. The ROC AUC results are compared with the results of [5], [4] and [3] which use UMN dataset. The resulting AUC values are shown below. In [4], trajectories of the crowd behavior are extracted using particle advection method on 2x2 sub windows in each 10-frame clip. From

these trajectories chaotic features are extracted. Test videos are also divided into 10-frame clips. In [5], Spatial-temporal Co-occurrence Gaussian Mixture Models (STCOG) is applied. They split the video frames into non-overlapping local areas of 20x20. The behaviors in these areas are modeled. In [3], Social force model is applied to characterize the crowd behavior. 10-frame clips with 5x5 patches are used in training and testing. In this thesis, each frame is modeled in training and labeled as abnormal and normal in test phase.

Table 6: AUC result of different feature vector combinations and scenes.

	Scene1	Scene2	Scene3	Total
all-features	0.99	0.66	0.82	0.73
velocity-features	0.99	0.95	1	0.97
direction-count-features	0.96	0.84	0.57	0.78
feature-point-count-features	1	0.7	0.88	0.87

In the Table 6, AUC values for different feature combinations are displayed; hence the best feature combination can be selected by looking at this table. The best results are obtained for velocity data. In the *feature-point-count-features*, first, rate of change calculation applied data is tested. However, the detection rate is around 0 most of the time for this data. Although, the abnormal and normal parts are different according to distribution difference test, according to the pdf values, the threshold value chosen by *perfcurve* function that is described in Model Fitting section is not enough to detect abnormalities. Therefore, this data does not give any information about the crowd behavior (Figure 13). Then, unprocessed *feature-point-count-features* data which is not stationary for scene 3 is used (Figure 20). The result in the table belongs to that data. The reason of non-stationarity is that moving area detection algorithm is running in a wider space since people are entering or exiting from the different place of the scene 3 (Figure 21). On the other hand, in other scenes, the movement is happening in specific places (Figure 22). The increase in the area of movement causes the SIFT detecting more matching feature points and increase in the feature point count.

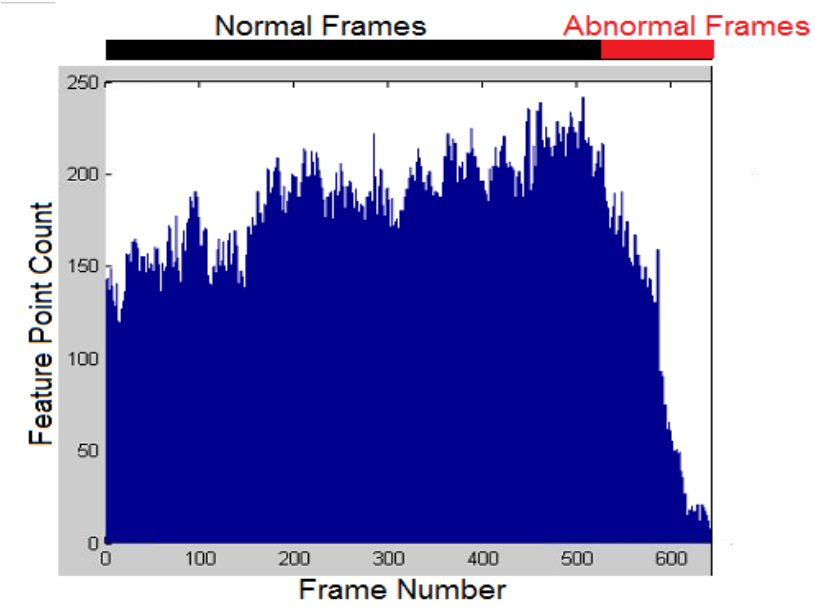

Figure 20: non-stationary V3.1 *FPC* data

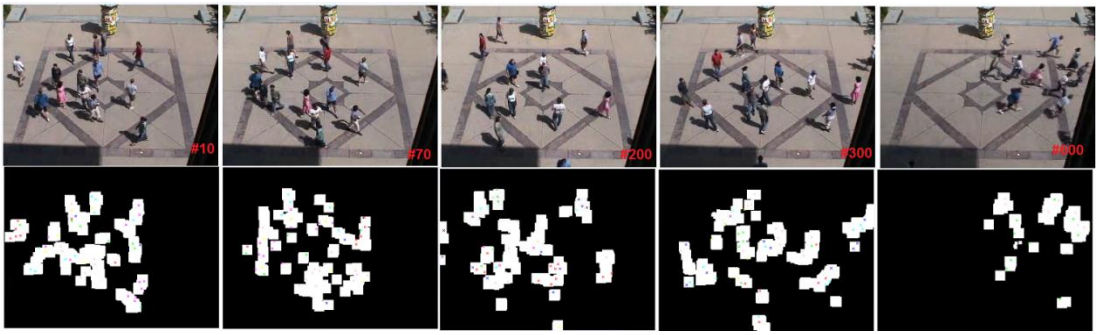

Figure 21: Video frames from scene 3, actual frames and moving pixel maps

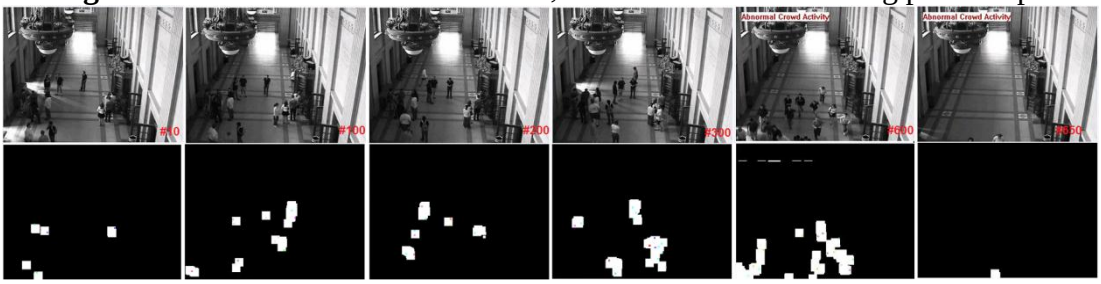

Figure 22: Normal frames: 10, 100, 200, 300, abnormal frames: 600, 650

For scene1, the AUC values are very high for all feature combinations. For scene 2, The AUC value of *all-feature* is lower than other scenes' AUC value. On the other hand, *direction-count-feature*'s AUC value is higher than *all-features'* in scene 2. However, when the pdf values of *direction-count-feature* are observed, the pdf values are not consistent. In addition, in *feature-point-count-features* combination of

scene 2, although AUC value is not very low, TPR is very low which is 0.41. For scene 3, *direction-count-features* combination has the lowest AUC value among other scene's AUC values. Therefore, it can be concluded that counting the feature points in each direction (*direction-count-features*) and counting the feature points in a frame (*feature-point-count-features*) does not give steady information about the crowd. Moreover, it is observed that, the AUC value of these feature vectors are very dependent to the moving pixel area which can be increased or decreased using the β of equation 4. As a result, velocity data is the most suitable data to detect global abnormalities. The reason of these results can be observed in data plots in FigureFigure 23 23. In the velocity plot Figure 23-a, the differences between abnormal and normal parts are observed clearly. Since the Euclidean distance between the matched feature points between consecutive frames are getting larger when people start running, the sudden increase seen in the plots in velocity plots like Figure 23-a is expected. Hence, due to this obvious distribution differences of the abnormal and normal data, the abnormalities are detected easily using only velocity data. In order to make direction count feature a more suitable data for abnormality detection, instead of quantizing the direction information into 8 bins, the bin numbers may be increased in order to obtain a more precise information about crowd direction.

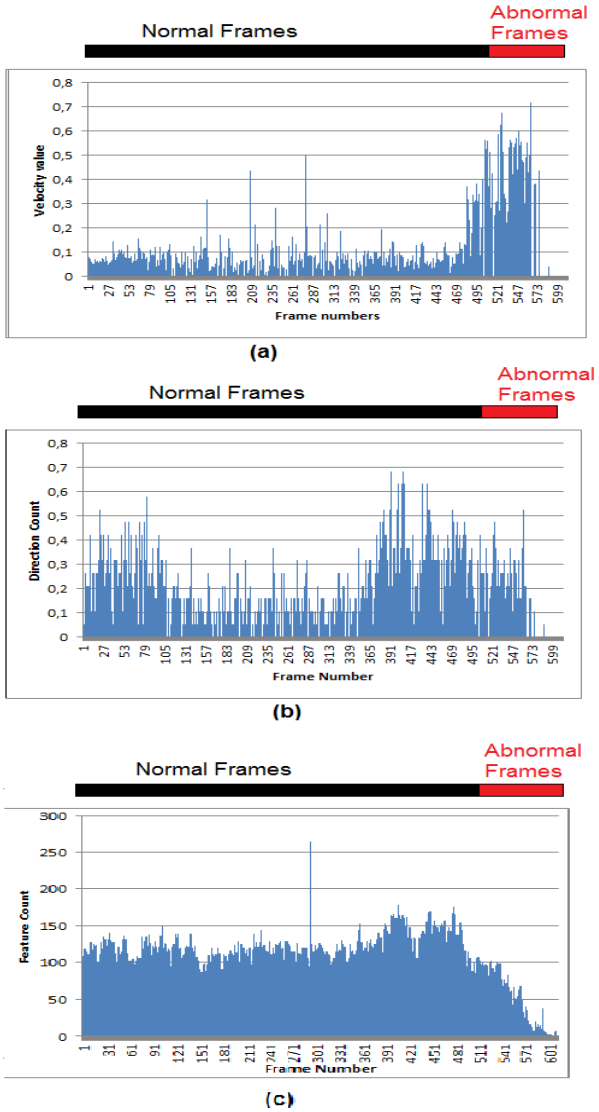

Figure 23: (a) V_0 (b) DC_0 (c) *FPC* of V1.1. y-axis shows the velocity, x-axis shows the frame number

Table 7: Precision, recall, AUC value and frame per second (fps) comparison with [5]

	Method proposed in [5]				Our method, *velocity-features*			
Scenes	Precision	Recall	AUC	Processing speed	Precision	Recall	AUC	Processing speed
Scene 1	0.99	0.95	0.94	8-9 fps	0,86	1	0.99	28.93 fps
Scene 2	0.86	0.96	0.78		0,93	0,93	0.95	
Scene 3	1	0.92	0.97		0,89	1	1	

Table 8: AUC value comparison with state-of-art methods

	Social Force[3]	Optical Flow[3]	Chaotic Invariance[4]	Our Method, *velocity features*
Training method	5x5x10 volumes for training.	10-frames clips are used for training.	2x2x10 volumes for trajectory extraction.	Global behavior in each frame is modeled.
Testing method	Each frame is labeled as normal or abnormal.	Each frame is labeled as normal or abnormal.	10-frame clips are labeled as normal or abnormal.	Each frame is labeled as normal or abnormal.
Real-time	Not stated	Not stated	Not stated	34.57 ms/frame 28.93 fps
AUC	0.96	0.84	0.99	0.97

In the Table 7 and Table 8, the comparison of AUC values of our method with different methods are demonstrated. In Table 7, the precision and recall values are also compared with the values of [5]. Since in other methods [3] and [4], precision and recall values are not given, comparison for these values is only made with [5]. In this table, AUC values can be compared for each scene, since an overall AUC value for all scenes is not given in [5]. In Table 8, overall AUC value for all scenes are compared with other methods' overall AUC result, since, in these methods, scene by scene AUC values are not given.

The best results are obtained when only velocity feature vectors are used. When these results are compared with [5], our method outperforms this method. In addition, except in the videos of scene 2, our method gives close results to the method that use chaotic invariance [4], which is a very effective method for crowd behavior analysis. This method uses trajectories as features and they preserve spatial information of these trajectories while in this thesis, we don't make use of spatial information not used completely. This finding indicates that the proposed method could have better performance, if the spatial information is used also. Moreover, according to Table 8, the proposed method outperforms Social Force and optical flow implementations' of [3].

Moreover, the computational performance of the current algorithm is measured in order to prove that the proposed method is able to run in real time. The test is applied on 320x240 UMN videos. If it is assumed that a camera process 25 frames per second, one frame should be processed in 1000 millisecond/25=40 milliseconds (ms) at most. The computation of SIFT-based features (velocity, direction frequency and feature point count) is implemented in Visual Studio C++. In this implementation, SIFT method is implemented in GPU-CPU hybrid way by the author of [65] and the computations that are added by the author of this thesis are implemented in CPU. Since Matlab contains implemented data mining functions, the pdf computation of test data is implemented in Matlab. Hence the time measurement is done in Matlab for this part. The training part is not measured since it is done once at the beginning and once the parameters are learned; there is no need to run this part again.

Table 9: Time measurements of the proposed method

SIFT-GPU [65]	VSAM+Movning pixel map extraction [72]	Pdf calculation and abnormality detection	Total
33.73 ms/frame	0.14 ms/frame	0.7 ms/frame	34.57 ms/frame (28.93 fps)

According to [5] results, they state that they archived 8-9 fps running on an Intel Duo 2.33GHZ CPU. Proposed method's fps value is 28.93. Hence, proposed method outperforms the method of [5] in terms of computation speed. However the computation environments are different. The proposed method's is measured in a much faster computer (Intel Core i7 2.8 GHz CPU) and GPU is used for parallelization of some part of the method. In addition, the proposed method is only measured for the video frames with the size of 320x240. In addition, the crowd's density in the videos is not very high. Therefore, the proposed method may not run in real-time for video frames with size higher than 320x240 and more congested crowds. Other compared methods do not give any information about their computational performance and whether real-time operation is possible or not.

CHAPTER 5

CONCLUSION AND FUTURE WORK

According to AUC results, the most promising feature is the velocity of the crowd while detecting the global abnormalities using SIFT features. However, even in that case, some false negative alarms occur as seen in scene 2. The reason of the high false alarm rate in scene 2 is due to the rough blob extraction in those frames and this occasion causes to wrong SIFT feature matches between consecutive frames. In other scene where the crowd blobs are more precise, better results are obtained. Hence, for scene 2, the blob extraction process must be done more carefully.

Moreover, direction frequency data would be more useful if it is used with local information. The generalization of this data by counting in overall frame does not give much information for detecting the crowd abnormal behavior.

In this thesis, although no spatio-temporal information is used which is a common method that is applied in the state-of-art techniques; the close results to those methods are obtained in global abnormality detection. Hence, it proves that SIFT feature tracking is a promising method for detecting abnormalities in crowd. However, this work also shows that there should be done more investigation on how to use these SIFT features.

The contributions of this thesis:

- We proposed a method which enables real-time crowd abnormality detection. According to the test result, the real-time operation is possible.
- The justifiability and repeatability of the methods can be observed in Methodology chapter. All pre-processing techniques are applied according to

the result of the statistic. As opposed to the other methods in literature, most of which require manually selected parameters, the proposed technique is adaptive and does not require any preset rules to be set.

- The result of the proposed method are comparable those of more complex ones in the literature.

For future work, local information also should be used. The frame could be divided into spatio-temporal patches and the SIFT features in those volumes might be used to model the behavior in those patches. This method might be investigated to understand how precise information will be captured about global and local crowd behavior.

In addition, feature point count data should be dealt in a different way. Instead of using the feature point count in abnormality detection directly, the abrupt change in density of feature point count could be used as additional information for the detection of abnormal situation. Moreover, for direction count data, direction information could be quantized in a higher number of bins for better precision.

Moreover, the methodology could be automatized more by get rid of the remaining thresholds like *FCT* and *FCT_normal* as described in Model Fitting section.

Also, in the matching descriptor algorithm of SIFT method, taking neighborhood constraint of feature points into consideration can give more accurate feature matching results. Therefore, the Euclidean distance threshold, which is used in velocity and direction count calculation for eliminating noisy feature points, may not be necessary. Additionally, the algorithm's computational performance may be increased by applying SIFT method in only the moving pixel areas.

The size of the training dataset may be increased. The high false positive rate in scene 2 indicates the need of more data for training. If the current datasets are not enough to create training models for global and local abnormalities, simulated data may be helpful in this case.

APPENDIX

QUALITATIVE RESULTS

Figure 24: Qualitative results of V1.2, black parts are the normal frames and red parts are the abnormal frames".

Figure 25: Qualitative results of V2.2

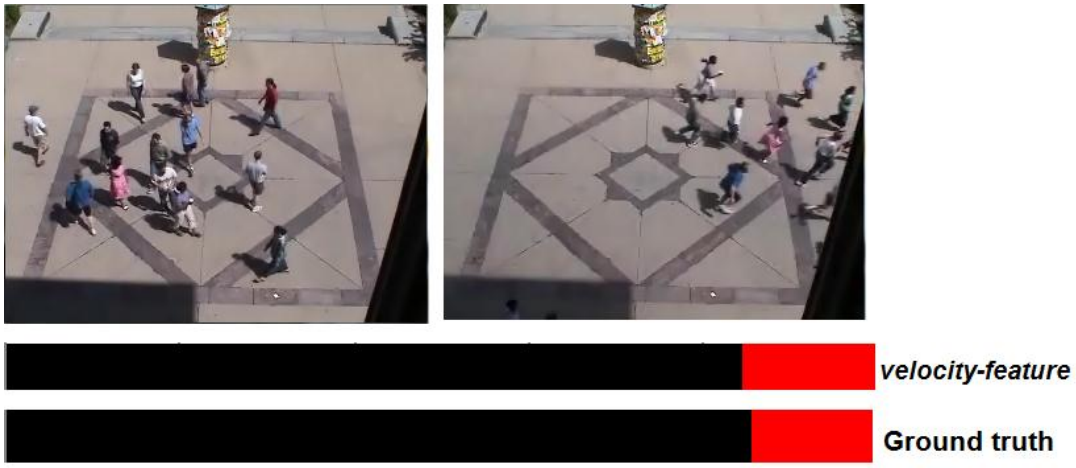

Figure 26: Qualitative results of video 3.2

REFERENCES

[1] Love Parade stampede. *Wikipedia.* [Online] [Cited: 01 September 2012.] http://en.wikipedia.org/wiki/Love_Parade_stampede

[2] Jacques Junior, J.C.S., Musse, S.R. and Jung, C.R. (2010). Crowd analysis using computer vision techniques. *Signal Processing*, 27(5), 66–77.

[3] Mehran, R., Oyama, a., & Shah, M. (2009). Abnormal crowd behavior detection using social force model. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, (2), 935–942. doi:10.1109/CVPR.2009.5206641

[4] Wu, S., Moore, B. E., & Shah, M. (2010). Chaotic invariants of Lagrangian particle trajectories for anomaly detection in crowded scenes. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2054–2060. doi:10.1109/CVPR.2010.5539882

[5] Shi, Y., Gao, Y., & Wang, R. (2010). Real-Time Abnormal Event Detection in Complicated Scenes. *2010 20th International Conference on Pattern Recognition*, (i), 3653–3656. doi:10.1109/ICPR.2010.891

[6] Garate, C., Bilinsky, P., & Bremond, F. (2009). Crowd Event Recognition Using HOG Tracker. *2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS-Winter)*. 1–6.

[7] Cisar, P. (2009). Event detection using local binary pattern based dynamic textures. *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 38–44. doi:10.1109/CVPRW.2009.5204204

[8] Ma, Y., & Cisar, P. (2008). Activity representation in crowd. *Structural, Syntactic, and Statistical Pattern Recognition*, 107–116.

[9] Ma, Y., Cisar, P., & Kembhavi, A. (2009). Motion segmentation and activity representation in crowds. *International Journal of Imaging Systems and Technology*, 19(2), 80–90. doi:10.1002/ima.20184

[10] Chan, A. B., Morrow, M., & Vasconcelos, N. (2009). Analysis of crowded scenes using holistic properties. *Performance Evaluation of Tracking and Surveillance workshop at CVPR 2009*, 101–108, Miami, Florida.

[11] Mahadevan, V., Li, W., Bhalodia, V., & Vasconcelos, N. (2010). Anomaly detection in crowded scenes. *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. doi:10.1109/CVPR.2010.5539872

[12] Reddy, V., Sanderson, C., & Lovell, B. C. (2011). Improved anomaly detection in crowded scenes via cell-based analysis of foreground speed, size and texture. *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*. doi:10.1109/CVPRW.2011.5981799

[13] Xu, J., Denman, S., Fookes, C., & Sridharan, S. (2011). Unusual Event Detection in Crowded Scenes Using Bag of LBPs in Spatio-Temporal Patches. *2011 International Conference on Digital Image Computing: Techniques and Applications*, 549–554. doi:10.1109/DICTA.2011.98

[14] Ryan, D., Denman, S., Fookes, C., & Sridharan, S. (2011). Textures of optical flow for real-time anomaly detection in crowds. *2011 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 230–235. doi:10.1109/AVSS.2011.6027327

[15] Turkay, C., Koc, E., & Balcisoy, S. (2009). An information theoretic approach to camera control for crowded scenes. *The Visual Computer*, *25*(5-7), 451–459. doi:10.1007/s00371-009-0337-1

[16] Zhong, Z., Yang, M., Wang, S., Ye, W., & Xu, Y. (2007). Energy Methods for Crowd Surveillance. *2007 International Conference on Information Acquisition*, 504–510. doi:10.1109/ICIA.2007.4295785

[17] Zhong, Z., Ding, N., Wu, X., & Xu, Y. (2008). Crowd surveillance using Markov Random Fields. *Automation and Logistics*, 2008, (September), 1822–1828.

[18] Cao, T., Wu, X., Guo, J., Yu, S., & Xu, Y. (2009). Abnormal crowd motion analysis. *2009 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 1709–1714. doi:10.1109/ROBIO.2009.5420408

[19] Luvison, B., Chateau, T., Sayd, P., Pham, Q., & Lapreste, J. (2009). An Unsupervised Learning based Approach for Unexpected Event Detection. (A. Ranchordas & H. Araujo, Eds.)*VISAPP 2009 - Proceedings of the Fourth International Conference on Computer Vision Theory and Applications*, 509–513. Lisboa, Portugal: INSTICC Press.

[20] Cheriyadat, A., & Radke, R. (2008). Detecting dominant motions in dense crowds. *Selected Topics in Signal*, 2(4), 568–581.

[21] Hu, M., Ali, S., & Shah, M. (2008). Learning motion patterns in crowded scenes using motion flow field. *2008 19th International Conference on Pattern Recognition*, 1–5. doi:10.1109/ICPR.2008.4761183

[22] Andrade, E. (2006). Modelling crowd scenes for event detection. *Pattern Recognition*, 18–21.

[23] Chen, D.-Y., & Huang, P.-C. (2011). Motion-based unusual event detection in human crowds. *Journal of Visual Communication and Image Representation*, *22*(2), 178–186. doi:10.1016/j.jvcir.2010.12.004

[24] Feng, J., Zhang, C., & Hao, P. (2010). Online Learning with Self-Organizing Maps for Anomaly Detection in Crowd Scenes. *2010 20th International Conference on Pattern Recognition*, *i*, 3599–3602. doi:10.1109/ICPR.2010.878

[25] Thida, M., Eng, H., Dorothy, M., & Remagnino, P. (2011). Learning video manifold for segmenting crowd events and abnormality detection. *Computer Vision–ACCV*, 1–12.

[26] Krausz, B., & Bauckhage, C. (2011). Automatic detection of dangerous motion behavior in human crowds. *Advanced Video and Signal-Based*, 224–229.

[27] Li, N., & Zhang, Z. (2011). Abnormal Crowd Behavior Detection Using Topological Methods. *2011 12th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, 13–18. doi:10.1109/SNPD.2011.21

[28] Saxena, S., Brémond, F., Thonnat, M., & Ma, R. (2008). Crowd Behavior Recognition For Video Surveillance. *ACIVS '08 Proceedings of the 10th International Conference on Advanced Concepts for Intelligent Vision Systems*, 970–981.

[29] Dee, H. M., & Caplier, A. (2010). Crowd behaviour analysis using histograms of motion direction. *Image Processing (ICIP), 2010 17th IEEE International Conference on*. doi:10.1109/ICIP.2010.5653573

[30] Wang, S., & Miao, Z. (2010a). Anomaly detection in crowd scene. *IEEE 10th Internatıonal Conference On Sıgnal Processıng Proceedıngs*, 1220–1223. doi:10.1109/ICOSP.2010.5655356

[31] Sharif, M., Uyaver, S., & Djeraba, C. (2010). Crowd Behavior Surveillance Using Bhattacharyya Distance Metric. In R. Barneva, V. Brimkov, H. Hauptman, R. Natal Jorge, & J. Tavares (Eds.) Computatıonal Modelıng Of Objects Represented In Images (Vol. 6026, pp. 311–323). Springer Berlin / Heidelberg. doi:10.1007/978-3-642-12712-0_28

[32] Chen, D.-Y., & Huang, P.-C. (2010). Dynamic human crowd modeling and its application to anomalous events detcetion. *Multimedia and Expo (ICME), 2010 IEEE International Conference on*. doi:10.1109/ICME.2010.5582938

[33] Xu, L.-Q., & Anjulan, A. (2008). Crowd behaviours analysis in dynamic visual scenes of complex environment. *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on.* doi:10.1109/ICIP.2008.4711678

[34] Wang, X., Ma, X., & Grimson, W. E. L. (2009). Unsupervised Activity Perception in Crowded and Complicated Scenes Using Hierarchical Bayesian Models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*. doi:10.1109/TPAMI.2008.87

[35] Raghavendra, R., & Bue, A. D. (2011). Optimizing interaction force for global anomaly detection in crowded scenes. *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, 136–143.

[36] Wang, S., & Miao, Z. (2010). Anomaly detection in crowd scene using historical information. *Intelligent Signal Processing and Communication Systems (ISPACS), 2010 International Symposium on*. doi:10.1109/ISPACS.2010.5704770

[37] Liao, H., Xiang, J., Sun, W., Feng, Q., & Dai, J. (2011). An Abnormal Event Recognition in Crowd Scene. *2011 Sixth International Conference on Image and Graphics*, 731–736. doi:10.1109/ICIG.2011.66

[38] Zaharescu, A., & Wildes, R. (2010). Anomalous Behaviour Detection Using Spatiotemporal Oriented Energies , Subset Inclusion Histogram Comparison and Event-Driven Processing, *Computer Vision – ECCV 201*, 563–576.

[39] Jiang, F., Yuan, J., Tsaftaris, S. a., & Katsaggelos, A. K. (2011). Anomalous video event detection using spatiotemporal context. *Computer Vision and Image Understanding*, *115*(3), 323–333. doi:10.1016/j.cviu.2010.10.008

[40] Pathan, S. S., Al-Hamadi, A., & Michaelis, B. (2010). Crowd behavior detection by statistical modeling of motion patterns. *2010 International Conference of Soft Computing and Pattern Recognition*, 81–86. doi:10.1109/SOCPAR.2010.5686403

[41] Ihaddadene, N., & Djeraba, C. (2008). Real-time crowd motion analysis. *2008 19th International Conference on Pattern Recognition*, 1–4. doi:10.1109/ICPR.2008.4761041

[42] Chiu, W.-Y., & Tsai, D.-M. (2010). A Macro-Observation Scheme for Abnormal Event Detection in Daily-Life Video Sequences. *EURASIP Journal on Advances in Signal Processing*, *2010*(1), 525026. doi:10.1155/2010/525026

[43] Ali, S., & Shah, M. (2007). A Lagrangian Particle Dynamics Approach for Crowd Flow Segmentation and Stability Analysis. *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 1–6. doi:10.1109/CVPR.2007.382977

[44] Kratz, L., & Nishino, K. (2009). Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 1446–1453. doi:10.1109/CVPR.2009.5206771

[45] Wang, B., Ye, M., Li, X., Zhao, F., & Ding, J. (2011). Abnormal crowd behavior detection using high-frequency and spatio-temporal features. *Machine Vision and Applications*, *23*(3), 501–511. doi:10.1007/s00138-011-0341-0

[46] Luvison, B., Chateau, T., Lapreste, J.-T., Sayd P. and Pham Q. C. (2011). Automatic Detection of Unexpected Events in Dense Areas for Videosurveillance Applications, Video Surveillance, Weiyao Lin (Ed.), ISBN: 978-953-307-436-8, *InTech*. [Online]. Available from: http://www.intechopen.com/books/video-surveillance/automatic-detection-of-unexpected-events-in-dense-areas-for-videosurveillance-applications

[47] Shechtman, E., & Irani, M. (2005). Space-time behavior based correlation. Computer Vision and Pattern Recognition. *CVPR 2005. IEEE Computer Society Conference on*. doi:10.1109/CVPR.2005.328

[48] Sun, X., Yao, H., Ji, R., Liu, X., & Xu, P. (2011). Unsupervised fast anomaly detection in crowds. *MM '11 Proceedings of the 19th ACM international conference on Multimedia*, (92), 1469–1472.

[49] Benezeth, Y., Jodoin, P.-M., & Saligrama, V. (2011). Abnormality detection using low-level co-occurring events. *Pattern Recognition Letters*, *32*(3), 423–431. doi:10.1016/j.patrec.2010.10.008

[50] Seidenari, L., Bertini, M., & Del Bimbo, A. (2010). Dense spatio-temporal features for non-parametric anomaly detection and localization. *Proceedings of the first ACM international workshop on Analysis and retrieval of tracked events and motion in imagery streams - ARTEMIS '10*, 27. doi:10.1145/1877868.1877877

[51] Allain, P., Courty, N., & Corpetti, T. (2010). Crowd flow characterization with optimal control theory. *Computer Vision–ACCV 2009*.

[52] Rodriguez, M., Sivic, J., Laptev, I., & Audibert, J.-Y. (2011). Data-driven crowd analysis in videos. *2011 International Conference on Computer Vision*, 1235–1242. doi:10.1109/ICCV.2011.6126374

[53] Pathan, S., Al-Hamadi, A., & Michaelis, B. (2010). Incorporating social entropy for crowd behavior detection using SVM. *Advances in Visual Computing*, 153–162.

[54] Ke, Y., Sukthankar, R., & Hebert, M. (2007). Event Detection in Crowded Videos. *2007 IEEE 11th International Conference on Computer Vision*, 1–8. doi:10.1109/ICCV.2007.4409011

[55] Wu, X., Ou, Y., Qian, H., & Xu, Y. (2005). A detection system for human abnormal behavior. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1204–1208. doi:10.1109/IROS.2005.1545205

[56] Cohen, C. J., Morelli, F., & Scott, K. A. (2008). A Surveillance System for the Recognition of Intent within Individuals and Crowds. *Technologies for Homeland Security, 2008 IEEE Conference on*. doi:10.1109/THS.2008.4534514

[57] Calderara, S., Heinemann, U., Prati, A., Cucchiara, R., & Tishby, N. (2011). Detecting anomalies in people's trajectories using spectral graph analysis. *Computer Vision and Image Understanding*, *115*(8), 1099–1111. doi:10.1016/j.cviu.2011.03.003

[58] Hoogs, A., & Bush, S. (2008). Detecting semantic group activities using relational clustering. *Motion and video Computing, 2008. WMVC 2008. IEEE Workshop on*, 1–8.

[59] Jiang, F., Wu, Y., & Katsaggelos, A. K. (2009). Detecting contextual anomalies of crowd motion in surveillance video. *Image Processing (ICIP), 2009 16th IEEE International Conference on*. doi:10.1109/ICIP.2009.5414535

[60] Ge, W., & Chang, M.-C. (2012). Group context learning for event recognition. *2012 IEEE Workshop on the Applications of Computer Vision (WACV)*, 249–255. doi:10.1109/WACV.2012.6163009

[61] Hommes, S., State, R., Zinnen, A., & Engel, T. (2011). Detection of abnormal behaviour in a surveillance environment using control charts. *2011 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 113–118. doi:10.1109/AVSS.2011.6027304

[62] Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, *60*(2), 91–110. doi:10.1023/B:VISI.0000029664.99615.94

[63] Bastanlar, Y., Temizel, a., & Yardımcı, Y. (2010). Improved SIFT matching for image pairs with scale difference. *Electronics Letters*, *46*(5), 346. doi:10.1049/el.2010.2548

[64] Teke, M., Vural, M. F., Temizel, A., & Yardımcı, Y. (2011). High-resolution multispectral satellite image matching using scale invariant feature transform and speeded up robust features. *Journal of Applied Remote Sensing*, *5*(1), 053553. doi:10.1117/1.3643693

[65] Wu, C. (2007). Sift-GPU: A GPU Implementation of Scale Invariant Feature Transform (SIFT). [Online]. Available from: http://cs.unc.edu/~ccwu/siftgpu/

[66] Sinha, S., Frahm, J., Pollefeys, M., & Genc, Y. (2006). GPU-based video feature tracking and matching. EDGE, *Workshop on Edge Computing Using New Commodity Architectures.*

[67] Vedaldi, A. (2006). SIFT++. [Online]. Available from: http://www.vlfeat.org/~vedaldi/code/siftpp.html

[68] Wu, X., Liang, G., Lee, K. K., & Xu, Y. (2006). Crowd Density Estimation Using Texture Analysis and Learning. *Robotics and Biomimetics, 2006. ROBIO '06. IEEE International Conference on*. doi:10.1109/ROBIO.2006.340379

[69] Arandjelovic, O. (2008). Crowd Detection From Still Images. *Electronic Proceedings of the 19th British Machine Vision Conference*.

[70] Siva, P., & Xiang, T. (2010). Action Detection in Crowd. *Procedings of the British Machine Vision Conference 2010*, 9.1–9.11. doi:10.5244/C.24.9

[71] Shi, P. And Tsai, C.-L. (2007). Extending the Akaike Information Criterion to Mixture Regression Models. *Journal of the American Statistical Association*, 102 (477), 244–254.

[72] Saglam, A., & Temizel, A. (2009). Real-Time Adaptive Camera Tamper Detection for Video Surveillance. *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*.

[73] Collins R.T., Lipton A.J., Kanade T., Fujiyoshi H., Duggins D., Tsin Y., Tolliver D., Enomoto N., Hasegawa O., Burt P., Wixon L. (1998). A system for video surveillance and monitoring: VSAM final report. s.l. : Carnegie Mellon University, Technical Report CMURI-TR-00-12.

[74] Beckert, W.. Non-Stationary Series. [Online]. Available from: http://www.ems.bbk.ac.uk/for_students/bsc_FinEcon/fin_economEMEC007U/adf.pdf

[75] Kwiatkowski, D., & Phillips, P. C. B. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root How sure are we that economic time series have a unit root ?, *Cowles Foundation Discussion Papers 979, Cowles Foundation for Research in Economics, Yale University*.

[76] kpsstest. *Matlab.* [Online] [Cited: 01 September 2012.] http://www.mathworks.com/help/toolbox/econ/kpsstest.html

[77] Statistical significance. *Wikipedia.* [Online] [Cited: 01 September 2012.] http://en.wikipedia.org/wiki/Statistical_significance

[78] Rate of Change (ROC). *Stockharts*. [Online] [Cited: 01 September 2012.] http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:rate_of_change

[79] Massey, F. J. (1951). The Kolmogorov-Smirnov Test for Goodness of Fit. *Journal of the American Statistical Association*, 46(253), 68–78.

[80] kstest. *Matlab.* [Online] [Cited: 01 September 2012.]

http://www.mathworks.com/help/toolbox/stats/kstest.html

[81] 5 How to analyze your data. *AO Publishing*. [Online] [Cited: 01 September 2012.] http://www.aopublishing.org/handb_stat/Sample_HBStatistic.pdf

[82] Mann H. B. and Whitney D . R . (1947). On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other, *The Annals of Mathematical Statistics*, 18(1), 50–60.

[83] Alpar, H. (2001). *Spor Bilimlerinde Uygulamalı Istatistik.* İstanbul: Nobel Yayın Dağıtım.

[84] ranksum. *Matlab.* [Online] [Cited: 01 September 2012.]
http://www.mathworks.com/help/toolbox/stats/ranksum.html

[85] Resch, B. Mixtures of Gaussians Mixtures of Gaussians Formulas and Definitions. [Online]. Available from:
http://www.spsc.tugraz.at/system/files/mixtgaussian.pdf

[86] Do, C. B., & Batzoglou, S. (2008). What is the expectation maximization algorithm? *Nature biotechnology*, *26*(8), 897–9. doi:10.1038/nbt1406

[87] gmdistribution.fit. *Matlab*. [Online] [Cited: 01 September 2012.]
http://www.mathworks.com/help/toolbox/stats/gmdistribution.fit.htm

[88] Bishop C.M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc..

[89] perfcurve. *Matlab*. [Online] [Cited: 01 September 2012.]
http://www.mathworks.com/help/toolbox/stats/perfcurve.html

[90] Fawcett, T. (2003). ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. [Online]. Available from:
http://www.hpl.hp.com/techreports/2003/HPL-2003-4.pdf

[91] Unusual crowd activity dataset made available by the University of Minnesota. [Online] [Cited: 01 September 2012.] http://mha.cs.umn.edu/movies/crowdactivity-all.avi.

ODTÜ
ENFORMATİK ENSTİTÜSÜ

YAZARIN

Soyadı : Güler.....

Adı : Püren.....

Bölümü : Bilişim Sistemleri.....

TEZİN ADI (İngilizce) : AUTOMATED CROWD BEHAVIOR ANALYSIS FOR VIDEO SURVEILLANCE APPLICATIONS
.....

.....

.....

.....

.....

TEZİN TÜRÜ : Yüksek Lisans ...X.... Doktora .....

1) Tezimden fotokopi yapılmasına izin vermiyorum ☐

2) Tezimden dipnot gösterilmek şartıyla bir bölümünün fotokopisi alınabilir ☐

3) Kaynak gösterilmak şartıyla tezimin tamamının fotokopisi alınabilir ☐

Yazarın imzası ..... Tarih ....