Motion based Event Analysis

A Thesis
Submitted for the Degree of
Master of Science (Engineering)
in the Faculty of Engineering

by
Sovan Biswas

Supercomputer Education and Research Center
Indian Institute of Science
Bangalore – 560 012 (INDIA)
DECEMBER 2014
DEDICATED TO

Baba and Ma
Acknowledgements

This thesis has been a great journey for me which not only has helped me to evolve technically but also as a human being. It will be unfair to name and thank a few who has helped me along the way.

Foremost, I would like to express my deepest gratitude to my advisor Dr. R. Venkatesh Babu, for his immense patience, encouragement and guidance. I feel extremely fortunate and grateful to have him as my advisor. It was his passion and dedication to research that inspired me every moment. I have learned so much from him about how to be a good researcher. Thank you Sir for believing in me, and for the support and guidance that have made this thesis possible.

My labmates deserve much thanks for making my student life an enjoyable and memorable one. Thanks Avinash Ramakanth for being there from the beginning, for being a true friend and setting such a good example to follow. Thanks MK Reddy, Srinivas, Nikita, Manu, Praveen, Paras and others for all the valuable technical and non-technical inputs. Without your ideas and endless criticisms, this is thesis would never have been possible. Thanks Nivedita and all my batchmates for making my graduate studies at IISc memorable journey. A special thanks to Ravi Ranjan Singh and other friends in helping me to cope up with work pressure of IISc.

Finally, this thesis would not have been possible without the unconditional love and support of my family: my father Samar Kumar Biswas, my mother Reba Biswas, and my brother Souvick Biswas. I dedicate my master’s thesis to them.
Abstract

Motion is an important cue in videos that captures the dynamics of moving objects. It helps in effective analysis of various event related tasks such as human action recognition, anomaly detection, tracking, crowd behavior analysis, traffic monitoring, etc. Generally, accurate motion information is computed using various optical flow estimation techniques. On the other hand, coarse motion information is readily available in the form of motion vectors in compressed videos. Utilizing these encoded motion vectors reduces the computational burden involved in flow estimation and enables rapid analysis of video streams. In this work, the focus is on analyzing motion patterns, retrieved from either motion vectors or optical flow, in order to do various event analysis tasks such as video classification, anomaly detection and crowd flow segmentation.

In the first section, we utilize the motion vectors from H.264 compressed videos, a compression standard widely used due to its high compression ratio, to address the following problems. i) Video classification: This work proposes an approach to classify videos based on human action by capturing spatio-temporal motion pattern of the actions using Histogram of Oriented Motion Vector (HOMV) ii) Crowd flow segmentation: In this work, we have addressed the problem of flow segmentation of the dominant motion patterns of the crowds. The proposed approach combines multi-scale super-pixel segmentation of the motion vectors to obtain the final flow segmentation. iii) Anomaly detection: This problem is addressed by local modeling of usual behavior by capturing features such as magnitude and orientation of each moving object. In all the above approaches, the focus was to reduce computations while retaining comparable accuracy to pixel domain processing.

In second section, we propose two approaches for anomaly detection using optical flow. The first approach uses spatio-temporal low level motion features and detects anomalies based on the reconstruction error of the sparse representation of the candidate feature over a dictionary of usual behavior features. The main contribution is in enhancing each local dictionary by applying an appropriate transformation on dictionaries of the neighboring regions. The other algorithm aims to improve the accuracy of anomaly localization through short local trajectories
Abstract

of super pixels belonging to moving objects. These trajectories capture both spatial as well as temporal information effectively. In contrast to compressed domain analysis, these pixel level approaches focus on improving the accuracy of detection with reasonable detection speed.
Publications from the Thesis


Contents

Acknowledgements i

Abstract ii

Publications from the Thesis iv

Contents v

List of Figures x

List of Tables xiii

1 Introduction 1

1.1 What is an Event? ................................................. 1
1.2 Motion: A key component? ..................................... 3
1.3 Compressed Domain Video Analysis ................................. 4
  1.3.1 Video compression standards ................................ 4
  1.3.1.1 H.261 .................................................. 5
  1.3.1.2 MPEG-1 ............................................... 5
  1.3.1.3 MPEG-2 ............................................... 5
  1.3.1.4 H.263 .................................................. 5
  1.3.1.5 H.264/AVC or MPEG-4 (Part 10) .......................... 5
  1.3.2 H.264 Standard Overview ................................... 5
  1.3.3 Compressed Domain Video Analysis and Applications ......... 7
  1.3.3.1 Video Indexing and Retrieval ............................ 7
  1.3.3.2 Moving Object Tracking ................................. 7
  1.3.3.3 Moving Object segmentation .............................. 8
1.4 Pixel-Level Video Analysis ......................................... 8
CONTENTS

1.4.1 Features in pixel-level analysis ........................................ 8
1.4.2 Extracting Flow Information ........................................... 9
1.4.3 Pixel Level Analysis and Applications ............................... 10
  1.4.3.1 Object Tracking .................................................. 10
  1.4.3.2 Video Synopsis .................................................. 10
  1.4.3.3 Crowd Analysis .................................................. 10
1.5 Organization of Thesis .................................................... 11

2 Video Classification .......................................................... 12
  2.1 Introduction ............................................................. 12
  2.2 Related Work ............................................................ 13
  2.3 Proposed Algorithm ...................................................... 14
    2.3.1 Preprocessing ...................................................... 14
    2.3.2 Feature extraction ............................................... 15
      2.3.2.1 Space-time cubes generation ................................ 15
      2.3.2.2 Region of interest features ................................. 16
      2.3.2.3 Histogram of Oriented Motion Vector (HOMV) ............... 17
      2.3.2.4 Orientation Normalization ................................ 17
    2.3.3 Video feature ..................................................... 18
  2.4 Experiments and Discussion ............................................ 19
    2.4.1 Datasets ............................................................ 19
  2.5 Conclusion .............................................................. 21

3 Crowd Flow Segmentation: A super-pixel based approach ............... 22
  3.1 Introduction ............................................................. 22
  3.2 Related Work ............................................................ 23
  3.3 Problem formulation .................................................... 24
  3.4 The proposed approach .................................................. 25
    3.4.1 Preprocessing of Motion Vectors ................................ 25
    3.4.2 Clustering of Motion Vectors by Super-pixels .................... 26
    3.4.3 Detection of true edges using Confidence scores ............... 27
    3.4.4 Final crowd flow segmentation .................................. 28
  3.5 Results and discussions ................................................ 29
    3.5.1 Experiments ........................................................ 29
    3.5.2 Analysis ............................................................ 30
4 Anomaly Detection via Modeling local Motion
  4.1 Introduction .................................................. 33
  4.2 Related Work .................................................. 34
  4.3 Proposed Algorithm ......................................... 35
    4.3.1 Modeling Usual Behavior .............................. 36
      4.3.1.1 Pre-processing .................................. 36
      4.3.1.2 Usual Behavior (UB) Modeling ................ 37
    4.3.2 Detecting Abnormal Pattern .......................... 40
  4.4 Experiments and Results .................................. 42
    4.4.1 Evaluation Procedure ................................ 42
    4.4.2 Quantitative Performance Analysis .................. 44
  4.5 Conclusion .................................................. 47

5 Anomaly Detection via Local Dictionary Enhancement
  5.1 Introduction ................................................ 49
  5.2 Related Work ................................................ 50
  5.3 Proposed Framework ....................................... 50
  5.4 Detailed Algorithm ....................................... 52
    5.4.1 Descriptor extraction ................................. 53
      5.4.1.1 Foreground pixel occupancy .................... 53
      5.4.1.2 Histogram of optical flow (HOF) ............... 53
      5.4.1.3 Motion magnitude ............................... 53
    5.4.2 Learning data transformation across neighboring blocks ........ 53
    5.4.3 Anomaly detection ................................. 55
  5.5 Experiments ................................................. 56
    5.5.1 Implementation details .............................. 56
    5.5.2 Quantitative evaluation ........................... 57
    5.5.3 Computation complexity ............................ 58
  5.6 Conclusion ................................................ 59

6 Anomaly Detection via Short Local Trajectory
  6.1 Introduction ............................................... 60
  6.2 Problem Formulation ..................................... 61
  6.3 Proposed Approach ....................................... 62
6.3.1 Descriptor - Short Local Trajectories ........................................ 62
6.3.2 Usual Behavior Modeling - Hidden Markov Model (HMM) ................. 64
  6.3.2.1 Hidden Markov Model .................................................. 65
  6.3.2.2 Generating symbol sequences for $T^f$ for HMM ...................... 66
6.3.3 Anomaly Detection ................................................................... 66
  6.3.3.1 Stage 1: Hidden Markov Model Based Measure ....................... 66
  6.3.3.2 Stage 2: Spatial Consistency Measure ................................. 67
  6.3.3.3 Joint Measure and Final Detection: .................................... 68
  6.3.3.4 Spatio-Temporal Filtering: ............................................. 68
6.4 Results and Analysis .................................................................. 69
  6.4.1 Implementation Details ....................................................... 69
  6.4.2 Quantitative Evaluation ....................................................... 69
    6.4.2.1 Effect of length of trajectory: ....................................... 71
  6.4.3 Experiment on Dense Crowd Scenarios .................................... 71
6.5 Conclusion .............................................................................. 73

7 Summary .................................................................................. 75

Bibliography ............................................................................... 77
List of Figures

1.1 Frames of Video from Weizmann Action Dataset [41]. Even though 1st frame is similar later frames are different due to temporal variation in the video: a) Action - Skip b) Action - Run .................................................. 3

2.1 Block diagram of the proposed approach .................................................. 14

2.2 Space-time cubes. a) In $1 \times 1 \times 1$, space-time cubes are only temporally overlapping (level 1) b) In $3 \times 1 \times 1$, space-time cubes are spatially as well as temporally overlapping as shown. (level 2) (Images are best viewed in color) .................................. 16

2.3 Organization of orientation bins. ............................................................... 17

2.4 A Part of Datasets. [2] [4] .............................................................. 19

2.5 Confusion Matrix for Datasets .......................................................... 20

3.1 Examples Scenarios of videos involving high density crowded scenes in [11] . . . 23

3.2 Overview of the proposed approach ......................................................... 25

3.3 Output at the intermediate stages of the proposed approach (a) Input video sequence (b) Color coded version (c) Confidence score $C_1$ (d) Confidence score $C_2$ (e) Final confidence score $C_f$ (f) Flow segmentation ........................................... 27

3.4 Experimental results of some of the videos in the database. First column of images shows input video sequence, second column of images show results of Ali et al.[11], whereas third column shows the output of the proposed approach . . . 30

4.1 Motion Pyramid with 4 Levels from L0 to L3 ........................................ 36

4.2 a) Original variable macroblocks present in a single frame b) Motion replication upto its $4 \times 4$ constituent macroblocks. c) Motion pyramid with multiple frame resolution. .................................................. 36
4.3 a) Sample density for the scene b) Variable kernel size at three different points $P_0$, $P_1$ and $P_2$. $P_0$ has highest number of training samples followed by $P_1$ and $P_2$. Thus, the kernel size are $P_2$ is biggest followed by $P_1$ and $P_0$. 

4.4 i) Usual behavior histogram ii) Sorted histogram values iii) Plot of cumulative sum of the sorted histogram values used to find $b^*$ based on $\gamma_L$. 

4.5 Detection from level L1 to L0. Anomaly (circle) is suspected at coarse level L1 is further processed at finer level L0. 

4.6 ROC curve for the two data-sets. 

4.7 Results on Ped1 and Ped2. Anomalies are marked as anomalies. 

4.8 Equal Error Rate (EER) vs $\gamma_{L1}$ plot for Ped1 dataset. ‘*’ denotes optimal minimum error while varying the thresholds. 

4.9 Frames of UMN data-set wrongly marked a) Abnormal frame marked as normal frame b) Normal frame marked as abnormal frame. 

4.10 Result comparison on UMN data-set: Labels of each test frame a) Original GT bar b) Modified GT bar c) Actual detection bar; Green: Normal frame, Red: Abnormal frame. 

4.11 ROC curve for UMN data-sets a) Based on Original Ground Truth b) Based on Modified Ground Truth. 

4.12 Results on UMN data-set (Yellow colored Frames are detected as abnormal Frames). 

5.1 Frame Blocks: Statistics in a block is similar and can be accumulated to form a local dictionary and subsequent sparse solving. 

5.2 Representative results for different video sequences where each row is from a single sequence. The anomalies are marked in ‘Red’ color. (Best viewed in color) (More results available at http://val.serc.iisc.ernet.in/AnomalyResultsICIP14). 

5.3 a,b) Performance of the different approaches tested for the frame-level anomaly detection on the Ped1 and Ped2 respectively. c) Frame level comparison with and without dictionary enhancement on Ped2 d) Pixel-level comparison on Ped1. 

6.1 Comparison of Short Local Trajectories (SLT) of normal and abnormal behaviors on UCSD Ped1 (Marked by blue). Normal trajectories are marked in green whereas abnormal trajectories are in red. The trajectories of both objects are distinct even when they are in same vicinity. 

6.2 Trajectory generation stages: original frame, super-pixel output using ERS (Top-Right) [62] and trajectories generated for only moving objects (Bottom-Left).
6.3 Block diagram for Short Local Trajectories: $T_{s_i}^k$ and $T_{s_i}^f$ is marked by '×'...

6.4 a) Block diagram for different trajectories: $S_1$, $S_2$, and $S_3$ are spatial neighbor trajectories. $S_1$ is of different length compared to others. $S_3$ and $S_4$ are trajectories of same region albeit different frames and thus form two independent trajectories. b) Generation of symbol sequence from optical flow using circular quantization grid. An optical flow (denoted by red) is replaced by the bin number ‘x’ in the symbol sequence. $u_t$ and $v_t$ are horizontal and vertical components of the optical flow.

6.5 Comparison of trajectory similarity when the same object (marked with ellipse of blue) is at different position. a) Near b) Middle and c) Far. As can be observed, the trajectory and its properties change depending upon their position in the video.

6.6 Performance of the different approaches tested for a) Frame-level anomaly detection on the Ped1 b) Frame-level anomaly detection on the Ped2 and c) anomaly localization in the Ped1.

6.7 Few Outputs from different video sequences where each row is from a single sequence. The anomalies are marked in ‘Red’ color. Note: All the outputs are generated by using a constant threshold.

6.8 Performance of the different approaches tested on CHUK-Crowd Anomaly Dataset (Best viewed in color).

6.9 Few Outputs from CHUK-Crowd Anomaly dataset video sequences where each row is from a single sequence. The anomalies are marked in ‘Red’ color. (Best viewed in color).
List of Tables

2.1 Results Comparisons: Accuracy (Execution Time per video) .......................... 20
2.2 Effect of various parameters on UCF 50 ....................................................... 21
3.1 Jaccard Index Similarity Measure for [11] and our proposed approach for some of the videos in the database with reference to ground truth ......................... 31
4.1 Equal Error Rate (EER) on Ped Data-sets .................................................... 45
4.2 Rate of Detection (RD), Area Under the curve (AUC) and Detection Rate for Ped 1 .......................................................... 45
4.3 Variation of Equal Error Rate with varying $\gamma_{L1}$ ..................................... 46
4.4 Comparative results with other approaches on UMN ..................................... 48
5.1 Frame Level Comparison: Equal Error Rate (EER) on Ped data-sets ............. 58
5.2 Pixel Level Comparison on Ped 1: Rate of detection (RD), Area under the curve (AUC) and Detection speed .......................................................... 58
6.1 Effect of length of trajectory on frame level EER of Ped1 dataset .................. 70
6.2 Frame Level Equal Error Rate (EER) on Ped Datasets .................................. 71
6.3 Rate of Detection (RD), Area Under the curve (AUC) and Detection Rate for Ped 1 .......................................................... 71
6.4 Frame Level Equal Error Rate (EER) on CHUK-Crowd Anomaly Dataset .... 73
Chapter 1

Introduction

In 20th century, the world has seen faster technical developments than anyone could ever imagine. The developments are not only limited to a single field but spread over many fields such as electronic devices, communication, computational devices etc. Among all of these, the invention of computers is considered to be the epitome of the technological developments. Though the older computers were supposed to be used for computational purposes, but the capability of the computers alongwith inventions in physical devices enabled them to perform many complex tasks. Today, computers are used in almost every facet of human life.

Recently, computers are being used for analysing various situations related to human life in order to emulate the intelligence of human brain. One among the many such task is that of understanding visual content. Human brain from eternity has been effective in analysing visual content. But a imparting similar capability to a computational system is a major challenge due to large variations in visual features of similar content. For example, cars could have different color and design but yet conceptually they are the same. The problem of understanding the visual content is further complicated by the associated semantics in visual content. For example, differentiating a racing car from a normal car is very challenging and can only be possible through attaching high level semantics to the visual content.

The wide variety of objectives and high level semantic meanings associated with the understanding of visual content pose a major challenge to today’s researchers. Among all of them, event analysis is the one of the very challenging problems posed to vision group.

1.1 What is an Event?

The word ‘Event’, in English, has multiple meanings such as
• ‘A thing that happens or takes place’

• ‘A social gathering’

• ‘A occurrence located in space-time’

However, to put the objective into perspective, we have defined an *Event* as a occurrence located in *space-time*. Thus, it can refer to a crowd gathering or a crowd disturbance or an action performed by a certain human being. The important characteristic of an event is its relation with time i.e. a point of occurrence on a time frame.

Due to its generic definition, an event can be used to indicate an activity performed by a single person or by a group of people present in the crowd. Incorporating all these types of events under a single roof becomes challenging as the features for each event vary in accordance to the number of people present. Hence, automated event analysis poses a huge challenge to current vision research community.

An event can be effectively captured by using any visual medium such as an image or a video. An image being only a snapshot of spatial information at a specific time instant, fails to capture the relation of an event with its time frame. Due to the lack of the temporal information in images, detecting an dynamic event becomes very challenging.

On the other hand, a video is a coherent sequence of images, commonly referred as frames, placed in a temporal order to impart continuity in visual perception as shown in Fig. 1.1. These images can be recorded, copied or displayed in the temporal sequence. Due to its temporal extension, capturing the temporal dynamics is relatively easier compared to static images.

Event analysis captured by videos is a subset of a bigger problem of video content analysis and plays a very critical role in today’s large scale intelligent video surveillance systems. The main aim of event analysis is to perceive the semantic information of the event captured in video through carefully laid out steps. For example, we perceive a number of events like crowd gathering, crowd running, etc., everyday as part of our daily routine where each of them involves complex movements of various objects, with large number of variations. These variations could be due to changes along time, place, circumstances and human behavior. Even though it seems easy for a human brain, but for a machine it is far from done. This is because human brain is well trained to recognize basic to complex patterns that helps in understanding the high level semantics of the real world.
1.2 Motion: A key component?

As videos are sequences of frames, all the techniques for image content analysis can be extended easily to videos. The basic low level features such as color, texture, etc. too can provide useful information about the visual content of a video. But to understand the high level semantic meaning of the video content, understanding the temporal relation between frames is very important. For example in Fig 1.1, the 1st frames of both video sequences have similar human posture content and cannot be distinguished based on single frame information. It is only after considering future frames in the sequence, that the actual action is recognizable. (See 1st and last frame of Fig. 1.1a and 1.1b).

![Figure 1.1: Frames of Video from Weizmann Action Dataset [41]. Even though 1st frame is similar later frames are different due to temporal variation in the video: a) Action - Skip b) Action - Run](image)

These temporal relations can be captured in variety of ways using low level features such as frame differencing, history image, etc. But most of these approaches are limited to sparsely crowd and static background videos that affect the efficacy of the automated content analysis. However, motion captures the movement of various objects that play a critical role in analysing content of the video. Human beings have an intuitive understanding of the concept of motion that helps them immensely in perceiving the high level semantic. The focus of this thesis is to perform video based event analysis by extracting and analysing these motions to solve different objectives such as classification, crowd analysis, etc.

Even though motion across frames captures movement of objects, this motion is very different from real world three-dimensional (3D) motion in a scene. This is due to the 2D projection of 3D scene on image plane. So, the motion information derived from a image sequence is the 2D displacement of the projection of the objects present in a 3D scene. Nevertheless, this
projection based motion information is still key aspect as the temporal changes on the 2D projections can also provide a good insight about the event.

Event analysis in videos can be classified into two categories based on the domain of analysis. These categories are a) compressed domain video analysis and b) pixel level video analysis. The compressed domain video analysis is preferred for speed by sacrificing the accuracy during analysis. Whereas, pixel-level video analysis provides high accuracy due to the presence of highly accurate pixel informations of a video. In the subsequent sections, we will provide a very brief survey of both video analysis categories and their merits and demerits focusing predominantly on motion analysis.

1.3 Compressed Domain Video Analysis

As videos are sequences of images/frames, the raw stacking of frames can result in huge amount of data. In a bid to reduce the size of the video, the redundant information within and across frames is exploited for compression of the video. This spatio-temporal redundancy is reduced by using various image transforms and motion compensation leading to a higher compression.. These transforms and compensation vary from standard to standard that harnesses various information redundancy differently using pre-laid set of rules. As from a laymen perspective, one needs to decompress these videos before displaying and accessing the pixel information in them. Only then any subsequent feature extraction can be performed for automated video analysis. Many researchers such as Babu et al. [16], avoided this decompression overhead by exploiting meaningful patterns among the compression parameters to perform suitable video analysis tasks. This laid the foundation of compressed video analysis.

The objective of the compressed video analysis is to rapidly analyze the video with lesser amount of data exploiting various coded compression parameters. In the following sections, we first discuss about the various compression standards (1.3.1) followed by a short discussion on H.264 compression standard in 1.3.2. In the end (1.3.3), we present a brief list of compressed video analysis problems tackled in past.

1.3.1 Video compression standards

The compression standards have evolved over time with varying objectives. For example MPEG-1 was meant for storing videos on CD whereas the more recent H.264 is focused to cater the needs of internet streaming. Over past two decades, these objectives have lead to the development of complex encoders that achieve higher compression rate. The following is a brief
overview of the various compression standards (Refer [18] for further details):

1.3.1.1 H.261

H.261 [49] standard is one of the oldest digital video coding standards designed primarily for videos transmitted over telephone such as video conferencing that operates for bitrates in the range 40 kbit/s to 2 Mbit/s. The standard supports two video resolutions viz., QCIF (176 × 144) and CIF (352 × 288).

1.3.1.2 MPEG-1

MPEG-1 [46] supports resolutions up to 4095 × 4095 (12 bits). MPEG-1 was designed for coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbit/s, but can even go up to 100 Mbit/s.

1.3.1.3 MPEG-2

MPEG-2 [47] was developed for higher quality video at bit rates more than 4 Mbps for digital broadcast applications such as NTSC (720 × 480), PAL (720 × 576), etc. and storage devices such as DVDs.

1.3.1.4 H.263

H.263 [87] was designed mainly for video conferencing and other audio-visual services transmitted on Public Switched Telephone Networks (PSTN). The standard is also a potential candidate for internet-based video applications like flash videos, aiming at low bit-rate applications less than 64 kbps.

1.3.1.5 H.264/AVC or MPEG-4 (Part 10)

Compared to the predecessors, H.264/AVC [86] is the most recent and widely used standard. It offers better video quality at lower bit-rates. More than 50% bit rate savings can be achieved by replacing MPEG-2 videos with H.264 compression and hence can be employed for applications ranging from internet streaming to digital broadcasting.

1.3.2 H.264 Standard Overview

H.264 [86] compression standard is the most complex commercially used encoder. The standard exploits various redundant information one way or the other while encoding. This exploitation
of redundant information during H.264 encoding can be classified into: a) Intra coding (Spatial redundancy) and b) Inter coding (Temporal redundancy). The brief explanation of these coding techniques in H.264 standard are as follows:

Spatial redundancy is due to the correlation of nearby pixels in a given image frame. Usually, a spatial segment is indicated using a fixed size rectangular block referred as macroblock in all the compression standards. These macroblocks, being of fixed size, cover smaller region of a frame and may have huge similarity with its neighbouring macroblocks. For example, in case of a frame showing a landscape, all the macroblocks corresponding to sky have high visual similarity among themselves and contain huge amount of redundant information. H.264 exploits this spatial redundancy by predicting the current macroblock from the previously encoded neighboring macroblocks based on predefined prediction modes. The prediction error is encoded using Discrete Cosine Transform (DCT) followed by Quantization and Entropy coding. The technique of harnessing spatial redundancy is referred as ‘Intra coding’. Due to the non dependency of intra coding of a frame on any other frame during compression, it is used to compress the first frame of the video. Despite the advantage, the amount of compression achieved is less and thus this is mainly restricted to compress first and few other strategically placed frames in a video called Intra frame (I-frame).

On the other hand, the temporal redundancy is due to the high correlation across consecutive frames of a video. ‘Inter coding’ exploits this redundancy by estimating the movement of each macroblock to next frame that reduces the number of encoding bits for a given video quality. The movement of each macroblock with respect to reference frame is captured by the motion vector. The residuals left after prediction of the current block from the referenced block is very less consisting of high number of zeros. These residuals are subsequently encoded using Discrete Cosine Transform (DCT) followed by Quantization and Entropy coding. H.264 standard further specifies the usage of flexible macroblock sizes (from 16 × 16 to 4 × 4) during inter prediction to encode frame efficiently. The blocking artifacts that may have been incurred after DCT-based compression, are suppressed via a deblocking filter in H.264/AVC for better visual appearance. As correlation is high across consecutive frames, the compression ratio achieved by inter coding is very high. However due to its dependency on other frames, motion vectors are estimated for Predicted frames (P-frames) and Bi-predicted frames (B-frames) using multiple reference frames (usually I or P frame, depending upon the profile used). This is known as multi-picture inter prediction.

Apart from these features, various other advanced features adopted in H.264/AVC that includes context adaptive entropy coding, addition of switching slices, quantization optimization, lossless macroblock coding etc. Additionally, the standard supports 21 different profiles and
quite a few levels for application-oriented usage.

Apart from providing high visual quality and high compression, the motion vectors in H.264 provide better estimate about the object motion. Thus, the motion vectors can be correctly used for various video and event analysis problems. The improved quality of H.264 motion vectors are attributed to mainly two factors: a) Variable block size and b) Quarter pixel motion prediction. Variable block size ensures the block size of H.264 changes depending on bits required to encode the residual error. The standard specifies block sizes can vary from $4 \times 4$ to $16 \times 16$. Accuracy of motion vectors is further improved by accurate prediction of the motion vector using quarter pixel motion prediction.

1.3.3 Compressed Domain Video Analysis and Applications

Various video analysis challenges have been tackled in recent times using compression parameters such as motion vectors, macroblock size, quantization parameter, etc. However keeping ourself aligned to the objective of the thesis, the following are the lists of compressed video analysis techniques proposed for various compression standards, predominantly using motion vectors (motion information). For detailed survey on various compressed domain video analysis techniques, refer [18].

1.3.3.1 Video Indexing and Retrieval

With huge amount of video being uploaded on internet per minute, it becomes a key challenge to index videos for quick retrieval. Many researchers have used various compression parameters to define the content of the video. Yu et al. [115] were among the first few who used partial pixel domain features such as color and edge information alongwith compression parameters such as motion vectors and DC image in MPEG-1 for video retrieval and indexing. Later, Babu et al. [15] extracted the object based and global features from compressed MPEG video using the motion vector information. Recently, Mehrabi et al. [69] used color histogram feature of DC-pictures (derived from I frames) for content-based information retrieval in H.264/AVC compressed domain.

1.3.3.2 Moving Object Tracking

One of the leading research areas of computer vision is object tracking with wide applications in surveillance, navigation, transportation monitoring, human computer interaction, and robotics. One of the initial works in tracking was put forward by Favalli et al. [39] in MPEG-2 compressed domain. The object/region to be tracked, at macroblock level, was first identified manually.
The macroblock information in adjacent frames was utilized to find the new position of the tracker. This was done using the motion vectors associated to the macroblocks. Achanta et al. [9] used color information from intra frames to identify the object to be tracked and forward motion vectors in the P and B frames to track the object. As opposed to the related works till that time, object was tracked in the B frames too. Recently, Kas et al. [51] proposed an unsupervised motion vector based trajectory estimation approach for moving objects in H.264 compressed videos.

1.3.3.3 Moving Object segmentation

Moving object detection and segmentation is a key module for any object analysis problem in video. Thus, many compressed domain approaches have been developed with that respect. In H.264/AVC standard, Zeng et al. [116] utilized motion vectors for object segmentation in H.264 compressed videos. Moving objects were extracted from the motion field using the Markov Random Field (MRF). Later, Hong et al. [42] proposed a moving object segmentation approach in H.264 compressed domain considering moving camera scenarios. They have used cues only from the block partition modes and motion vectors in the compressed bit stream.

As discussed in literature, motion information from motion vectors are key in many compressed domain video analysis. We extended the capability of using motion information for various other event analysis applications such as action recognition, crowd flow segmentation and anomaly detection in crowd videos.

1.4 Pixel-Level Video Analysis

After decompression, videos are three dimensional (3D) matrices of integer values that indicate the true visual content captured during events. Each integer values in the 3D matrix are referred as ‘pixels’. Due to the effectiveness of pixel in capturing true visual content, automated analysis of an event using pixel level processing is better in terms of accuracy. However, the process is computationally complex. This is evident as the amount of information that is needed for processing is many times more than that of compressed domain analysis.

1.4.1 Features in pixel-level analysis

Due to in-detail information available in pixel-level analysis various features have been discussed in literature. Many of them also played a key role in video analysis. All these features can be
divided predominantly into static image features or motion based video features as discussed below.

- **Static Image Features**: These low level features such as SIFT [64], HoG [36], color histograms, etc. used for video analysis have been extended from image analysis. As the images lack temporal component, one needs to solely rely on the spatial and temporal distribution and representation of these static features to understand the content. In literature, people have used SIFT to accurately represent local structure of interest points. On the other side, HOG is used to detect people in crowd. To capture the temporal behavior of people in the video, the temporal sequencing of these static features features are analysed.

- **Motion based Video Features**: Contrary to the mentioned static features, motion based video features take the temporal aspect into consideration and thus are more suitable for video analysis. These set of features can be further classified into short duration features or long duration features. STIP [56] and HOF [57] are short duration features. STIP captures the interest features in the space-time framework. HOF defines the motion information for space-time cubes. Whereas, long duration features such as trajectories [103] of humans provide detailed analysis about the movement of people in the video. These can be used for action recognition and anomaly detection.

### 1.4.2 Extracting Flow Information

Motion features such as Histogram of Optical Flow (HOF) requires flow information which can be pixel or sub-pixel accuracy. In pixel level analysis, this flow information is referred as ‘optical flow’. According to [96], the accuracy of the flow information or optical flow depends upon the objective function and optimization framework used during its computation.

The criticality of objective function in computing optical flow can be understood by the fact the traditional brightness based objective function proposed by Horn and Schunck [43] results in average angular error (AAR) of 30 degrees. Other major functions involve oriented smoothness [95], image segmentation [58], etc.

On the other side, optimization framework reduces the error in initial computation of the optical flow. Current best framework involves coarse-to-fine estimation. The framework helps to deal with large motions [21, 27], texture decomposition [107, 106] or high-order filter constancy [26]. Other frameworks suggest post filtering such as median or bilateral to remove noisy estimation and introduce spatial consistency [112].
1.4.3 Pixel Level Analysis and Applications

Due to presence of detailed information, very large number of video analysis problem have been explored. Various features have been exploited as discussed. However with respect to motion information keeping in-line with objective of the thesis, following is a brief listing of pixel based video analysis applications.

1.4.3.1 Object Tracking

Object tracking is one of the major video analysis problems that is heavily sensitive to variations in object and background. Many algorithms such as \[12, 68\] uses template matching techniques to track objects. However, sometimes motion information is also used as prior as in particle filter approach \[75\] for object tracking.

1.4.3.2 Video Synopsis

According to Rav-Acha et al. \[82\], video synopsis (or abstraction) is a temporally compact representation that aims to enable video browsing and retrieval. Majority of the works \[13, 35\] in this field revolve around extraction of key frames that represent each shot. With respect to motion based key frame extraction, one of the first work by Narasimha et al. \[73\] identifies key frame based on the motion intensity descriptor and spatial activity descriptor. More recently, Mendi et al. \[71\] selects the key frames based on two different optical flows. The key frames correspond to frame based motion extrema in the video. These key frame based abstraction generates static storyboard and thus are less preferred over moving-image abstracts that are merely a collection of image sequences. Ma et al. \[65\] is among the very few who proposed attention model based on motion information to generate moving-image abstracts.

1.4.3.3 Crowd Analysis

For proper planing and management of crowd and to ensure safety and security, automated crowd analysis is of utmost requirement. According to \[117\], majority of the crowd analysis solutions such as \[121\] involve modeling crowd behavior based on its motion. One of the first work by Chan et al. \[29\] tries to segment videos of crowded environments. This quickly followed by Ali et al. \[11\] who performed crowd flow segmentation. Later many other crowd analysis algorithms for applications such as anomaly detection \[53, 20\], data driven crowd modelling \[88\], dominant flow detection \[33\], etc. have been proposed.
1.5 Organization of Thesis

The main aim of the thesis is to analyze the effect of motion in describing the events captured in videos. The content of the thesis is divided over 7 chapters that present an overview of the approaches tackling event analysis problems using the motion information. The brief summary of each chapter is as follows:

- In Chapter 2, we present a novel solution to action/event based video classification in compressed domain using motion vector information resulting in faster computations.

- In Chapter 3, we study the problem of crowd flow segmentation using motion vectors that can be used for modeling the movement of a crowd.

- Chapter 4 demonstrates the effectiveness of motion vectors in describing motion information for anomaly detection.

- From Chapter 5, we explore pixel domain event analysis for anomaly detection. In this chapter we deployed short motion features for the problem of anomaly detection to obtain better accuracy.

- In Chapter 6, trajectory based feature has been explored for anomaly detection in pixel domain. Due to the use of trajectory information we have demonstrated highest detection accuracy.

- Chapter 7 concludes the thesis with an overall summary of motion based event analysis and chalks out the possible future application that can be potentially solved using motion information.
Chapter 2

Video Classification

2.1 Introduction

With ever growing number of videos on web, the problem of video classification has grabbed the attention of vision researchers all over the world. On an average, 72 hours of videos are added per minute to youtube itself all across the world [8]. So, if we assume each video is of 3 minutes on average, then around $72 \times 60/3 = 1440$ videos are added every minute that accumulate to $1440 \times 60 \times 24 = 2073600$ videos every day! With such large amount of data being uploaded every day, the need for video classification on a larger scale becomes crucial for efficient video retrieval, annotations, etc.

Generally, videos can be classified based on three modalities namely visual, audio and text material associated with it [25]. However, here we are mainly focusing on visual (event/action content) based classification. Recently, visual content based video classification have undergone significant developments that resulted in better and accurate classification. But, almost all of them require decompressed video and pixel level processing [105, 81] to extract features. As videos are stored in one or other compressed format, it is intuitive to develop algorithms in compressed domain for faster analysis by avoiding huge amount of pixel data and decoding step. To further analyse the overhead of decompression, let us consider if one starts decoding each video for classification, then just decoding all videos, uploaded in a day, in itself will take more than a day! Hence, the apparent need to look for approaches that handle compressed video classification.

Since H.264 is the widely used video compression standard, we propose extraction of features using H.264 motion vectors that represent the action content of video efficiently. The proposed classification approach is based on the fact that similar motion patterns occur among videos of
a category whereas these motion patterns vary across categories. These motion vector patterns are captured through proposed *Histogram of Oriented Motion Vector (HOMV)* feature to fulfil the objective of large scale video classification based on action content.

## 2.2 Related Work

According to the survey of Brezeale et al. [25], one can classify videos using different modalities. Modalities used for video classification can broadly be divided into three major categories - text based classification, audio based classification and visual content based classification.

Much of the research on video classification is based on combining multiple modalities. Huang et al. [45] suggested the use of Hidden Markov Model (HMM) on multiple modalities rather than single modality. On the other hand, Wang et al. [104], suggested the use of a hybrid approach which involved different models for different modalities like SVM for text and Gaussian Mixture Model (GMM) for audio-visual features.

Among video content based approaches, Dimitrova et al. [37] used face and text to represent the content of the video and used it with HMM to classify or retrieve videos. In motion based analysis, Wang et al. [105] proposed a combined model for spatial and temporal layout of motion pattern for video classification. Rapantzikosa et al. [81] demonstrated the importance of salient region detection to compute effective features during video classification. Later, Chaudhry et al. [31] proposed Histogram of Optical Flow (HOOF) feature alongwith Binet Cauchy Kernel on non dynamical system for action recognition (in effect ‘Action’ being content of the video). The proposed HOMV feature is similar to HOOF feature only in terms of feature descriptor, but differ significantly in extraction process such as region of interest and hierarchical spatio-temporal representation.

In compressed video analysis, majority of research is confined to moving object segmentation in surveillance setup. Babu et al. [17] was among the first few authors who used motion vectors of compressed MPEG video for segmentation. More recently, Poppe et al. [77] and Verstockt et al. [100] introduced macro block size and macro block type of H.264 stream as new reliable features for moving object segmentation. Moving to action recognition, Babu et al. [16] proposed HMM modeling of MPEG motion vector features to define the action content of a video [14]. Whereas, a motion similarity measure based on motion vectors of H.263 compressed video was harnessed by Yeo et al.[114] to perform action recognition. But, almost all of these approaches in action classification involved surveillance videos without camera motion.

In short, due to presence of variety of challenges such as camera motion, scale of object, etc., video classification based on content/action still remains an open problem. The problem
is further complicated due to the apparent need of real-time classification on a large scale. The proposed approach is a novel attempt to achieve faster classification by analysing a video in compressed domain that has lesser amount of data compared to pixel-level processing.

### 2.3 Proposed Algorithm

In this section, we present the details of the proposed algorithm to classify videos based on the action content using H.264 motion vectors. The proposed approach has mainly three stages a) Preprocessing, b) HOMV feature extraction, and c) Video feature extraction. The block diagram (Fig. 2.1) summarizes the training and testing modules of the proposed approach for video classification.

![Block diagram](image)

(a) Training module

(b) Testing module

Figure 2.1: Block diagram of the proposed approach

#### 2.3.1 Preprocessing

As motion vectors are aimed at harnessing the temporal redundancy across frames to minimize the number of encoding bits, few of them may not follow true object motion and are considered as noise with respect to the objective of the approach. Furthermore, in presence of camera motion, motion vectors dominantly represent background motion which act as a distractor for video classification. Thus, these motion vectors require a preprocessing step to align them to true object motion by removing the noisy motion vectors, estimating the camera parameters and finding the region of interest in the video that corresponds to foreground object.
Generally, most of the noisy motion vectors have huge magnitude in the order of size of the video frame. Thus, motion vectors which are of length more than 10% of the frame size are truncated to zero, effectively neglecting noisy motions in further computations.

Noiseless motion vectors are used in subsequent camera parameter estimation step. This camera estimation step helps in compensating the motion distortion due to the camera movement. Camera parameters are estimated using Eq. (2.1), where $s$ is the scale factor, $p_3$ and $p_4$ are pan rate and tilt rate respectively. $(x, y)$ and $(x', y')$ being current and future locations of the blocks [98].

$$
\begin{pmatrix}
x'
\end{pmatrix}
= s \begin{pmatrix}
x
\end{pmatrix}
+ \begin{pmatrix}
p_3 \\
p_4
\end{pmatrix}
\tag{2.1}
$$

This is followed by motion vector compensation using the camera parameters estimated in Eq. (2.1).

The content of a video is essential for our objective and thus picking up the effective content among others (viz. region of interest) is very important aspect. We employ a simple region of interest (ROI) estimation based on motion orientation gradient and motion magnitude gradient. The temporal accumulation of the above gradients are performed to achieve final ROI. In effect, ROI is the region where motion vectors change its characteristics over a temporal bunch of frames. Mathematically,

$$ROI = \begin{cases}
1 \text{ if } \left( \sum_{i-k}^{i} (\nabla(M) + \nabla(O)) \right) > Th \\
0 \text{ otherwise}
\end{cases}
\tag{2.2}
$$

where, $M$ and $O$ are magnitude image and orientation image with values normalized between 0 and 1. $\nabla$ denotes image gradient. $i$ is current frame and $k$ being number of previous frames used (we have used $k = 7$). Though very naive, this helps in selecting better features.

2.3.2 Feature extraction

Features are key in defining the content. Thus, intelligent extraction of features is important for classification. Here, feature extraction step involves three stages, namely hierarchical space-time cube generation, HOMV extraction and orientation normalization stage.

2.3.2.1 Space-time cubes generation

Space-time cubes are small segments of a video that together define the video content. Hence, effective arrangement of space-time cube through hierarchy can jointly represent the global
and local motion patterns. To capture the temporal variation in motion patterns, preprocessed motion vectors of a video are divided into small temporal cubes of partially overlapping $b$ frames. Each temporal cube is further divided into space-time cubes accordingly at three hierarchy of spatial division that capture various local arrangement at different scale. At first level, the space-time cubes are inclined to capture global motion patterns. Hence, the temporal cubes as retained as they are ($1 \times 1 \times 1$ at level 1 (coarse)). This is followed by dividing the temporal cubes into 3 and 5 space-time cubes along rows for $3 \times 1 \times 1$ at level 2 (medium) and $5 \times 1 \times 1$ at level 3 (finer) respectively. The space-time cubes, here, are partially overlapped along rows to extract interesting localized motion patterns. This is done as two different categories of video can have similar global motion pattern, but may vary significantly in localized motion patterns. For example Pullups and Pushups have similar global motion pattern but have distinctive local patterns at upper and lower space-time cubes respectively at finer levels. However, we do not divide the video along columns as dividing the columns will remove the left right symmetry among videos of same class. Additionally, we observed in our experiments that any further division of a video results in low spatial relation and reduces the classification accuracy. Fig. 2.2 illustrates the spatial and temporal distribution of the motion vectors in space-time cubes.

![Figure 2.2: Space-time cubes. a) In $1 \times 1 \times 1$, space-time cubes are only temporally overlapping (level 1) b) In $3 \times 1 \times 1$, space-time cubes are spatially as well as temporally overlapping as shown. (level 2) (Images are best viewed in color)](image)

### 2.3.2.2 Region of interest features

Global motion patterns include residual camera movements (after camera compensation), etc., that are captured at coarser level through features based on motion vector at all locations. But finer levels (level 2 and level 3) are intended to capture local motion pattern, hence we form features based on motion vectors present in ROI only at these levels. This helps to emphasize
more on local motion patterns through pruning of motion information corresponding to non-ROI regions and predominantly focusing on motion of foreground object.

2.3.2.3 Histogram of Oriented Motion Vector (HOMV)

Usually, similar videos have mirror properties i.e., a person walking left to right is equivalent to person walking right to left (left-right symmetry). Thus we have defined HOMV as weighted histogram of orientation of motion vectors binned on primary angle ($-\pi/2$ to $\pi/2$ anti-clockwise and $3*\pi/2$ to $\pi/2$ clockwise) where weight of each sample motion vector is equal to its magnitude. This weighing of samples enable us to discriminate a faster movement from a slower one. For example, running and walking can have similar orientation histogram (if a person is moving in same direction in both cases) but weighing the motion sample with respect to magnitude can discriminate both the patterns (running has higher motion magnitude than walking). Here, the dimension of HOMV is $n$, where $n$ is the number of orientation bins used. Fig. 2.3 further illustrates the organization of orientation bins.

![Figure 2.3: Organization of orientation bins.](image)

The proposed HOMV feature are computed for each space-time at every hierarchical level to represent the motion patterns with left-right symmetry as explained.

2.3.2.4 Orientation Normalization

Orientation normalization is required to accommodate the minor variations in orientation at global pattern. This is achieved by wrapping around all the orientation bins with respect to the maxima of coarser HOMV bin, i.e., wrapping around occurs at coarser level followed by finer
Algorithm 1  

HOMV Feature

**Input:** motion vectors for each Space-Time cubes. \( n \) = number of orientations.

**Output:** HOMV.

\[
MV = \text{motion vector for Space-Time cubes.}
\]

\[
\text{orientation} = \left\lfloor \tan^{-1}(MV_y/MV_x) \ast n/\pi \right\rfloor.
\]

\[
\text{magnitude} = \sqrt{(MV_x^2 + MV_y^2)}.
\]

**initialize:** feature = \(0_{1 \times n}\)

**for all** orientation at location \((x, y)\) in \(MV\) **do**

\[
\text{feature(orientation}(x, y)) = \text{feature(orientation}(x, y)) + \text{magnitude}(x, y)
\]

**end for**

\[
\text{HOMV} = \text{feature}/\|\text{feature}\|_1
\]

levels using the same orientation bin arrangement irrespective of the histogram maxima at corresponding level. This aligns the feature according to global motion pattern but simultaneously captures the variation of local motion pattern with respect to global one.

2.3.3 Video feature

Given HOMV features for each space-time cube, we build a bag of features (BOF) for each hierarchical level independently. This requires building codebook for each hierarchy. In our experiments, we have randomly sampled HOMV features from each training video. This forms a subset for each hierarchical level. We then optimized these subsets to form optimal codebook using k-means clustering at various hierarchical level.

Each space-time cube is then represented by the corresponding hierarchical level HOMV codebook using the nearest word (Euclidean distance). Later, this is latter used to form histogram of words for each hierarchical level and then concatenate them to form the video level feature. We provided higher weight for finer level and lower weight for coarser levels.

\[
F = [0.25 \ast f_{\text{coarse}}, 0.5 \ast f_{\text{medium}}, f_{\text{fine}}]
\]

where, \(F\) is the video feature. And, \(f_{\text{coarse}}, f_{\text{medium}}\) and \(f_{\text{fine}}\) denotes BOF feature at each level.
2.4 Experiments and Discussion

In this part, we describe the evaluation procedure and the datasets used. We have conducted experiments on two large video databases to demonstrate the robustness of our algorithm in handling wide range of variations. Even though both of them are action dataset, we have used them as clips for video classification based on action content. However these datasets are not encoded in H.264 format, so we encoded them in H.264 using baseline profile with 1 reference frame. Group of pictures (GOP) length was set to 30 and videos were encoded at a rate of 25 frame per sec. In the end, for classifying the testing videos based on the computed video feature, we have used libsvm (Support Vector Machine) [30] with RBF kernel.

2.4.1 Datasets

(a) UCF50

(b) HMDB

Figure 2.4: A Part of Datasets. [2] [4]

UCF 50 is an action dataset having 50 actions [83, 4]. Each action class has more than 100 sample videos of different variations in lighting, camera motion, view point changes, scale of the object, etc. But for our experiments, we randomly selected 100 videos from each action class and divided them into non-overlapping 70:30 set for training and testing respectively. Additionally, we have performed 3-fold cross validation.

Human Motion Database 51 (HMDB51) [2] is the one of the recent and most challenging database for action recognition. The evaluation criteria used for the dataset is same as that of UCF 50.
Table 2.1: Results Comparisons : Accuracy (Execution Time per video)

<table>
<thead>
<tr>
<th>Approaches</th>
<th>UCF50</th>
<th>HMDB51</th>
</tr>
</thead>
<tbody>
<tr>
<td>GIST[74]</td>
<td>38.8 (-)</td>
<td>13.4 (-)</td>
</tr>
<tr>
<td>HOG/HOF[101]</td>
<td>47.9 (-)</td>
<td>20.2 (-)</td>
</tr>
<tr>
<td>C2[54]</td>
<td>- (-)</td>
<td>23.2 (-)</td>
</tr>
<tr>
<td>Action Bank[90]</td>
<td>57.9 (12210 secs)</td>
<td>26.9 (-)</td>
</tr>
<tr>
<td><strong>Proposed</strong></td>
<td><strong>40.1 (2.7 secs)</strong></td>
<td><strong>18.3 (1.7 secs)</strong></td>
</tr>
</tbody>
</table>

Tab. 2.1 presents the comparison of the proposed approach with recent state-of-the-art algorithms and benchmarks. According to the best of our knowledge, all the benchmarked results documented in the comparative table require decompressed and pixel level processing. Whereas, our proposed approach in compressed domain attains huge reduction in overall computations with a minor penalty in the form of reduction in classification accuracy in comparison to recent pixel domain approaches. Fig. 2.5 illustrates the confusion matrix for both datasets. UCF50 achieves dominant diagonal for confusion matrix with highest classification rate of about 80% when compared to HMDB that achieves a maximum classification rate of about 60%.

To compare the effect of ROI on accuracy of classification, we have experimented with and without ROI computation on UCF 50 dataset. The results are shown in Tab. 2.2. Clearly, the use of ROI provides better representation of video content.

Motion compensation (MC) is introduced to compensate the camera motion. The results suggest only a slight improvement due to MC. This could be due to the fact that resultant
camera motions sometime also provides important cue for classification. For example ‘Military Parade’ (UCF 50) will mostly associated with panning motion which helps in better identification of parade videos (refer Tab. 2.2).

Orientation normalization (ON) with respect to the highest global HOMV of the video is found to be important aspect. As depicted through the results of the experiments, ON normalizes direction for HOMV features which help in better classification (refer Tab. 2.2).

Parameters like number of keywords ($K$) and number of orientation bins ($n$), are empirically set to 100 and 10 respectively. But, the results are highly dependent on temporal size of space-time cubes. With increase in temporal size from $b = 6$ to 10 (see 2.3.2.1), the results drop from 40.1% to 36.9%. We have set $b = 6$ for all our experiments.

### 2.5 Conclusion

In this chapter, we have proposed a simple compressed domain technique to classify action/event videos by using novel Histograms of Oriented Motion Vectors (HOMV). This feature along with its novel extraction process for various space-time cubes helps in defining the action/event content of the video for classification purposes. In the end, this is verified by experiments on two large scale datasets that demonstrate comparable classification accuracy to the state-of-the-art even in compressed domain.
Chapter 3

Crowd Flow Segmentation: A super-pixel based approach

3.1 Introduction

With the increase in population, there is an imperative need to monitor and model large gatherings such as religious festivals, concerts, etc in order to avoid any catastrophic events such as stampedes, etc. In the recent years, analyzing and modeling the videos involving high density crowded scenes have drawn attention of several researchers from various perspectives, where computer vision algorithms have played a significant role to address this problem. However, the performance of these traditional vision approaches gets deteriorated with the increase in crowd density that introduces complex dynamic among the different individual. Hence there is an immense need to address the problem of analyzing and modeling high density crowded scenes such as shown in Fig. 3.1.

Most of the existing approaches for the problem of crowd flow modeling was done in pixel domain. In this work we have proposed an approach for crowd flow segmentation in H.264 compressed domain. As decoding the compressed videos and subsequent feature extraction are additional computations, we have used the readily available motion vectors as features for crowd flow segmentation. This significantly reduces computation costs and simultaneously provides initial segmentation to built on for further pixel level processing.

Though the MVs convey significant info related to the dynamics of crowd behaviour, they are very sparse and noisy. Hence MVs should be preprocessed to get more reliable motion information. With these processed information, we have achieved better crowd flow segmentation.
3.2 Related Work

Over the past few years, several vision researchers have explored different problems related to crowd analysis from various perspectives. Most of the work done so far in crowd analysis is mainly focussed on crowd detection, tracking of individuals [88], measuring crowd collectiveness [122], detection of anomalous behavior [66, 23], etc. Jacques et al. [50] and Zhan et al. [118] provided a detailed review of various vision approaches in order to tackle different problems in crowd analysis. In this proposed approach, we have focussed on the problem of segmenting the dominant and dynamically meaningful flow segments in crowded scenarios involving high density moving objects.

Wu et al. [110] exploited translational flow to approximate local crowd motion and developed a region growing scheme for crowd flow segmentation based on optical flow field. They had also proposed another approach for crowd flow segmentation based on fuzzy $c$-means clustering [111]. Later, Wu et al. [109] had proposed another approach for crowd flow partitioning by perceiving this problem as a problem of scattered motion field segmentation by assuming the local crowd motion as translational motion field. They had implemented their approach on real crowded sequences and shown better results in segmenting homogeneous regions in the crowded scenes. In another work by Li et al. [60], the dynamic region that corresponds to the crowd flow is extracted and the flow segmentation is performed based on the orientation histograms of the foreground velocity field. Anil et al. [32] had outlined the review of various computer vision algorithms dealing with crowded scenarios and proposed a system for the automatic detection of dominant patterns of crowd flow in dense crowd scenarios by tracking the low level object features using the optical flow algorithm. Kuhn et al. [55] had developed a frame work for extracting the motion patterns by combining the optical flow with Lagrangian analysis of time dependent vector fields and shown its applicability for crowd analysis like automated detection of abnormal events in the video sequences. Another significant work to address this problem was reported by Ali et al. [11], where they have modeled this problem from the perspective of
fluid dynamics. A grid of particles was superimposed on the video and trajectories of these particles over the frames of the video was obtained using optical flow and further processed to compute the final crowd flow segmentation. They had also extended their approach for the detection of instabilities such as anomalous behaviors in the crowded scenes.

All the above mentioned approaches have dealt this problem in pixel domain where optical flow is computed for the crowd video sequence and further processed to achieve the crowd flow segmentation. Since it is computationally expensive to obtain the optical flow, it may not be useful for real time applications. Hence we propose a novel framework of achieving flow segmentation in the compressed domain using motion vectors. The prospect of achieving crowd flow segmentation in the compressed domain was explored in [40]. However, on the pursuit of improving the computational efficiency, a novel framework with less computational complexity, still retaining comparable accuracy, is proposed and demonstrated on the publicly available dataset.

### 3.3 Problem formulation

We have perceived the problem of crowd flow segmentation from the perspective of determining the number of physically and dynamically meaningful flow segments using flow vectors ($F$). In order to obtain the number of flow segments, the delineation between the dynamically meaningful segments have to be efficiently captured while discarding the spurious regions which do not contribute to the demarcation of semantic crowd flow segments. In the proposed approach, this is performed through multi-scale image representation obtained using super-pixel segmentation with varying number of segments.

Let $E_k$ be the set of edges that corresponds to the scale $k$ where, $1 \leq k \leq S$, $S$ represents the number of scales (number of super-pixels) which is shown in Eq. (3.1)

$$E_k = \{x_i : \forall x_i \in \partial \mathcal{R}_k\}$$  \hspace{1cm} (3.1)

where, $x_i$ represents the pixels along the edges of the super-pixels at scale $k$, $\mathcal{R}_k$ denotes the segments and $\partial \mathcal{R}_k$ denotes the boundaries of the super-pixel segments.

Let $E$ be the entire set of edges at all the scales which constitutes both weak (spurious edges) and strong edges as shown in Eq. (3.2)

$$E = \bigcup_{k=1}^{S} E_k$$  \hspace{1cm} (3.2)
The detection of the significant edges helps in computing the final crowd flow segmentation. Now the problem has converged to computing the edge-set $E_f$ which is the subset of $E$ ($E_f \subset E$) where, the edge set $E_f$ denotes the edges that distinguishes the significant crowd flow in the video sequence as shown in Eq. (3.3)

$$E_f = \{x_i : \forall x_i \in \partial R_g\}$$  \hspace{1cm} (3.3)

where, $x_i$ represents the pixels of the edges of the final edge map, $R_g$ denotes the ground truth segments and $\partial R_g$ denotes the boundaries of the regions of the ground truth for crowd flow segmentation.

### 3.4 The proposed approach

The proposed approach constitutes of four stages. The overall block diagram of the proposed approach is shown in Fig. 3.2.

![Figure 3.2: Overview of the proposed approach](image)

#### 3.4.1 Preprocessing of Motion Vectors

Motion Vectors of H.264 compressed videos are aimed at video compression through exploring the temporal redundancy to optimize the number of encoding bits. As this optimization depends upon the underlying compression profile and algorithm used, they may not follow true object motion across consecutive frames and can be noisy. So the noisy motion vectors which shoots out at the static region of the video has to be eliminated before further processing of motion vectors. It plays a pivotal role in computing the accurate crowd flow segmentation. Hence the motion vectors that contribute to the crowd flow are retained and erroneous motion vectors
are discarded by taking spatio-temporal median filtering with a neighborhood of $5 \times 5 \times 5$ for spatio-temporal cube. Still there can be some erroneous motion vectors which are discarded by eliminating the motion vectors that occurs only for the few frames. Empirically, the motion vectors which occurs less than 10 percent of the number of frames at a specific location are discarded.

Secondly, there can be camera motion which leads to nonzero motion vectors for the static region resulting in distortion of the background. Hence the motion vectors in the background is eliminated by incorporating camera motion compensation process as mentioned in [98] which is shown in Eq. (3.4)

$$
\begin{pmatrix}
  x' \\
  y'
\end{pmatrix} = s \begin{pmatrix}
  x \\
  y
\end{pmatrix} + \begin{pmatrix}
  p_3 \\
  p_4
\end{pmatrix}
$$

where, $s$ is the scale factor, $p_3$ and $p_3$ are the pan rate and tilt rate respectively. $(x, y)$ and $(x', y')$ represents the current and future locations of the blocks. After discarding the erroneous motion vectors and compensating for the camera motion, a motion pattern is generated by considering the collective representation of the motion vectors, which is obtained by taking the median of the motion vectors temporally over all the frames. In other words, the resultant motion vector at a particular location $(i, j)$ is obtained by taking the median of all the motion vectors temporally at that particular location. Subsequently, a motion mask is generated corresponding to the regions with motion information.

### 3.4.2 Clustering of Motion Vectors by Super-pixels

The resultant motion vectors obtained from the previous step are transformed to color space as mentioned in [19] and the color coded map is smoothed in order to avoid the blocky effect due to the sub-macro block level motion compensation. Now the super-pixel segmentation is performed on this color coded map at various levels/scales by varying the number of super-pixels. Several researchers have explored various algorithms for super-pixel segmentation. However, in this proposed work we have used the super-pixel segmentation proposed by Li et al. [63]. As we increase the number of super-pixels, it results in clustering the motion vectors from coarser to finer level and subsequent extraction of edges of clustered motion through super-pixel boundaries. But the strong edges that separates the distinctive flow patterns of the scene and the delineation that discriminates the orientation of the motion vectors is captured in most of the levels. Hence these strong edges which contribute to the delineation of the crowd flow segmentation is retained by exploiting the strength of these edges and orientation of the motion vectors.
on both sides of the corresponding edges. The intermediate stages of the proposed approach is shown in Fig. 3.3.

Figure 3.3: Output at the intermediate stages of the proposed approach (a) Input video sequence (b) Color coded version (c) Confidence score $C_1$ (d) Confidence score $C_2$ (e) Final confidence score $C_f$ (f) Flow segmentation

3.4.3 Detection of true edges using Confidence scores

The segmented output of boundaries of the super-pixels at various levels are integrated together and a collective representation of the segmented output was obtained which contains the edges at all the levels. We have introduced two confidence scores to retain the true edges of the super-pixels and prune the spurious edges at various levels. The first confidence score is based on the strength of the edges. This is obtained by integrating all the corresponding edges at various levels of the super-pixels as shown in Eq. (3.5)

$$\mathcal{E}_s(x_i) = \sum_{k=1}^{S} E_k(x_i)$$  \hspace{1cm} (3.5)
where, $E_s$ denotes the sum of edge strength across all the levels, $E_k$ represents the boundaries of super-pixels at level $k$, $S$ denotes the number of scales and $x_i$ is a indicator value that represents the boundary pixels of a region.

Since the true edges which contribute to the crowd flow segmentation are retained in most of the levels, it will have higher strength compared to the spurious edges which occurs only in a few levels. The first confidence score ($C_1$) is obtained by normalizing the resultant integrated edge map as shown in Eq. (3.6)

$$C_1 = \frac{E_s}{S} \quad (3.6)$$

where, $E_s$ is the edge strength for all edges as defined in Eq. (3.5). $S$ is the number of scales.

Secondly, the second confidence score ($C_2$) is measured based on the orientation of the motion vectors on both sides of the edges. The difference of orientation of the motion vectors on both sides of the edges is obtained to discard the edges that contribute to over segmentation. The equation is shown as Eq. (3.7).

$$C_2 = \frac{D(\theta_{S_i} - \theta_{S_j})}{\pi}, \text{ s.t. } S_i \in N(S_j) \quad (3.7)$$

where, $\theta_{S_i}$, $\theta_{S_j}$ represents the orientation on both sides of the corresponding edge and $D(\theta_{S_i} - \theta_{S_j})$ represents the angular distance between the orientations on both sides of the edges. $N(S_j)$ denotes neighbors of $S_j$. If the orientation on both sides of the motion vectors are coherent to each other, then the difference of orientation of the corresponding edge will be very low and vice-versa. Now the final confidence score for each edge is obtained by multiplying the two confidence scores as shown in Eq. (3.8)

$$C_f = C_1 \times C_2 \quad (3.8)$$

### 3.4.4 Final crowd flow segmentation

Now the final confidence score from the previous step is refined by the motion mask obtained during preprocessing (see Sec. 3.4.1) to suppress any spurious edges emerging in the static region of the scene. The resultant final score is then thresholded to $R_\tau$, where $R_\tau$ represents the set of edges whose strength is more than $\tau$ as shown in Eq. (3.9)

$$R_\tau (x, y) = \begin{cases} 
0 & \text{if } C_f < \tau \\
1 & \text{otherwise} 
\end{cases} \quad (3.9)$$
where, $\tau$ is varied from 0 to 1 to extract the corresponding edge set $R_{\tau}$.

Now the true edges that corresponds to the contribution of crowd flow segmentation ($E_f$) is obtained by following the steps given in the algorithm 2.

**Algorithm 2**

**Input**: Threshold ($T$) = $\{\tau_i\}$ : $1 \leq i \leq t$ and corresponding $R_{\tau_i}$

**Output**: $E_f$ as defined in Eq. (3.3)

```
for i = 1 to t do
    $J_i = \max_{E_k} J(R_{\tau_i}, E_k)$ where, $J_i$ is the maximum Jaccard measure between $R_{\tau_i}$ and $E_k$
    $\forall 1 \leq k \leq S$
    $E_{k_i}^* = \arg\max_{E_k} J(R_{\tau_i}, E_k)$
end for

$E_f = \arg\max_{E_i^*} J_i$
```

### 3.5 Results and discussions

#### 3.5.1 Experiments

The proposed approach is evaluated on the dataset provided by Ali et al. [11], which contains a wide range of dynamics. Since, the dataset was not encoded in H.264 format, we have first encoded the same in H.264 format using x264\(^1\) baseline profile (only I and B frames) with 1 reference frame and group of pictures (GOP) length is set to 30. Since B frames are not used, baseline profile is ideal for network cameras and video encoders to achieve low latency [6]. The performance comparison of the proposed approach with Ali et al. [11] is bench-marked using Jaccard measure with the manually generated groundtruth. The motion vectors are extracted by partially decoding each H.264 video sequence and preprocessed as described in section. 3.4.1. These extracted motion vectors vary from sub-macro block of $4 \times 4$ to $16 \times 16$, but for ease of computation we replicate the motion for each macroblocks to their constituent $4 \times 4$ blocks. The number of super-pixels are varied in the range of 3 to 10 for all the experiments. Few of sample crowd flow segmentation obtained through algorithm 2 are shown in Fig. 3.4. All the experiments were performed using MATLAB on single core 3.4 GHz processor.

\(^1\)available at : http://www.videolan.org/developers/x264.html
Figure 3.4: Experimental results of some of the videos in the database. First column of images shows input video sequence, second column of images show results of Ali et al.\textsuperscript{[11]}, whereas third column shows the output of the proposed approach.

### 3.5.2 Analysis

Since we are using only motion vectors of H.264 compressed videos, the proposed approach runs faster for this challenging dataset. Some of the qualitative results of flow segmentation for the videos in the dataset are shown in Fig. 3.4. The Jaccard measure obtained for the
video sequences are shown in Tab. 3.1 (in the same order as that of the Fig. 3.4). Since Ali et al.’s [11] method is a global approach, the output flow segments extend beyond its actual demarcations, whereas our proposed approach appropriately captures the demarcations of the dynamic regions of the flow. Hence the proposed approach, even with noisy motion vectors shows better accuracy than that of [11] as shown in Tab. 3.1. For instance, the traffic video sequence (Seq. 2) shown in Fig. 3.4 clearly shows that the proposed approach captures the dynamic flow of the crowded scene with better precision than that of the [11]. Sequence 4 in the dataset is a complex flow pattern where people are moving alternatively in random directions. Since Ali et al.’s [11] approach performs segmentation at the global level, it merges the diverse pattern and gives only the dominant flows, whereas the proposed approach could capture the diverse variation of the complex flow pattern as shown in Fig. 3.4.

It is observed in Tab. 3.1, the performance degrades for Sequence 5 and 7. This is due to the fact the motion vectors in these videos could not capture very fine motion information at sub-pixel accuracy. Sequence 5 is of frame size $188 \times 144$ compared to $480 \times 360$ for rest of videos. For Sequence 7 (mecca sequence), most of the motion vectors are not retrieved due to the adjacent noisy motion vectors. Hence as long as the motion information is captured by motion vectors, the proposed approach performs better or equivalent to [11].

The execution time for the proposed approach is approximately 5 sec. for a video with 100 frames of size $480 \times 360$ with un-optimized Matlab code compared to 30 sec. by Ali et al. [11], without optical flow computation, executed on the same machine.

<table>
<thead>
<tr>
<th>Video Sequences</th>
<th>Jaccard Similarity Measure</th>
<th>Timings (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence 1</td>
<td>0.63</td>
<td>0.60</td>
</tr>
<tr>
<td>Sequence 2</td>
<td>0.28</td>
<td>0.67</td>
</tr>
<tr>
<td>Sequence 3</td>
<td>0.57</td>
<td>0.74</td>
</tr>
<tr>
<td>Sequence 4</td>
<td>0.67</td>
<td>0.68</td>
</tr>
<tr>
<td>Sequence 5</td>
<td>0.78</td>
<td>0.24</td>
</tr>
<tr>
<td>Sequence 6</td>
<td>0.41</td>
<td>0.62</td>
</tr>
<tr>
<td>Sequence 7</td>
<td>0.54</td>
<td>0.15</td>
</tr>
</tbody>
</table>

Table 3.1: Jaccard Index Similarity Measure for [11] and our proposed approach for some of the videos in the database with reference to ground truth
3.6 Conclusion

We have proposed a novel approach for crowd flow segmentation by using only motion vectors in the compressed domain. The preprocessed motion vectors are clustered using super-pixel segmentation at various levels by varying the number of super-pixels, and the crowd flow segmentation was achieved by retaining the strong motion boundaries and discarding the spurious ones. The proposed approach was found to perform faster with better accuracy than competitive pixel domain approaches.
Chapter 4

Anomaly Detection via Modeling local Motion

4.1 Introduction

As discussed in chapter 3, due to the increasing importance to physical security, analysis of crowd behavior has been a point of focus for many researchers in recent times. The objective of crowd behavior analysis spans from anomaly detection in surveillance scenarios to global crowd pattern analysis for safety planning in highly populated regions of which former is of particular interest to security personnel. As ‘A stitch in time saves nine’ is the main motive for security personnel, the focus is on real-time anomaly detection in surveillance videos. With more and more locations being equipped with camera surveillance, the problem of real-time analysis and monitoring becomes a huge challenge.

With recent advancements, computer vision algorithms could handle the problem of anomaly detection to large extent with sufficient accuracy. New algorithms are being developed to improve the accuracy of anomaly detection [66, 24, 48, 20]. However, most of these algorithms are computationally expensive making them unsuitable for real time applications. As majority of the current algorithms work with fully decompressed videos having pixel level information, there is an apparent need of decompressing the video before processing. Apart from the delay due to video decompression from any compressed format (MPEG-2, MPEG-4, etc.), huge amount of data and more complex feature extraction in pixel domain leads to slower execution speed. The problem of decompression overhead is trivial for few short videos or online processing of a feed from a single camera, but becomes humongous for large scale anomaly detection, both online and offline on a single centralized system. For example, a building being surveilled
24×7 by a single CCTV camera, recording at 25 frames per second, results in 24×3600×25 = 2160000 frames per day. During offline processing, assuming decompressing algorithms achieve 100 to 300 frames per second to decode, one would require 7200 to 21600 seconds ≈ 2 to 6 hours only to decode 24 hours of video! Even in online processing, decoding overhead becomes a concern when hundreds of video feeds are processed simultaneously for anomaly detection on a centralized system. The focus of this proposed work is to reduce these computational overheads by performing compressed video anomaly detection for H.264 videos. Even though the accuracy is compromised to some extent, this can provide fast initial screening for pixel domain analysis on uncompressed videos aimed for better accuracy.

Apart from compression, H.264 motion vectors also help in various video analysis tasks such as anomaly detection [23]. Usually, macroblock corresponding to anomalous region behave differently and exhibit different characteristics (e.g. motion magnitude and orientation) with respect to spatially and temporally neighboring macroblocks. The proposed algorithm computes the statistics of magnitude and orientation of motion vectors for a macroblock region over training videos. Anomaly is detected based on the probability of occurrence of a candidate motion pattern at that macroblock region.

4.2 Related Work

In one of the pioneer works involving optical flow, Adam et al. [10] used multiple local monitors which modeled flow information using histograms. Anomaly was detected through careful integration of multiple alerts from different monitors. Kim et al. [52] captured the distribution of the typical optical flow through Mixture of Probabilistic Principal Component Analyzers (MPPCA), then used Markov Random Field (MRF) to enforce local consistency. Mehran et al. [70] proposed Social force concept compared to Lagrangian trajectories by Wu et al. [108] to define the crowd movement. Ryan et al. [89] proposed textures of optical flow and measured the uniformity of the flow field to detect abnormality. As with any pixel domain algorithms compared to compressed domain approaches, these approaches fail to cater the need to real-time anomaly detection solutions.

Many methods in recent times were developed bypassing the optical flow computation and relied on object size and appearance. Mahadevan et al. [66] proposed mixture of dynamic textures (MDT) [28], which jointly modeled the appearance and dynamics of crowd scenes. Though it is better in terms of accuracy, processing each frame to extract textures resulted in huge computation and low execution speed. To overcome the same, recently Reddy et al. [84] proposed to divide the frame into non overlapping blocks for independent processing.
The detection of anomaly at each block was based on features generated using combination of motion, size and texture of the region.

All the aforementioned set of anomaly detection algorithms require training from video sequences containing normal behavior only. However, Sun et al. [97] proposed an unsupervised approach for anomaly detection in crowds based on motion saliency and attractive motion disorder concept.

The proposed method is different from others as we can model the normal behavior for each location individually via pyramidal approach to reduce the computation. The following section presents the proposed algorithm in detail.

### 4.3 Proposed Algorithm

Anomaly is defined as departure from usual characteristics. Mathematically, let \( y = [x_1, \ldots, x_n] \) be a candidate event at a particular region/location, where \( x_i \) is its \( i^{th} \) feature. So, the probability of occurrence of an event directly depends on the probability of occurrence of its features, i.e., \( P(y) = P(x_1, \ldots, x_n) \). Without loss of generality, we can assume independence among its features, resulting in

\[
P(y) = \prod_{i=1}^{n} P(x_i)
\]

where, \( P(y) \) is the probability of occurrence of the event. Thus, anomaly is defined as

\[
P(y) \leq \tau
\]

where \( \tau \) is the decision threshold.

In general, a behavior could be normal in one feature space but abnormal in another. This reduces the problem to extract relevant features that could discriminate the usual from unusual behaviors.

Typically in any crowd video, the motion is consistent during normal scenario and tend to have a distinctive motion patterns/features. Representing the motion through its magnitude and orientation, together, is found to be adequate to learn the usual pattern and detect anomaly. For example, a person riding a cycle on footpath has different magnitude than that of a person walking, even though direction of motion is same. On the other side, in case of sudden suspicious activity, there is sudden change in crowd movement magnitude and orientation than usual.

Motion Vectors (MVs) in H.264/AVC are defined for a minimum of \( 4 \times 4 \) block, which reduces the analysis computation by sixteen times than pixel level processing. But with increase in video
resolutions, computation increases proportionally. Though handling high resolutions videos are computationally expensive, it can be processed effectively by utilizing pyramidal representation. So, one can begin processing at coarser level (reduced frame size) and in case of ambiguity, move to finer level (actual frame size) to resolve it. This hierarchical processing leads to different levels of motion vector representation (created using smoothing and sub-sampling), which we refer as Motion Pyramid (Fig. 4.1).

In this approach, we tried to tap the local variations in magnitude and orientation of motion vectors (feature) to classify normal and abnormal events by modeling usual behavior for each location. The major modules of the proposed algorithm include: a) Modeling usual behavior b) Detecting abnormal pattern.

4.3.1 Modeling Usual Behavior

The proposed algorithm is trained for usual observations either from training videos or from the initial frames of each videos containing usual behavior pattern. Training is performed for each pyramidal level to learn the usual behavior patterns. Primarily, training is divided into two stages: i) Pre-processing and ii) Usual behavior modeling.

4.3.1.1 Pre-processing

Figure 4.2: a) Original variable macroblocks present in a single frame b) Motion replication upto its $4 \times 4$ constituent macroblocks. c) Motion pyramid with multiple frame resolution.
Motion vector replication: H.264/AVC encoding is performed on variable block size for better prediction and reduction in overall bit rate. So, motion vectors are estimated for variable blocks depending upon the content of the video. Instead of handling the variable block size motion vectors explicitly, we replicated the motion vectors for higher size macroblocks to its $4 \times 4$ constituents, resulting in MVs for every $4 \times 4$ blocks. This without any loss of content creates matrix of uniform size which are easier to handle (Refer Fig. 4.2b). Additionally assuming linearity in the motion prediction, if current P frame is encoded using any past $k$ previously encoded frames as reference, existing literature [72, 79] suggested rescaling motion vectors proportionally with respect to the distance between the reference frame and current P frame in the video sequence in order to obtain normalized motion vector.

Finer Level ($L_0$): As motion vectors are aimed at reducing the amount of bits required for encoding specific macroblock, it does not always reflect the true object motion. In order to minimize the effect of noise in motion vectors, a spatio-temporal median filter is applied on the motion vectors as shown in Fig. 4.2c (Eq. (6.13)). Additionally, median filtering guarantees motion interpolation for I-frames. For effective filtering, temporal range is divided into equal amount of past and future frames.

$$x[m, n, t] = \text{median}\{\tilde{x}[p, q, r], (p, q, r)\in w\}$$

(4.2)

where, $x$ and $\tilde{x}$ are the filtered motion vectors and raw motion vectors respectively. $w$ represents a neighborhood centered around location $(m, n, t)$ in the spatio-temporal cube. As median filtering is a computationally expensive step, we have used a small cube of $5 \times 5 \times 5$.

Coarser Level (Reduced Size) ($L_i$): Coarse Level ($L_i$) is obtained from previous level by spatial bi-linear interpolation of higher resolution raw motion vectors of each complete frame to one fourth of previous level size (keeping in tandem with minimum macroblock size of 4). Though, motion vectors are noisy, interpolation reduces the noise drastically making additional filtering futile. However, to have temporal consistency in coarser levels, motion vectors are averaged out temporally at each location. Furthermore, averaging ensures approximate motion estimation for I-frames at coarser levels as shown in Fig. 4.2c.

4.3.1.2 Usual Behavior (UB) Modeling

As behavior is characterized by motion vectors, UB modeling is done by forming histograms of motion magnitude and orientation at each pyramidal level $L$. Typically, motion vectors vary from one region to another in a frame. For example, motion vectors for regions away from the camera exhibit lower motion magnitude than those near to camera. The variation is not only
limited to motion magnitude but also the motion orientations, as direction of movement at each location can be very different and depends upon topography spanned by camera. Marginalized histograms are formed from temporal observation of both motion magnitude and orientations, respectively. We deliberately refrain from joint statistics as joint histograms at every location at a particular pyramidal level $L$ would require large amount of memory for storage.

![Image](image.png)

Figure 4.3: a) Sample density for the scene b) Variable kernel size at three different points $P_0$, $P_1$ and $P_2$. $P_0$ has highest number of training samples followed by $P_1$ and $P_2$. Thus, the kernel size are $P_2$ is biggest followed by $P_1$ and $P_0$

Though ideally, capturing the motion variations should result in true statistics. But, due to insufficient and inconsistent information at each location in training videos, histograms are noisy and need to be corrected. For example in Fig. 4.3a, locations $P_0$, $P_1$ and $P_2$ have different statistics. $P_0$ has high density of information compared to others whereas $P_2$ has the least density. Recent research in approximate nearest neighbor fields (ANNF mapping [80]) based on coherency have demonstrated that two neighboring location tend to exhibit same property with high probability. Therefore, accumulated statistics are further refined by interpolating based on neighbor characteristics. A Balloon estimator (Adaptive Kernel Density Estimator (AKDE)) is applied to achieve interpolation to form individual histograms. Since, $P(x_{i,j,k±a})$ and $P(x_{i±b,j±b,k})$ have effect on $P(x_{i,j,k})$, we use Gaussian kernel in AKDE (Eq. (4.3)) both spatially and across histogram bins with a variable kernel size where, $x_{i,j,k}$ denotes count of motion magnitude and orientation at spatial location $[i,j]$ falling in $k^{th}$ bin.

$$f_h(z) = \frac{1}{n} \sum_{i=1}^{n} K_h(z - z_i) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{z - z_i}{h}\right)$$ (4.3)

where,

$$K(\bullet) = (2\pi)^{-k/2} \left|\Sigma\right|^{-1/2} \exp \left(-\frac{1}{2}(z - \mu)^T \Sigma^{-1}(z - \mu)\right),$$
k = 3, \mu = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \Sigma = \begin{pmatrix} \sigma_s^2 & 0 & 0 \\ 0 & \sigma_s^2 & 0 \\ 0 & 0 & \sigma_b^2 \end{pmatrix}, h = \frac{c}{g(z)}

\( K(\bullet) \) is 3D Gaussian kernel with \( \sigma_s \) spatial standard deviation and \( \sigma_b \) across histogram range. \( z \) being the data. \( h \) is the variable kernel size based on the density of the samples \( g(z) \) at that location. \( c \) (set to 1) and \( n \) are scaling factors for bandwidth \( h \) and number of samples respectively. This is based on the assumption that any highly used regions by crowd will have large number of samples and will capture proper statistics. So, it is intuitive to use a smaller kernel around the high density region compared to other regions. (Refer Fig. 4.3b).

Marginalized histograms, for motion magnitude and orientation, are subsequently normalized to obtain respective probability density function. Usual behavior probability is computed using Eq. (4.1), where motion magnitude and orientation are two features.

\[
P(y) = P(x_{\text{magnitude}}) * P(x_{\text{orientation}})
\]

(4.4)

So, at each pyramidal level \( L \) and location \( p \), the anomaly is defined as

\[
P(y_L(p)) \leq \tau_L(p)
\]

where, \( y_L(p) \) and \( \tau_L(p) \) denote the event and decision threshold at location \( p \) and pyramidal level \( L \) respectively.

Due to varying topography captured by the field of view of camera, the motion statistics (motion magnitude and orientation) change from one location to another. Thus, at each pyramidal level \( (L) \), the decision threshold \( \tau_L \) varies based on locations. So instead of computing different thresholds for each location \( p \) through trial and error, we further define \( \tau_L(p) \) based on a constant value.

Let, \( H(p) = \{h_u(p)\}_{u=1...m} \) be the \( l_1 \) normalized histograms of a usual behavior feature at location \( p \) and \( u \) be the bin values. Then,

\[
V(p) = \text{sort}(|\{h_u(p)\}_{u=1...m}| \% \text{descending order}
\]

where, \( V(p) \) denotes the sorted probability (histogram) values at location \( p \). Subsequently,

\[
b^* = \text{argmin}( \{ \sum_{i=1}^{b} V_i(p) \} \geq \gamma_L ) \text{ where, } b = 1...m
\]

\[
\tau_L(p) = V_{b^*}(p)
\]
where, $\sum_{i=1}^{b} V_i(p)$ denotes the cumulative sum. $\gamma_L$ is a constant value for all locations at a particular pyramidal level $L$. (Refer Fig. 4.4 for details). In case of multiple features,

$$\tau_L(p) = \prod_{j=1}^{n} V_{b^*}(p)$$  (4.5)

where, $n$ is the number of features. Here, $n = 2$ as features are motion magnitude and orientation. Thus, the probability of an event being anomaly is now defined as

$$P(y_L(p)) \leq \tau_L(p) = F(\mathcal{H}(p), \gamma_L)$$  (4.6)

where, $F$ is the above explained function to obtain the thresholds.

In real time scenarios, the training set is accompanied by validation set. Unlike the training set that only has positive instances, the validation set consists of both positive and negative instances for estimating the decision parameters $\gamma_L$ for all levels. Usually, the majority of selection or rejection of candidates for anomaly detection occur at coarsest scale ($L_f$, where $f$ is the coarsest level) followed by finer levels in motion pyramid. So, we propose a greedy strategy to estimate the value for $\gamma_{L_f}$ followed by other intermediate levels according to motion pyramid. Using extreme levels of motion pyramid (ie. level $L_f$ and $L_0$), the value of $\gamma_{L_f}$ is fixed by maximizing anomaly detection on the validation set. Later each level $i$ from level $f - 1$ to level 1 is added to the motion pyramid and corresponding $\gamma_{L_i}$ is fixed by maximizing anomaly detection on the validation set with previously fixed $\gamma$ values for coarser level.

![Figure 4.4: i) Usual behavior histogram ii) Sorted histogram values iii) Plot of cumulative sum of the sorted histogram values used to find $b^*$ based on $\gamma_L$](image)

### 4.3.2 Detecting Abnormal Pattern

The use of motion pyramid plays a major role in reducing the computation without affecting the quality. In a nutshell, detection is started at coarsest level and moves to finer level only
if anomaly is suspected at current level (see Fig. 4.5). Thus effectively, it is a single frame detection, without any relation to future and past frames.

![Diagram](image)

**Figure 4.5: Detection from level L1 to L0.** Anomaly (circle) is suspected at coarse level L1 is further processed at finer level L0.

In the proposed approach, features (motion magnitude and orientation) of each location of a test video frames are compared to usual behavior to detect the region of anomaly. But, in a bid to increase detection rate in test videos, raw MVs are pre-processed only at coarsest level (similar to training phase) and detection is performed based on probability of occurrence of an event at every location, frame-by-frame. Any, pre-processing (filtering) and detection at finer pyramid levels are delayed till a probable anomalous candidate region is suspected at previous coarser level (Fig. 4.5). To further reduce computation, preprocessing (filtering) is performed only at suspected candidate locations. Anomaly is suspected at location $p$ if $P(y_{Li}(p)) \leq \tau_{Li}(p)$, where $y_{Li}(p)$ represents probability of occurrence of an event at coarse level ($Li$) and $\tau_{Li}(p)$ acts as decision threshold computed using $\gamma_{Li}$. In case an anomalous candidate is found at a particular location in previous level, preprocessing and detection is performed only at that location at current level. This hierarchical processing across levels results in huge computational savings. At finest level ($L0$) and location $p$, $P(y_{L0}(p)) \leq \tau_{L0}(p)$ indicates abnormality (Ref. Eq. (4.1)), where $y_{L0}(p)$ and $\tau_{L0}(p)$ represents event and decision thresholds respectively. The proposed steps could still result in some amount of spectral noise. These are anomalous detections mainly at the edges of moving body due to improper orientation. As our goal is to find a consistent anomaly movement rather than a false detection in a single frame, we additionally perform median filtering on $P(y_{L0}(p))$. This ensures that the anomaly is detected consistently rather than detecting a noisy motion boundary.
The proposed pyramidal approach provides a great boost up for high resolution videos as majority of the computation is bypassed at coarser levels. In low resolution videos, processing is done only at original level as coarser levels fail to capture the motion magnitude variations satisfactorily, leading to many mis-detection.

4.4 Experiments and Results

In this section, we introduce the datasets used for evaluation as well as describe the evaluation procedure. We have conducted experiments on three video databases to demonstrate the capability of the proposed algorithm to handle wide range of variations with comparable accuracy to the state-of-the-art techniques. Since these datasets were not encoded in H.264 format, we encoded the same in H.264 format using baseline profile (only I and P frames) with 1 reference frame and group of pictures (GOP) length is set to 30. Baseline profile is ideal for network cameras and video encoders since low latency is achieved because of the absence of B-frames [6]. We have used x264\textsuperscript{1} and JM15.1 [1] for encoding and partial decoding of the compression parameters respectively. All the experiments were performed using MATLAB on single core 3.4 GHz processor.

4.4.1 Evaluation Procedure

Algorithm is evaluated on two aspects namely global anomaly detection and localized anomaly detection. Ped1 [3] (HD videos) is used to test global and localized anomaly detection, whereas Ped2 [3] and UMN [5] crowd datasets are used for global anomaly detection only.

Ped1 contains training set of 34 clips, whereas Ped2 has a set of 16 clips. The testing set consist of 36 clips for Ped1 and 12 clips for Ped2. We have increased the size of Ped1 and Ped2 datasets from $158 \times 238$ and $240 \times 320$ to $480 \times 720$ respectively. This is done to validate our pyramidal approach. Anomalies are divided into two categories a) Non-pedestrians among the pedestrians and b) Pedestrians moving into unusual regions. The aim is to detect abnormality in a frame and localize the corresponding regions.

Similar to Mahadevan et al. [66], evaluation is performed on Ped1 for two aspects: frame level anomaly detection and pixel level anomaly detection (anomaly localization), whereas only frame level anomaly detection for Ped2. a) Frame Level Anomaly: A frame is anomalous (true positive) if at least one pixel of the frame is detected anomalous in comparison to a certain threshold. The threshold is subsequently varied to determine ROC curve. b) Pixel Level

\textsuperscript{1}available at: http://www.videolan.org/developers/x264.html
Anomaly: On the other side, anomaly localization is measured by comparing the detected anomaly of the given algorithm with respect to the ground truth anomalous region for each frame. If at least 40% of the truly abnormal locations are detected, then the frame is considered to be anomalous (True positive). Whereas, a frame is considered false positive if ground truth indicates it to be normal but one or more of its pixels are detected as anomalous. The decision threshold is varied to obtain anomaly localization ROC curve, as mentioned above.

UMN dataset consists of single video with 11 sequences of 3 different scenes of frame size 320 × 240, where an abrupt crowd anomaly occurs at the end of each sequences. The first scene has 2 sample sequences, whereas second and third scenes have 6 and 3 sample sequences each. The basic intention is to detect abnormal frames. The ground truth abnormal frames are marked through tags at the upper left corner.
Figure 4.7: Results on Ped1 and Ped2. Anomalies are marked as anomalies.

4.4.2 Quantitative Performance Analysis

The proposed algorithm is capable of achieving real-time prediction by compromising some accuracy. The number of pyramidal levels generated differs from videos to videos but regulated by the restriction on size of the coarsest level to a minimum of $30 \times 30$. Other parameters are set for different datasets are explained below.

*Ped1 and Ped2*: Typically, the statistics (motion magnitude and orientation histograms) are different for each location $p$ at a particular pyramidal level $L$, so varying $\tau_L(p)$ is defined using a constant $\gamma_L$ (Eq. (4.6)). Currently, the proposed algorithm uses two levels of pyramids for Ped1 and Ped2, where coarse level is obtained by reducing the finer level by a fraction of 4. Thus, there are two decision thresholds $\gamma_{L1}$ and $\gamma_{L0}$. The anomaly evaluation is performed by varying $\gamma_{L0}$ to obtain receiver operating characteristic (ROC). This evaluates the effect of $\gamma_{L0}$ on the anomaly detection. But to evaluate effect $\gamma_{L1}$ on the output, we studied the variation
Table 4.1: Equal Error Rate (EER) on Ped Data-sets

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Ped 1</th>
<th>Ped 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF [70, 66]</td>
<td>31%</td>
<td>42%</td>
</tr>
<tr>
<td>MPPCA [52, 66]</td>
<td>40%</td>
<td>30%</td>
</tr>
<tr>
<td>SF-MPPCA</td>
<td>32%</td>
<td>36%</td>
</tr>
<tr>
<td>MDT [66]</td>
<td>25%</td>
<td>25%</td>
</tr>
<tr>
<td>Sparse [34]</td>
<td>19%</td>
<td>-</td>
</tr>
<tr>
<td>LSA [92]</td>
<td>16%</td>
<td>-</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td>24.77%</td>
<td>21.23%</td>
</tr>
</tbody>
</table>

Table 4.2: Rate of Detection (RD), Area Under the curve (AUC) and Detection Rate for Ped 1

<table>
<thead>
<tr>
<th>Approaches</th>
<th>RD</th>
<th>AUC</th>
<th>Detection Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF [70, 66]</td>
<td>21%</td>
<td>17.9%</td>
<td>-</td>
</tr>
<tr>
<td>MPPCA [52, 66]</td>
<td>18%</td>
<td>20.5%</td>
<td>-</td>
</tr>
<tr>
<td>SF-MPPCA</td>
<td>18</td>
<td>21.3%</td>
<td>-</td>
</tr>
<tr>
<td>MDT [66]</td>
<td>45%</td>
<td>44%</td>
<td>0.04 fps</td>
</tr>
<tr>
<td>Sparse [34]</td>
<td>46%</td>
<td>46.1%</td>
<td>0.25 fps</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td>44.08%</td>
<td>40.06%</td>
<td>70 fps</td>
</tr>
</tbody>
</table>

The proposed algorithm has achieved frame-level anomaly detection of equal error rate (EER) 24.77% on Ped1 (Fig. 4.6(a)) and 21.23% on Ped2 (Fig. 4.6(b)), which is comparable to Cong et al. [34] and MDT [66](Tab. 6.2), with better localization of anomaly than any existing methods. This is demonstrated by rate of detection of 44.08% on Ped1 (Fig.4.6(c)). Refer Tab. 6.3 for comparisons. Additionally, it is able to reduce computational complexity resulting in...
<table>
<thead>
<tr>
<th>$\gamma_{L1}$</th>
<th>EER</th>
<th>Detection Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.999</td>
<td>0.999</td>
<td>30.47</td>
</tr>
<tr>
<td>0.99</td>
<td>24.77</td>
<td>70 fps</td>
</tr>
<tr>
<td>0.98</td>
<td>24.99</td>
<td>65 fps</td>
</tr>
<tr>
<td>0.95</td>
<td>25.30</td>
<td>58 fps</td>
</tr>
</tbody>
</table>

Table 4.3: Variation of Equal Error Rate with varying $\gamma_{L1}$

real time detection. We have achieved around 70 frames per sec for 480 × 720 resolution video.

In comparison to 0.04 frames per sec by MDT [66], and comparison to 0.25 frames per sec by sparse approach [34], we have achieved around 1700 × speedup and 250 × speedup respectively. Few of the sample results are shown in Fig. 6.7.

Figure 4.9: Frames of UMN data-set wrongly marked a) Abnormal frame marked as normal frame b) Normal frame marked as abnormal frame

**UMN data-set**: Before discussing the experiments on UMN data-set, we observed some interesting points about UMN data-set sequences and its abnormal frame marking. In Fig. 4.9a, though abnormality has already begun but the ground truth indicates otherwise. Additionally, in Fig. 4.9b, the scene is empty (static frame), whereas ground truth indicates abnormality. On evaluating the algorithm to detect global anomaly detection on UMN data-set, we observed that the proposed algorithm is able to detect all the abnormality but by a shift, which was expected because of rationale discussed earlier. Since the feature extraction of the proposed algorithm depends on the motion vectors, the algorithm fails to detect anomaly if there is no motion. We thus modified the ground truth by considering static frames as normal and accommodating other aforementioned reasons. There are 11 video sequences in UMN dataset. For training we have used first sequence of each scene and left the remaining sequences in the same scene for testing. For learning usual behavior model, we have only relied upon normal frames and neglected the abnormal frames. The evaluation is done on each scene independently. Interestingly, each of these sequences have movements only corresponding to particular region
while training but tend to move in other regions as well during testing. On comparing our results with original ground truth we have achieved ROC curve with area under the curve (AUC) 68.67%, but with corrected ground truth we achieved AUC of 94.28%. (Refer Fig. 4.11 and Tab. 4.4). Since, resolution of UMN dataset is low, we have processed only at finer level ($L0$). Even then, we achieved around 70 frames per sec (a speedup of 90× compared to sparse approach [34]). Fig. 4.10 shows comparison on UMN-dataset. The occurrence of few false positives are due to unavailability of training samples at unused regions during training. Few of detection results on UMN dataset are shown in Fig. 4.12.

![Figure 4.10: Result comparison on UMN data-set: Labels of each test frame a) Original GT bar b) Modified GT bar c) Actual detection bar; Green: Normal frame, Red: Abnormal frame](image)

Figure 4.10: Result comparison on UMN data-set: Labels of each test frame a) Original GT bar b) Modified GT bar c) Actual detection bar; Green: Normal frame, Red: Abnormal frame

![Figure 4.11: ROC curve for UMN data-sets a) Based on Original Ground Truth b) Based on Modified Ground Truth](image)

Figure 4.11: ROC curve for UMN data-sets a) Based on Original Ground Truth b) Based on Modified Ground Truth

### 4.5 Conclusion

We have proposed a compressed domain approach in H.264/AVC framework to detect anomalies in surveillance videos. The training phase involves local modeling of the motion vector magnitude and orientation at various locations across multiple resolutions using motion pyramid. The novel usage of motion pyramid enables the proposed approach to bypass much of
<table>
<thead>
<tr>
<th>Approaches</th>
<th>ROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Force [70]</td>
<td>96</td>
</tr>
<tr>
<td>Nearest Neighbor [34]</td>
<td>93</td>
</tr>
<tr>
<td>Sparse [34]</td>
<td>97.8</td>
</tr>
<tr>
<td>Ours Without GT correction</td>
<td>68.67</td>
</tr>
<tr>
<td><strong>Ours - With GT correction</strong></td>
<td><strong>94.28</strong></td>
</tr>
</tbody>
</table>

Table 4.4: Comparative results with other approaches on UMN

Figure 4.12: Results on UMN data-set (Yellow colored Frames are detected as abnormal Frames)

Due to this bypassing, the proposed method achieves huge reduction in computation time with having on-par detection accuracy to many recent approaches.
Chapter 5

Anomaly Detection via Local Dictionary Enhancement

5.1 Introduction

In comparison to motion vectors of a compressed video, motion in pixel level processing is mainly estimated using optical flow. Optical flow field provides accurate information regarding the pixel motion across two successive frames. Due to its ability to define motion information for each pixels, optical flow is highly accurate in representing true event motion captured by video. It is due to this accuracy in representing the true object motion, optical flow are being used extensively in variety of video content analysis problems such as video classification, crowd flow segmentation, video anomaly detection, tracking, etc.

The rest of the chapter presents a proposed novel approach in motion based anomaly detection in videos. The main contribution is in transferring the usual behaviour dictionary from neighbouring regions to the current location through appropriately learned transformations. Section 5.2 presents related works with respect to pixel-level anomaly detection with more focus on video features such as Histogram of Optical Flow (HoF). Section 5.3 presents the proposed framework followed by the details of the proposed algorithm in Section 5.4. To further demonstrate the effectiveness of the proposed algorithm, experiments were conducted on the widely used datasets as described in Section 5.5. This is followed by Section 5.6 that summarizes the proposed approach in the end.
5.2 Related Work

According to Ce Li et al. [59], all the anomaly detection algorithms can be coarsely classified into two categories, namely trajectory analysis and spatio-temporal features based analysis such as optical flow, etc. In trajectory analysis [59, 76, 94, 85], usual behavior is learned through tracking of normal objects/persons and interaction of the tracked objects/person. On the other side, video feature based analysis [10, 28, 66, 70] involves abnormality detection based on the feature of a space-time cube. Due to the features used the proposed approach can be classified as as spatio-temporal feature based algorithm in the mentioned taxonomy. However, the focus of the proposed algorithm is to improve the anomaly detection by better modeling of regular behavior in sparse dictionaries.

One of the earlier anomaly detection works involving low level video features was proposed by Itti and Baldi [48]. The authors used Poisson modeling of feature descriptors computed at every location. But, due to the lack of temporal cues the algorithm fails in capturing temporal behaviors. More recent algorithms such as [53, 66] started harnessing the spatio-temporal consistency of the video to remove any false localized detections. Varadarajan et al. [99] explored topic models for anomaly detection by modeling recurrent activities in long video sequences. Whereas, Saligrama et al. [92] assume anomaly has significant local spatio-temporal signatures that occur for a very small interval and compute the likelihood based on the similarity of a space-time cube with respect to its spatial and temporal neighbors.

The most similar work to the proposed approach is that of Cong et al. [34]. The authors used sparse reconstruction cost (SRC) to evaluate the normalness of the testing sample using a local dictionary (which is a set of local spatio-temporal motion patches obtained from training normal videos). However, the proposed approach is aimed at enhancing this dictionary for a region. This is done by applying appropriately learned transformations on the neighbouring region dictionary.

5.3 Proposed Framework

Anomaly is defined as departure from usual statistics as defined in Sec. 4.3. In other terms, given a set of usual $d$-dimensional data $\mathcal{X} = \{x|x \in \mathbb{R}^{d \times 1}\}$, a candidate $y \in \mathbb{R}^{d \times 1}$ is abnormal if it deviates from usual pattern $\mathcal{X}$. The amount of abnormality ($e$) can be defined in terms of measure of deviation/distance. Mathematically,

$$e \propto \text{dist}(\mathcal{X}, y)$$
where, \( \text{dist}(\bullet, \bullet) \) indicates the deviation/distance between the test data \( y \) and usual pattern \( \mathcal{X} \).

The deviation can be represented in many forms. One of the recent approaches represents the given candidate data as sparse linear combinations of the optimal subset of usual features referred as ‘dictionary’ \( (\mathcal{D} \in \mathbb{R}^{d \times m}) \), where \( m \) is the number of atoms in the dictionary.

\[
\mathcal{D} = f(\mathcal{X})
\]  

(5.1)

where, \( f(\bullet) \) is a dictionary learning function on usual data \( \mathcal{X} \).

The anomaly can be defined in terms of reconstruction error that indicates the measure of deviation. The reconstruction error of any abnormal test sample \( y \), using the corresponding dictionary, would result in large error compared to that of usual test samples.

\[
e \propto \| y - \mathcal{D}\alpha \|_2
\]  

(5.2)

where, \( \alpha \in \mathbb{R}^{m \times 1} \) denotes the sparse coefficients. \( e \) is the reconstruction error which indicates the measure of anomaly.

Even though the solution is simple, it is restricted by the learning of appropriate dictionary \( \text{(Eq. (5.1))} \) that depends on the amount of data available for learning. In this work, we propose to enhance the currently learned dictionary by appending dictionaries learned with another set of similar data. Typically, different sets of data are related under some constrained transformation as in Eq. \( (5.3) \).

\[
\tilde{\mathcal{X}}_j = \Psi_{ji} \mathcal{X}_i
\]  

subject to \( \tilde{\mathcal{X}}_j \tilde{\mathcal{X}}_j^T = \tilde{\mathcal{X}}_j \tilde{\mathcal{X}}_j^T \)

(5.3)

where, \( \Psi_{ji} \) is the transformation from data \( \mathcal{X}_i \) to data \( \tilde{\mathcal{X}}_j \). Each candidate sample is a linear combination of the corresponding dictionary. Thus, we can consider the dictionaries and underlying data to be related by the same transformation \( \text{(Eq. (5.4))} \) for a given distribution of coefficients as below.

\[
\mathcal{X}_i \approx \mathcal{D}_i \alpha
\]

\[
\Psi_{ji} \mathcal{X}_i \approx \Psi_{ij} \mathcal{D}_i \alpha
\]

\[
\tilde{\mathcal{X}}_j \approx \tilde{\mathcal{D}}_j \alpha
\]

where, \( \tilde{\mathcal{D}}_j = \Psi_{ji} \mathcal{D}_i \)

(5.4)
Thus, transformed dictionary ($\tilde{D}$) enhances the existing dictionary ($D$) and helps in better representation of the normal pattern. Subsequently, the reconstruction error ($e$) can be expressed by the weighted combination original and enhanced dictionary.

$$e \propto w_1 \ast \|y - D\alpha\|_2 + w_2 \ast \|y - [D, \tilde{D}]\alpha\|_2$$

(5.5)

where, $w_1$ and $w_2$ are weights associated with the reconstruction error from original dictionary and enhanced dictionary (concatenation of original and transformed dictionary i.e. $[D, \tilde{D}]$) respectively, with following constraint: $w_1 \leq w_2$. Reconstruction error is the weighted average of both original dictionary and enhanced dictionary to measure the error with a higher penalty of reconstruction error to enhance dictionaries compared to original dictionary.

## 5.4 Detailed Algorithm

The proposed approach detects anomaly based on the reconstruction error by solving $l_1$ minimization by sparsely representing a candidate using a dictionary of usual behavior. Typically, the behavior change widely depends upon factors such as layout of the scene, distance from camera, topology of paths, etc. For instance, the flow magnitude near to the camera is different from that far from the camera. Hence, it is not suitable to have a global dictionary for the entire scene. The issue can be resolved by dividing the space into homogeneous motion regions and then solving $l_1$ minimization problem independently based on dictionaries learned at each location. As it makes the proposed algorithm highly sensitive to location information, we have proposed a local dictionary for each $k \times k$ size block representing a larger region of the frame (Fig. 5.1).

The proposed approach constitutes the following four main stages: a) Descriptor extraction
b) learning data transformation c) local dictionary formation and enhancement d) anomaly detection.

5.4.1 Descriptor extraction

Descriptors play an important role in anomaly detection. Extracting descriptors based on the type of anomaly can effectively help in anomaly detection. For our experiments, we have used histogram of optical flow (HOF), motion magnitude and foreground pixel occupancy for ‘dense’ space time cube of $m \times m \times n$. The descriptors are explained in detail below:

5.4.1.1 Foreground pixel occupancy

Foreground pixels are obtained using the method [7] which performs background subtraction. Foreground pixel occupancy in the space time cube indicate the importance of the space time cube in capturing the foreground object.

5.4.1.2 Histogram of optical flow (HOF)

Histogram of optical flow provide an important indication about the motion pattern in a space-time cube. Thus, we have used $l_1$ normalized HOF descriptor where flow information is quantized into $p$ bins ($p = 10$). The algorithm proposed by Liu [61] is used for estimating the optical flow. Each bin contains flow information corresponding to foreground region only.

5.4.1.3 Motion magnitude

Motion magnitude can directly indicate anomaly. For example, even though the directional flow for a skater and a walking person at a location are similar, but they differ drastically in flow magnitude. Thus, an additional bin is introduced for mean magnitude flow corresponding to foreground region in a space-time cube. The mean magnitude is further normalized based on the median statistics in the $k \times k$ block (Fig. 5.1).

Thus, we have a 12-dimensional descriptor for each space-time cube. Subsequently, we $l_2$ normalize the descriptor prior to transformation learning and subsequent dictionary enhancement.

5.4.2 Learning data transformation across neighboring blocks

The local usual dictionary is learned on data samples/descriptors at each block. But, the data samples at current block are related to data samples of other blocks under some constrained
transformations such as Eq. (5.3).

Consider, two blocks $i$ and $j$ with corresponding $l_2$ normalized data $\mathbf{X}_i \in \mathbb{R}^{d \times m}$ and $\mathbf{X}_j \in \mathbb{R}^{d \times n}$. If the data samples of both blocks are $l_2$ normalized that is required to preserve dot product among different data samples, the transformation matrix $\Psi_{ji} \in \mathbb{R}^{d \times d}$ should have the following properties:

- $\det(\Psi_{ji}) = 1$ or $-1$
- $\Psi_{ij} = \Psi_{ji}^{-1}$

where, $d = 12$ is the dimension of the features as described earlier.

As,

$$\mathbf{X}_j \mathbf{X}_j^T = \tilde{\mathbf{X}}_j \tilde{\mathbf{X}}_j^T$$

$$\mathbf{X}_j \mathbf{X}_j^T = (\Psi_{ji} \mathbf{X}_i)(\Psi_{ji} \mathbf{X}_i)^T$$

On Eigenvalue decomposition,

$$\Omega_j \Lambda_j \Omega_j^T = \Psi_{ji} \Omega_i \Lambda_i \Omega_i^T \Psi_{ji}^T$$

(5.6)

where, $\Omega_i$ and $\Omega_j$ are eigenvectors of $\mathbf{X}_i \mathbf{X}_i^T$ and $\mathbf{X}_j \mathbf{X}_j^T$ respectively.

If two blocks $i$ and $j$ have similar variation in data $\mathbf{X}_i$ and $\mathbf{X}_j$, then the corresponding eigenvalues $\Lambda_i$ and $\Lambda_j$ would be highly similar. Thus,

$$\Omega_j = \Psi_{ji} \Omega_i$$

$$\Psi_{ji} = \Omega_i \Omega_j^{-1}$$

(5.7)

subject to $\|\Lambda_i - \Lambda_j\|_2 \leq \epsilon$

So, the best transformation is achieved between two blocks if $\|\Lambda_i - \Lambda_j\|_2$ is minimum and below some threshold $\epsilon$. We compute $\|\Lambda_i - \Lambda_j\|_2$ between block $j$ and neighboring block $i$, where $j = i + h$ and $h$ being the spatial neighborhood offset. Typically, $\|\Lambda_i - \Lambda_j\|_2$ is very small in neighborhood and increases across blocks placed far apart.

For optimal dictionary creation, we have used Mairal et al. [67] dictionary learning algorithm which tries to learn dictionary by optimizing Eq. (5.8). However, the proposed approach is
suitable for any dictionary learning algorithm with $l_2$ normalized atoms.

$$\min_{D \in C, \alpha \in \mathbb{R}^{d \times n}} \frac{1}{n} \sum_{i=1}^{n} \left( \frac{1}{2} \|x_i - D\alpha_i\|_2^2 + \lambda \|\alpha_i\|_1 \right)$$  \hspace{1cm} (5.8)$$

where, $D$ is the learned optimal dictionary among all matrices $C$. $x_i$ is the $l_2$ normalized $i^{th}$ data sample in $X$ and $\alpha_i$ are its sparse coefficients over dictionary $D$.

The learned dictionary is suitable to represent the underlying data in optimal way, but fails to cater the need of unknown normal variation which might not be present in the training sample in current block but occurs in the neighboring blocks. We enhance the current dictionary by learning the transformation ($\Psi$) based on the similarity of the underlying data. The transformed dictionary, defined as $\tilde{D}_j = \Psi_{ji}D_i$ in Eq. (5.4), is also used to find the reconstruction error.

![Video Sequence Ped1 Test 001](image1.png)

![Video Sequence Ped1 Test 019](image2.png)

(a) Video Sequence Ped1 Test 001
(b) Video Sequence Ped1 Test 019

Figure 5.2: Representative results for different video sequences where each row is from a single sequence. The anomalies are marked in ‘Red’ color. (Best viewed in color) (More results available at http://val.serc.iisc.ernet.in/AnomalyResultsICIP14)

### 5.4.3 Anomaly detection

The local dictionary and enhanced dictionary both contain the normal behavior for a localized block. The anomaly is now defined using reconstruction error ($e$) of the new candidate feature ($y$) in a given block with respect to both dictionaries. We compute the sparse coefficients through solving

$$\min_{\alpha} \frac{1}{2}\|y - D\alpha\|_2^2 + \lambda_1\|\alpha\|_1 + \frac{\lambda_2}{2}\|\alpha\|_2^2$$  \hspace{1cm} (5.9)$$

where, $\lambda_1$ and $\lambda_2$ are two regularizing parameters. $D$ is the concatenation of $D$, $\tilde{D}$ and $T$ dictionaries. $T$ is the trivial dictionary consisting of $[I - I]$. Extending the reconstruction error
defined in Eq. (5.5), we compute the final reconstruction error \( e \) as:

\[
e = w_1 \| y - D\alpha \|_2 + w_2 \| y - [D; \tilde{D}]\alpha \|_2 + w_3 \| \alpha_T \|_1
\]  

(5.10)

where, \( w_1, w_2 \) and \( w_3 \) are weights such that \( w_3 \leq w_1 \leq w_2 \). And, \( \alpha_T \) denotes the coefficients corresponding to trivial dictionary only.

We have used a combination of atoms from original dictionary, transformed dictionary and trivial dictionary to compute the sparse coefficient in Eq. 5.9. However, the final anomaly measure, which is based on error computation/deviation in the Eq. 5.10, contains both the original and enhanced dictionaries. This is done to differentiate the motion pattern that could be exactly reconstructed using original dictionary from those patterns that require extra information for better reconstruction. In an ideal scenario, transformed dictionaries are not required for anomaly detection as the original dictionary has all the possible variations. But in real scenarios, due to lack to appropriate training samples, the normal behaviour model needs to be boosted by enhancement during training like the proposed approach. But, enhancement by the proposed dictionary transformation may introduce unwanted behaviours patterns which reduce the ability to detect certain abnormalities. Providing equal weight for original and enhanced dictionary based error computation/deviation for patterns would then reduce the discrimination between normal behaviour and those abnormalities. However, the current equation as suggested maintains this relative difference.

5.5 Experiments

The focus of the proposed approach is on detection of anomaly in a video through enhancing the local dictionary by learning transformations across neighbors. We demonstrate the effectiveness of the approach through evaluating the algorithm on two widely used anomaly datasets named UCSD Ped1 and Ped2 [66] on their original frame size of 238 × 158 and 360 × 240 respectively. We have performed all the experiments on MATLAB (with mex implementation for optical flow computation) on single core 3.4 GHz Intel i7 processor with 8GB RAM.

5.5.1 Implementation details

For experiments, the space time cube for feature extraction is empirically set to a size of 10 × 10 × 7 for Ped1 and 15 × 15 × 7 for Ped2.

We have kept the number of the atoms in dictionary as 50 for a localized block of 10 × 10
for both dataset.

![ROC curves for different approaches](image)

**Figure 5.3:** a,b) Performance of the different approaches tested for the frame-level anomaly detection on the Ped1 and Ped2 respectively. c) Frame level comparison with and without dictionary enhancement on Ped2 d) Pixel-level comparison on Ped1

### 5.5.2 Quantitative evaluation

The proposed algorithm has achieved frame-level anomaly detection with area under curve (AUC) of 85.85% and equal error rate (EER) of 19.22% on Ped1 (Fig. 5.3a and Tab. 5.1). Even without employing spatio-temporal basis to detect anomaly, the proposed algorithm performs...
Table 5.1: Frame Level Comparison: Equal Error Rate (EER) on Ped data-sets

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Ped1</th>
<th>Ped2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF [70, 66]</td>
<td>31%</td>
<td>42%</td>
</tr>
<tr>
<td>MPPCA [52, 66]</td>
<td>40%</td>
<td>30%</td>
</tr>
<tr>
<td>SF-MPPCA</td>
<td>32%</td>
<td>36%</td>
</tr>
<tr>
<td>MDT [66]</td>
<td>25%</td>
<td>25%</td>
</tr>
<tr>
<td>Sparse [34]</td>
<td>19%</td>
<td>-</td>
</tr>
<tr>
<td>LSA [92]</td>
<td>16%</td>
<td>-</td>
</tr>
<tr>
<td>Ours (No Enhancement)</td>
<td>19.53%</td>
<td>21.26%</td>
</tr>
<tr>
<td>Ours (With only enhanced dictionary)</td>
<td>19.27%</td>
<td>-</td>
</tr>
<tr>
<td>Ours (With both dictionary)</td>
<td>19.22%</td>
<td>20.38%</td>
</tr>
</tbody>
</table>

Table 5.2: Pixel Level Comparison on Ped 1: Rate of detection (RD), Area under the curve (AUC) and Detection speed

<table>
<thead>
<tr>
<th>Approaches</th>
<th>RD</th>
<th>AUC</th>
<th>Detection speed (frame per sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF [70, 66]</td>
<td>21%</td>
<td>17.9%</td>
<td>-</td>
</tr>
<tr>
<td>MPPCA [52, 66]</td>
<td>18%</td>
<td>20.5%</td>
<td>-</td>
</tr>
<tr>
<td>MDT [66]</td>
<td>45%</td>
<td>44%</td>
<td>0.04 fps</td>
</tr>
<tr>
<td>Sparse [34]</td>
<td>46%</td>
<td>46.1%</td>
<td>0.25 fps</td>
</tr>
<tr>
<td>Ours (With Enhancement)</td>
<td>51.02%</td>
<td>50.63%</td>
<td>~ 3 fps</td>
</tr>
</tbody>
</table>

competitively with Local statistical aggregates based anomaly detection [92] and Sparse [34]. To quantify localization of anomaly, Mahadevan et al. [66] proposed that a frame is considered abnormal if atleast 40% of the anomalous region is detected. The proposed algorithm achieves 50.63% AUC and 51.02% Rate of Detection (RD), outperforming other recent anomaly detection algorithms in pixel-level accuracy as shown in Fig. 5.3d and Tab. 5.2.

On Ped2, the proposed approach performs with EER of 20.38% and area under ROC of 85.95% as depicted in Fig. 5.3b and Tab. 5.1, outperforming other algorithms evaluated on Ped2. The proposed approach of dictionary enhancement is also compared with regularly learned local dictionary without changing the parameters (Fig. 5.3c).

5.5.3 Computation complexity

The computational complexity of the fastest $l_1$ minimization algorithm is $O(kM^2 + kMn)$ [113] where $M$ is the number of atoms in the dictionary; $n$ is the dimension of the feature and $k$ is the
number of iterations before convergence (ie. upper limit for number of iterations in real applications). We have used $M = 50$ and $n = 12$ in our experiments. But, with respect to dictionary enhancement, the additional computations are computing the eigenvalue decomposition in Eq. 5.6 ie $O(n^3)$ and $l_1$ minimization with learned dictionary atoms ie additional $O(kM^2 + kMn)$. Thus, the new complexity is $O(k(2M)^2 + k(2M)n + n^3) = O(4kM^2 + 2kMn + n^3)$ which is a combination of computations for solving Eq. 5.9 and computations for eigenvalue decomposition in Eq. 5.6.

5.6 Conclusion

In this chapter, we have proposed a spatio-temporal feature based algorithm using sparse representation framework to detect anomalies in surveillance videos. The major contribution of the proposed approach lies in the enhancement of dictionaries containing usual behavior features. The enhancement of the local dictionaries was performed by appending regular local dictionaries along with transformed dictionaries that were obtained from underlying transformation between sets of usual features across neighboring regions. The experiments conducted on widely used datasets UCSD Ped1 and Ped2 datasets, demonstrate better anomaly detection using enhanced dictionary compared to regularly learned local dictionary.
Chapter 6
Anomaly Detection via Short Local Trajectory

6.1 Introduction

In comparison to video feature based abnormality detection, trajectories are able to capture temporal dynamics effectively. This was shown by Hu et al. [44] in their work where the authors introduced Hierarchical self-organizing neural network model to learn the motion patterns of single moving object for anomaly detection and object behavior prediction. Another single object trajectory based abnormal action was proposed by Porikli et al. [78]. The authors characterized coordinates, orientation and speed of a moving object using Hidden Markov model (HMM) parameters to measure the similarity between two trajectories and subsequently used clustering algorithm to extract multiple clusters of the event. Moving from single object anomaly detection, Zhang et al. [119] clustered trajectories extracted through tracking to identify primitive events. The authors proposed Minimum Description Length (MDL) principle based grammar induction algorithm on these trajectories, which handled the problem of variable length trajectory, to infer rules for the primitive events. Similarly, single-class Support Vector Machine (SVM) based fixed length trajectory clustering was proposed by Piciarelli et al. [76]. Any candidate trajectory forming an outlier to the existing trajectory clusters is considered as anomaly. More recently, Sparse Reconstruction Analysis (SRA) [59] was used for candidate trajectories for comparing with accumulated normal trajectories as dictionary. The minimal residual was used to detect the anomaly.

Though these trajectory based algorithms capture the motion information efficiently, they lack in capturing the shape information to localize the detected anomalies. Further, the tra-
Trajectories are usually generated by tracking the moving objects. Thus the tracking algorithm is required to be robust across various challenges such as crowd densities, occlusion, etc. To overcome these problems, we have proposed a super-pixel based short trajectory generation approach that captures both shape and motion information for anomaly detection and localization.

![Figure 6.1: Comparison of Short Local Trajectories (SLT) of normal and abnormal behaviors on UCSD Ped1 (Marked by blue). Normal trajectories are marked in green where as abnormal trajectories are in red. The trajectories of both objects are distinct even when they are in same vicinity.](image)

### 6.2 Problem Formulation

Consider a random vector $y = (x_1, \ldots, x_n)^T$, defined on $\mathcal{F} \in \mathbb{R}^n$. This random vector is an instantaneous information captured at any time instant $t$ and is denoted as $y_t$. We define a random event as sequence of such random vectors $\mathcal{T} = \{y_t-(k-1), \ldots, y_t\}$, where $k$ is the number of past observations considered. The sequence $\mathcal{T}$ incorporates the temporal information of the event.

Anomaly is defined as a random event whose likelihood of occurrence is less than certain threshold. Conversely, a random event is anomalous if probability of abnormality of the event is higher than certain threshold i.e.,

$$P(\mathcal{T}) \leq \tau \quad (6.1)$$

$$P(\overline{\mathcal{T}}) = 1 - P(\mathcal{T}) \geq 1 - \tau = \tau_2 \quad (6.2)$$
Figure 6.2: Trajectory generation stages: original frame, super-pixel output using ERS (Top-Right) [62] and trajectories generated for only moving objects (Bottom-Left).

where, $P(\mathcal{F})$ and $P(\bar{\mathcal{F}})$ denote the probability of normality and abnormality of the event respectively and $\tau_2 = 1 - \tau$ is the decision threshold.

6.3 Proposed Approach

With the above formulation in mind, the proposed anomaly detection algorithm is formulated as combination of two stages. The first stage computes abnormality based on Hidden Markov Models (HMM) learned on normal training samples. Though most anomalies are captured at this stage, they result in few false detections due to sensitivity of HMM towards noisy observations. The effect of the noisy observations is overcome by measuring spatial consistency of a trajectory with respect to its spatial neighbors at second stage. The final detection is the combination of the both stages.

6.3.1 Descriptor - Short Local Trajectories

Trajectories or tracklets are created through tracking of moving objects [120, 91]. Though trajectories capture the dynamics of the object, they are limited by the shortcomings of the associated tracking algorithms. The major challenges involved include performance of tracking algorithm across different crowd densities and robustness of the tracking algorithm during occlusions, illumination and scale changes. To overcome these issues of trackers, trajectories
are computed for interest points across multiple frames [102]. Though trajectories generated using interest points are better, they fail for textureless regions due to lack of interest points. Furthermore, each trajectory is generated for a single pixel making it unsuitable for localization of the moving objects.

In this work, we have computed the trajectories at frame $t$ by traversing the median flow of a region (represented by a super-pixel) in reverse order up to past $k$ frames as explained in Eq. (6.4). The key criterion to obtain a correct trajectory is to have a coherent movement for a defined region. Regions with multiple motions result in erroneous trajectory extraction. Instead of using blocks/patches to define the region, we have used super-pixels [62], which are sets of characteristically similar pixels and are highly probable to have similar motion irrespective of crowd density in the video. Further improvement to trajectories is performed by discarding the regions that do not belong to foreground/moving objects and restricting the trajectory generation to foreground regions (Refer Fig. 6.2). The background is modeled as median of all the training frames (Eq. (6.3)).

$$B(x, y) = \tilde{I}(x, y, t_0 - t_n)$$
$$F(x, y, t) = ||I(x, y, t) - B(x, y)|| > Th$$ (6.3)

where, $I(x, y, t)$ and $F(x, y, t)$ denote pixel value and foreground mask at location $(x, y)$ at time $t$. $\tilde{I}$ denotes temporal median of the frames. $B(x, y)$ denotes the background model.

Let the region occupied by any foreground super-pixel ($s_i$) at frame $t$ be $r^i_t$.

$$r^i_t = \{x : \forall x \in s_i(t)\}$$
$$V_t(r^i_t) = \begin{bmatrix} \tilde{u}_t(r^i_t) \\ \tilde{v}_t(r^i_t) \end{bmatrix}$$
$$r^i_{t-1} = r^i_t - V_t(r^i_t)$$
$$\mathcal{T}^h_{s_i}(t) = (\tilde{r}^i_{t-(k-1)}, \ldots, \tilde{r}^i_t)$$
$$\mathcal{T}^f_{s_i}(t) = \left( \frac{V_{t-(k-1)}(r^i_{t-(k-1)})}{M_{t-(k-1)}(r^i_{t-(k-1)})}, \ldots, \frac{V_t(r^i_t)}{M_t(r^i_t)} \right)$$ (6.4) (6.5)

where, $r^i_{t-1}$ denotes the possible region occupied by super-pixel $s_i(t)$ in previous frames. $\tilde{r}^i_t$ refers to the centroid of the region occupied by $s_i$ at frame $t$ and $\tilde{r}^i_{t-(k-1)}$ denotes centroid of the possible region occupied at frame $t - (k - 1)$. $\tilde{u}_t$ and $\tilde{v}_t$ denote median of $x$ and $y$ components of optical flow at frame $t$. $V_t(r^i_t)$ and $M_t(r^i_t)$ denote the median flow and maximum magnitude of motion obtained from training videos for a region $r^i_t$ at frame $t$ respectively. Thus, $\mathcal{T}^h_{s_i}(t)$
contains the location information, whereas $T_f^s(t)$ contains the normalized motion information for the super-pixel $s_i$, along the trajectory, at current frame $t$ from past $k$ frames. We denote $T_h^s(t)$ as $T_h^s$ and $T_f^s(t)$ as $T_f^s$ hereafter, unless explicitly mentioned otherwise. (Refer Fig. 6.3 for details)

Existing methods either form a single trajectory for an object till it exists in the scope of the video or a single tracklet for an object for some small duration of time followed by joining multiple such tracklets to form a single long trajectory. Forming such trajectories essentially requires temporal matching of super-pixels of each frame with its previous frames’ super-pixels, which can be difficult due to various occlusions and crowd interaction. Instead, we form trajectories for each super-pixel ($s_i$) at frame ($t$) independent of super-pixels in previous frame ($t-1$) irrespective of both super-pixels belonging to the same object. As shown in Fig. 6.4(a), super-pixels $S_3$ and $S_4$ belong to same region across consecutive frames but generate two independent trajectories. Due to lack of explicit matching of super-pixels across consecutive frames and presence of sub-pixel flows, the trajectory becomes highly erroneous with increase in its length. Thus, we compute short trajectories upto length $k$ and truncate the trajectories to a lesser length than $k$ if the super-pixel region has multiple flows or if the number of pixels in a super-pixel is not adequate to reliably compute the trajectories. This results in variable length trajectories with a maximum length of $k$. Furthermore, the topography of the region in video plays an important role in variation of trajectories across different locations. The trajectories of same object at different locations are different due to projection on 2D image plane (see Fig. 6.5). Since, these trajectories are small and location dependent, we call these trajectories as short local trajectories (SLTs).

These SLTs capture both the motion (Eq. (6.5)) and location information (Eq. (6.4)) of the foreground super-pixels upto past $k$ frames. Compared to earlier trajectories computed using interest points [102], SLTs inherit the shape information, through underlying super-pixels, that follow same movement resulting in better localization of the coherent moving parts and subsequent abnormality localization.

### 6.3.2 Usual Behavior Modeling - Hidden Markov Model (HMM)

As explained, SLT can be very different from SLTs of other super-pixels at different locations resulting in varying $T_f^s$. This is due to varying topography of the scene captured by the surveillance camera. For example, the motion at the left of the frame is in different direction than other regions (See Fig. 6.5(a)). To avoid the effect of such patterns, we propose individual modeling of motion information corresponding to usual SLTs belonging to a particular location.
6.3.2.1 Hidden Markov Model

A HMM model, $\lambda = (A, B, \pi)$, is computed for each location, where $A$ is the state transition matrix, $B$ is symbol output probability and $\pi$ is the prior probability of each hidden state. The model is computed using all the usual $T^{s_i}_k$ generated at a specified region. Since most of the
Figure 6.5: Comparison of trajectory similarity when the same object (marked with ellipse of blue) is at different position. a) Near b) Middle and c) Far. As can be observed, the trajectory and its properties change depending upon their position in the video.

features computed in spatial neighborhood of image depict high similarity, we have proposed a single model for each non-overlapping $5 \times 5$ block.

6.3.2.2 Generating symbol sequences for $T^f$ for HMM

The trajectory containing motion information $T^f_{si}$ is data observation sequence from $\mathbb{R}^2$. We generate symbol sequence belonging to $\mathbb{R}$ by plotting the normalized optical flow, onto a circular quantization scale as shown in Fig. 6.4(b) and the corresponding bin number is used as the symbol.

6.3.3 Anomaly Detection

During extraction of SLTs, a single object is divided into multiple super-pixels and thus multiple SLTs. As all these SLTs belong to a single object, these should be spatially and temporally consistent. But sometimes flow computations result in noisy motion which could result in wrongly perceived anomalous SLTs in a frame. These false SLTs are not consistent with their spatial neighbor SLTs. So we have proposed two stage method to compute abnormality.

6.3.3.1 Stage 1: Hidden Markov Model Based Measure

The training models learned through Hidden Markov Model (HMM) at particular location $x$ are used to find the measure of ‘normalness’. At first, $T^f_{si}$ of a SLT is extracted for individual frames. These $T^f_{si}$ are then compared with corresponding location models. Usually, the probability of ‘normalness’ is measured through $P(T^f) = P(y_{t-(k-1)}, \ldots, y_t|\lambda_x)$, where $P(y_{t-(k-1)}, \ldots, y_t|\lambda_x)$.
is computed as

\[
P(y_t, \ldots, y_{t-(k-1)} | \lambda) = \sum_{q_t} P(y_t, \ldots, q_{t-(k-1)}, q_t | q_{t-(k-1)}, \ldots, q_{t-1}, \lambda) \times P(q_{t-(k-1)}, \ldots, q_t | \lambda)
\]

(6.6)

where, \( \mathcal{Q} \) denotes all the possible hidden sequences and \( q_{t-(k-1)}, \ldots, q_t \) denotes the associated hidden state sequence. But, since the length of the SLTs are different, the probability estimate vary for different SLTs. So, we compute normalized estimate of the normalness as the probability of current observation conditioned upon the previous observations and underlying HMM:

\[
P(y_t, \ldots, y_{t-1}, \lambda) = \frac{P(y_t, \ldots, y_{t-(k-1)}, \ldots, y_{t-1} | \lambda)}{P(y_t, \ldots, y_{t-1} | \lambda)}
\]

(6.7)

Anomalies could happen anywhere along the SLT and needed to be captured. As Eq. (6.7) captures the estimate of normalness given past behavior, we divide the SLT into its sub-trajectories and compute their normalness. The minimum measure among sub-trajectories correspond to unusual behavior along the SLT. For example, a skater (an anomaly) can stop skating and start walking (normal behavior) but should still be considered as an anomaly. So, the estimate of normalness for \( T^f \) to capture the temporal behavior is modified as:

\[
P_1(T^f_{s_i}) \approx \min(P(\{S^f_{s_i}\})) = \min(\frac{P(y_t, \ldots, y_{t-(k-1)}, \ldots, y_{t-1}, \lambda_x)}{P(y_t, \ldots, y_{t-k-2}, y_{t-(k-1)}, \lambda_x)})
\]

(6.8)

where, \( \{S^f_{s_i}\} \) is the set of sub-trajectories of \( T^f_{s_i} \). So based on Eq. (6.8) and Eq. (6.2), we define the abnormality as :

\[
P_1(\overline{T}^f_{s_i}) = 1 - P_1(T^f_{s_i})
\]

(6.9)

\( P_1(\overline{T}^f_{s_i}) \) is the abnormality measure obtained from 1st stage.

6.3.3.2 Stage 2: Spatial Consistency Measure

As discussed, the Stage 1 could result in false detections due to approximate flow computations and erroneous \( T^f_{s_i} \). Stage 2 removes these anomalies based on spatial consistency measure with respect to neighboring similar \( T^h_{s_i} \) predominantly belonging to same foreground object.

Since, the SLTs are computed for each super-pixel, all the SLTs belonging to a single person/object should move coherently. In other terms, consistency with respect to the neighborhood (spatial) SLTs is essential for accurate detection of anomaly. Thus, we compute the
probability of spatial consistency based on similarity of SLTs with neighborhood SLTs. The neighborhood SLTs selected for comparison are based on their position ($\Omega$) and similarity of $T_h$ these SLTs. Let $\tilde{r}_i$ be the centroid of the super-pixel $s_i$ under consideration, then:

$$\mathcal{I}_{s_j} = \{ T_{s_k}^f | \forall \tilde{r}_k \in \Omega \}$$

$$D_{ij} = \| T_h^s - T_h^s \|_2$$

$$S_{ij} = \exp(-\| \tilde{r}_i - \tilde{r}_j \|_2 - D_{ij})/\sigma$$ (6.10)

where, $\mathcal{I}_{s_j}^f$ is the set of neighboring trajectories and $\Omega$ is the neighborhood. $D_{ij}$ is the average distance between trajectory locations $T_h^s$ and $T_h^s$ in the past; $S_{ij}$ is the similarity of the trajectory $T_{s_i}^f$ with neighboring trajectory $T_{s_j}^f$. $\sigma$ is a scaling factor. The measure of spatial consistency is defined as in Eq. (6.11)

$$P_2(T_s^h) = \max(S_{ij}) \quad \text{where, } \quad i \neq j \quad (6.11)$$

$P_2(T_s^h)$ is the spatial consistency measure obtained from 2nd stage.

### 6.3.3.3 Joint Measure and Final Detection:

The Hidden Markov Model based measure indicates the abnormality in the trajectory of super-pixel, whereas spatial consistency is measured in stage 2. The final probability of the trajectory being abnormal is obtained as in Eq. (6.12)

$$P(T_s) = P_1(T_{s_i}^f) \times P_2(T_s^h)$$

$$\text{Anomaly} = \begin{cases} 
1 \text{ if } P(T_s) \geq \tau_2 \\
0 \text{ otherwise}
\end{cases} \quad (6.12)$$

### 6.3.3.4 Spatio-Temporal Filtering:

Spatio-temporal median filtering is performed as a post processing step to further reduce the noise and compute temporally consistent abnormality measure.

$$P[x, y, t] = \{ \tilde{P}[x', y', t'], (x', y', t') \in w \} \quad (6.13)$$

where, $P$ and $P'$ are the median filtered probability estimate and raw estimate, respectively. $w$ represents a neighborhood centered around location $(x, y, t)$ in the spatio-temporal cube.
6.4 Results and Analysis

The focus of the proposed approach is to localize the anomaly effectively with less false positives. We demonstrate the effectiveness of the approach by evaluating the algorithm on two widely used anomaly datasets named UCSD Ped1 and Ped2 [66]. All the experiments were performed using MATLAB (with mex implementation for optical flow and super-pixel computation) on single core 3.4 GHz Intel i7 processor with 8GB RAM.

6.4.1 Implementation Details

We have used OpenCV implementation of optical flow [38] alongwith implementation of ‘Entropy Rate Super-pixel Segmentation’ [62] for super-pixel computation. Figure 6.2 illustrates super pixels obtained using ERS. The number of super-pixels for each frame is set to 200 for both datasets.

As mentioned in Sec. 6.3, the history is limited only to \( k \) past frames. The value of \( k \) is set to 7. Even then, some trajectories at the initial frames and others at boundary might have smaller length. For example trajectories of 2\textsuperscript{nd} frame are of size 2 whereas those of later frames are longer.

6.4.2 Quantitative Evaluation

The proposed algorithm has achieved frame-level anomaly detection of equal error rate (EER) 18.37\% and area under the ROC curve (AUC) of 85.05\% on Ped1 (Fig. 6.6(a) and Tab. 6.2). Only Local Statistical Aggregate (LSA) based anomaly detection [92] performs better in frame-level anomaly detection (frame is considered to be abnormal even if one pixel is classified as abnormal) on Ped1, but Saligrama et al. [92] do not provide comparison on localization of anomaly. To quantify localization of anomaly, Mahadevan et al. [66] proposed that a detection is correct if atleast 40\% of the region in the ground truth is accurately detected. The proposed algorithm achieves 56.9\% as rate of detection, outperforming other recent anomaly detection algorithms as shown in Fig. 6.6(c) and Tab. 6.3.

On Ped2, the proposed approach achieves EER of 15.70\% and area under ROC curve of 84.75\% as depicted in Fig. 6.6(b) and Tab. 6.2, outperforming other algorithms evaluated on Ped2.

Additionally, we have achieved a speed of approximately 1 sec. for each frame on Ped1 and 1.8 sec. per frame on Ped2. This is due to the fact that the frame size of the Ped2 is more than Ped1. On an average, we are able to achieve a speedup of around \( 4 \times \) than Sparse
Reconstruction based anomaly detection (Cong et al. [34]), which takes approximately 4 sec. per frame on Ped1.

Figure 6.6: Performance of the different approaches tested for a) Frame-level anomaly detection on the Ped1 b) Frame-level anomaly detection on the Ped2 and c) anomaly localization in the Ped1
Table 6.2: Frame Level Equal Error Rate (EER) on Ped Datasets

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Ped1</th>
<th>Ped2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF[70, 66]</td>
<td>31%</td>
<td>42%</td>
</tr>
<tr>
<td>MPPCA[52, 66]</td>
<td>40%</td>
<td>30%</td>
</tr>
<tr>
<td>SF-MPPCA</td>
<td>32%</td>
<td>36%</td>
</tr>
<tr>
<td>MDT[66]</td>
<td>25%</td>
<td>25%</td>
</tr>
<tr>
<td>Sparse[34]</td>
<td>19%</td>
<td>-</td>
</tr>
<tr>
<td>LSA[92]</td>
<td>16%</td>
<td>-</td>
</tr>
<tr>
<td>Proposed</td>
<td>18.37%</td>
<td>15.70%</td>
</tr>
</tbody>
</table>

Table 6.3: Rate of Detection (RD), Area Under the curve (AUC) and Detection Rate for Ped 1.

<table>
<thead>
<tr>
<th>Approaches</th>
<th>RD</th>
<th>AUC</th>
<th>Detection Rate (frame per sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SF[70, 66]</td>
<td>21%</td>
<td>17.9%</td>
<td>-</td>
</tr>
<tr>
<td>MPPCA[52, 66]</td>
<td>18%</td>
<td>20.3%</td>
<td>-</td>
</tr>
<tr>
<td>SF-MPPCA</td>
<td>18</td>
<td>21.3%</td>
<td>-</td>
</tr>
<tr>
<td>MDT[66]</td>
<td>45%</td>
<td>44%</td>
<td>0.04 fps</td>
</tr>
<tr>
<td>Sparse[34]</td>
<td>46%</td>
<td>46.1%</td>
<td>0.25 fps</td>
</tr>
<tr>
<td>Proposed</td>
<td>56.9%</td>
<td>57.32%</td>
<td>1 fps</td>
</tr>
</tbody>
</table>

6.4.2.1 Effect of length of trajectory:

The length of the trajectory plays an important role. The result varies with respect to length of the trajectory. Very short trajectories fail to capture the anomaly over longer period where a person was abnormal in the past but changes back to normal behavior in the end. For example, a skater skating in past starts walking, is classified as normal after only few frames with very short trajectories. (Refer Tab. 6.1). Thus, with increase in length of trajectories, the anomaly detection is more robust. But, as length of the trajectory increases beyond certain value, the trajectory computation becomes erroneous resulting in wrong anomaly detections as show in Tab. 6.1.

6.4.3 Experiment on Dense Crowd Scenarios

The proposed approach is generic enough to work on various kinds of crowd scenarios. We have added another set of experiments on a newly proposed CHUK-Crowd Anomaly Dataset. This dataset is subset of CHUK crowd dataset [93] selected for anomaly detection. The type of anomalies include incorrect directional movements, fast moving objects and performing im-
Figure 6.7: Few Outputs from different video sequences where each row is from a single sequence. The anomalies are marked in 'Red' color. Note: All the outputs are generated by using a constant threshold.

proper actions such as jumping etc. in dense crowd environment. In addition, the dataset throws various challenges in form of crowd densities, occlusion to moving objects and less training data.

The new dataset contains 14 training videos and 10 testing videos. All these videos have been divided into 9 different sets depending upon the scene. Each of the video clip contains 90 to 150 frames of size 856 × 480. To measure the performance we have used same frame-level anomaly measure as that used in UCSD Ped1 and Ped2.
The qualitative results of the proposed approach on CHUK-Crowd Anomaly Dataset are shown in Fig. 6.9. Quantitatively, the proposed algorithm performs better with an EER % of 31.28% and AUC % of 71.29% on CHUK-Crowd Anomaly Dataset when compared to 40.81% and 64.38% of Biswas et al. [22] (Ch. 5). Refer Fig. 6.8 and Tab. 6.4 for details.

![Performance Graph](image)

**Figure 6.8:** Performance of the different approaches tested on CHUK-Crowd Anomaly Dataset (Best viewed in color)

<table>
<thead>
<tr>
<th>Approaches</th>
<th>EER</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biswas et al. [22]</td>
<td>40.81%</td>
<td>64.38%</td>
</tr>
<tr>
<td>Proposed</td>
<td>31.28%</td>
<td>71.29%</td>
</tr>
</tbody>
</table>

**Table 6.4:** Frame Level Equal Error Rate (EER) on CHUK-Crowd Anomaly Dataset

### 6.5 Conclusion

We have proposed an anomaly detection algorithm based on short local trajectories that capture shape and motion information under a single framework. The approach uses Hidden Markov Model to capture the usual patterns and subsequently perform anomaly detection. The novelty of the proposed approach lies in the unique way of capturing the trajectories that helps in better localization of moving objects irrespective of the density of the crowd. Experiments were performed on UCSD Ped1 and UCSD Ped2 datasets, where the proposed method achieved state-of-the-art results demonstrating the high localization capability.
Figure 6.9: Few Outputs from CHUK-Crowd Anomaly dataset video sequences where each row is from a single sequence. The anomalies are marked in ‘Red’ color. (Best viewed in color)
Chapter 7

Summary

In this thesis we have focused on analyzing the motion patterns, retrieved from motion vectors and optical flow, in order to tackle three event analysis problems, namely human action recognition, crowd flow segmentation and anomaly detection. For video classification based on action content in compressed domain, Histogram of Oriented Motion Vector (HOMV) feature is proposed to capture the spatio-temporal motion patterns of the videos. To segment the various crowd flows in a compressed video, multi-scale super-pixel segmentation of the motion vectors is proposed which helped in obtaining the meaningful segments without having prior knowledge about the number of segments. In order to detect anomalies in compressed domain, usual behavior is modeled locally by capturing features such as magnitude and orientation of motion vectors. Any deviation from this learned model is considered as an anomaly.

Compressed domain approaches typically have low accuracies due to availability of coarsely and noisy motion information available at macroblock. The accuracy, however, is improved in pixel domain by using more accurate optical flow information. This is demonstrated by the two proposed techniques for anomaly detection to attain better accuracies. The first approach is based on widely used sparse reconstruction framework. The proposed algorithm enhances each local dictionary by applying appropriate transformation on dictionaries of the neighboring regions. This enhancement of dictionaries helps in better representation of usual behavior and subsequent detection of anomalies. The accuracy of anomaly detection is further improved by the another approach which uses short local trajectories of foreground super-pixels to model usual behavior. These trajectories capture both spatial as well as temporal information resulting in better detection and localization of anomalies. The proposed approach achieves state of the art anomaly detection accuracy and localization.

In conclusion, motion information helps in understanding various event analysis related problems. Thus, it plays a critical role in understanding the content of video as it captures the
dynamics of moving objects and their interactions.
Bibliography


[59] Ce Li, Zhenjun Han, Qixiang Ye, and Jianbin Jiao. Visual abnormal behavior detection based on trajectory sparse reconstruction analysis. Neurocomputing, 119:94–100, 2013. 50, 60


