Incremental Behavior Modeling and Suspicious Activity Detection

by

Kan Ouivirach

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science

Examination Committee:	Dr. Matthew N. Dailey (Chairperson) Dr. Nitin V. Afzulpurkar Dr. Supavadee Aramvith (External Expert) Dr. Sanparith Marukatat (External Expert)
External Examiner:	Prof. James J. Clark Dept. of Electrical and Computer Engineering McGill University, Canada
Nationality:	Thai
Previous Degree:	Master of Engineering in Computer Science Asian Institute of Technology, Thailand
Scholarship Donor:	Royal Thai Government Fellowship

Asian Institute of Technology
School of Engineering and Technology
Thailand
December 2013

This dissertation is dedicated to my parents.

Acknowledgments

This research work was accomplished through the support of many people. My most humble and sincere thanks to:

As always, first and foremost, my beloved parents, who inspire me and offer endless unconditional love and support. My deepest thanks and heartfelt appreciation cannot go to anyone else but them.

My tireless and trusted advisor, Dr. Matthew N. Dailey, for giving me invaluable advice, excellent guidance, strong encouragement, and support throughout my work with unbridled kindness and support. I also thank him for being very patient with me and always willingly answering all my questions.

My committee members, Dr. Nitin V. Afzulpurkar, Dr. Supavadee Aramvith, and Dr. Sanparith Marukatat, for their valuable insight, suggestions and constructive criticism.

My external examiner, Prof. James J. Clark, for his meticulous review and constructive commentary.

All of my friends at the AIT Vision and Graphics Lab (VGL) for enthusiastic discussions and comments on this work. In particular, Shashi Gharti, who conducted extensive research in areas related to the implementation of human tracking and feature extraction, and Phattanapon Rhienmora, who has been encouraging and supportive during the time I spent in pursuit of my Ph.D., and even to this day. I am eternally grateful to have been a part of this lab.

CSIM secretaries, Ms. Pintip Thavornchan, Ms. Sireekant Thanwongpan, and Ms. Siriporn Nanthasing, for their irreplaceable assistance during my studies, and without whom all things descend into chaos.

AITians for healthy conversations and their true friendship while I was living on campus at AIT.

Last but not least, the Royal Thai Government for the graduate scholarship that facilitated my studies at AIT. Without the scholarship, my studies at AIT would have been impossible.

Abstract

Video surveillance systems have become widespread tools for monitoring and law enforcement in public areas. Due to increasing crime, video surveillance systems are being deployed in more and more places. Video surveillance systems are needed to help security personnel prevent and respond to criminal activity in a timely fashion. However, human monitoring becomes increasingly expensive and ineffective as the amount of video data increases. Also, manual review of all video footage is time-intensive and error-prone.

Automated anomaly detection can help to improve the effectiveness of human observers by separating the video stream into a sequence of "normal" and "unusual" events. However, much of the existing work has many limitations in this direction. Oftentimes, separate models for each distinct a priori known class of "normal" behavior are assumed. This could lead to the problem of typical behavior evolving and becoming more diverse to the point that the false alarm rates increase. The naive solution would be to retain all of the observation data and retrain the system periodically, but this requires storing all of the incoming data, requiring too much disk space. In this dissertation, we therefore propose and evaluate an efficient method for incremental automatic identification of suspicious behavior in video surveillance data.

First, we develop a blob extraction method that segments blobs and an appearance-based blob tracking method that uses a forward-backward overlap method and color coherence vectors (CCVs) to maintain identity through blob merging and splitting cases.

Second, we propose a new method for detecting shadows using a simple maximum likelihood formulation based on HSV color information. We find that the method outperforms standard shadow detection methods on three different real-world video surveillance data sets.

Third, we propose a new method for clustering human behaviors in the context of video surveillance that is suitable for bootstrapping an anomaly detection module for intelligent video surveillance systems. We show that the method is extremely effective in separating anomalous from typical behaviors on real-world testbed video surveillance data.

Fourth, we propose a *semi-supervised* batch anomaly detection method that self-calibrates itself from a small bootstrap set in which each bootstrap sequence is manually labeled as normal or suspicious by a human operator. Our method proves extremely effective, with a very low false alarm rate at a 100% hit rate.

Finally, we propose an effective behavior modeling and suspicious activity detection method extending the batch method that incrementally learns scene-specific statistical models of human behavior without requiring storage of historical data. The incremental method's false alarm rate drops below that of the batch method on the same data.

In experimental evaluations on real-world testbed video surveillance data sets, the proposed methods prove to be practical and effective at inducing scene-specific statistical models useful for bringing suspicious behavior to the attention of human security personnel.

Table of Contents

Chapter	Title	Page
	Title Page	i
	Acknowledgments	iii
	Abstract	iv
	Table of Contents	vi
	List of Figures	viii
	List of Tables	x
	List of Symbols	xi
	List of Acronyms	xii
1	Introduction	1
	1.1 Background and Motivation	1
	1.2 Review of State of the Art in Surveillance Systems	3
	1.3 Contributions	8
	1.4 Organization of Dissertation	10
2	Blob-Based Motion Analysis	11
	2.1 Introduction	11
	2.2 Literature Review	11
	2.3 Global Motion Detection	20
	2.4 Blob Extraction	20
	2.5 Appearance-Based Blob Tracking	20
	2.6 Blob Feature Vector Discretization	23
	2.7 Discussion	24
3	Extracting the Object from the Shadows: Maximum Likelihood Object/Shadow Discrimination	25
	3.1 Introduction	25
	3.2 Maximum Likelihood Classification of Foreground Pixels	28
	3.3 Experimental Results	30
	3.4 Discussion	32
4	Clustering Human Behaviors with Dynamic Time Warping and Hidden Markov Models	35
	4.1 Introduction	35
	4.2 Human Behavior Pattern Clustering	37
	4.3 Experimental Results	41
	4.4 Discussion	44
5	Automatic Suspicious Behavior Detection from a Small Bootstrap Set	47
	5.1 Introduction	47
	5.2 Behavior Model Bootstrapping	48

5.3 Anomaly Detection 50
5.4 Experimental Results 51
5.5 Discussion 55

6 Incremental Behavior Modeling and Suspicious Activity Detection 59

6.1 Introduction 59
6.2 Related Work 60
6.3 Behavior Model Bootstrapping 61
6.4 Anomaly Detection 61
6.5 Incremental Behavior Modeling for Anomaly Detection 62
6.6 Experimental Results 65
6.7 Discussion 72

7 Conclusion and Recommendations 73

7.1 Conclusion 73
7.2 Recommendations 74

8 References 76

List of Figures

Figure	Title	Page
1.1	CCTV monitoring room.	2
1.2	Example of a scene captured by ZoneMinder.	3
1.3	Screen shot of SecureCam.	4
1.4	Screen shot of Vitamin D Video.	5
1.5	Screen shot of SuperTrack.	6
2.1	Example results for single and MoG background modeling methods.	13
2.2	Example MoG and improved MoG background modeling results.	13
2.3	Mesh feature calculation.	14
2.4	Example star skeletons.	15
2.5	Space-time image representation.	16
2.6	Example motions along with their motion history images (MHIs).	16
2.7	Example filtered images.	17
2.8	Sample 3D tracking results from Rosales and Sclaroff's method.	18
2.9	Sample tracking results from Girondel et al.'s method.	19
2.10	Sample tracking results from Lv et al.'s method.	19
2.11	Sample tracking results from Senior et al.'s method.	20
2.12	Sample blob tracking results for typical simple cases	22
2.13	Sample blob tracking results for a complex case.	22
3.1	Example distributions over the difference in hue, saturation, and value components for true object pixels, extracted from our hallway dataset.	28
3.2	Example distributions over the difference in hue, saturation, and value components for true shadow pixels, extracted from our hallway dataset.	29
3.3	Sample frames from the Hallway, Laboratory, and Highway video sequences.	30
3.4	Results for an arbitrary frame in each video sequence.	34
4.1	Block diagrams for the proposed method.	37
4.2	Sample foreground extraction and shadow removal results.	38
4.3	Processing flow of the use of HMM clustering method.	40
4.4	Examples of common human activities in our testbed scene.	41
4.5	Example of linear HMM with bypass transitions, as used in all experiments.	42
4.6	Partial dendrogram constructed in Experiment I.	43
5.1	Block diagram of our overall system.	48
5.2	Examples of common human activities in our testbed scene.	51
5.3	Subset of model configuration selection results.	56

5.4 Anomaly detection ROC curves. Solid red, dotted green, and dashed blue lines represent ROCs for the proposed method in Experiment I, Mahalanobis classifier for anomaly detection in Experiment III, and SVM-based anomaly detection in Experiment IV, respectively. 57
5.5 Example anomaly detected by the proposed method in Experiment I. 58
6.1 Anomaly detection ROC curves. Solid red, dotted green, dashed blue, and dash-dot magenta lines represent ROCs for methods I, II, III, and IV, respectively. 68
6.2 Comparison of batch learning and incremental learning over time. 70

List of Tables

Table	Title	Page
3.1	Comparison of shadow detection results between the proposed, DNM, and NCC methods.	32
4.1	Clustering results for Experiment I (DTW+HMMs).	45
4.2	Clustering results for Experiment II (HMMs only).	46
5.1	Example human behavior pattern bootstrapping results.	53
5.2	Anomaly detection results for the proposed method, k-NN anomaly detection method, Mahalanobis classifier for anomaly detection method, and SVM-based anomaly detection method in Experiment I, II, III, and IV, respectively.	54
6.1	Example human behavior pattern bootstrapping results.	66
6.2	Anomaly detection results for the local method with the z-scoring method and the likelihood ratio test, and the global method with the z-scoring method and likelihood ratio test.	69
6.3	Anomaly detection results on average for incremental HMM learning for our local method and the global method over 5 trials.	71

List of Symbols

$\mathbf{f}_i^t$	Feature vector of track i at time t.
V	Set of discrete categories (cluster IDs).
U	Number of categories.
Θ	Model estimate for shadows.
sh	Shadow.
obj	Object.
L_i	Log of the probability of a sequence O_i given a trained HMM.
$\mathbf{O_i}$	Observation sequence i.
T_i	Number of observations in sequence i.
M_c	HMM that models the sequences in cluster c.
p_c	Optimal rejection threshold for cluster c.
N_c	Number of deviant patterns allowed in cluster c.
z	z-score.
θ_z	Global alarm threshold.

List of Acronyms

CCTV	Closed-Circuit Television.
MoG	Mixture of Gaussian.
CCV	Color Coherence Vector.
DNM	Deterministic Nonmodel-Based Shadow Detection.
NCC	Normalized Cross-Correlation.
ML	Maximum Likelihood.
IML	Incremental Maximum Likelihood.
EM	Expectation Maximization.
DBN	Dynamic Bayesian Network.
k-NN	k-Nearest Neighbors.
SVM	Support Vector Machine.
PCA	Principal Component Analysis.
MDL	Minimum Description Length.
GMM	Gaussian Mixture Model.
HMM	Hidden Markov Model.
DTW	Dynamic Time Warping.
LRT	Likelihood Ratio Test.
ROC	Receiver Operating Characteristic.
EER	Equal Error Rate.

TP	Number of True Positives.
FP	Number of False Positives.
TN	Number of True Negatives.
FN	Number of False Negatives.
TPR	True Positive Rate.
FPR	False Positive Rate.

Chapter 1

Introduction

This dissertation explores the possibilities and develops algorithms for modeling human behaviors and detecting anomalies without requiring large databases of training data. In this chapter, I discuss the background of, and motivation for my work. Additionally, I review real-world surveillance systems, present the contributions of the work, and provide an outline of this dissertation.

1.1 Background and Motivation

Due to the increase in demand for video surveillance in public areas, video surveillance is becoming ubiquitous in our lives. It becomes increasingly important for preventing and responding to terroristic threats and other criminal activities. However, the resulting proliferation of surveillance cameras is making it increasingly difficult to monitor all channels continuously. Human monitoring is therefore becoming increasingly expensive and ineffective as the torrent of video data increases. For instance, in a CCTV monitoring room (see Figure 1.1), security operators are required to monitor 24 hours a day and be ready to take action when an alarm occurs. Security will be enhanced if there is an intelligent surveillance system able to perform filtering, archive "normal" events, and automatically raise alarms for possible "abnormal" events or presenting such events to human security personnel for consideration as a security threat. The problems to consider in a surveillance system include pedestrian detection and tracking, unattended object detection, human behavior modeling, and anomaly detection.

Human behavior understanding is one of the most intricate and complex facets of intelligent video surveillance. Clearly, behavioral understanding can be problematic because it requires considerable time to study each individual subject, and often cannot predict "one-time" malicious behavior before it happens. The problem is difficult and remains unsolved, due to the wide range of activities possible in any given context and the large amount of variability within any particular activity.

One limitation of much of the existing work is that it learns in batch mode and creates separate models, which remain static once trained, for each distinct class of behavior. It may have an advantage in explaining data that are well-defined, but it needs sufficiently large samples of behavior patterns before training the system. Another limitation is that the number of "normal" behavior patterns needs to be known beforehand because we cannot learn a model for "abnormal" behavior that is rare and diverse. Therefore, we must instead detect deviations from typical behavior.

The ambiguity of human behaviors makes the problem more challenging since it is scene-dependent and can change over time. In particular, a behavior considered normal in one context might be considered unusual in another context depending on the type of behavior and where and when it is observed. The characteristics of typical behavior also vary from scene to scene.

Many researchers have attempted to build surveillance systems able to interpret and understand human behaviors. However, most of the work learns behaviors in an offline fashion and keeps the model

Figure 1.1: CCTV monitoring room. Reprinted from the Twenty First Security Web site (`http://www.twentyfirstsecurity.com.au/`).

static once trained. This is impractical for real-time surveillance applications, since offline training requires a great deal of computational time and resources. In addition, the number of behaviors needs to be known beforehand. We therefore need to find a way to initially recognize and model each of the common behavior groups in a particular scene as well as to incrementally learn scene-specific models as new behavior is observed.

Many open source and commercial video surveillance products are available. Most of the products do not implement any intelligent features; therefore, they cannot learn and understand the events occurring in a scene. Moreover, a great deal of configuration must be done before a user can deploy the system to operate in a real situation. For instance, a user may need to define restricted zones or rules for a specific scene.

Three main challenges for human behavior understanding and anomaly detection are as follows.

1. It is impractical to store all data in a system; therefore, we need an efficient approach that learns scene-specific statistical models of human behavior without requiring storage of large databases of training data.

2. Real human behavior is sometimes ambiguous; therefore, we need to keep humans such as security personnel in the loop to interpret it.

3. Unusual behavior is rare and diverse, and it is also impractical to acquire sufficient data for a good model for it; therefore, we need to detect deviations from typical behavior instead.

In this dissertation, we propose to explore, implement, and evaluate an efficient method for automatic identification of suspicious behavior in video surveillance data that incrementally learns scene-specific statistical models of human behavior without requiring storage of large databases of training data. The method is based on hidden Markov models (HMMs) with sufficient statistics and an optimal threshold on the likelihood of an event according to the human behavior model. We begin by building an initial set of models explaining the behaviors occurring in a small bootstrap data set. The bootstrap procedure partitions the bootstrap set into clusters then assigns new observation sequences to clusters based on statistical tests of HMM log likelihood scores. Cluster-specific likelihood thresholds are

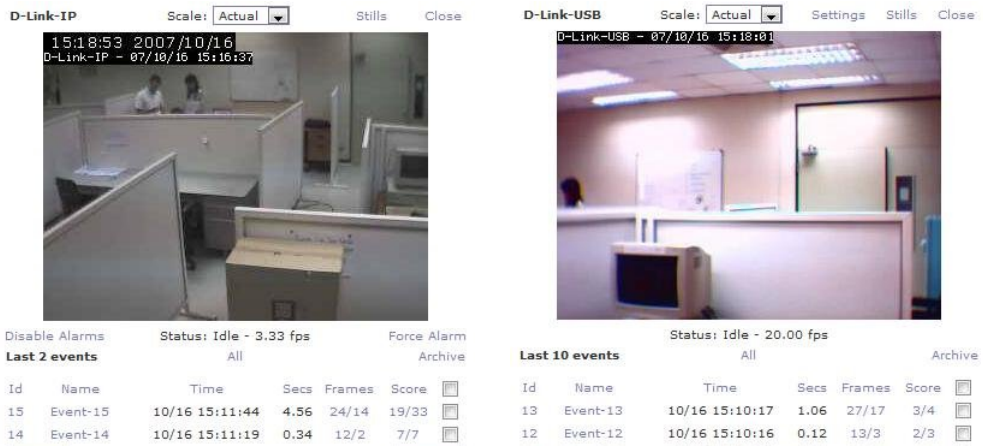


Figure 1.2: Example of a scene captured by (a) an IP camera and (b) a USB camera from ZoneMinder.

learned rather than set arbitrarily. After bootstrapping, each new sequence is used to incrementally update the sufficient statistics of the HMM it is assigned to. Our method is an effective solution to the problem of inducing scene-specific statistical models useful for bringing suspicious behavior to the attention of human security personnel.

1.2 Review of State of the Art in Surveillance Systems

We divide this section into two parts. The first part describes the review of open source and commercial products. The second part describes the review of the state of the art in surveillance systems.

1.2.1 Open Source and Commercial Products

ZoneMinder (ZM; Coombes, 2007) is an open source video surveillance system for Linux. It has been released under the terms of the GNU general public license (GPL). ZM supports both IP cameras and USB cameras. Figure 1.2 shows an example of a scene captured by an IP camera and a USB camera. The system consists of many independent modules including motion detection. Each component is designed to use as few resources as possible, maximizing the efficiency of the machine. It provides a PHP-based Web interface that allows us to control the cameras and monitor a scene from anywhere. Besides controlling and monitoring, we can also review, archive, or delete the events through its Web interface.

SecureCam (Bedecs, 2009) is an open source software project for the Windows platform providing a friendly user interface as shown in Figure 1.3. Similar to ZoneMinder, SecureCam supports multiple cameras and provides a few simple and fast algorithms such as motion detection. It has e-mail and

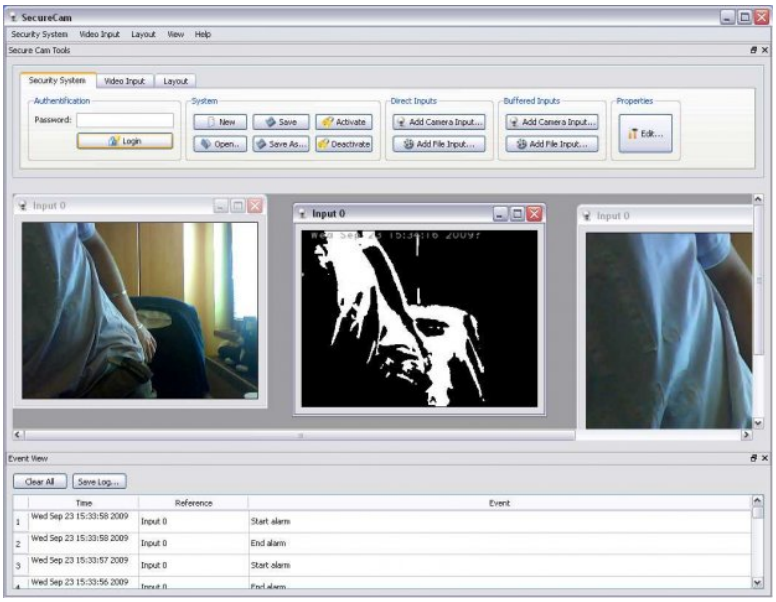


Figure 1.3: Screen shot of SecureCam. Reprinted from the SecureCam Web site (`http://sourceforge.net/projects/securecam/`).

sound notification features, and it also allows users to record video for later review.

Vitamin D Video (Vitamin D, 2010) is a commercial video surveillance system. It supports USB webcams and network cameras and provides advanced features such as creating rules for specific events. For example, the user could create a rule to record a clip only when a person opens a door, or the user could create a rule to notify via email when the system senses some movement in a defined region. The system also allows users to monitor a scene and to search for videos of interest. Figure 1.4 shows a screen shot of the software.

It is interesting that Vitamin D Video does not use motion detection, but rather uses people detection. It uses hierarchical temporal memory (HTM; Hawkins, 2007) to detect people. This algorithm attempts to apply the concept of how the brain recognizes a human being.

Video Analytics (DVTel, 2009) is a video surveillance software package that includes many different modules. Sample modules are intrusion detection, left luggage detection, and autonomous pan-tilt-zoom (PTZ) tracking. However, some modules such as intrusion detection do not have any intelligent features.

SuperTrack (Stratech, 2009) is an intelligent video surveillance system. SuperTrack is designed for automatically analyzing and monitoring events without the need for human attention and interaction. This system provides automatic object detection, abnormal behavior detection, tracking, and so on. However, it requires the user to configure and maintain a set of alarm trigger rules in advance. SuperTrack also supports various types of cameras such as USB cameras and network cameras. Figure 1.5 shows a screen shot of the system.

The review on open source and commercial video surveillance products makes us realize the impor-

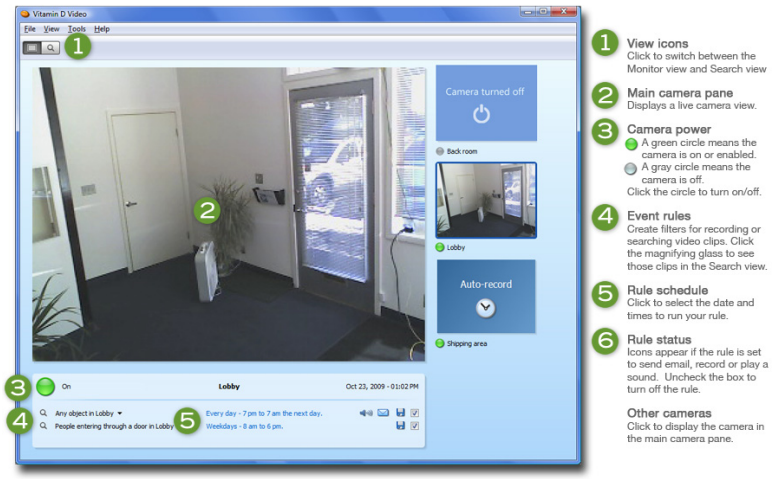


Figure 1.4: Screen shot of Vitamin D Video. Reprinted from the Vitamin D Video Web site (`http://www.vitamindinc.com/`).

tance of and demand for human behavior modeling and anomaly detection.

1.2.2 State of the Art Systems

In general, open source and commercial video surveillance products lag behind state-of-the-art systems that are currently in development in academic research. Despite the vast number of systems available, most do not use much intelligent analysis. Therefore, they cannot understand the events occurring in a scene or learn human activities. In addition, users are required to set a large number of configuration parameters before they can deploy a system to operate in real-world situations. For instance, in most commercial and open source systems, restricted zones or rules for a scene need to be defined for the system to operate optimally.

There have been a number of significant advanced research with novel approaches for visual surveillance in recent years, including automatic human activity interpretation and wide-area surveillance to reduce maintenance costs, human interaction, and the propensity for human error. Remagnino et al. (2007) introduce a journal issue that covers many novel concepts and challenges for research on intelligent video surveillance. One of the most challenging problems in this area is the analysis of human activities. Various approaches can be applied to resolve the issues.

Several researchers have proposed space-time methods, which model a human activity using features extracted from a 3D shape in a spatio temporal vector space, to recognize human activities from video sequence.

Campbell and Bobick (1995) propose a method to represent human body movements as curves in low-dimensional "phase spaces." They define a phase space as a space that has axes representing an independent parameter of the body based on a 3D model estimated for each frame. The authors use

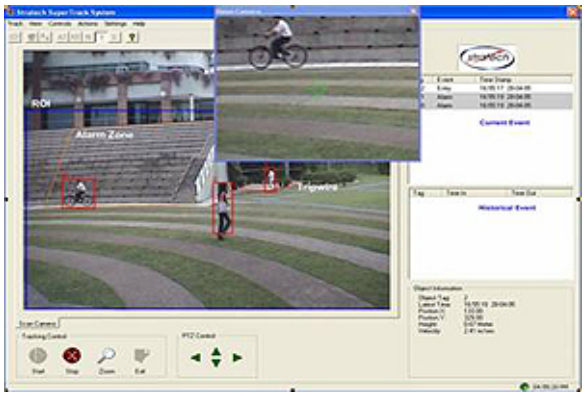


Figure 1.5: Screen shot of SuperTrack. Reprinted from the Stratech Web site (`http://www.stratechsystems.com/iv_supertrack.asp`).

this representation to learn new movements by searching for constraints that are in effect during the movement to be learned and not in effect during other movements. The learned representation is used for recognizing body movements.

Shechtman and Irani (2005) propose an approach to estimate motion flows to recognize human actions using 3D space-time features. The authors compare human activity representations in terms of their patches. They extract a small space-time patch around every 3D point. They then represent each space-time patch as a flow of a particular local motion. Finally, they compute a matching score from the correlation between two patches in a template and video at the same location. Given a new video, the authors use a sliding window template to search for all possible 3D patches that best match with the template.

Ke et al. (2007) classify segmented spatio-temporal volumes extracted from human activities using support vector machines (SVMs). The authors use a hierarchical mean shift method to cluster colored voxels then obtain segmented volumes. They recognize each activity by finding the best match of a model to a subset of over-segmented spatio-temporal volumes.

As an alternative to space-time approaches, template-based approaches can be used to represent human activities by maintaining a sequence of templates or a set of sample action sequences. Given a new video input, the method extracts features from the video, constructs a sequence of feature vectors, and then measures the similarity between the constructed sequence and a sequence of templates in the recognition process.

Darrell and Pentland (1993) propose a dynamic time warping (DTW)-based gesture recognition method using multiple template images or models to represent the poses or the dynamics of articulated objects. They find the correspondences between observed frames and models using the DTW algorithm.

D. M. Gavrila and Davis (1995) develop a 3D model-based body part tracking method using a DTW algorithm to recognize human actions. Multiple cameras have been used to obtain 3D models of a human that is represented as a stick figure model. The authors treat each joint angle as a feature

characterizing human movement, then they use the DTW algorithm to analyze a sequence of angles and compare the score with a pre-trained reference sequence per action.

Yacoob and Black (1998) use PCA-based modeling to represent an activity as a linear combination of a set of eigenvectors. They compute the projections of the activity observed from the input and action templates onto the eigenvector basis, then they compare the projections of the two.

Bobick and Davis (2001) propose a template-based human action recognition method using human silhouette in subsequent frames to track shape changes over time. Each action is represented by a template that is composed of two 2D images: a motion energy image (MEI) and a motion history image (MHI), constructed from a sequence of foreground images. They recognize human actions by searching for the best match between human silhouette extracted from an image and the template, which is a pair of images, both MEI and MHI.

Vaswani et al. (2003) recognize people interaction with an airplane using a single-layered exemplar-based sequential approach. A group activity is represented as a shape change over time. The authors extract point objects and construct a polygon in each frame then track those points. Their method extracts an activity shape from an input and measures the similarity with a maintained model in a tangent space to recognize activities.

Yang et al. (2010) use an exemplar-based pose representation called a "poselet." A poselet is a set of patches with similar 3D pose configurations, which is expressed as a latent variable. The authors treat the action recognition problem as inference over the latent variables.

Statistical approaches are also widely used in activity recognition. These approaches use statistical state-based models to recognize human activities from historical data. Dynamic Bayesian networks (DBNs) and hidden Markov models (HMMs) are very well-known for their effectiveness in this analysis.

Yamato et al. (1992) adopt standard HMMs to recognize tennis activities. They extract a feature vector containing the number of foreground pixels that are present in each cell of grid. They then treat the feature vectors as a sequence of observations generated by an activity model, each of which is modeled using a HMM. Given a new sequence, the authors measure the likelihoods of each HMM and then use the likelihood results to classify the action.

Bobick and Wilson (1997) use state models for gesture recognition. Each gesture is represented by a 2D trajectory that describes the location changes of a hand. They compute the transition probabilities in the similar way as the transition probabilities of HMMs. To recognize each gesture, they perform a dynamic programming algorithm to measure the optimal matching cost between the new observation and each gesture prototype.

Oliver et al. (2000) extend standard HMMs to the "coupled" HMM (CHMM) to model human activity interactions. Since the basic HMM is unsuitable for modeling activities involving two or more persons, the authors then build a CHMM by coupling multiple HMM models, where each HMM models the motion performed by one person, to model human-human interactions.

Cupillard et al. (2002) use a finite state machine to recognize group activities. This approach is equivalent to a fully observable Markov model.

H. Zhong et al. (2004) use a clustering-based similarity approach for detecting abnormal activity in video. They evaluate the approach on several real world surveillance videos such as a video from surveillance camera overlooking a road adjacent to a fenced facility.

Park and Aggarwal (2004) use a DBN for gesture recognition for two interacting people. They construct a hierarchical DBN to gain an advantage of the dependence of nature among body parts' motion. The authors model each pose using a set of features including locations of skin regions, maximum curvature points, and the ratio and orientation of each body part, from the corresponding body part to recognize gestures.

Vaswani et al. (2005) use a statistical model such as a continuous-state HMM for the configuration of a set of landmark shape dynamics in an activity. They identify an abnormal activity defined as a change in the shape activity model. Their approach is used in an airport scenario with people deplaning and moving toward the terminal.

Park and Trivedi (2007) individually model body part gestures including upper body gestures, lower body gestures, and torso translations based on the assumption that each body part is independent of the others. Then the authors recognize gestures using the independent sets of HMMs.

The knowledge-based approach is another well-known approach in which predefined rules or a priori knowledge about the observed environment and video events are used.

Georis et al. (2004) built a real-time bank agency monitoring system. Scenarios are modeled by domain expert knowledge. Then end-user feedback is collected to improve the scenario models.

Georis et al. (2007) divide video events into three parts: all objects present in the scene, sub-events involved in the event, and relations between objects and sub-events. The authors then use a formalism (Vu et al., 2003) to model events in the scene.

Fusier et al. (2007) propose an automated video understanding system that is able to recognize activities occurring in environments. They use the contextual knowledge about the observed environment to build a formalism for event modeling. The formalism is designed to allow the experts to easily describe and define the event models.

Other related work can be found in the survey papers (Cedras & Shah, 1995; D. Gavrila, 1999; Aggarwal & Cai, 1999; Krüger et al., 2007; Turaga et al., 2008) that also discuss the essential aspects of and recent research trends in activity recognition approaches designed for human activity analysis.

1.3 Contributions

The contributions of this dissertation are as follows.

1. We propose an intelligent video surveillance system that is practical and fairly close to the ideal surveillance system, which should monitor activity and automatically trigger alarms to security personnel;

2. We develop an algorithm for extracting and tracking multiple moving foreground blobs in a scene using an appearance model;

3. We propose a shadow detection method that uses a simple maximum likelihood classification approach based on color information;

4. We propose a new method for clustering human behaviors that is suitable for bootstrapping an anomaly detection module for intelligent video surveillance systems;

5. We propose and develop a new and effective algorithm for semi-supervised learning of common human behaviors and detect anomalies in video sequences;

6. We introduce a new and more accurate algorithm for incrementally profiling human behaviors in video sequences without requiring storage of large databases of training data;

7. We provide the source code and datasets online[1] for researchers interested in evaluating or extending our work.

1.3.1 List of Publications

We provide a list of publications as part of this dissertation. We also include the list of works currently under review and preparation for submission.

Published Works

– **Extracting the Object from the Shadows: Maximum Likelihood Object/Shadow Discrimination**
Kan Ouivirach and Matthew N. Dailey
International Conference on Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON), pages 1–5, 2013
Publisher: IEEE Computer Society
Included in Chapter 3

– **Clustering Human Behaviors with Dynamic Time Warping and Hidden Markov Models for a Video Surveillance System**
Kan Ouivirach and Matthew N. Dailey
International Conference on Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON), pages 884–888, 2010
Publisher: IEEE Computer Society
Included in Chapter 4

– **Automatic Suspicious Behavior Detection from a Small Bootstrap Set**
Kan Ouivirach, Shashi Gharti, and Matthew N. Dailey

[1] See `http://www.kanouivirach.com/#downloads`.

International Conference on Computer Vision Theory and Applications (VISAPP), volume 1, pages 655–658, 2012
Publisher: Springer-Verlag
Included in Chapter 5

– **Incremental Behavior Modeling and Suspicious Activity Detection**
Kan Ouivirach, Shashi Gharti, and Matthew N. Dailey
Pattern Recognition, 46(3): 671–680, 2013
Publisher: Elsevier
Included in Chapter 6

Manuscripts Currently in Preparation

– **Incorporating Geometric and Shadow Region Shape Information for Shadow Detection**
Kan Ouivirach and Matthew N. Dailey
To be submitted in *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013
Publisher: IEEE Computer Society

1.4 Organization of Dissertation

I organize the rest of this dissertation as follows.

In Chapter 2, I describe our blob-based motion analysis methods including blob extraction and appearance-based blob tracking.

In Chapter 3, I present a new method for detecting shadows using a simple maximum likelihood classification method based on color information.

In Chapter 4, I propose a new method for clustering behaviors in a scene.

In Chapter 5, I present an automatic suspicious behavior detection method that uses a small bootstrap set.

In Chapter 6, I describe our extension of the suspicious behavior detection approach for incremental learning in which security personnel feedback is incorporated.

Finally, in Chapter 7, I conclude and discuss the possible further extensions of my dissertation.

Chapter 2

Blob-Based Motion Analysis

In this chapter, I review existing blob-based motion analysis approaches including blob extraction, blob feature extraction, and blob tracking. I provide details of our methods, and then I discuss the feasibility of further improvement.

2.1 Introduction

Motion analysis is the first key step in human behavior understanding, since the results of later processes such as activity recognition rely on the analysis. Motion analysis provides a detailed synopsis of object movement in the scene, on which foundation we can interpret the activities in a surveillance video. This could save a tremendous amount of human effort in monitoring or retrieving videos for review.

This dissertation focuses exclusively on blob-based motion analysis methods. Our method assumes the camera is fixed relative to the scene in which objects/people are moving.

In the following sections, I first present a literature review of existing methods for blob-based motion analysis including blob extraction, blob feature extraction, and blob tracking. Second, I describe the details of our blob-based motion analysis approach. Finally, I present a discussion of possible further improvements to our approach.

2.2 Literature Review

This section provides a review of existing methods for blob-based motion analysis. I divide the section into three parts. The first part is blob extraction, the second part is blob feature extraction, and the last part is blob tracking.

2.2.1 Blob Extraction

Many approaches can be used to extract blobs or objects from a scene, assuming the camera is fixed. The simplest approach is image differencing. Yamato et al. (1992) use this method to extract blobs representing humans in a scene based on the following conditions:

$$\begin{array}{ll} \text{if} & |I_a(x,y) - I_b(x,y)| < T,\ I_e(x,y) = 0 \\ \text{else} & I_e(x,y) = I_a(x,y). \end{array}$$

$I_e(x, y)$ is the extracted human image, $I_a(x, y)$ is the original image, $I_b(x, y)$ is a background image without any objects, and T is a predefined threshold. This method works well under specific conditions but is not robust to noise, due to the use of a static background image. In real-world situations, the background scene changes over time.

Nair and Clark (2002) improve the image differencing technique by updating the background model and threshold over time for every frame using temporal averaging. They first model the background by taking the average of five consecutive frames without any moving objects in the scene. The labeled pixels are considered as foreground if they satisfy the following condition:

$$|I_n(x, y) - B_n(x, y)| > T_n(x, y),$$

where $I_n(x, y)$ is the current frame, $B_n(x, y)$ is the current background model, and $T_n(x, y)$ is a threshold function. The authors initialize the threshold value to 50 at all pixel locations. The background model and threshold function are updated as follows:

$$B_{n+1}(x, y) = \begin{cases} B_n(x, y) & \text{if } (x, y) \text{ is foreground,} \\ \alpha B_n(x, y) + (1 - \alpha) I_n(x, y) & \text{otherwise,} \end{cases}$$

$$T_{n+1}(x, y) = \begin{cases} T_n(x, y) & \text{if } (x, y) \text{ is foreground,} \\ \alpha T_n(x, y) + 2(1 - \alpha)|I_n(x, y) - B_n(x, y)| & \text{otherwise,} \end{cases}$$

where α is a constant to determine how fast $B_n(x, y)$ and $T_n(x, y)$ adapt to changes. The update to $T_{n+1}(x, y)$ increases the threshold when large variations in the background pixel distribution are observed and decreases when little variation is observed.

Although the image differencing technique is fast and commonly used in many surveillance applications such as ZoneMinder (Coombes, 2007), a single model with a simple threshold per pixel may not allow adaptation to multi-modal background distributions.

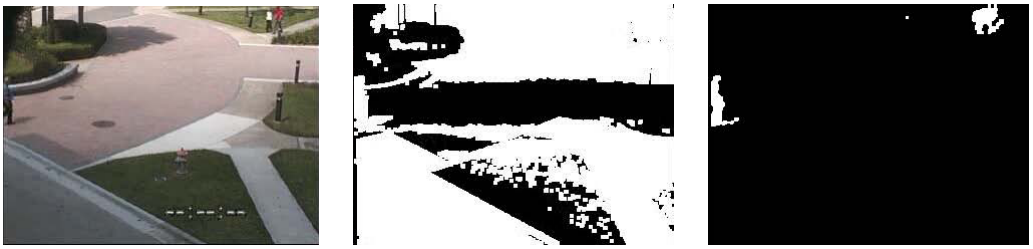
Many researchers have attempted to improve background modeling to deal with cluttered dynamic scenes. Wren et al. (1997) model people and the background scene separately using a single Gaussian distribution per pixel in the YUV color space. Each pixel is represented by a vector $[x, y, Y, U, V]^T$, where (x, y) is the pixel location and (Y, U, V) is the color component. They classify each pixel by calculating a likelihood for each class and selecting the best class. Based on the classification result, they update the new model's parameters for each person and for the background scene accordingly. Using this background modeling technique with the proper initial parameters yields reasonable results for both indoor and outdoor scenes. Figure 2.1(b) shows sample results for the single Gaussian background modeling method. Although this technique can deal with noise, it is unsuitable for more cluttered dynamic background scenes, especially outdoor scenes in which multi-modal background distributions occur more frequently.

Several additional methods have been proposed to model backgrounds in recent years. Examples include an eigenvalue decomposition-based approach (Oliver et al., 2000), a kernel density estimation-based approach (Elgammal et al., 2002), and a median-based approach (H. Li et al., 2006).

One of the best-known background modeling techniques is that of Stauffer and Grimson (1999), who propose a method that models multi-modal background distributions using a mixture of Gaussian (MoG) distributions. This technique works well in many cluttered dynamic scenes. It also outperforms the single Gaussian background modeling method outdoors, since multi-modal background

(a) (b) (c)

Figure 2.1: Example results for single and MoG background modeling methods. (a) Original image. (b) Foreground pixels using the single Gaussian background modeling method. (c) Foreground pixels using the MoG background modeling method. Reprinted from Tuan Anh (2008).



(a) (b) (c)

Figure 2.2: Example MoG and improved MoG background modeling results. (a) Original image. (b) Foreground pixels using MoG background modeling. (c) Foreground pixels using improved MoG background modeling. Reprinted from Poppe et al. (2007).

distributions occur frequently in outdoor scenes. The authors model the recent history of each pixel using a mixture of K Gaussian distributions. K is determined by the available memory and computational power. They classify pixel values that do not fit one of the background distributions as foreground until one of the Gaussian distributions supports those values. Figure 2.1(c) shows sample results for the MoG background modeling method.

However, the MoG background modeling method faces problems when dealing with gradual illumination changes, as seen in Figure 2.2(b). Poppe et al. (2007) propose a new version of the MoG background modeling method to solve this problem. The idea is to store the previous pixel value and previous matching model number and use them in the matching step. In the matching step, each pixel is checked against the K Gaussian distributions. For the model that matches the pixel in the previous frame, if the difference between the current pixel value and previous pixel value is smaller than a threshold, an intermediate match is defined; otherwise, the normal matching step is followed. If the matched model represents the background, the model number and the current pixel value are stored; otherwise, they retain their values. By doing this, foreground objects will not affect the recent history. Figure 2.2 shows example results of using the MoG background modeling method and Poppe et al.'s improved MoG background modeling method.

Once the foreground is segmented from the background, moving blobs can be extracted as connected components in the foreground mask.

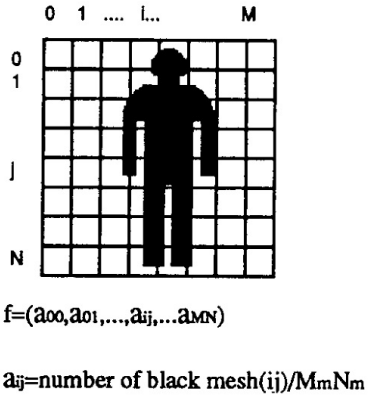


Figure 2.3: Mesh feature calculation. Reprinted from Yamato et al. (1992).

2.2.2 Blob Feature Extraction

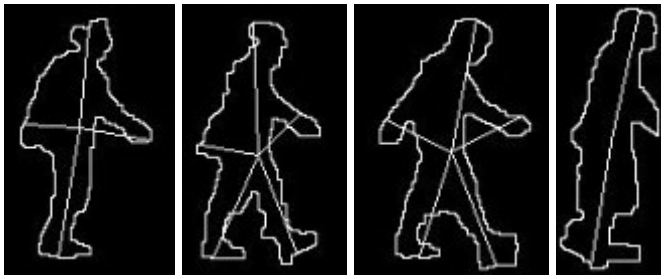
After segmenting blobs, we need to extract features to describe the shape and dynamics of each blob. Extracted features are commonly used in the blob tracking and behavior understanding processes. In this section, I review blob feature extraction methods used in existing systems.

Yamato et al. (1992) extract mesh features from foreground blobs. Mesh features are low-level image features that have also been successfully applied to complex 2D patterns such as multi-font characters (Umeda, 1982). Each binarized image ($M \times N$ pixels) is divided into a mesh with $M_M \times N_M$ cells. They represent each binarized image by a feature vector in which each element is the proportion of black pixels in each cell. Figure 2.3 shows how mesh features are calculated.

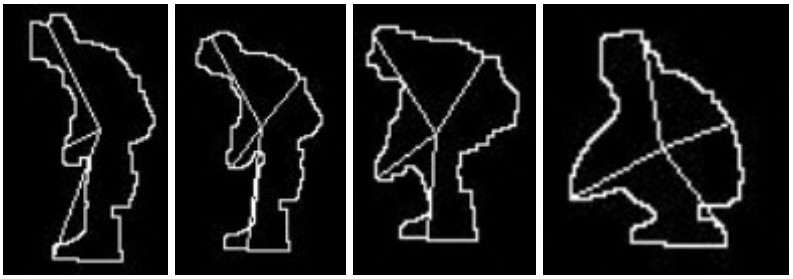
Nair and Clark (2002) represent an extracted blob by a feature vector that contains the coordinates of the blob's center of mass, average color, and height. They use a k-means algorithm to compute a set of codebook vectors from a set of feature vectors in the training data. Once the k-means model is trained, they map a feature vector to the index of the nearest codebook vector using Euclidean distance. They then generate and output a sequence of codebook vectors, which are then input to behavior modeling.

Star skeletonization is a technique that represents the gross extremity structure of a blob. For humans, the gross extremities would normally be the head, arms, and legs. Fujiyoshi and Lipton (1998) use this technique for analyzing human motion. Figure 2.4 shows an example of two different star skeletons extracted from two human motions.

Some feature extraction methods have the ability to capture motion history rather than only capturing current static features. One such method is space-time image features. H. Li et al. (2006) propose a method based on space-time image features for automatic behavior modeling and recognition. Their method is simple and insensitive to noise, and it does not need to track parts of the human body. They represent object shape using space-time images. After the foreground is extracted, they equidistantly divide the bounded rectangle of the foreground into sub-blocks then normalize the value of each sub-block by the maximum number of foreground pixels of each sub-block. Finally, they construct the space-time image representation (see Figure 2.5) and extract the statistical features from it using



(a)



(b)

Figure 2.4: Example star skeletons extracted from two human motions. (a) Walking. (b) Leaning forward and sitting down. Reprinted from Tuan Anh (2008).

Gabor filtering.

J. W. Davis and Bobick (1997) propose a feature extraction method using a motion history image (MHI). The MHI is a static image template where pixel intensity is a function of the recency of motion in the video sequence. The idea is to represent how the object is moving. The MHI is defined as:

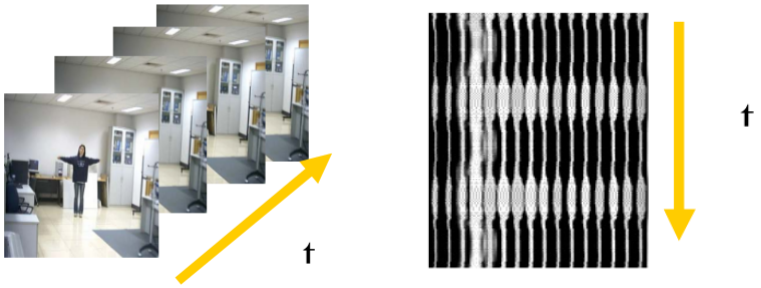
$$H_\tau(x,y,t) = \begin{cases} \tau & \text{if } D(x,y,t) = 1, \\ \max(0, H_\tau(x,y,t-1) - 1) & \text{otherwise,} \end{cases}$$

where $H_\tau(x,y,t)$ represents the MHI for a pixel (x,y) at time t. The more recently moving pixels will be brighter. Figure 2.6 presents example MHIs.

Based on the local intensity history of pixels, Xiang et al. (2002) propose a new low-level primitive applicable to human representation that can describe human actions efficiently, namely, the pixel change history (PCH). The PCH is capable of characterizing pixel-wise temporal visual information in order to detect pixel-level events. It is a representation of pixel-wise changes based on a combination of the pixel signal energy (PSE) proposed by Ng and Gong (2003) and the MHI. The PCH is computed as follows:

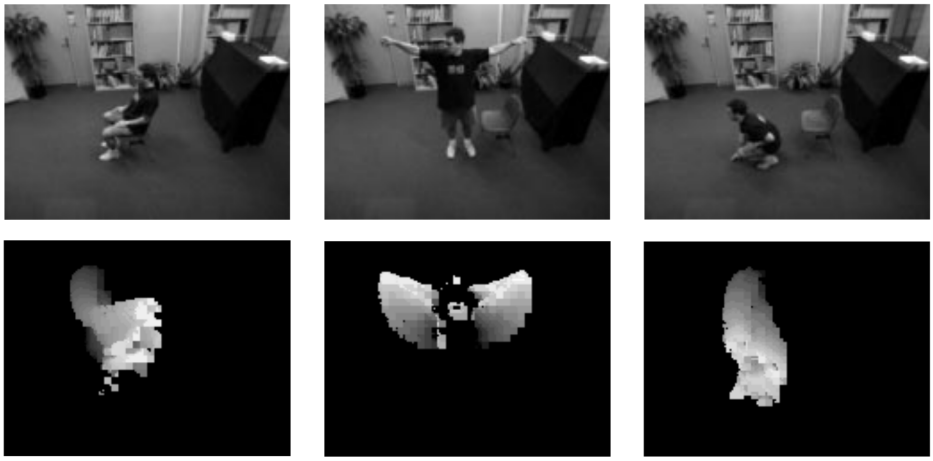
$$P_{\varsigma,\tau}(x,y,t) = \begin{cases} \min(P_{\varsigma,\tau}(x,y,t-1) + \frac{255}{\varsigma}, 255) & \text{if } D(x,y,t) = 1 \\ \max(P_{\varsigma,\tau}(x,y,t-1) + \frac{255}{\tau}, 0) & \text{otherwise,} \end{cases}$$

where $P_{\varsigma,\tau}(x,y,t)$ represents the PCH for a pixel (x,y) at time t, $D(x,y,t)$ indicates foreground regions at time t, ς is an accumulation factor, and τ is a decay factor. When the accumulation factor is set to 1, the PCH over the entire image is equivalent to the MHI.



(a) (b)

Figure 2.5: Space-time image representation. (a) Original video sequence. (b) Space-time image for the "hand clapping" behavior. Reprinted from H. Li et al. (2006).



sit-down arms-wave crouch-down

Figure 2.6: Example motions along with their motion history images (MHIs). Reprinted from J. W. Davis and Bobick (1997).

Xiang and Gong (2008b) use a discrete scene-wide event-based approach for behavior pattern representation. This approach is effective for cluttered scenes compared to the trajectory-based representations used in most existing approaches. Xiang and Gong model the foreground pixels using PCH (Xiang et al., 2002), group them into blobs, and define a scene event if the average PCH value of a blob is larger than a defined threshold. They then represent a scene event as a 7-dimensional feature vector

$$\mathbf{f} = [\bar{x}, \bar{y}, w, h, R_f, M_p x, M_p y],$$

where $(\bar{x}, \bar{y})$ is the centroid of the blob, w and h are the width and height of the bounding box associated with the blob, respectively, R_f is the filling ratio of foreground pixels within the bounding box, and $(M_p x, M_p y)$ are a pair of first-order moments of the PCH image within the bounding box.

Recursive filtering (Masoud & Papanikolopoulos, 2003) is a technique that encodes motion information such as the speed of the motion within a short period of time. The idea is to represent the "recency" of the motion. This technique is conceptually similar to the MHI. Let I_t be the frame at

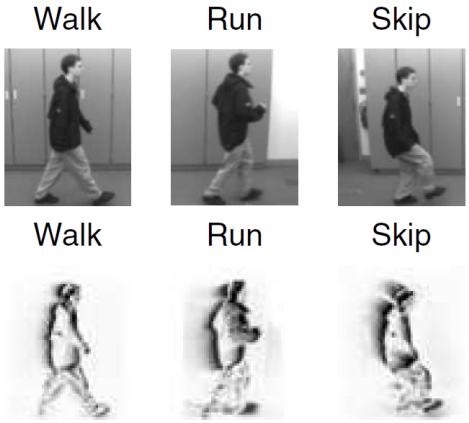


Figure 2.7: Example filtered images. The coefficient β is set to 0.5. Reprinted from Masoud and Papanikolopoulos (2003).

time t. The filtered image F_t at time t is defined as follows:

$$F_t = |I_t - M_t|$$

$$M_t = (1 - \beta)M_{t-1} + \beta M_t$$

$$M_0 = I_0 = \text{Background},$$

where $t = 1, 2, \cdots, n_i$. If $\beta = 0$, the filtered image F_t will be the foreground and if $\beta = 1$, F_t will be equivalent to image differencing. Examples of filtered images are presented in Figure 2.7.

I describe the blob feature representation used in this dissertation in Section 2.5.

2.2.3 Blob Tracking

Tracking is one of the most active research areas in computer vision. It can be used in a variety of applications, e.g. video indexing, sports video enhancement, and visual surveillance. In the area of tracking, occlusion ambiguity is the most fundamental difficulty. It makes associating blobs with individuals or moving objects that are partially or fully occluded in different frames highly challenging, especially in complex scenes.

In video surveillance applications, person/object tracking is a critical step prior to behavior modeling. Extracted information such as the motion trajectory can be used to better understand activities occurring in a scene.

The Kalman filter (Kalman, 1960) is a well-known algorithm that has been extensively applied to tracking in many domains. In visual surveillance, Rosales and Sclaroff (1999) propose a method for tracking multiple people in a 3D space. They assume that the system uses a single uncalibrated video camera to observe multiple moving people. They use an extended Kalman filter (EKF) to estimate relative 3D motion trajectories up to a scale factor. They select only two feature points, the two opposite corners of the person's bounding box, to reduce the complexity of the tracking problem. The

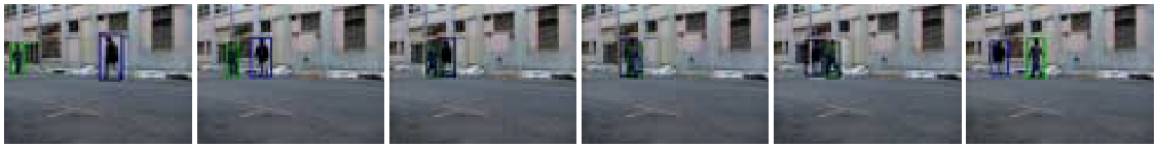


Figure 2.8: Sample 3D tracking results from Rosales and Sclaroff's method. Two people walking along different trajectories then occluding each other. Reprinted from Rosales and Sclaroff (1999).

3D size of the person's bounding box is assumed to remain approximately the same or at least vary smoothly. People are considered as planar objects, so the depth at the two feature points are assumed to be the same. The state vector then becomes:

$$\mathbf{x} = (x_0, y_0, x_1, y_1, z\beta, \dot{x}_0, \dot{y}_0, \dot{x}_1, \dot{y}_1, \dot{z}\beta)^T,$$

where $(x_0, y_0, z\beta)^T$ and $(x_1, y_1, z\beta)^T$ are the corners of the 3D planar bounding box, and $(\dot{x}, \dot{y}, \dot{z}\beta)^T$ represents a corner's 3D velocity relative to the camera. The experimental results demonstrate that Rosales and Sclaroff's 3D trajectory-based estimation method significantly improves robustness over 2D trajectory-based estimation methods. Figure 2.8 shows an example of 3D tracking results for this method.

Bodor et al. (2003) develop an automated system to track the positions of individual pedestrians using a Kalman filter and alert security personnel when a pedestrian enters a secure area.

Niu et al. (2003) propose a method using a second-order Kalman filter to track each person individually and handle occlusion problems. They define a state space that includes position, velocity, and acceleration. When a new person is detected, they store a unique label for that person including the height, centroid, and the average intensity of the region of the person. They then use this information to keep track of the person in subsequent frames.

Girondel et al. (2004) propose a multiple human tracking method using the Kalman filter. They create a Kalman filter for each person then predict each person's bounding box, the person's face's bounding box, and the speed on each frame. They use a state vector of ten components for each Kalman filter: the corners of the bounding boxes of the person and face plus two components for face speed. They use skin detection to detect the face and hands and assume that the face is a bigger blob that moves more steadily than the hands. The authors use a combination of a forward tracking phase and a backward tracking phase to track detected objects in two consecutive frames. The authors call this technique "forward and backward overlapping." I describe this technique in more detail in Section 2.5. Figure 2.9 shows an example of the tracking results from Girondel et al.'s method.

Lv et al. (2006) use a Kalman filter to track moving blobs. The authors use a color histogram as an appearance model to resolve the identity problem of blob merging. If the blob in the current frame matches with multiple existing tracks, they use the color histogram of each matched track to find the best-corresponding blob. Figure 2.10 shows an example of blob tracking results from Lv et al.'s method. Color histograms are a simple and efficient method that can be used to compare the similarity of images. However, they lack spatial information. For instance, the color histogram of an image with a single large block of red pixels will be similar to that of the image with same number of scattered red pixels.

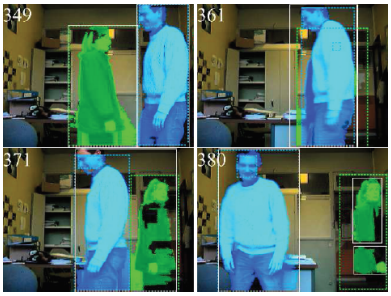


Figure 2.9: Sample tracking results from Girondel et al.'s method. Reprinted from Girondel et al. (2004).

Figure 2.10: Sample tracking results from Lv et al.'s method. Reprinted from Lv et al. (2006).

Yu et al. (2007) propose an appearance model that incorporates structural information for establishing correspondence between tracks of people in video. The authors construct an appearance model based on the color and path-length features of a pixels inside the silhouette of a person. Path-length is the length of the shortest path from a distinguished point, for example, the top of the head, to the pixel inside the silhouette. Their experimental results show that the combination of the color and path-length features is important for a discriminative appearance model.

Senior et al. (2006) use an appearance model to solve occlusion problems by modeling moving objects. The model can be used to localize objects during partial occlusions, detect complete occlusions, and resolve depth ordering of objects during occlusions. They use a RGB color model as an appearance model with an associated probability mask for each foreground blob. The probability mask records the likelihood of the object being observed at that pixel. The appearance model of each blob is then updated on subsequent frames. They also construct a distance matrix based on a bounding box distance measure. They use the distance matrix to determine the association between tracks and foreground blobs. This technique is similar to the forward and backward overlap method. The authors evaluate the proposed method on the PETS 2001 data set, and the results show that their algorithm successfully deals with complex real-world interactions. Figure 2.11 shows an example of the tracking results from Senior et al.'s method.

In the following sections, I provide details of our blob-based motion analysis: I describe our global motion detection in Section 2.3, our blob extraction in Section 2.4, our appearance-based blob tracking method in Section 2.5, and our blob feature vector discretization method in Section 2.6. I then finally discuss the limitations and possible future improvements to our method.

Figure 2.11: Sample tracking results from Senior et al.'s method. Reprinted from Senior et al. (2006).

2.3 Global Motion Detection

In order to decrease processing and storage time, I remove any video frames in which no motion occurs. We use simple frame-by-frame subtraction. When the difference image has a number of pixels whose intensity differences are above a threshold, we start a new video segment. We use a threshold large enough not to trigger event processing in a static scene, e.g. when a breeze makes leaves of a tree move. We also buffer the no-motion frames for a period of time and include some number of frames before and after the deleted motion comprising the event. This method avoids oversegmentation of events when moving objects stop moving briefly. In all of the experiments reported in this dissertation, we start a new video segment when the number of changed pixels in the 320×240 input frame exceeds 300.

Through some preliminary empirical experimentation I found that 300 was a good threshold. While this simple approach worked well in our experimental setup, it would not work as well in a situation where the camera is subject to vibration, causing a large number of pixels to change values significantly.

2.4 Blob Extraction

After discarding the no-motion video segments, we use Poppe et al.'s (2007) background modeling technique to segment foreground pixels from the background. As previously mentioned, Poppe et al. extend the standard mixture of Gaussians background model (Stauffer & Grimson, 1999) to handle gradual illumination changes.

2.5 Appearance-Based Blob Tracking

In this step, we take at time t a list of blobs detected by the previous step, a set of tracks updated at time $t - 1$, and a merged track association matrix. We output an updated track list and merged

Algorithm 1 Appearance-Based Blob Tracking

Input: B: set of all current blobs

Input: T: set of all current tracks

Input: M: merged track association matrix

Output: $\widetilde{T}$: set of all revised tracks

Output: $\widetilde{M}$: revised merged track association matrix

$\widetilde{T} \leftarrow \emptyset$; $\widetilde{M} \leftarrow \emptyset$; $L \leftarrow \emptyset$

$A \leftarrow$ GET-OVERLAP-AREA-MATRIX(B, T)

for each $t \in T$ **do**

if t is marked as processed **then** continue

$B' \leftarrow \{b' \mid A(b', t) > 0\}$ {B' contains candidate blobs for track t.}

$T' \leftarrow \{t\} \cup \{t' \mid M(t, t') = 1\}$ {T' contains all tracks currently merged with t.}

if $|B'| \geq 1$ **then**

for each $t' \in T'$ **do**

Let $b = \underset{b' \in B'}{\operatorname{argmax}} S(b', t')$

$L \leftarrow L \cup \{(t', b)\}$

MARK-TRACK-AS-PROCESSED(t')

end for

end if

end for

for each $(t_i, t_j) \in T \times T$ **do**

If $\exists b$ s.t. $(t_i, b) \in L \wedge (t_j, b) \in L$, $\widetilde{M}_{ij} \leftarrow 1$, otherwise $\widetilde{M}_{ij} \leftarrow 0$

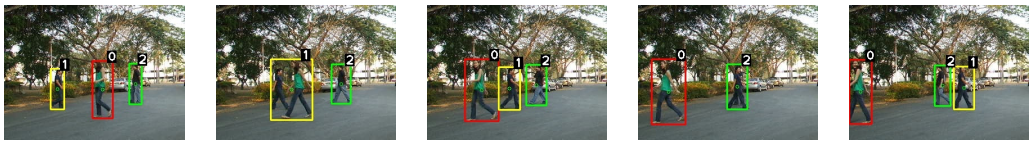
end for

$T^* \leftarrow \{t^* \mid \neg\exists b \in B$ s.t. $(t^*, b) \in L\}$ {T^* contains tracks for which "stale count" will be increased.}

$\widetilde{T} \leftarrow$ UPDATE-OR-DELETE-STALE-TRACKS(T, T^*)

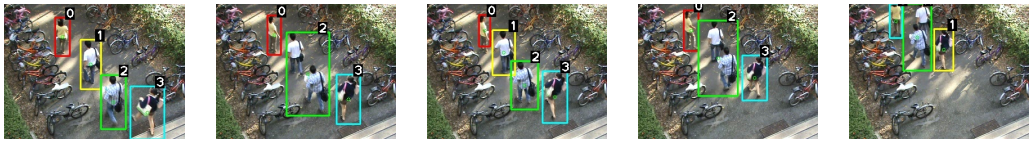
$B^* \leftarrow \{b^* \mid \neg\exists t \in T$ s.t. $(t, b^*) \in L\}$ {B^* contains blobs with no tracks assigned.}

$\widetilde{T} \leftarrow$ ADD-NEW-TRACKS-FOR-NOT-LINKED-BLOBS$(\widetilde{T}, B^*)$



Frame 90 Frame 94 Frame 103 Frame 110 Frame 116

Figure 2.12: Sample blob tracking results for typical simple cases. In frame 94, track 0 and track 1 have merged. In frame 103, our method has correctly associated each motion blob with the correct track after they split. Similarly, frame 110 shows the result of another merge, and frame 116 shows the (correct) result when the merged motion blob splits.



Frame 125 Frame 131 Frame 133 Frame 141 Frame 160

Figure 2.13: Sample blob tracking results for a complex case. In frame 131, track 1 and track 2 have merged. In frame 133, our method has correctly associated the current motion blobs with tracks after the merged blob splits. However, when the merged blob shown in frame 141 splits (in frame 160), we observe a track ID switch error in associating blobs with tracks.

track association matrix. Algorithm 1 is a pseudocode summary of our approach. We first construct a matrix in which each element indicates the overlap area of the bounding box of a current blob with an existing track. When a blob is found to correspond to a single unique track, the track update is simple; we just associate the blob with the track. If a track is no longer visible for some period of time, it will be considered *stale* and deleted. Special handling is required for cases in which a new blob overlaps multiple old tracks (*merges*) or multiple new blobs overlap the same old track (*splits*). To handle these cases, our method evaluates a similarity function $S(b, t)$ for each candidate blob-track pair (b, t). $S(b, t)$ is based on the color coherence vector (CCV; Pass et al., 1996); when tracks merge, we group them, but keep their identities separate, and when tracks split, we associate the new blobs with the correct tracks or groups of tracks by comparing their CCVs, on the assumption that when an object eventually separates from the group, its appearance will be similar to its appearance before the merge. Our tracking method performs very well on typical simple cases such as those shown in Figure 2.12 involving clear trajectories of each moving person or object. However, it still has some problems with more complex cases such as those shown in Figure 2.13 involving large groups of people or objects moving together and interacting for some period of time.

Once the blob-to-track association is performed, we represent each track i at time t by a feature vector

$$\mathbf{f}_i^t = \begin{bmatrix} x_i^t & y_i^t & s_i^t & r_i^t & dx_i^t & dy_i^t & v_i^t \end{bmatrix},$$

where (x_i^t, y_i^t) is the centroid of the blob associated with track i. s_i^t is the size or area, in pixels, of the detected blob. r_i^t is the aspect ratio of the blob's bounding box, calculated by dividing the width of the bounding box by its height. (dx_i^t, dy_i^t) is the unit motion vector for the blob associated with track i at time t compared to track i at time $t-1$. v_i^t is a temporally smoothed version of the speed of the blob associated with track i, calculated as

$$v_i^t = r\frac{\sqrt{(x_i^t - x_i^{t-1})^2 + (y_i^t - y_i^{t-1})^2}}{\Delta t} + (1-r)v_i^{t-1},$$

where r is a constant (we use $r = 0.5$ in our experiments), Δt is the capture time difference between the frames at time t and $t-1$.

A preliminary report on the results for this appearance-based blob tracking method appeared in Gharti (2010). I worked with Gharti to produce the initial version of the method. Since the initial implementation was simplistic in some cases and some work was redundant, I improved the code to get better blob matching. In particular, the current implementation does not need to create and label each group of merged tracks; it uses only the information gained from the merged track association matrix. I also eliminated unnecessary cases in the blob-to-track association process.

Kalman filtering or other methods may be more optimal for filtering blob position and velocity over time, and tracking blobs with such filters could indeed improve the tracking results. In this dissertation, we find that they are unnecessary in practice because we observe relatively little noise in our blob position from the tracker, and we map the feature vectors to discrete symbols fairly coarsely in the next step.

2.6 Blob Feature Vector Discretization

For simplicity, we use discrete-observation hidden Markov models (HMMs) in this dissertation. This means that each feature vector $\mathbf{f}_i^t$ must be mapped to a discrete category (cluster ID) in the set $V = \{v_1, v_2, \ldots, v_U\}$, where U is the number of categories. We use k-means clustering based on a training set to map feature vectors to discrete clusters. Prior to clustering, we normalize each feature by a z-score as in Equation 2.1.

$$z = \frac{x - \mu}{\sigma} \qquad \text{(Equation 2.1)}$$

where μ is the mean and σ is the standard deviation of each feature. The scale factors are independent for all features except x_i^t and y_i^t, for which we apply a single isotropic scale factor.

In the training stage, we run k-means with U cluster centers then save the cluster centers as a codebook for later use. We select U based on the results of a model configuration selection procedure (to be discussed in Section 6.6.1). At run time, each blob feature vector is mapped to the nearest cluster center according to Euclidean distance then replaced by the corresponding cluster ID. This means that a behavior sequence is finally represented as a sequence of cluster IDs.

In the rest of this dissertation, when we mention the term "observation sequence," we mean a sequence of cluster IDs. The outputs to the next step are a list of blob feature vectors with the corresponding cluster IDs and bounding boxes, as well as the current frame and foreground mask.

2.7 Discussion

In this chapter, we segment each frame into background vs. foreground and extract blobs with features using Poppe et al.'s improved mixture of Gaussian background modeling method. We then link blobs in successive images to create distinct tracks for each moving blob in the scene using the forward-backward overlap technique. For occlusion handling, we use the color coherence vector (CCV) as an appearance model. When tracks merge, we group them, but keep their identities separate, and when tracks split, we attempt to associate the new blobs with the correct tracks or groups of tracks by comparing their CCVs.

The main limitation of our blob tracking method is that it is not robust for complex events involving multiple people. The method also does not allow evolution of the learned bootstrap model over time. Due to noise, fragmented blobs may affect the blob extraction, blob feature extraction, and blob tracking results. An effective method to handle blob fragmentation would be recommended for further improvement. In future work, we plan to address these limitations.

Chapter 3

Extracting the Object from the Shadows: Maximum Likelihood Object/Shadow Discrimination

In this chapter, we propose and experimentally evaluate a new method for detecting shadows using a simple maximum likelihood formulation based on color information. We first estimate, offline, a joint probability distribution over the difference in the HSV color space between pixels in the current frame and the corresponding pixels in a background model, conditional on whether the pixel is an object pixel or a shadow pixel. Given the learned distribution, at run time, we use the maximum likelihood principle to classify each foreground pixel as either shadow or object. In an experimental evaluation, we find that the method outperforms standard methods on three different real-world video surveillance data sets. We conclude that the proposed shadow detection method would be an extremely effective component in an intelligent video surveillance system.

3.1 Introduction

In many video surveillance applications, detecting and tracking moving objects is an important issue. A common approach to detect moving objects is to apply background a subtraction algorithm. However, background subtraction algorithms share one major disadvantage: shadows tend to be misclassified as part of the foreground object. This can lead to many undesirable consequences in the detection step while segmenting and extracting features of moving objects. For example, any estimate of the size of the detected object would be an overestimate due to the misclassification of shadow pixels as foreground pixels. Additionally, during object segmentation, shadows misclassified as moving objects could lead to merging otherwise separate blobs representing different people walking close to each other. This would make isolating and tracking people in a group much more difficult than necessary.

Since shadow removal can significantly improve the performance of computer vision tasks such as tracking, segmentation, and object detection, shadow detection has become an active research area in recent years.

Several well-known algorithms for shadow detection already exist. Most of the work is based on background modeling and color difference information. Generally, some model of the background is estimated, then the difference between the background and current image is used to identify changed pixels, then the changed pixels are further classified into object and shadow. Shadow pixels tend to have similar chromaticity but lower luminance than the corresponding background pixel. In the RGB color space, chromaticity and luminance are not orthogonal, but lighting differences can be controlled for in the normalized RGB color space (Finlayson et al., 1998). Some research work thus utilizes the normalized RGB space in the background subtraction and shadow removal algorithm (?, ?; Elgammal et al., 2002; Hong & Woo, 2003).

Mikić et al. (2000) observe that in the normalized RGB color space, shadow pixels tend to be more blue and less red than illuminated pixels. They apply a probabilistic model based on the normal-

ized red and blue features to classify shadow pixels in traffic scenes. The authors assume that the background and shadow values follow Gaussian distributions and foreground values follow a uniform distribution. They iteratively estimate the posterior probabilities of a given pixel belonging to each of three classes: background, shadow, and foreground, until one of the probabilities reaches a fixed threshold. The pixel is then classified accordingly. If none of the three probabilities reaches the threshold, the pixel is classified as background.

Salvador et al. (2004) propose a new method for detecting cast shadows. They first identify the presence of shadows in the RGB color space based on the fact that the surface on which shadows cast by the light is darkened. The detected regions are further verified based on the color invariance and geometric properties expected of shadows.

Havasi et al. (2006) illustrate that color-based methods work well for weak shadows but not strong shadows. Hence, they integrate geometric information into the detection process, resulting in an iterative Bayesian framework combining both color and geometric information that improves detection results.

One well-known problem with the normalized RGB space is that normalization of pixels with low intensity results in unstable chromatic components (Kender, 1976). Cucchiara et al. (2001) and Chen et al. (2008) propose a HSV color-based method to distinguish shadows from moving objects that eliminates this concern. Their approach is based on the assumption that only the intensity of the area covered by shadows will significantly change. Therefore, they detect shadows using the so-called deterministic nonmodel-based (DNM) approach as follows:

$$SP_t(x,y) = \begin{cases} 1 & \text{if } \alpha \leq \frac{I_t^V(x,y)}{B_t^V(x,y)} \leq \beta \\ & \quad \wedge (I_t^S(x,y) - B_t^S(x,y)) \leq T_S \\ & \quad \wedge \left| I_t^H(x,y) - B_t^H(x,y) \right| \leq T_H \\ \\ 0 & \text{otherwise,} \end{cases}$$

where $SP_t(x,y)$ is the resulting binary mask for shadows at each pixel (x,y) at time t. I_t^H, I_t^S, I_t^V, B_t^H, B_t^S, and B_t^V are the H, S, and V components of foreground pixel $I_t(x,y)$ and background pixel $B_t(x,y)$ at pixel (x,y) at time t, respectively. They prevent foreground pixels from being classified as shadow pixels by setting two thresholds, $0 < \alpha < \beta < 1$. The four thresholds α, β, T_S, and T_H are empirically determined.

Some researchers have investigated color spaces besides RGB and HSV. Blauensteiner et al. (2006) use an "improved" hue, luminance, and saturation (IHLS) color space for shadow detection to deal with the issue of unstable hue at low saturation by modeling the relationship between them. They then perform simple background subtraction method based on the IHLS color space and saturation-weighted hue statistics. Their experimental results show that detecting shadows in this color is more reliable than in normalized RGB or HSV color spaces in several video sequences.

Another alternative color space is YUV. Some applications such as television and videoconferencing use the YUV color space natively, and since transformation from YUV to HSV is time-consuming, Schreer et al. (2002) operate in the YUV color space directly, developing a fast shadow detection algorithm based on approximated changes of hue and saturation in the YUV color space.

There has been some work using texture-based methods such as the normalized cross-correlation (NCC) technique (Tian et al., 2005; Jacques Jr. et al., 2005). This method detects shadows based on the assumption that the intensity of shadows is proportional to the incident light, so shadow pixels should simply be darker than the corresponding background pixels. Under this assumption, shadow patches should be scaled versions of the corresponding background patches. This assumption is most valid in scenes with visible background texture inside the shadows. The method computes the NCC between the neighborhood of a pixel in the foreground mask and the neighborhood of the corresponding pixel in the background model. For each pixel (i, j) of the foreground mask, it considers a $(2N + 1) \times (2N + 1)$ template T_{ij} defined by $T_{ij}(n, m) = I(i + n, j + m)$ for $-N \leq n \leq N$ and $-N \leq m \leq N$ (N is empirically determined). If $B(i, j)$ is the background model, the NCC value at pixel (i, j) is defined as follows:

$$NCC(i, j) = \frac{ER(i, j)}{E_B(i, j)E_{T_{ij}}},$$

where

$$ER(i, j) = \sum_{n=-N}^{N} \sum_{m=-N}^{N} B(i + n, j + m)T_{ij}(n, m),$$

$$E_B(i, j) = \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} B(i + n, j + m)^2},$$

$$E_{T_{ij}} = \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} T_{ij}(n, m)^2}.$$

For a pixel in a shadow region, the NCC value should be large (close to one) and the $E_{T_{ij}}$ for the region around (i, j), i.e., its magnitude, should be smaller than $E_B(i, j)$. Consequently, a pixel is classified as shadow if

$$NCC(i, j) \geq L_{NCC}$$

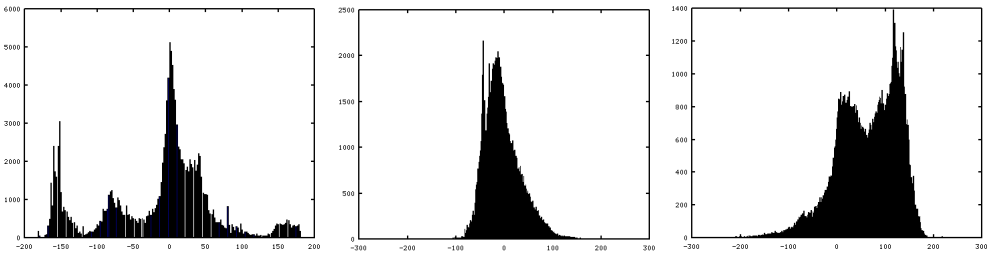
and

$$E_{T_{ij}} < E_B(i, j),$$

where L_{NCC} is an empirical threshold.

However, the texture-based method tends to misclassify foreground pixels as shadow pixels when the foreground region has a similar texture to the corresponding background region. Xu et al. (2005) propose a hybrid shadow removal technique that combines color and texture-based procedures to detect shadows. Since chromaticity in a shadow region should be the same as the corresponding background region, and since the texture in a shadow region should be the same as the corresponding background region, the authors first classify pixels based on a set of thresholds for brightness and color distortion then perform speckle removal filtering to reconstruct the final foreground shapes.

Here we propose a new method for detecting shadows using maximum likelihood estimation based on color information. We extend the deterministic nonmodel-based approach to parametric statistical model-based approach. Our method estimates the joint distribution over the difference in the HSV color space between pixels in the current frame and the corresponding pixels in a background model, conditional on whether the pixel is an object pixel or a shadow pixel. At run time, we simply use the maximum likelihood principle to classify each foreground pixel as either shadow or object given



(a) (b) (c)

Figure 3.1: Example distributions over the difference in (a) hue, (b) saturation, and (c) value components for true object pixels, extracted from our hallway dataset.

the estimated model. Experimental results demonstrate that our proposed method outperforms the standard methods (DNM and NCC) on three different real-world video surveillance data sets. Our method is thus effective and also has the potential to improve the object detection and motion analysis module in intelligent video surveillance systems.

In the rest of this chapter, I provide details of the proposed method and the overall process in Section 3.2, demonstrate the effectiveness of the shadow detection method with an experimental evaluation in Section 3.3, and then conclude and point to future work in Section 3.4.

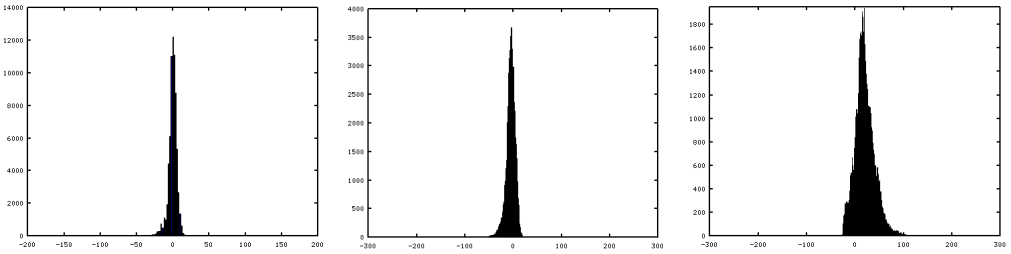
3.2 Maximum Likelihood Classification of Foreground Pixels

We divide our method into two phases. In the first, offline, phase, we acquire training video, construct a background model from the first few frames, perform foreground extraction on the remaining frames, then manually label the extracted pixels as either object pixels or shadow pixels. I previously described these steps in Sections 2.3 and 2.4. After that, we construct a joint probability model over the difference in the HSV color space between pixels in the current frame and the corresponding pixels in the background model, conditional on whether the pixel is an object pixel or a shadow pixel.

During the second, online, phase, we perform the same background modeling and foreground extraction procedure and further classify foreground pixels as either shadow or object using the maximum likelihood approach. I describe each of these steps in more detail in the following sections.

3.2.1 Offline Phase

After foreground extraction, we manually label pixels as either shadow or object. We then observe the distribution over the difference in hue (H_{diff}), saturation (S_{diff}), and value (V_{diff}) components in the HSV color space between pixels in the current frame and the corresponding pixels in the background model. Figure 3.1 shows examples of these distributions for object pixels, and Figure 3.2 shows examples of these distributions for shadow pixels.



(a) (b) (c)

Figure 3.2: Example distributions over the difference in (a) hue, (b) saturation, and (c) value components for true shadow pixels, extracted from our hallway dataset.

Clearly, in all three cases, the distributions for object pixels and shadow pixels are very different. We thus introduce a measurement probability distribution conditional on whether the assignment for a pixel is object or shadow. In this work, we assume that the individual component difference distributions are conditionally independent given the assignment.

We define the probability of measurement for pixel (x, y) given its assignment as follows.

$$\begin{aligned} P(M_{\text{xy}} \mid A_{\text{xy}} = \text{sh}) \quad = \quad & P(H_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) \times \\ & P(S_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) \times \\ & P(V_{\text{diff}} \mid A_{\text{xy}} = \text{sh}), \end{aligned} \qquad \text{(Equation 3.1)}$$

where M_{xy} is a tuple containing the HSV value for pixel (x, y) in the current image as well as the HSV value for pixel (x, y) in the background model for pixel (x, y), and A_{xy} is the assignment of pixel (x, y) as object or shadow. "sh" stands for shadow.

We assume that the distributions over the components on the right hand side in Equation 3.1 follow Gaussian distributions, defined as follows.

$$P(H_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) = \mathcal{N}(H_{\text{diff}}; \mu_{h_{\text{diff}}^{\text{sh}}}, \sigma^2_{h_{\text{diff}}^{\text{sh}}})$$

$$P(S_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) = \mathcal{N}(S_{\text{diff}}; \mu_{s_{\text{diff}}^{\text{sh}}}, \sigma^2_{s_{\text{diff}}^{\text{sh}}})$$

$$P(V_{\text{diff}} \mid A_{\text{xy}} = \text{sh}) = \mathcal{N}(V_{\text{diff}}; \mu_{v_{\text{diff}}^{\text{sh}}}, \sigma^2_{v_{\text{diff}}^{\text{sh}}})$$

Although the distributions for the true object pixels are clearly non-Gaussian as seen in Figure 3.1, I plan to extend this chapter in future work, but in this version, I simply use the sample mean and variance. However, on extension of the work, we will definitely investigate better estimators and different distributions.

Similarly, the probability of measurement given its assignment for object pixels can be computed as follows.

$$\begin{aligned} P(M_{\text{xy}} \mid A_{\text{xy}} = \text{obj}) \quad = \quad & P(H_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) \times \\ & P(S_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) \times \\ & P(V_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) \end{aligned} \qquad \text{(Equation 3.2)}$$

Here "obj" stands for object. As for the shadow pixel distributions, we assume Gaussian distributions

(a) (b) (c)

Figure 3.3: Sample frames from the (a) Hallway, (b) Laboratory, and (c) Highway video sequences.

over the components on the right hand side in Equation 3.2, as follows.

$$P(H_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) = \mathcal{N}(H_{\text{diff}}; \mu_{h_{\text{diff}}^{\text{obj}}}, \sigma^2_{h_{\text{diff}}^{\text{obj}}})$$

$$P(S_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) = \mathcal{N}(S_{\text{diff}}; \mu_{s_{\text{diff}}^{\text{obj}}}, \sigma^2_{s_{\text{diff}}^{\text{obj}}})$$

$$P(V_{\text{diff}} \mid A_{\text{xy}} = \text{obj}) = \mathcal{N}(V_{\text{diff}}; \mu_{v_{\text{diff}}^{\text{obj}}}, \sigma^2_{v_{\text{diff}}^{\text{obj}}})$$

We estimate the parameters $\Theta = \{\mu_{h_{\text{diff}}^{\text{sh}}}, \sigma^2_{h_{\text{diff}}^{\text{sh}}}, \mu_{s_{\text{diff}}^{\text{sh}}}, \sigma^2_{s_{\text{diff}}^{\text{sh}}}, \mu_{v_{\text{diff}}^{\text{sh}}}, \sigma^2_{v_{\text{diff}}^{\text{sh}}}, \mu_{h_{\text{diff}}^{\text{obj}}}, \sigma^2_{h_{\text{diff}}^{\text{obj}}}, \mu_{s_{\text{diff}}^{\text{obj}}}, \sigma^2_{s_{\text{diff}}^{\text{obj}}}, \mu_{v_{\text{diff}}^{\text{obj}}}, \sigma^2_{v_{\text{diff}}^{\text{obj}}}\}$ directly from training data during the offline phase.

3.2.2 Online Phase

Given the model estimate Θ, we use the maximum likelihood approach to classify a pixel as a shadow pixel if

$$P(M_{\text{xy}} \mid A_{xy} = \text{sh}; \Theta) > P(M_{\text{xy}} \mid A_{xy} = \text{obj}; \Theta). \qquad \text{(Equation 3.3)}$$

Otherwise, we classify the pixel as an object pixel.

We could add the prior probabilities to the shadow model and the object model in Equation 3.3 to obtain a maximum a posteriori classifier. In our experiments, we assume equal priors.

3.3 Experimental Results

In this section, we present experimental results for our proposed maximum likelihood (ML) classification method and compare the results with two other methods from the literature, namely the deterministic nonmodel-based (DNM) method (Kender, 1976; Cucchiara et al., 2001) and the normalized cross-correlation (NCC) method (Tian et al., 2005; Jacques Jr. et al., 2005).

We performed the experiments on three video sequences. Figure 3.3 shows sample frames from the three video sequences. The video sequences include both indoor and outdoor scenes. The *Hallway*

sequence[1] shows a hallway scene. For this video, we mounted a CCTV camera to record in an academic building. The *Laboratory* sequence shows a laboratory room, and the *Highway* sequence shows a traffic scene. The last two video sequences were first introduced in Prati et al.'s work.

To evaluate the performance of the methods, we compute the two metrics proposed by Prati et al. (2003), defining the shadow detection rate η and the shadow discrimination rate ξ as follows:

$$\eta = \frac{TP_{\text{sh}}}{TP_{\text{sh}} + FN_{\text{sh}}};\ \xi = \frac{TP_{\text{obj}}}{TP_{\text{obj}} + FN_{\text{obj}}},$$

where the subscript "sh" and "obj" stand for shadow and object, respectively. TP and FN are the number of true positive (i.e., the shadow or object pixels correctly identified) and false negative (i.e., the shadow or object pixels classified incorrectly) pixels. η expresses the proportion of shadow pixels correctly detected, and ξ expresses the proportion of object pixels correctly detected. η and ξ can also be thought of as the true positive rate (sensitivity) and true negative rate (specificity) for detecting shadows, respectively. In the experiment, we also compare the methods with the additional two metrics: precision and F_1 score.

3.3.1 Preparation

Ground truth data are provided with the *Laboratory* and *Highway* video sequences in Sanin et al.'s work. They used a standard Gaussian mixture (GMM) background model to extract foreground pixels for the two videos. They selected 20 frames including objects from the *Laboratory* sequence arbitrarily for labeling. For the *Highway* sequence, they labeled one out of every twenty frames including objects for a total of 20 frames. For our *Hallway* video sequence, to prepare similar ground truth data, we selected one out of every ten frames including objects for 20 frames and manually labeled each pixel of each frame as object, background, or shadow. We used the previously mentioned extended version of the GMM background model for foreground extraction, but the results were not substantially different from those of the standard GMM.

To find the best parameters for each model while avoiding overfitting, for each of the three models and each of the three data sets, we performed five-fold cross validation using 10 of the training frames, reserving the remaining 10 frames for the final test. The 10 frames in each case were the second of every two frames in sequential order. We varied the parameter settings for each method on each video dataset and selected the setting that maximized the F_1 score (a measure combining both precision and recall) over the cross validation test sets. Finally, we tested on the remaining 10-frame final test set for each video sequence.

[1]Freely available for others to experiment with at: `http://www.kanouivirach.com/#downloads`.

Table 3.1: Comparison of shadow detection results between the proposed, DNM, and NCC methods.

Method	Hallway				Laboratory				Highway			
	η	ξ	Precision	F_1	η	ξ	Precision	F_1	η	ξ	Precision	F_1
ML	95.36%	91.13%	91.41%	93.34%	98.41%	75.56%	88.52%	93.14%	93.38%	45.27%	51.58%	66.42%
DNM	90.71%	76.4%	79.17%	84.49%	96.59%	69.93%	85.27%	90.51%	48.77%	82.7%	62.78%	54.51%
NCC	84.94%	94.15%	93.95%	89.17%	79.42%	95.43%	96.73%	86.71%	9.45%	99.93%	98.62%	17.04%

3.3.2 Shadow Detection Performance

Table 3.1 compares the shadow detection results between the proposed, DNM, and NCC methods. Our method achieves the top performance for shadow detection rate η and F_1 score in every case. We also obtain a good shadow discrimination rate ξ and precision in all three video datasets. Figure 3.4 shows the results for an arbitrary frame in each video sequence. Green pixels are those labeled as object pixels and red pixels are those labeled as shadow pixels. The results in the figure confirm that our proposed method clearly outperforms the two standard methods in all three video datasets.

The DNM method has stable performance for all three videos, with good performance for all metrics. Both the DNM method and our proposed method suffer from the problem that the object colors can be confused with the background color. In the *Highway* sequence (third row in Figure 3.4), we clearly see this situation. Our method detects shadows well but misclassifies some object pixels as shadow, whereas DNM sometimes better discriminates the shadow from the object. However, the overall performance of our proposed method is superior.

The NCC method achieves the best shadow discrimination rate ξ and precision. However, as can be seen in Figure 3.4, this is because it classifies nearly every pixel as object. This gives NCC an advantage for shadow precision and ξ but on the other two metrics, shadow detection rate η and F_1 score, NCC performs extremely poorly in all cases. This is due to unclear background texture inside the shadows, particularly on the *Highway* sequence.

3.4 Discussion

We propose a new method for detecting shadows using a simple maximum likelihood approach based on color information. We extend the deterministic nonmodel-based approach, designing a parametric statistical model-based approach. Our experimental results show that our proposed method is extremely effective and superior to the standard methods on three different real-world video surveillance data sets.

In some cases, our method misdetects shadow pixels due to similar color between the object and the background and unclear background texture in shadow regions. Incorporating geometric or shadow region shape priors would potentially improve the detection and discrimination rates.

Additionally, the distributions over the difference in HSV color space for true object pixels and true shadow pixels remain static once trained. Therefore, the current approach may be inappropriate in

the long run. A more adaptive approach should be considered.

In future work, we plan to address these issues, further explore the feasibility of combining our method with other useful shadow features, and integrate our shadow detection module with a real-world open source video surveillance system (Coombes, 2007).

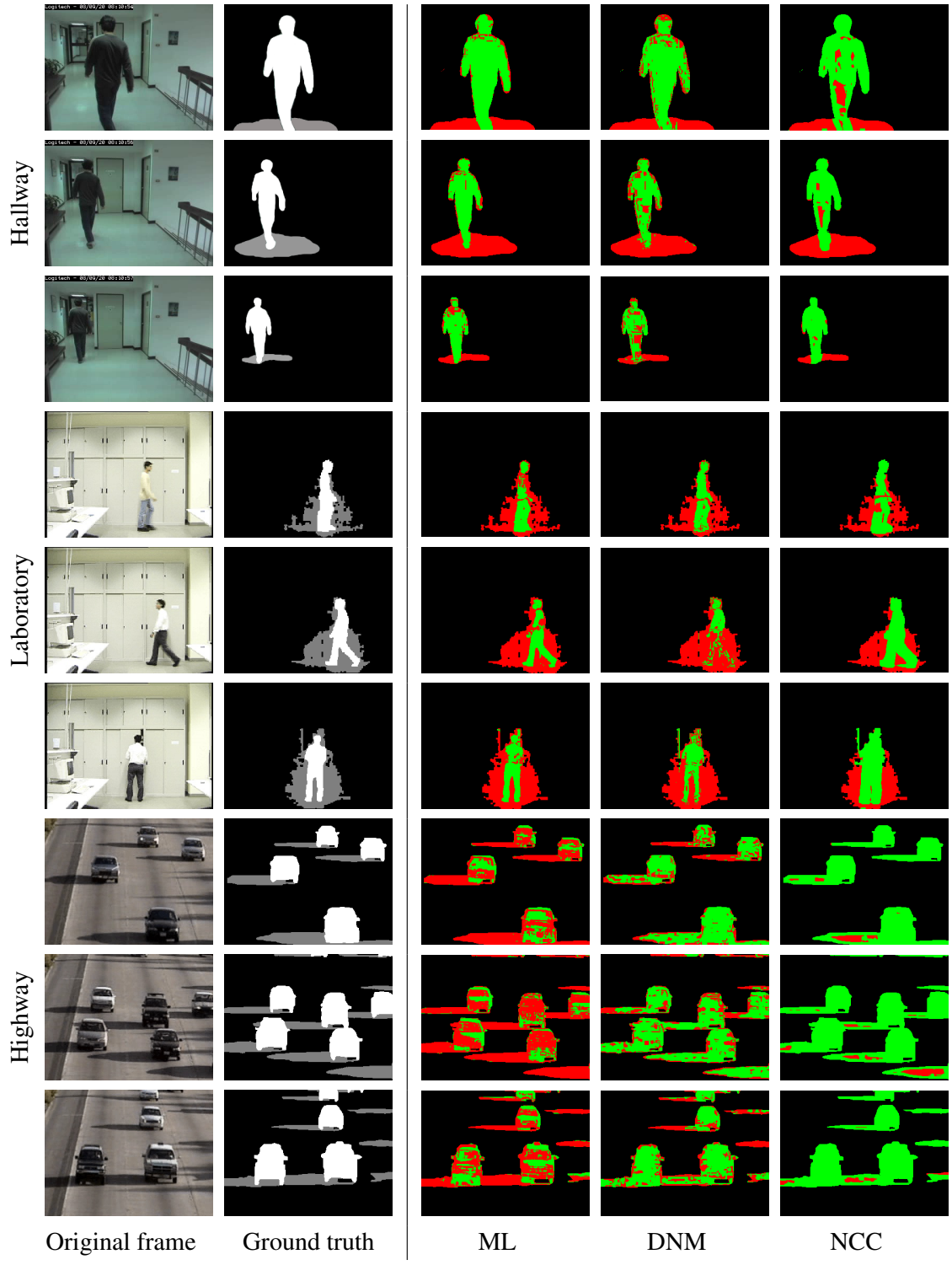


Figure 3.4: Results for an arbitrary frame in each video sequence. The first column contains an example original frame for each video sequence. The second column shows the ground truth for that frame, where object pixels are labeled in white and shadow pixels are labeled in gray. The remaining columns show shadow detection results for each method, where pixels labeled as object shown in green and pixels labeled as shadow are shown in red.

Chapter 4

Clustering Human Behaviors with Dynamic Time Warping and Hidden Markov Models

In this chapter, we propose and experimentally evaluate a new method for clustering human behaviors that is suitable for bootstrapping an anomaly detection module for intelligent video surveillance systems. The method uses dynamic time warping, agglomerative hierarchical clustering, and hidden Markov models to provide an initial partitioning of a set of observation sequences then automatically identifies where to cut off the hierarchical clustering dendrogram. We show that the method is extremely effective, providing 100% accuracy in separating anomalous from typical behaviors on real-world testbed video surveillance data.

4.1 Introduction

Human behavior understanding is an important component of a wide variety of desirable intelligent systems. However, the problem is very difficult, due to the wide range of activities possible in any given context and the large amount of variability within any particular activity. Many researchers have attempted to build systems able to interpret and understand human behaviors. The most classic work is from Yamato et al. (1992), who model tennis actions using hidden Markov models (HMMs). Du et al. (2006) present an approach to recognize interaction activities using dynamic Bayesian networks (DBNs) that outperforms conventional HMMs. Gao et al. (2004) use a single mixture-of-Gaussian HMM, in which the states represent the stages of dining activities, to monitor the eating behavior of elderly people in a nursing home.

As video monitoring is becoming more ubiquitous in our lives, research on advanced video surveillance analysis is increasingly important. To help security personnel work reliably and efficiently, we would like to filter out typical events, and in cases of anomalous events, automatically raise an alarm or present the event to a human operator for consideration as a security threat. Most existing work assumes that the number of "normal" behavior patterns need to be known beforehand. For example, Nair and Clark (2002) built an automated video surveillance system using HMMs, each modeling a common, predefined activity in a scene.

In more recent years, research has started to focus on unsupervised analysis and clustering of behaviors in a particular scene for a variety of purposes including anomaly detection, surveillance, and classification. H. Zhong et al. (2004) treat video segments as documents and cluster the documents based on the co-occurrence information. H. Li et al. (2006) cluster human gestures by constructing an affinity matrix using dynamic time warping (DTW; Sakoe, 1978) then apply the normalized-cut approach to cluster the gestures. Hautamäki et al. (2008) apply DTW and use the pairwise DTW distances as input to a hierarchical clustering process in which k-means is used to fine-tune the output.

Here we use a combination of clustering and HMMs to group human behaviors in a scene. There is some recent related work using graphical models such as HMMs to cluster behavior patterns. C. Li and Biswas (1999) use the Bayesian information criterion (BIC) for HMM model selection and

construct a binary hierarchical clustering dendrogram to initialize data partitions based on a sequence-to-model likelihood distance measure. They then compare each pair of clusters using a partition mutual information (PMI) criterion (Bahl et al., 1986) to find the optimal number of clusters.

Xiang and Gong (2005) model the distribution of activity data in a scene using a Gaussian mixture model (GMM) and also employ the BIC to select the optimal number of behavior classes prior to HMM training.

Swears et al. (2008) propose hierarchical HMM-based clustering to find and cluster motion trajectories and velocities in a highway interchange scene. They build up a set of HMMs incrementally. For each new trajectory, they first test the likelihood of the trajectory according to each existing HMM model. If the new observation is not fit by any existing model, it is considered deviant and is grouped with other deviant observations to form a new HMM.

Alon et al. (2003) propose a method to discover groupings of similar object motions. They apply a finite mixture of HMMs where the number of mixture components is assumed to be known. They estimate the number of clusters using the minimum description length (MDL) criterion (Rissanen, 1998), a penalized likelihood measure.

In this dissertation, we propose a new method for clustering human behaviors in the context of video surveillance. After extracting sequences of features representing individual human behaviors in a given scene, we use DTW to measure the pairwise similarity between sequences. Then we construct an agglomerative hierarchical clustering dendrogram based on the DTW similarity measure. To find the optimal set of behavior clusters, we start at the root of the tree, train a HMM on the patterns in that cluster, and determine how well the HMM models the set of patterns in the cluster. When we find that a HMM is an insufficient representation of the patterns in a given cluster, we throw away that HMM and recursively consider each of the child clusters according to the pre-calculated DTW-based dendrogram. Our method is able to automatically find the common human behaviors occurring in a given scene. In an experiment with a testbed video surveillance data set, we find that the method separates typical behaviors and abnormal behaviors into separate sets of clusters with 100% accuracy.

The most similar related work is that of Oates et al. (2001), who first proposed the idea of using the DTW with HMMs to cluster time series. They use the DTW dendrogram cut off at an arbitrary depth as an initial partition of the training sequences, then they train HMMs on each partition iteratively until they have a set of HMMs that models all of the training sequences. They apply their method to simulated time series with good results but report obtaining poor clustering results in an experiment with real robot sensor data.

As we shall see in later chapters, the method improves upon the state of the art in intelligent video surveillance applications by bootstrapping human behavior classification and anomaly detection modules in a given installation. Once a set of initial clusters and corresponding HMMs representing typical behavior is determined, we can easily find which cluster a new sequence should fall into by performing statistical tests on the sequence's likelihood according to each HMM model. When the likelihood is low according to all of the pre-existing clusters, we can consider the sequence to be anomalous and alert a human operator. When the likelihood is sufficiently high for one of the pre-existing models, we can simply incrementally update the sufficient statistics for that model. This approach would allow raising alerts for behaviors inconsistent with the automatically-derived typical behavior profile for the scene while providing adaptation to gradual changes in typical behavior pat-

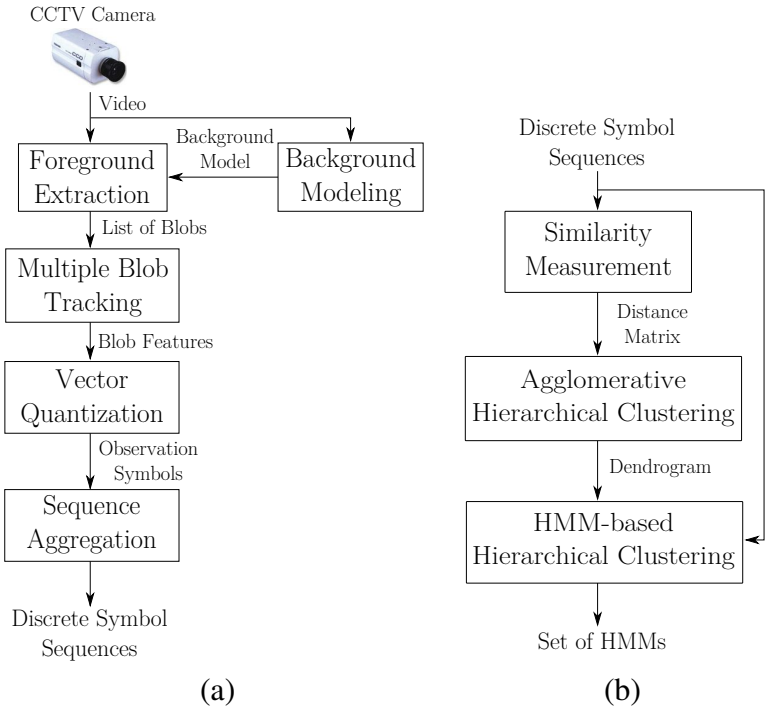


Figure 4.1: Block diagrams for the proposed method. (a) Blob extraction flow. (b) Behavior clustering flow.

terns over time. We do not focus on these incremental learning and anomaly detection issues in this chapter. We consider these issues in later chapters.

In the rest of this chapter, I provide the details of our human behavior clustering algorithm in Section 4.2, demonstrate the feasibility of the algorithm with an experimental evaluation in Section 4.3, and then conclude and point to future work in Section 4.4.

4.2 Human Behavior Pattern Clustering

4.2.1 Overview

Figure 4.1 provides an overview of the architecture of our proposed method. In this chapter, for simplicity, we evaluate our method only with a single moving blob. The blob extraction phase (Figure 4.1a) generates observation sequences from videos as follows:

1. Grab a few initial frames from an the input video to model the background scene.
2. Perform foreground extraction to get a list of blobs.
3. Find the single largest blob in the scene, remove any pixels likely to be shadow pixels, and extract feature vector $\mathbf{f}_t$ for the blob at time t. This could lead to situations where multiple objects

(a) (b) (c)

Figure 4.2: Sample foreground extraction and shadow removal results. (a) Original image. (b) Foreground pixels according to background model. (c) Foreground pixels after shadow removal.

are tracked as one, depending on the camera's perspective. For this reason, the experiments in this chapter are limited to sequences only containing a single blob.

4. Apply vector quantization using the k-means algorithm to convert features into symbols.
5. Aggregate symbol sequences and store for batch cluster analysis.

In the behavior clustering phase (Figure 4.1b), for the set of all discrete symbol sequences, we perform the following steps:

1. Apply DTW to the set of sequences to construct a similarity-based distance matrix.
2. Run agglomerative hierarchical clustering on the distance matrix to get a dendrogram.
3. Perform HMM-based hierarchical clustering using the set of sequences and the dendrogram to get a set of HMMs modeling the typical behaviors in the scene.

4.2.2 Blob Extraction

Here we use the motion detection and blob extraction methods previously described in Sections 2.3 and 2.4, respectively.

In the work reported in this chapter, we use a simple normalized cross correlation (NCC) shadow elimination method described in Chapter 3 to eliminate shadows cast by moving objects. We compute the grayscale correlation between the foreground pixels and a background image constructed as the mean over each mixture of Gaussian distribution. Any foreground pixels whose NCC with the background are above some threshold are removed. We choose this method because it works well in outdoor scenes where background texture inside shadows is visible. The sample results from the foreground extraction and shadow removal procedures are shown in Figure 4.2. We use the NCC method for shadow removal through the rest of this dissertation.

We next apply morphological opening and closing operations by probing an image with a disk structure element of radius 5 to remove noise and connect foreground regions. The opening operation

of a binary image A by the predefined structuring element B is obtained by the erosion of A by B, followed by dilation of the resulting image by B, which is defined as

$$A \circ B = (A \ominus B) \oplus B,$$

The closing operation is of A and B is obtained by the dilation of A by B, followed by erosion of the resulting structure by B, which is defined as

$$A \bullet B = (A \oplus B) \ominus B.$$

We then obtain the connected foreground components (blobs) and filter out any components whose size is below threshold. In this chapter, for simplicity, we evaluate our clustering method with videos containing a single moving blob. We have extended the algorithm to handle multiple moving blobs in next chapters.

We therefore follow the process previously described in Section 2.5 for representing a blob (connected foreground component) by a feature vector, normalizing each feature independently, and quantizing the feature vectors into discrete symbols using k-means.

Currently, we empirically tune the free parameters (frame buffer length, thresholds, and number of k-means clusters) to the training data.

In particular, we set up our system and observed the scene for a few days before we started collecting data. We manually fine-tuned the frame buffer length and motion detection thresholds for triggering events to achieve good results per manual observation. Other parameters critical to the modeling such as the number of k-mean clusters were set through cross validation. We hope to automate the blob extraction parameter selection process in future work.

4.2.3 Behavior Clustering

We model the common behaviors in a scene by clustering a set of observation sequences acquired over some period of time. First, we apply dynamic time warping (DTW) to estimate the similarity between every pair of training sequences despite variations in length and speed, to obtain a similarity matrix. Second, we use the similarity matrix for hierarchical agglomerative clustering by first combining the most similar two sequences into a single cluster then repeatedly merging clusters until just one cluster is left at the root of the tree or dendrogram. To determine the similarity of two clusters during this step, we use the similarity of the most similar pair of sequences between the two clusters.

The resulting DTW dendrogram provides a convenient representation of the similarity structure within a set of time sequences, but hierarchical clustering always comes with the practical issue of determining the optimal cutoff or number of clusters to use in a particular application. We solve this problem using HMMs as described below.

The flow of the algorithm is summarized in Figure 4.3. We begin at the root of the hierarchical clustering dendrogram and attempt to model the sequences in that cluster (all training sequences, for the root) using a HMM. When there are more than N sequences in parent cluster c whose per-

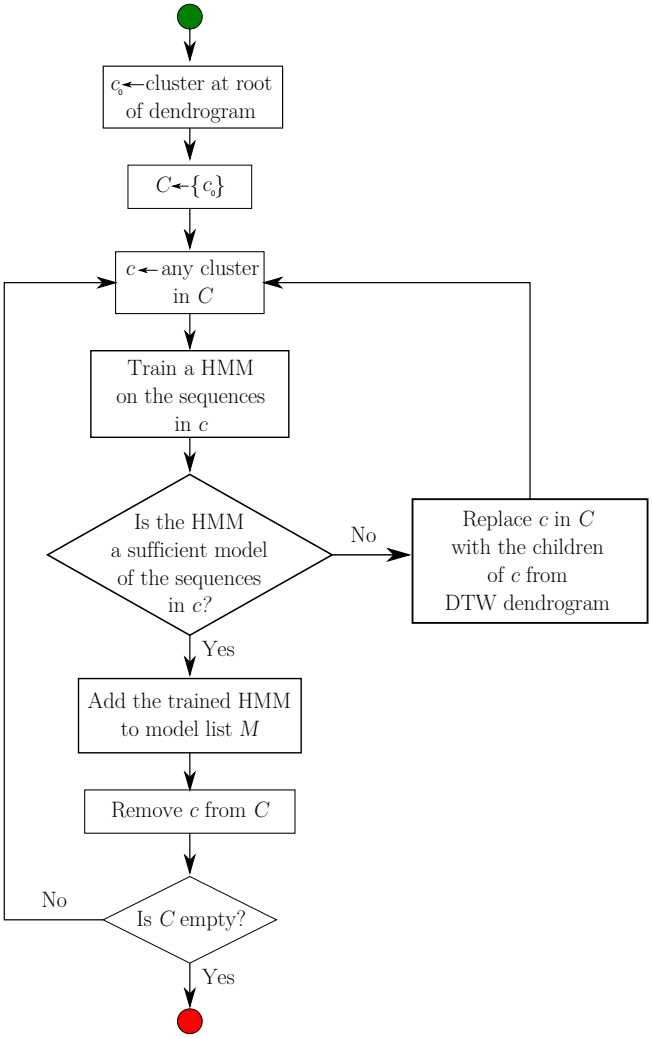


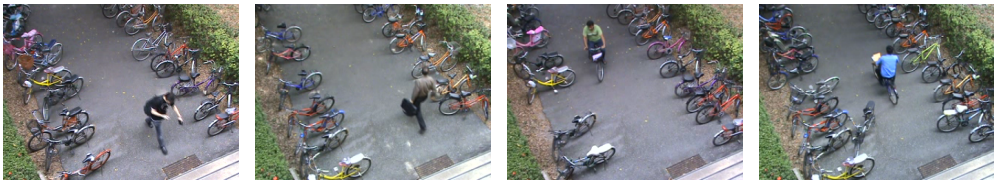
Figure 4.3: Processing flow of the use of HMM clustering method.

observation log likelihood is less than a threshold p_c, we consider the HMM to be inadequate, throw it away, and then recursively attempt to model each of c's children in the DTW dendrogram. We use $N = 10$ in our experiments. The per-observation log of the probability of a sequence $\mathbf{O}_i = \{O_{i,1}, O_{i,2} \cdots O_{i,T_i}\}$ given the HMM M_c is

$$L_i = \frac{\log P(\mathbf{O}_i \mid M_c)}{T_i}, \qquad \text{(Equation 4.1)}$$

where M_c is the HMM that models the sequences in cluster c, T_i is the number of observations in sequence i, and $P(\mathbf{O}_i \mid M_c)$ is calculated using the forward algorithm (Rabiner, 1989).

To determine the optimal rejection threshold p_c for cluster c, we use an approach similar to that of Oates et al. (2001). We generate random sequences from the HMM and then calculate the mean μ_c



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

Figure 4.4: Examples of common human activities in our testbed scene. (a) Walking in. (b) Walking out. (c) Cycling in. (d) Cycling out.

and standard deviation σ_c of the per-observation log likelihood over the set of generated sequences. For the lengths of the generated sequences, we simply use the average length of the training patterns in cluster c. After obtaining the statistics of the per-observation log likelihood, we let p_c be $\mu_c - z\sigma_c$, where z is an experimentally tuned parameter that gives us convenient control over the probability of making Type I errors in classifying a particular sequence as having been generated by a particular HMM model.

4.3 Experimental Results

To create a testbed data set, we mounted a CCTV camera to view the scene in front of an academic building, as seen in Figure 4.2a. We recorded videos at a resolution of 320×240 and 25 frames per second over one week during working hours (9:00–17:00). To save disk space, we used a motion detection technique that automatically segments a raw video stream into separate videos containing motion. We obtained videos corresponding to over 500 motion events then manually selected the 298 videos containing only a single motion.

We found that there are at least four common behaviors in this scene: people walking into the building, walking out of the building, parking a bicycle, and riding a bicycle out. Figure 4.4 shows examples of each of these behaviors. Other less common activities include people walking into the scene then walking out or people walking while telephoning and leaving the scene. For purposes of evaluating the results of our algorithm, we hand-labeled each of the videos with the categories Walk-in, Walk-out, Cycle-in, Cycle-out, or Other.

We performed three experiments to evaluate our method. In Experiment I, we applied our proposed method, as previously described, to cluster the 298 single-motion videos in our testbed data set. In Experiment II, we used an alternative clustering method that only uses recursive modeling by HMMs, without DTW. In Experiment III, we used an alternative method combining HMMs with supervised learning. In every experiment, we evaluated the clustering results (or classification results in the case of the supervised system of Experiment III) according to how well the induced categories separate the anomalous sequences (hand-labeled with the category "Other") from the typical sequences (Walk-in, Walk-out, Cycle-in, Cycle-out). Our main hypothesis was that *using DTW as a pre-process prior to HMM-based clustering should improve the quality of the clusters* in terms of separating anomalous from typical behaviors. One might also have hypothesized that supervised learning (Experiment III) would be better than either of the unsupervised methods (Experiments I and II), but as we shall see, we obtained a somewhat surprising result to the contrary.

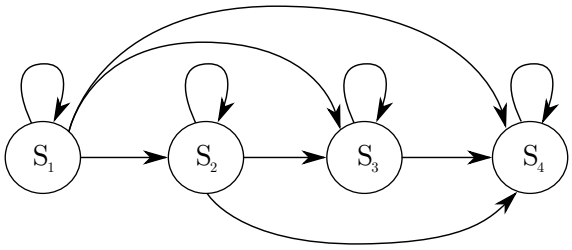


Figure 4.5: Example of linear HMM with bypass transitions, as used in all experiments.

In all three experiments, we used linear HMMs with four states and bypass transitions. That is, each HMM had transitions from state 1 to states 1, 2, 3, and 4, from state 2 to states 2, 3, and 4, from state 3 to states 3 and 4, and from state 4 to itself. See Figure 4.5 for an example of a 4-state HMM with bypass transitions. We chose this model structure based on our previous empirical experience (Ouivirach, 2008).

To find the distribution (parameters μ_c and σ_c) of the per-observation log likelihood for a particular HMM, we always generated 1000 sequences of 120 observations then used a z-threshold of 2.0, corresponding to a Type I error (probability of misclassifying a sequence generated by the HMM as not generated by the HMM) of 0.0228. We fixed the parameter N (the number of deviant patterns allowed in a cluster) to 10.

4.3.1 Experiment I (DTW+HMMs)

The clustering results are shown in Table 4.1. The method obtained 97 clusters. For the 17 clusters containing more than one sequence, we show the distribution of the activities represented by each sequence. For the 80 clusters containing only a single sequence, we summarize their distribution across the activity categories in the last row of the table.

It is clear from the results that the separation of anomalous and typical behaviors is excellent (100% accuracy), with the caveat that 80 sequences (26.8% of the data set) fall into single-sequence clusters that would have to be manually examined by a human operator if the method was used to bootstrap a real-world surveillance system.

It is worth pointing out that even though our method shows an excellent separation between typical and anomalous behaviors, it also creates a large number of single-sequence clusters. This could be due to the relatively unbalanced dendrogram structure constructed in the experiment as shown in Figure 4.6. Finding an algorithm to maintain a balanced dendrogram structure could improve the clustering results and reduce the number of single-sequence clusters.

Additionally, since the current system treats each (z-scaled) feature equally when it performs blob feature vector discretization, important fine distinctions for some features can be lost. Adding more weight to features such as blob size or aspect ratio could help distinguish similar behaviors such as Walking In and Cycling In.

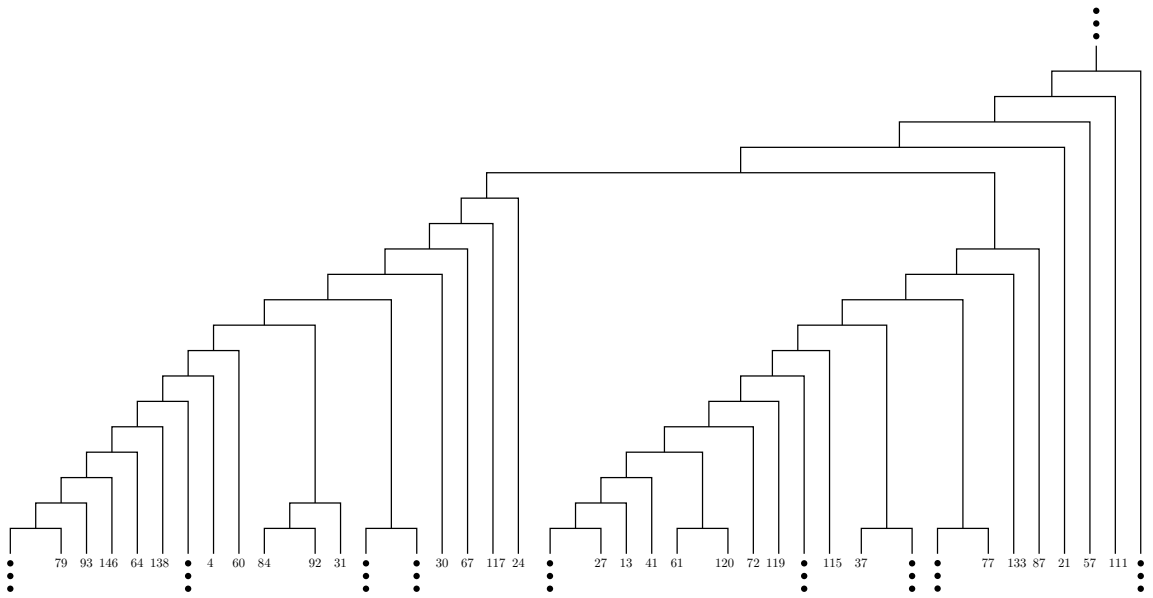


Figure 4.6: Partial dendrogram constructed in Experiment I. Numbers represent sequence numbers, and ⋮ indicates additional structure not shown.

4.3.2 Experiment II (HMMs only)

To determine the extent to which our system benefits from preprocessing using DTW, in this experiment, we used the concept of recursive modeling of the data using HMMs without using DTW as a pre-process. The method is similar to that of Swears et al. (2008). We begin by training a single HMM on all sequences and computing the distribution of the per-observation log likelihood for that HMM as previously described. We then assign every sequence with a per-observation log likelihood above threshold p_c to a cluster then repeat the process by training a new HMM on the remaining sequences. Similarly to the method of Experiment I, we stop splitting whenever the number of deviant sequences in the cluster is less than 10.

The results are shown in Table 4.2. The method obtains only three clusters, and the clusters are incapable of separating anomalous behaviors from typical behaviors. With manual assignment of all three clusters to the "typical" category, we would achieve a recall of 0, a precision of 100%, and an accuracy of 94.6%. By manually assigning cluster 1 to the anomalous category, we would achieve a recall of 100%, a precision of 8%, and an accuracy of 38.3%. These strikingly poor results confirm our main hypothesis, and we conclude that recursive HMM modeling without DTW-based preprocessing is useless for video surveillance.

4.3.3 Experiment III (Supervised classification with HMMs)

As an alternative approach to separate anomalous from typical behaviors, in this experiment, we trained four HMMs on 80% of each of the four typical behaviors observed in our testbed data set. We retained 20% of each typical sequences and the 16 anomalous sequences as a test set. After HMM

training, we manually determined the best per-observation log likelihood threshold for each HMM by maximizing the F1 value (a measure combining both precision and recall) for the separation between the positive and negative test patterns. The anomalous pattern detection rate of the combined classifier was 50% (8 patterns) with a false alarm rate of 24.6% (16 out of 65 normal testing patterns).

A priori, one might have hypothesized that supervised sequence classification should outperform the unsupervised method we have proposed in this chapter. To the contrary, we find that the simple-minded approach of training a single HMM on each typical behavior is far inferior. Clearly, there is more variation within the behavior categories than can be handled precisely by single simple HMMs. These supervised results could presumably be improved by applying our clustering method within each behavior category, thus obtaining a collection of HMMs for each behavior category. However, we have already demonstrated that our method achieves perfect separation of anomalous and typical behaviors *without any information about the labels*, so this approach would be unnecessarily laborious.

4.4 Discussion

In this chapter, we have proposed and evaluated a new method for clustering human behaviors. As we shall see, the method can be used to bootstrap an anomaly detection module for intelligent video surveillance systems. The combination of DTW partitioning with linear HMM training with bypass transitions turns out to be quite powerful; manual examination of the clusters obtained from our method shows a perfect separation between typical and anomalous behaviors on a real-world testbed video surveillance data set.

The combination of DTW with the type of linear HMMs with bypass transitions we use in this work is surprisingly effective. It is likely that the patterns DTW groups together are perfectly suited for modeling by this type of HMM. We plan to further explore this idea in future work.

There are two key limitations to our current method. The first is that we limited the data to single motion events. The remaining chapters will eliminate that constraint. The second is that although the method achieves 100% accuracy in separating typical from anomalous events, it does so at the cost of creating a fairly large number of single-sequence clusters that would have to be manually identified as typical or anomalous by a human operator in a real surveillance setting.

Table 4.1: Clustering results for Experiment I (DTW+HMMs).

Cluster #	Walk-in	Walk-out	Cycle-in	Cycle-out	Other
1	96	0	18	0	0
2	0	54	0	5	0
3	0	3	0	8	0
4	0	2	0	0	0
5	0	1	0	2	0
6	0	0	0	2	0
7	0	0	0	2	0
8	0	0	0	2	0
9	0	0	0	2	0
10	0	0	2	0	0
11	0	0	2	0	0
12	0	0	3	0	0
13	0	0	1	1	0
14	0	0	0	0	4
15	0	0	0	0	4
16	0	0	0	0	2
17	0	0	0	0	2
One-seq clusters	4	17	34	21	4

Table 4.2: Clustering results for Experiment II (HMMs only).

Cluster #	Walk-in	Walk-out	Cycle-in	Cycle-out	Other
1	15	77	49	43	16
2	80	0	11	2	0
3	5	0	0	0	0

Chapter 5

Automatic Suspicious Behavior Detection from a Small Bootstrap Set

In this chapter, we propose and evaluate a new method for automatic identification of suspicious behavior in video surveillance data. The approach works by constructing scene-specific statistical models explaining the behaviors occurring in a small bootstrap data set. It partitions the bootstrap set into clusters then assigns new observation sequences to clusters based on statistical tests of HMM log likelihood scores. Cluster-specific likelihood thresholds are learned rather than set arbitrarily. In an evaluation on a real-world testbed video surveillance data set, the method proves extremely effective, with a false alarm rate of 7.4% at a 100% hit rate. The method is thus a practical and effective solution to the problem of inducing scene-specific statistical models useful for bringing suspicious behavior to the attention of human security personnel.

5.1 Introduction

Video surveillance is ubiquitous in our lives, but the ongoing proliferation of surveillance cameras makes it increasingly difficult to monitor all channels continuously. As the amount of surveillance video increases, monitoring becomes more expensive and less effective. Security would be enhanced if it were possible to perform intelligent filtering of typical events and to automatically bring suspicious events to the attention of human security personnel.

The goal of automated suspicious event detection, however, requires some level of human behavior understanding. Many researchers have attempted to build intelligent systems able to interpret and understand human behaviors (L. S. Davis et al., 1998; Masoud & Papanikolopoulos, 2003; Cao et al., 2004; Sminchisescu et al., 2005; Du et al., 2006; H. Li et al., 2006; Wang et al., 2006). Pre-trained hidden Markov models (HMMs) and other dynamic Bayesian networks such as conditional random fields (CRFs) have been widely used in this area, ranging from visual activity recognition (Yamato et al., 1992; Sminchisescu et al., 2005; Xiang & Gong, 2005; Vail & Guestrin, 2007), gesture recognition (Wilson & Bobick, 2000; Gao et al., 2004), and unusual activity detection (Nair & Clark, 2002; Lee et al., 2003; Chan et al., 2004; Wu et al., 2005; Andrade et al., 2006; Snoek et al., 2006; Zhang et al., 2005; Arsić et al., 2007). However, the problem is difficult and remains unsolved, due to the wide range of activities possible in any given context and the large amount of variability within any particular activity. In the context of video surveillance, the problem is even more challenging because behavior considered normal in one scene might be considered unusual in another scene.

Here we propose a method for automatic suspicious behavior detection that utilizes a small bootstrap set in which observation sequences are manually labeled as normal or suspicious (warranting an operator's attention). Using the algorithm from Chapter 4, we partition the bootstrap set into clusters of similar sequences and model each cluster with a simple HMM. We then label each behavior cluster as normal or possibly suspicious based on the labels of the individual sequences mapped to the cluster.

After bootstrapping is complete, we assign new observation sequences to behavior clusters using

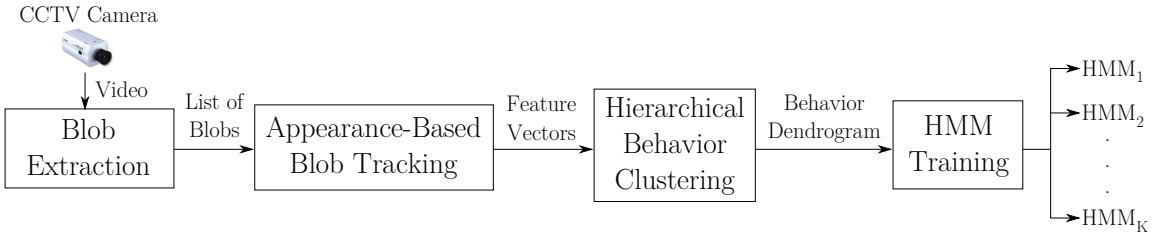


Figure 5.1: Block diagram of overall anomaly detection system.

statistical tests on the log likelihood of the sequence according to the corresponding HMMs. A sequence is considered suspicious if the most likely cluster is already labeled as suspicious or if its log likelihood according to the most likely cluster's HMM is too low. The cluster-specific likelihood threshold is learned rather than set arbitrarily.

In an evaluation on a real-world video surveillance situation, we find that, based on a bootstrap set of 150 human motion sequences, our method is extremely effective at identifying suspicious behavior, with a false positive rate of 7.4% at a hit rate of 100%. The proposed method dramatically outperforms traditional machine learning approaches using the same bootstrap set for training.

Our method is thus a practical and effective solution to the problem of inducing scene-specific statistical models useful for bringing suspicious behavior to the attention of human security personnel.

In the rest of this chapter, we provide the details of our human behavior modeling and bootstrapping algorithm in Section 5.2 and our anomaly detection method in Section 5.3. We demonstrate the feasibility of the algorithm with an experimental evaluation in Section 5.4, and then discuss and point to future work in Section 5.5.

5.2 Behavior Model Bootstrapping

Our method for bootstrapping a model of the specific behaviors in a scene is a batch procedure based on a training video stream acquired over a short period of time such as one week. We first perform moving blob tracking on the training video on the assumption that moving blobs of sufficient size are people or groups of people. For each blob, we extract a sequence of feature vectors describing the blob's trajectory and appearance over time. From these data, we automatically bootstrap a bank of linear HMMs with bypass transitions, each model specializing in one type of behavior.

Here we follow the processing steps for motion detection described in Section 2.3 and multiple blob tracking described in Section 2.4.

We define an "event" as a contiguous change or motion detected over some period of time. Extremely short events (occasionally occurring due to noise) are automatically removed before processing.

Next we use the tracking algorithm previously described in Section 2.5 to track the obtained blobs, extract a sequence of feature vectors describing each blob's trajectory and appearance over time, and

automatically bootstrap a bank of linear HMMs with bypass transitions, each model specializing in one type of behavior.

Note that one difference between the methods of this chapter and the previous chapter is that this chapter extends the simple single blob tracking method to the full multiple blob tracking method described in Chapter 2.

5.2.1 Behavior Clustering

After blob tracking, we obtain, from a given video, a set of observation sequences describing the motion and appearance of every distinguishable moving object in the scene. We next partition the observation sequences into clusters of similar behaviors then model the sequences within each cluster using a simple linear HMM with bypass transitions.

In this section, we use the method previously described in Chapter 4, which first uses dynamic time warping (DTW) to obtain a distance matrix for the set of observation sequences then performs agglomerative hierarchical clustering on the distance matrix to obtain a dendrogram (a binary tree expressing the similarity structure within the set of observation sequences). Once the set of all behaviors has been partitioned into groups containing similar sequences, as we shall see, each group can be modeled individually with a simple statistical model, the linear HMM with bypass transitions. Next, we can use the resulting bank of trained models for anomaly detection.

To determine where to cut off the dendrogram, we traverse the DTW dendrogram in depth-first order from the root and attempt to model the observation sequences within the corresponding subtree using a single linear HMM with bypass transitions. If, after training, the HMM is unable to "explain" (in the sense described below) the sequences associated with the current subtree, we discard the HMM then recursively attempt to model each of the current node's children. Whenever the HMM is able to explain the observation sequences associated with the current node's subtree, we retain the HMM and prune the tree.

A HMM is said to explain or model a cluster c of observation sequences if there are no more than N_c sequences in c whose per-observation log-likelihood is less than a threshold p_c. We use $N_c = 10$ in our experiments. The per-observation log likelihood of a sequence is previously defined in Section 4.2.3.

To determine the optimal rejection threshold p_c for cluster c, we use an approach similar to that of Oates et al. (2001). We generate random sequences from the HMM and then calculate the mean μ_c and standard deviation σ_c of the per-observation log likelihood over the set of generated sequences. For the lengths of the generated sequences, we simply use the average length of the sequences in the bootstrap set. After obtaining the statistics of the per-observation log likelihood, we let p_c be $\mu_c - z\sigma_c$, where z is an experimentally tuned parameter that gives us convenient control over the probability of making Type I errors in classifying a particular sequence as having been generated by a particular HMM model.

The clustering process results in a set of K different typical behavior clusters $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$ with a set of K corresponding HMMs $\mathcal{M} = \{M_1, M_2, \ldots, M_K\}$.

5.3 Anomaly Detection

In this section we describe our method for anomalous or suspicious behavior detection. In the *supervised* approach, one would construct a training set consisting of anomalous and normal behaviors, build a model, then use the model to classify new behavior sequences as anomalous or normal.

The pure supervised approach is obviously not suitable, however, when examples of anomalous behavior are sparse or nonexistent. In practical scenarios, the set of possible anomalous behaviors is infinite in its variety, making it very difficult to acquire a representative training set.

In the *unsupervised* approach, on the other hand, we would simply construct a generative model of the normal behavior patterns, then use the model to classify new behavior sequences as abnormal when they are "too far" in some sense from typical behavior. For example, we could use the algorithm described in Section 5.2.1 to construct a bank of HMMs explaining the normal sequences in a training set of patterns, then classify new sequences as abnormal or suspicious whenever the likelihood of the sequence according to the most likely HMM is below some specific threshold.

The difficulty with the pure unsupervised approach, however, is that there is no clear way to calibrate the parameters of the "too far" criterion. In practice, one would have to select a conservative initial distance threshold then fine-tune the threshold to achieve the best tradeoff between hit and false positive rates. An inappropriate initial cutoff could lead to disastrous misses of suspicious behavior or inundation with false positives.

Since both approaches have limitations, rather than the pure supervised method or the pure unsupervised method, we instead propose a *semi-supervised* method that self-calibrates itself from a small bootstrap set in which each bootstrap sequence is manually labeled as normal or suspicious by a human operator. Our method is simple. We acquire labels for the bootstrap patterns from the operator, then we apply the algorithm of Section 5.2.1 to *both the positive and negative sequences* in the bootstrap set. We identify each cluster as a "normal" cluster if *all* of the sequences falling into it are labeled as normal, or identify it as an "abnormal" cluster if *any* of the sequences falling into it are labeled as abnormal.

In the best case, if we updated the log likelihood for each model every time we receive a new observation token in constant time (this is feasible), the time complexity of the method would be O(nm), where n is the number of observations and m is the number of models. However, for convenience, we rerun the forward algorithm (Rabiner, 1989) on each observation for every model.

Algorithm 2 presents a pseudocode summary of the runtime anomaly detection method. New sequences are classified as normal if the most likely HMM for the input sequence is associated with a cluster of normal sequences *and* the z-scaled per-observation log of the probability of the sequence under that most likely model is greater than a global empirically determined threshold θ_z.

Algorithm 2 Anomaly Detection

Input: $\mathbf{O}$: behavior sequence

Input: $\mathcal{M}$: set of HMMs

$\mathcal{M}_{ab} \leftarrow \{M \mid M \in \mathcal{M}$ and M is marked abnormal$\}$

$(M_{ml}, L_{ml}) \leftarrow$ FIND-MOST-LIKELY-MODEL$(\mathbf{O}, \mathcal{M})$

if $M_{ml} \in \mathcal{M}_{ab}$ or $L_{ml} \leq \theta_z$ **then**

ALERT-SECURITY-PERSONNEL$(\mathbf{O})$

end if

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

Figure 5.2: Examples of common human activities in our testbed scene. (a) Walking in. (b) Walking out. (c) Cycling in. (d) Cycling out. (e) Other group activities.

5.4 Experimental Results

We created a testbed data set using a CCTV camera with a view of the front of a building, as seen in Figure 5.2(a). Videos were recorded at 25 frames per second with a resolution of 320×240 during working hours (9:00–17:00) for two weeks. We used the previously-described techniques to segment the raw video stream into separate videos containing motion, track blobs, extract features, discretize the feature vectors, and create observation sequences. We obtained 423 video segments containing 660 observation sequences. In total, 370 videos contain single sequences, and 53 videos contain multiple sequences. As previously mentioned, our blob tracking method is not perfect; some tracks are lost and some tracks are incorrectly associated with each other. See Figure 2.13 for an example of a tracking error.

We observed that there are four common activities in this scene: people walking into the building, people walking out of the building, people parking bicycles, and people riding bicycles out. Most of the remaining activities involve people walking past or having interactions with each other. Most multiple people interactions are considered normal because each of them includes multiple single activities. For instance, a person walking in has a conversation with another person walking out before he/she walks into the building. As a result, two single activities occur in this situation. Figure 5.2 shows examples of each of these behaviors. In our experiments, we considered all other behaviors, such as walking around looking for an unlocked bicycle, to be suspicious or abnormal.

To evaluate the effectiveness of our method, we divide the experiments into two parts: model configuration selection (finding an optimal set of HMMs to model the bootstrap set) and anomaly detection

based on the HMM bootstrap set. In all of the experiments, we used linear HMMs with bypass transitions. We chose this model structure based on our previous empirical experience (Ouivirach, 2008).

5.4.1 Model Configuration Selection

We first manually labeled each of the videos with the categories "normal" or "abnormal." For purposes of analyzing the results, we further subdivided the normal patterns into categories "walk-in," "walk-out," "cycle-in," and "cycle-out," but we did not use these labels for model selection or learning.

Towards model identification, we then performed a series of experiments with different bootstrap parameter settings and selected the configuration with the highest accuracy in separating the normal sequences from the abnormal sequences on the bootstrap sequence set, as measured by the false positive rate for abnormal sequences. (Every bootstrap cluster containing an abnormal sequence is considered abnormal, so we always obtain 100% detection on the bootstrap set; the only discriminating factor is the false positive rate.) The set of parameters that we varied were 1) the number of states in each bootstrap HMM (4–8), 2) the number of sequences used for bootstrapping (50–200), and the number of symbols used (4–8).

To find the distribution (parameters μ_c and σ_c) of the per-observation log likelihood for a particular HMM, we always generated 1000 sequences of 150 observations then used a z-threshold of 2.0. We fixed the parameter N_c (the number of deviant patterns allowed in a cluster) to 10.

A subset of the results of the model configuration selection experiments is shown in Figure 5.3. Based on the false positive rate criterion described above, we selected the model configuration consisting of HMMs with five states and seven tokens trained on 150 bootstrap sequences. We arbitrarily chose the model from one of the 10 trials with this configuration. Table 5.1 shows an example of the distribution of bootstrap sequences across behavior clusters with the selected model configuration. In this run, we found that, with 20 behavior clusters, anomalous and typical sequences are always mapped to different clusters, although the large number of one-sequence clusters means there would be a fairly large amount of manual work required to identify them.

We used this model with the remaining 510 sequences in the anomaly detection experiments described next.

5.4.2 Anomaly Detection

Here we describe four experiments to evaluate our anomaly detection method. In Experiment I, we applied our proposed method, as previously described, to detect anomalous events. In Experiments II, III and IV, for comparison, we applied traditional machine learning algorithms to the same problem. We use k-nearest neighbors (k-NN), a Mahalanobis classifier, and support vector machines (SVMs). For these methods, we use the 150-sequence bootstrap sequence set from Section 5.4.1 as training data and test on the remaining 510 sequences. Since the main concern in video surveillance is to

Table 5.1: Example human behavior pattern bootstrapping results. We used linear HMMs with five states and seven tokens and bypass transitions. The model consists of 20 clusters. "W" means "walk" and "C" means "cycle." For the seven clusters containing more than one sequence, shown is the distribution of the patterns in the cluster over the activities. The last row shows the distribution of the 13 clusters containing only a single sequence over the activity categories.

Cluster #	W-in	W-out	C-in	C-out	Other
1	44	0	20	0	0
2	0	37	0	19	0
3	0	0	2	0	0
4	0	0	3	0	0
5	0	0	0	4	0
6	0	0	0	0	6
7	0	0	0	0	2
One-seq clusters	0	2	6	2	3

detect every unusual event while minimizing the false positive rate, in every experiment, we calculate an ROC curve and select the detection threshold yielding the best false positive rate at a 100% hit rate. Our experimental hypothesis was that the proposed method for modeling scene-specific behavior patterns should obtain better false positive rates than the traditional methods.

5.4.2.1 Experiment I: Proposed method

The red line in Figure 5.4 represents the ROC curve for our method as we vary the likelihood threshold at which a sequence is considered anomalous. Note that the ROC does not intersect the point $(0, 0)$ because any sequence that is most likely under one of the HMMs modeling anomalous sequences in the bootstrap set is automatically classified as anomalous regardless of the threshold.

The ROC reveals that a threshold of -3.259 achieves zero false negatives at a false alarm rate of 0.074. Table 5.2 shows the detailed performance of this model, and Figure 5.5 shows an example of a sequence classified as abnormal.

I have performed some brief experimentation to determine how the size of bootstrap set affects the results. I found that with more bootstrap data we tend to discover more compact behavior classes, leading to better anomaly detection rates in practice.

Table 5.2: Anomaly detection results for the proposed method, k-NN anomaly detection method, Mahalanobis classifier for anomaly detection method, and SVM-based anomaly detection method in Experiment I, II, III, and IV, respectively. For the Mahalanobis classifier for anomaly detection method, we include 11 abnormal sequences from the bootstrap set in the test set, so the total number of positives is 35. TP, FP, TN, and FN are the number of true positives, false positives, true negatives, and false negatives, respectively. TPR and FPR stand for true positive rate and false positive rate, respectively.

Method	TP	FP	TN	FN	TPR	FPR
Ours	24	36	450	0	1	0.074
1-NN	19	1	485	5	0.792	0.002
2-NN	19	2	484	5	0.792	0.004
3-NN	16	0	486	8	0.667	0
4-NN	16	1	485	8	0.667	0.002
5-NN	14	0	486	10	0.583	0
Mahalanobis classifier	35	421	65	0	1	0.87
SVM	24	228	258	0	1	0.469

However, since a larger bootstrap set will generally contain more variable behavior, our approach consequently creates more models to handle the more variable set of patterns. The effect is that we require more interaction with the operator determine whether each behavior cluster is normal or abnormal.

In this sense, the "optimal" bootstrap set size would thus be a size sufficient to include most typical behaviors (1–2 days) but no more.

5.4.2.2 Experiment II: k-NN

In this experiment, we tested the ability of a more traditional machine learning algorithm, k-nearest neighbors, to detect the anomalies in our testbed data set using the same division of sequences into training and testing as in Experiment I. As the distance measure, we used the same DTW measure we used for hierarchical clustering of the bootstrap patterns in our method. We varied k from 1 to 5. The results are shown in Table 5.2. While the false positive rates are much lower than those obtained in our method, the hit rates are unacceptable. For k-NN to be a practical anomaly detection method, we would have to adjust it. For example, we could impose a distance threshold beyond which a pattern is considered anomalous even if a majority of the nearest neighbors are normal.

5.4.2.3 Experiment III: Mahalanobis Classifier

In this experiment, we classified sequences as normal or anomalous using a Gaussian density estimator derived from principal components analysis (PCA). Since PCA requires a fixed-length input vector, we calculated, for each sequence in the testbed data set, a summary vector consisting of the means and standard deviations of each observation vector element over the entire sequence. With seven features in the observation vector, we obtained a 14-element vector summarizing each the sequence. After feature summarization, we normalized each component of the summary vector by z-scaling. Then, since we are performing probability density estimation for the normal patterns, we applied PCA to the 139 normal sequences in the bootstrap set. We chose the number of principal components accounting for 80% of the variance in the bootstrap data. Finally, we classified the remaining 521 test sequences using a Mahalanobis classifier to calculate the Mahalanobis distance of each sequence's summary vector to the mean of the normal bootstrap patterns' summary vectors.

The green line in Figure 5.4 is the ROC curve obtained by varying the Mahalanobis distance threshold, and Table 5.2 shows detailed results for anomaly detection at a 100% hit rate. The high false positive rate at this threshold and the overall poor performance in the ROC analysis show that the Mahalanobis classifier is clearly inferior to our proposed method.

5.4.2.4 Experiment IV: SVMs

Here we used the same summary vector technique used in Experiment III but performed supervised classification using support vector machines. We used the radial basis function kernel implementation in LIBSVM (Chang & Lin, 2001) with grid search for the optimal hyperparameters using five-fold cross validation on the training set (150 sequences). The blue line in Figure 5.4 is the ROC curve obtained by varying the threshold on the signed distance to the separating hyperplane used for classification as normal or abnormal. Table 5.2 shows the detailed anomaly detection results for SVMs at a 100% hit rate. Although the results are clearly better than those obtained from k-NN or Mahalanobis classifier, they are also clearly inferior to those obtained in Experiment I.

In the experiments, we use SVM and Mahalanobis classifier to perform two-class classification (normal vs. abnormal). In our data set, both normal and abnormal patterns contains subclasses, so the results could be improved if we train the algorithms on those subclasses separately.

5.5 Discussion

In this chapter, we have proposed and evaluated a new method for bootstrapping scene-specific anomalous human behavior detection systems. The method requires minimal involvement of a human operator; the only required action is to label the patterns in a small bootstrap set as normal or anomalous. On a testbed data set acquired in a real-world video surveillance situation, with a bootstrap set of 150 sequences, the method achieves a false positive rate of merely 7.4% at a hit rate of 100%. The experiments demonstrate that with a collection of simple HMMs, it is possible to learn a complex set of varied behaviors occurring in a specific scene.

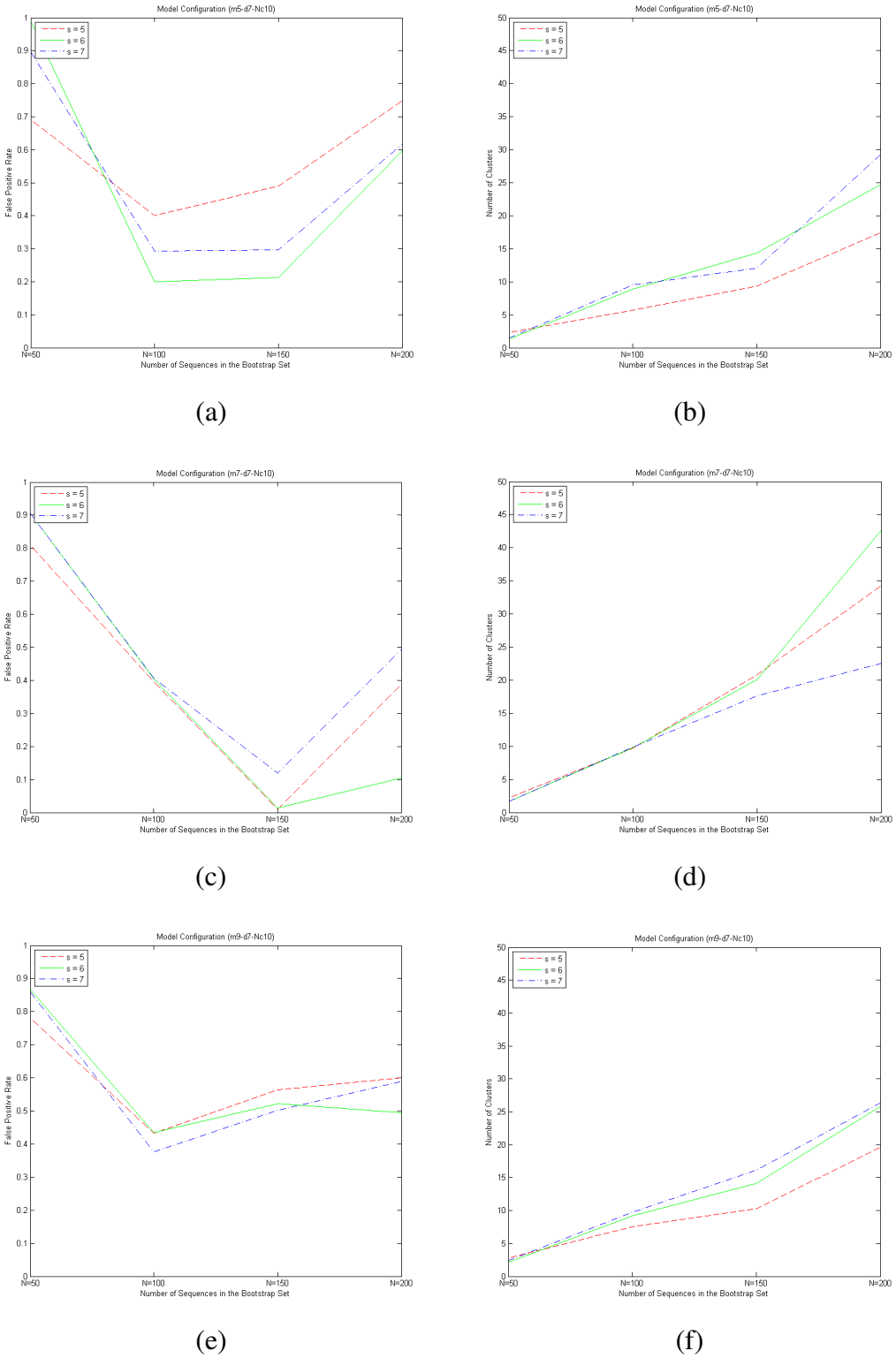


Figure 5.3: Subset of model configuration selection results. Model configuration with (a)–(b) five tokens, (c)–(d) seven tokens and (e)–(f) nine tokens. Dashed red, solid green and dash-dot blue lines represent models with five, six and seven states, respectively. Each point is an average over 10 trials.

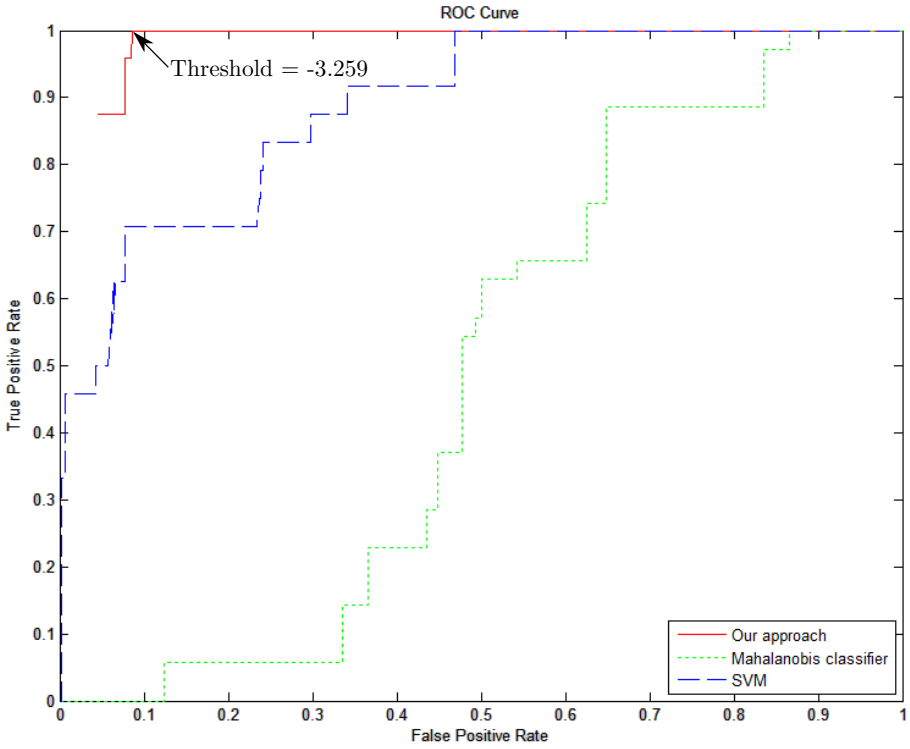


Figure 5.4: Anomaly detection ROC curves. Solid red, dotted green, and dashed blue lines represent ROCs for the proposed method in Experiment I, Mahalanobis classifier for anomaly detection in Experiment III, and SVM-based anomaly detection in Experiment IV, respectively.

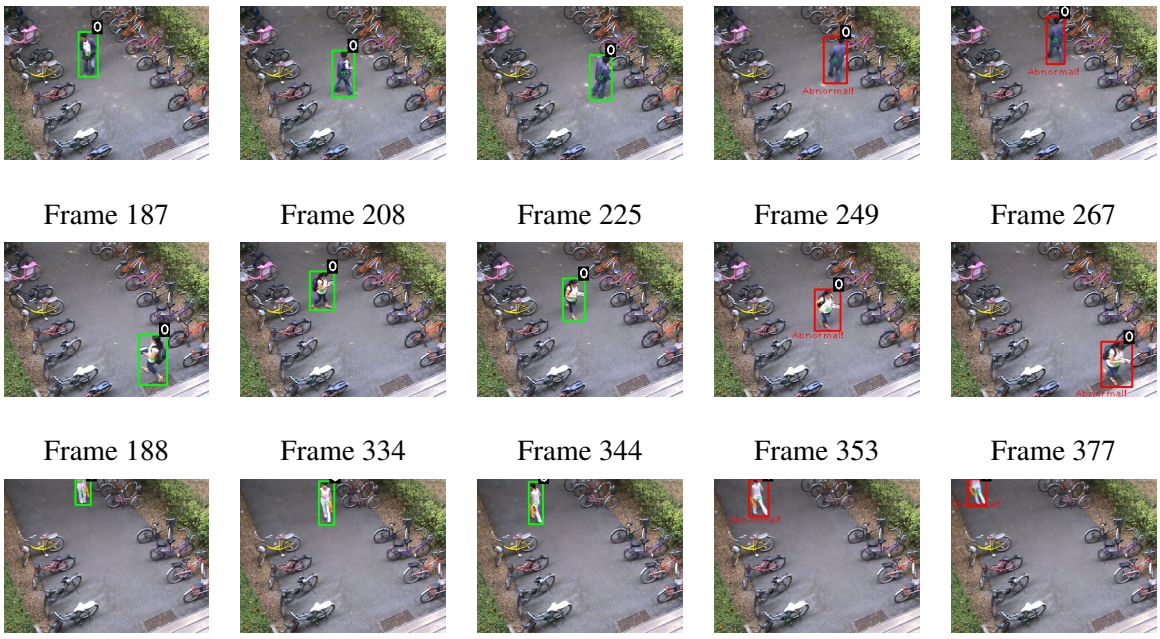


Figure 5.5: Example anomaly detected by the proposed method in Experiment I. The first row represents the sequence containing a person walking around looking for an unlocked bicycle. The second row represents the sequence containing a person walking out of the building then walking back into it. The last row represents a person walking in and looking around then walking out.

Chapter 6

Incremental Behavior Modeling and Suspicious Activity Detection

We propose and evaluate an efficient method for automatic identification of suspicious behavior in video surveillance data that incrementally learns scene-specific statistical models of human behavior without requiring storage of large databases of training data. The approach begins by building an initial set of models explaining the behaviors occurring in a small bootstrap data set. The bootstrap procedure partitions the bootstrap set into clusters then assigns new observation sequences to clusters based on statistical tests of HMM log likelihood scores. Cluster-specific likelihood thresholds are learned rather than set arbitrarily. After bootstrapping, each new sequence is used to incrementally update the sufficient statistics of the HMM it is assigned to. In an evaluation on a real-world testbed video surveillance data set, we find that within one week of observation, the incremental method's false alarm rate drops below that of a batch method on the same data. The incremental method obtains a false alarm rate of 2.2% at a 91% hit rate. The method is thus a practical and effective solution to the problem of inducing scene-specific statistical models useful for bringing suspicious behavior to the attention of human security personnel.

6.1 Introduction

Human monitoring of video surveillance channels is increasingly ineffective as the number of channels increases. *Anomaly detection* aims to alleviate this problem by filtering out typical events and only bringing suspicious events to the attention of human security personnel.

We propose a new incremental method for learning behavior models and detecting anomalies using hidden Markov model (HMM)-based clustering on a small bootstrap set of sequences labeled as normal or suspicious. After bootstrapping, we assign new observation sequences to behavior clusters using statistical tests on the log likelihood of the sequence according to the corresponding HMMs. We label a sequence as suspicious if it maps to an existing model of suspicious behavior or does not map to any existing model. After labeling, we either incrementally update the sufficient statistics for the most likely HMM's parameters or create a new HMM for the new sequence.

In an evaluation on a real-world video surveillance situation, we find that the method is very effective at identifying suspicious behavior. A batch method achieves a false positive rate of 8.6% at a 100% hit rate. The incremental method decreases the false alarm rate to 2.2% at a 91% hit rate, and detection threshold tuning can achieve an equal error rate of 95.8% or a false alarm rate of 3.7% at a 100% hit rate. The experimental results demonstrate that our incremental method is better able to learn new forms of normal behavior, leading in turn to more effective anomaly detection. The method is thus a practical and effective solution to the problem of inducing scene-specific statistical models useful for bringing suspicious behavior to the attention of human security personnel.

6.2 Related Work

In this section, we explore methods for incremental learning of behavior models and active detection of anomalous deviations from the learned typical behavior. We build upon previous research in human behavior understanding (L. S. Davis et al., 1998; Masoud & Papanikolopoulos, 2003; Cao et al., 2004; Sminchisescu et al., 2005; H. Li et al., 2006; Du et al., 2006; Wang et al., 2006), the use of dynamic graphical models such as hidden Markov models (HMMs) and conditional random fields (CRFs) for behavior understanding in specific scenarios (Yamato et al., 1992; Gao et al., 2004; Xiang & Gong, 2005; Sminchisescu et al., 2005; Vail & Guestrin, 2007), and classification of behavior in videos to a priori known categories (Nair & Clark, 2002; Lee et al., 2003; Wu et al., 2005; Arsić et al., 2007).

Several research groups have investigated methods for video clustering (Sakoe, 1978; H. Zhong et al., 2004; Xiang & Gong, 2005; Hautamäki et al., 2008; Xiang & Gong, 2008c; Swears et al., 2008; Xiang & Gong, 2008a). Especially effective are those methods using HMMs to model different classes of behavior in video surveillance data (Xiang & Gong, 2005; Swears et al., 2008; Xiang & Gong, 2008a).

There has been some work on anomalous time series detection outside the context of video surveillance using relatively simple statistical models (Turner et al., 2010; Yao et al., 2010). However, these methods learn single comprehensive models without addressing the special requirement in video surveillance to capture an extremely wide variety of typical behaviors. This requires construction of a collection of statistical models. Intrusion detection requires similar diversity in the statistical models. Some of the work uses ensembles of classifiers (Rasoulifard et al., 2008), but most of the research has focused on anomaly detection methods using incremental clustering (Burbeck & Nadjm-Tehrani, 2004, 2007; Ren et al., 2008; C. Zhong & Li, 2008; Ekman & Holst., 2008). All of these methods generally work by comparing a new pattern against a collection of clusters representing historically typical behavior and classifying the new pattern as an anomaly if its distance from the nearest cluster is above threshold. We take a similar approach, with a specific approach to clustering, distance measurement, and threshold learning specific to the case of surveillance video.

Some incremental learning approaches have been applied to human behavior modeling. Vasquez et al. (2009) incrementally model vehicle and pedestrian trajectories using growing hidden Markov models (GHMMs). The method builds a topological map and corresponding GHMM using an incremental variant of the Baum-Welch algorithm. For each new node in the map, a corresponding state and connectivity are added to the GHMM. Kulić and and Nakamura (2010) model movement primitives using HMMs and piece the primitives together with a higher level HMM. Similar to Vasquez et al., the method incrementally adds new hidden states when a new primitive model is built.

Besides human behavior modeling, Stenger et al. (2001) first propose the incremental Baum-Welch (IBW) algorithm on continuous observations for background modeling. This algorithm is a straightforward adaptation of the original Baum-Welch algorithm to incremental learning. They update the parameters by taking into account the sufficient statistics. Although they use a single observation sequence to update the parameters, but the sufficient statistics still can represent the information computed for all of the observed data. Later, Florez-Larrahondo et al. (2005) adapt this algorithm to model discrete observations. Cavalin et al. (2008) use the ensemble training (ET) algorithm (MacKay, 1997) to the incremental learning problem in alphanumeric character recognition. This algorithm has

never been used in incremental learning setting. The authors claim that it is easy to adapt to this kind of setting since the parameters of the final HMM are computed for each observation sequence independently.

The existing work most similar to ours is the incremental learning approach of Xiang and Gong (2008b), who also model activity in a scene incrementally after initialization from a small bootstrap dataset. They build explicit probabilistic models of normal and abnormal activity patterns in the bootstrap set then classify new observations using a likelihood ratio test. The models are global joint models over the entire scene. It is not clear how the likelihood ratio test could handle completely new anomalous events with low likelihoods under both models, and it is not clear whether a global model is appropriate for detecting isolated anomalous behavior in a scene.

Our method does not require an a priori model of anomalous human behavior. It constructs an ensemble of simple models, adding to the ensemble when the existing set is insufficient to represent new cases. We associate each new observation with an individual model using separate statistical tests on the observation's likelihood according to each individual model. Rather than globally modeling the entire scene, our method takes a more local approach, separately analyzing each individual moving object's behavior. In an experimental comparison of the two methods, we find statistical testing of the likelihood to be superior to likelihood ratio tests, and we find the local representation to be superior to the global representation.

The main contribution of the work described in this chapter is a new modeling method for detection of anomalous events in surveillance video based on simply-structured models and incremental learning that is demonstrably able to evolve the models over time to adapt to new behavior and also outperforms current techniques on the same data set.

6.3 Behavior Model Bootstrapping

Simply-structured statistical models cannot hope to capture the diversity of "normal" behavior in a given scene — multiple models are required. However, it is difficult to determine how many and what activities will occur in a scene a priori or based on manual monitoring. We therefore propose a method to automatically determine, from a small bootstrap set, the common similar behaviors occurring in a scene. The entire process of the method used in this section is previously described in Section 5.2 including blob extraction, appearance-based blob tracking, blob feature vector discretization, and behavior clustering.

This process results in a set of K different typical behavior clusters $\mathcal{C} = \{c_1, c_2, \ldots, c_K\}$ with a set of K corresponding HMMs $\mathcal{M} = \{M_1, M_2, \ldots, M_K\}$.

6.4 Anomaly Detection

We classify new sequences as abnormal if the most likely HMM for the input sequence is associated with one of the abnormal models, *or* the z-scored per-observation log likelihood of the sequence

under that most likely model is less than a global empirically determined threshold θ_z. We previously described this method in Section 5.3.

One problem with this approach is that it never learns any new typical behavior over time. Given a small bootstrap set, the method may work well for a short period of time, but as typical behavior evolves and becomes more diverse, the false alarm rate may increase.

The naive solution to this problem is to retain all of the observation data and retrain the system periodically. However, this would require storing all of the incoming data, requiring too much disk space. For lifelong anomaly detection, we need a more adaptive approach that learns new behaviors over time but does not require storing all of the historical data. We solve this problem in the next section.

6.5 Incremental Behavior Modeling for Anomaly Detection

In this section, we extend the basic anomaly detection method previously described in Section 6.4 using the incremental maximum likelihood (IML) algorithm for HMMs (Gotoh & Silverman, 1996). The incremental expectation maximization (EM) procedure was originally proposed by Neal and Hinton (1993). The basic idea is to update the sufficient statistics for the parameters of the HMMs as new events occur. In our approach, given an initial set of observation sequence clusters and corresponding HMMs modeling the behavior in the scene, when a new observation sequence arrives, we first determine which cluster it falls into by performing statistical tests on the sequence's likelihood according to each HMM model. Since likelihood scores for different HMMs cannot be compared directly, we select the most likely model based on the same z-scored per-observation log likelihood measure introduced in Section 4.2.3 for the initial behavior model clustering.

When the most likely model for a new observation sequence is a model for a cluster of atypical behaviors, or when the new observation sequence's likelihood according to the most likely model is not sufficiently high, we raise an alert. If the operator gives feedback that a new observation sequence marked as suspicious by the system is in fact not suspicious, we create a new typical behavior cluster and corresponding HMM for the false alarm sequence. In particular, we randomize the initial model then use the gradient descent with the Baum-Welch algorithm from that point regardless of the number of patterns.

If the likelihood according to the most likely model is sufficiently high for one of the pre-existing models and the operator confirms that the classification is correct, we incrementally update the sufficient statistics for that model; otherwise, we start a new cluster for the sequence, build a new HMM for the new cluster, and label it according to the operator feedback. This approach allows the system to raise alerts when unusual behaviors are detected while adapting to gradual changes in behavior patterns over time. The method is detailed in Algorithm 3.

When a new behavior sequence $\mathbf{O}$ arrives, we find the cluster k whose HMM M_{ml} assigns $\mathbf{O}$ the highest log likelihood L_{ml}. $\mathbf{O}$ is considered consistent with c_k if L_{ml} is above threshold θ_z and the operator does not mark $\mathbf{O}$ as a false positive. In this case, we use $\mathbf{O}$ to update the sufficient statistics $\mathcal{S}^{(k)} = \{\mathbf{\Gamma}^{(k)}, \mathbf{\Xi}^{(k)}\}$ for cluster k, allowing us to throw away the original observation sequence. $\mathbf{\Gamma}^{(k)}$ accumulates the $\gamma^{(k)}$ probabilities (defined in Rabiner's tutorial; Rabiner, 1989) needed to estimate

Algorithm 3 Anomaly Detection with Incremental Learning

Input: $\mathbf{O}$: behavior sequence

Input: $\mathcal{M}$: set of HMMs

Input: $\mathcal{S}$: set of sufficient statistics

Output: $\widetilde{\mathcal{M}}$: set of revised HMMs

Output: $\widetilde{\mathcal{S}}$: set of revised sufficient statistics

$\widetilde{\mathcal{M}} \leftarrow \mathcal{M};\ \widetilde{\mathcal{S}} \leftarrow \mathcal{S}$

$\mathcal{M}_{ab} \leftarrow \{M \mid M \in \mathcal{M}$ and M is marked abnormal$\}$

$(M_{ml}, S_{ml}, L_{ml}) \leftarrow$ FIND-MOST-LIKELY-MODEL$(\mathbf{O}, \mathcal{M})$

$d_{\text{feedback}} \leftarrow \emptyset$

if $M_{ml} \in \mathcal{M}_{ab}$ or $L_{ml} \leq \theta_z$ **then**

$d_{\text{feedback}} \leftarrow$ ALERT-SECURITY-PERSONNEL$(\mathbf{O})$

end if

if $L_{ml} > \theta_z$ and $d_{\text{feedback}} \neq$ false positive **then**

$(M, S) \leftarrow$ INCREMENTALLY-UPDATE(M_{ml}, S_{ml})

$\widetilde{\mathcal{M}} \leftarrow \{\widetilde{\mathcal{M}} \setminus M_{ml}\} \cup \{M\};\ \widetilde{\mathcal{S}} \leftarrow \{\widetilde{\mathcal{S}} \setminus S_{ml}\} \cup \{S\}$

else

$(M, S) \leftarrow$ CREATE-NEW-MODEL$(\mathbf{O})$

if $L_{ml} \leq \theta_z$ and $d_{\text{feedback}} \neq$ false positive **then**

Mark M as abnormal

end if

$\widetilde{\mathcal{M}} \leftarrow \widetilde{\mathcal{M}} \cup \{M\};\ \widetilde{\mathcal{S}} \leftarrow \widetilde{\mathcal{S}} \cup \{S\}$

end if

Algorithm 4 Incremental EM Algorithm

Input: $\mathbf{O}$: behavior sequence

Input: M: HMM model

Input: $\mathcal{S} = \{\mathbf{\Gamma}, \mathbf{\Xi}\}$: set of sufficient statistics

Output: M^*: revised HMM model

Output: $\mathcal{S}^* = \{\mathbf{\Gamma}^*, \mathbf{\Xi}^*\}$: set of revised sufficient statistics

$M^* \leftarrow M$

for $i = 1 \rightarrow I$ **do**

{E-step}

$(\gamma, \xi) \leftarrow$ COMPUTE-SUFFICIENT-STATISTICS$(\mathbf{O}, M^*)$

$\mathbf{\Gamma}^* \leftarrow \alpha\mathbf{\Gamma} + \gamma$; $\mathbf{\Xi}^* \leftarrow \alpha\mathbf{\Xi} + \xi$

{M-step}

$M^* \leftarrow$ REESTIMATE-MODEL-PARAMETERS$(M^*, \mathbf{\Gamma}^*, \mathbf{\Xi}^*)$

end for

the emission and transition probabilities of M_k. The update rule is simply

$$\mathbf{\Gamma}^{(k)}_{\text{new}} = \alpha\mathbf{\Gamma}^{(k)}_{\text{old}} + \gamma^{(k)},$$

where α is a forgetting factor (Nowlan, 1991). $\mathbf{\Xi}^{(k)}$ accumulates the $\xi^{(k)}$ probabilities needed to estimate the transition probabilities. It is updated the same way $\mathbf{\Gamma}^{(k)}$ is. In the experiments reported upon in this work, we use $\alpha = 0.9$.

When an HMM is first trained, we initialize the sufficient statistics $\mathcal{S}^{(k)}$ over all of the observation sequences in cluster c_k using batch training. During incremental learning, in each iteration of the E-step, we calculate the sufficient statistics $S^{(k)}$ for the single input observation sequence then update $\mathcal{S}^{(k)}$ accordingly to be used in the M-step. We repeat the E and M updates for a fixed number of iterations I. Neal and Hinton (1993) prove monotonic convergence of incremental EM under certain circumstances and find that in practice it is much faster than the batch EM algorithm. In our case, since we are incorporating a completely new observation sequence at each increment, the likelihood over all the sequences may increase or decrease. The method is summarized in Algorithm 4.

6.6 Experimental Results

We recorded video[1] from the scene in front of a building during working hours (9:00–17:00) for one week. We started a new event whenever the number of foreground pixels exceeded 300. We used a threshold of 0.995 for shadow detection NCC. We discarded any blobs smaller than 300 pixels in area. We obtained 423 video segments containing 660 observation sequences.

Figure 5.2 shows examples of four common activities: people entering the building, people leaving the building, people parking bicycles, and people riding bicycles out, and people walking pass by or having interactions with each other. For the purpose of analyzing results, we manually labeled each of the videos with the "normal" categories "walk-in," "walk-out," "cycle-in," and "cycle-out," or, for other activities such as walking around looking for an unlocked bicycle, with the "abnormal" category.

We performed experiments in three parts: model configuration selection (finding an optimal set of HMMs to model the bootstrap set), anomaly detection based on the HMM bootstrap set, and anomaly detection with incremental learning.

6.6.1 Model Configuration Selection

Towards model identification, we performed five-fold cross validation with different bootstrap parameter settings and selected the configuration with the highest accuracy in separating the normal sequences from the abnormal sequences on the bootstrap set, as measured by the false positive rate for abnormal sequences over all cross validation test folds. (Every bootstrap cluster containing an abnormal sequence is considered abnormal, so we always obtain 100% detection on the bootstrap set; the only discriminating factor is the false positive rate.) The set of parameters that we varied were 1) the number of states in each bootstrap HMM (3–8), 2) the number of symbols/k-means clusters (3–8), and 3) the number N_c of deviant patterns allowed in a bootstrap cluster (1–10). To find the distribution (parameters μ_c and σ_c) of the per-observation log likelihood for a particular HMM, we always generated 1,000 sequences of average length (the average length was computed over the bootstrap set; it was 205 observations in our experiment). For the rejection threshold p_c, we used a z-threshold of 2.0 (i.e., $p_c = \mu_c - 2\sigma_c$), corresponding to a Type I error rate (probability of misclassifying a sequence generated by the HMM as not generated by the HMM) of 0.0228.

The setting of the z-score threshold is important. With a z-threshold of 2, on different runs, we find that the method generates 15–30 clusters from our bootstrap set, usually perfectly classifies the bootstrap set, and performs well on the test set. With a lower threshold (0.5 to 1.5), only extremely similar series are clustered together, leading to a large number of clusters and a large number of false positives on typical behaviors that vary only slightly from what has been seen in the bootstrap set. With higher thresholds (2.5 to 3), patterns that are quite dissimilar get clustered, oftentimes leading to very few clusters that do not cleanly separate the bootstrap set and have low accuracy on the test

[1]Freely available for others to experiment with at: `http://www.kanouivirach.com/#downloads`.

Table 6.1: Example human behavior pattern bootstrapping results. We used linear HMMs with five states and seven symbols and bypass transitions. The model consists of 17 clusters. "W" means "walk" and "C" means "cycle." For the six clusters containing more than one sequence, shown is the distribution of the patterns in the cluster over the activities. The last row shows the distribution of the 11 clusters containing only a single sequence over the activity categories.

Cluster #	W-in	W-out	C-in	C-out	Other
1	44	0	20	0	0
2	0	38	0	24	0
3	0	0	3	0	0
4	0	0	2	0	0
5	0	0	0	0	6
6	0	0	0	0	2
One-seq clusters	0	1	6	1	3

set.

To reduce the variance in the measured false positive rate due to random initialization, we ran the entire cross validation procedure three times and averaged the results.

Based on the false positive rate criterion described above, we selected the model configuration consisting of five states with bypass transitions, seven symbols, and $N_c = 6$ and trained a new HMM ensemble on all 150 bootstrap sequences. We repeated the training process several times until we obtained a model that perfectly maps anomalous and typical sequences to different clusters on the bootstrap set. Table 6.1 shows the distribution of bootstrap sequences across this model's 17 behavior clusters. We used this model with the remaining 510 sequences in the anomaly detection experiments.

6.6.2 Anomaly Detection (Batch Processing)

Here we evaluate the ability of our anomaly detection method to identify anomalous events in the data not included in the bootstrap set. Since the main concern in video surveillance is to detect every unusual event while minimizing the false positive rate, we calculate an ROC curve and select the detection threshold yielding the best false positive rate at a 100% hit rate. We reported preliminary results for the proposed method compared to traditional machine learning algorithms in Chapter 5.

In this experiment, we compare the proposed method against HMM-based methods using alterna-

tive representations and scoring methods similar to those of Xiang and Gong (2008b). Xiang and Gong (2008b)'s method constructs a joint probabilistic model over the entire scene and incrementally builds explicit models of normal and abnormal activity models after initialization from a small bootstrap dataset. The method classifies new observations using a likelihood ratio test (LRT). One might categorize the Xiang and Gong approach as "global" and ours as "local." We describe how we compute the LRT below.

Given an observation sequence $\mathbf{O}$, the probability of the observation sequence given a set of normal models $\mathcal{M}_n$ is written as:

$$P(\mathbf{O} \mid \mathcal{M}_n) = \sum_{k=1}^{K_n} w_n^k P(\mathbf{O} \mid M_n^k),$$

where w_n^k is the weight of the k-th mixture component. We use the number of sequences in cluster k as the weight and normalize it so that $\sum_{k=1}^{K_n} w_n^k = 1$. K_n is the number of normal models, and M_n^k is the k-th HMM corresponding to the k-th normal behavior cluster. Similarly, the probability of an observation sequence given a set of abnormal models $\mathcal{M}_{ab}$ is written as:

$$P(\mathbf{O} \mid \mathcal{M}_{ab}) = \sum_{k=1}^{K_{ab}} w_{ab}^k P(\mathbf{O} \mid M_{ab}^k),$$

where w_{ab}^k is the weight of the k-th mixture component with $\sum_{k=1}^{K_{ab}} w_{ab}^k = 1$, K_{ab} is the number of abnormal models, and M_{ab}^k is the k-th HMM corresponding to the k-th abnormal behavior cluster.

To classify a new observation sequence, given a new observation sequence $\mathbf{O}_{\text{new}}$, we compute the LRT as follows.

$$\text{LRT}(\mathbf{O}_{\text{new}}) = \frac{P(\mathbf{O}_{\text{new}} \mid M_n)}{P(\mathbf{O}_{\text{new}} \mid M_{ab})}.$$

If $\text{LRT}(\mathbf{O}_{\text{new}}) > \theta_z$, O_{new} is considered normal. Otherwise, it is considered abnormal. This allows us to determine the extent to which different event representations (our local representation or Xiang and Gong's global representation) and different scoring methods (our z-scoring method or the standard LRT) contribute to anomaly detection performance on our test set. The four methods are as follows:

1. Method I: The proposed method.
2. Method II: The proposed method, but using Xiang and Gong's likelihood ratio test (Xiang & Gong, 2008b) rather than our z-scored likelihood method.
3. Method III: An HMM-based method using a global event representation similar to that of Xiang and Gong , with our z-scored likelihood method.
4. Method IV: The global method (Method III) using Xiang and Gong's likelihood ratio test rather than our z-scored likelihood method.

For method I and II, we used the 150-sequence bootstrap sequence set from Section 6.6.1 as training data and tested on the remaining 510 sequences. Figure 5.5 shows an example of a sequence classified as abnormal. When evaluating methods III and IV, we extract global features rather than individual blob features. The local method considers the single blob motion at a time, whereas the global

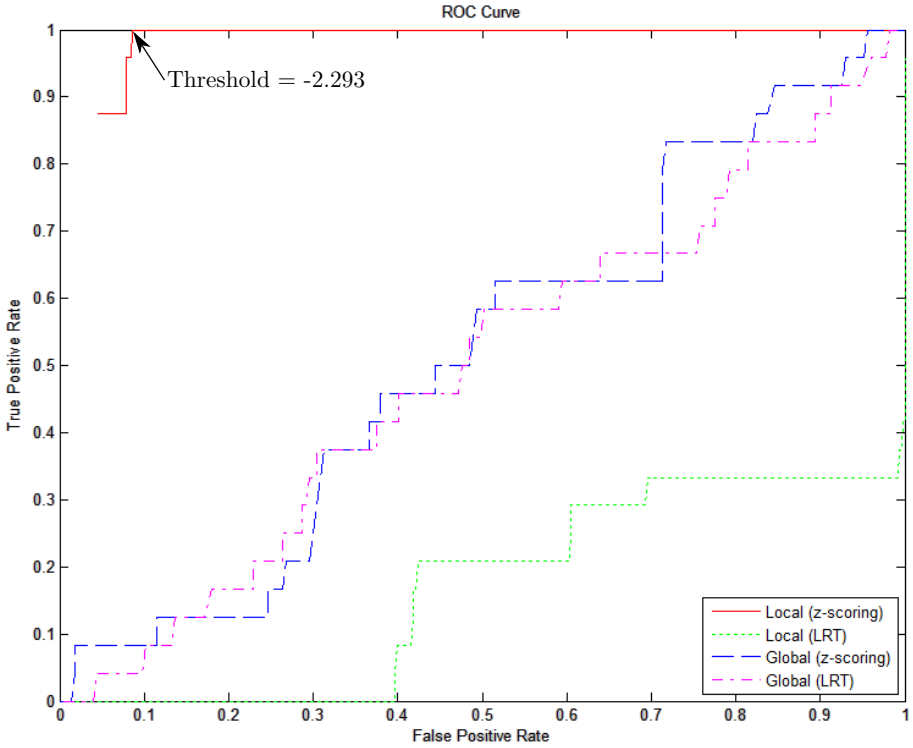


Figure 6.1: Anomaly detection ROC curves. Solid red, dotted green, dashed blue, and dash-dot magenta lines represent ROCs for methods I, II, III, and IV, respectively.

method simultaneously considers multiple blobs appearing at the same time. The data set for the global method comprises 401 sequences (366 negatives and 35 positives). We used the first 150 sequences (139 negatives and 11 positives) for the bootstrap set and the remaining 251 sequences (227 negatives and 24 positives) for the test set. In all four conditions, we performed a separate five-fold cross validation to select the best model configuration as described in Section 6.6.1.

Figure 6.1 shows ROC curves for methods I–IV. The solid red line represents the ROC curve for method I as we vary the threshold (z-score or likelihood ratio) at which a sequence is considered anomalous. Note that the ROC does not intersect the point $(0, 0)$ because any sequence that is most likely under one of the HMMs modeling anomalous sequences in the bootstrap set is automatically classified as anomalous regardless of the threshold. The ROC reveals that a z-score per-observation log likelihood θ_z of -2.293 achieves zero false negatives at a false alarm rate of 0.086. The dotted green line is the ROC curve for method II, the dashed blue line represents method III, and the dash-dot magenta line represents method IV. Table 6.2 shows the detailed performance of each detection method at 100% hit rate.

The ROC curves and FP rates show that our method clearly outperforms the other three methods, but there is also a surprising interaction: for the global method, z-scored likelihood and LRT are equally effective, but for our local event based method, z-scoring is much more effective than the LRT. This may reflect that the LRT's use of a mixture model is less appropriate for the local method than the global method.

The reason that the local method is better than the global method (albeit when combined with z-

Table 6.2: Anomaly detection results for the local method with the z-scoring method and the likelihood ratio test, and the global method with the z-scoring method and likelihood ratio test. TP, FP, TN, and FN are the number of true positives, false positives, true negatives, and false negatives, respectively. TPR and FPR stand for true positive rate and false positive rate, respectively.

Batch method	TP	FP	TN	FN	TPR	FPR
Local (z-scoring)	24	42	444	0	1	0.086
Local (LRT)	24	486	0	0	1	1
Global (z-scoring)	24	217	10	0	1	0.956
Global (LRT)	24	223	4	0	1	0.982

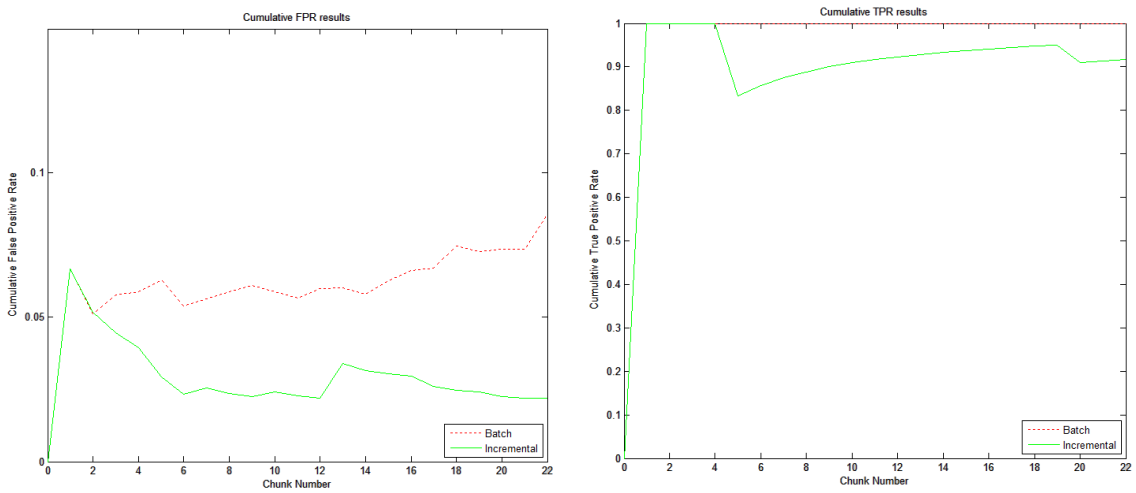
scoring of the likelihood) is that many of the abnormal sequences in the test set tend to be locally similar to the abnormal sequences in the bootstrap set but globally different. The local method treats concurrent but spatially separate local events as being independent, whereas the global method attempts to construct a joint model over the entire scene. The global method might be improved with a larger bootstrap set.

The log likelihood ratio test fails to detect completely new anomalous patterns when the pattern has a very low likelihood according to the abnormal model and the normal model but has a slightly higher likelihood according to the normal model. In order for the LRT test to detect such patterns, we need to adjust the LR threshold, causing higher false positives (increasing the number of false positives to about 30–40). Since our approach creates a new abnormal model for any low-likelihood observation, regardless of the relative likelihood, it does not suffer from this problem.

6.6.3 Anomaly Detection with Incremental Learning

In this experiment, we evaluate the use of incremental HMM learning for anomaly detection. We used the same settings and model configuration as in the batch method experiments described in Section 6.6.2. Since we incorporate a completely new observation sequence at each increment, the likelihood over all the sequences may increase or decrease. In the current work, we reestimate μ_c and σ_c after every update by generating 1,000 new sample sequences. The update process could obviously be optimized.

Overall results of incremental learning are shown in Table 6.3. Compared to the batch method, the incremental version of the proposed method (local method with z-scoring) obtains a nearly four-fold decrease in the false positive rate (2.2% vs. 8.6%) at the cost of a 9% decrease in the hit rate. The results shown in the table are based on a fixed z-score threshold of $\theta_z = 2.0$. To better enable comparison of the results for batch and incremental learning, we varied θ_z to obtain false positive rates at 100% detection rates and to obtain equal error rates (EERs). The incremental method achieves a



(a) (b)

Figure 6.2: Comparison of batch learning (dotted red line) and incremental learning (solid green line) over time. (a) Cumulative false positive rates. (b) Cumulative true positive rates. Each point is an average of the false positive or true positive rates over 5 trials.

3.7% false alarm rate at a 100% hit rate, compared to 8.6% for the batch method. It achieves an EER of 95.8%, compared to 91.8% for the batch method.

Overall, these results demonstrate that the incremental algorithm detects anomalies more effectively with lower false alarm rates. It is better able than the batch method to learn new forms of normal behavior. New behavior classes are always called anomalous at first, so completely new behaviors (suspicious or not) will always trigger a false positive and lead to the creation of a new model trained on the new sequence. These one-sequence HMMs tend to generate sequences with very low variance in the log likelihood. Such models do not often get updated with new sequences, but they do from time to time. In this experiment, new sequences were assigned to existing multi-sequence HMMs 466 times, existing single-sequence HMMs 44 times, and were used to create new sequences 5 times. The improvement in performance of the incremental method over the batch method thus comes first from tuning the parameters of the existing behavior classes (both typical and anomalous), but it also comes from the use of additional typical labels for the false positive anomalous series.

To further understand the performance differences between the batch and incremental methods, we partitioned the test data sequentially into chunks and computed the false and true positive rates for each chunk of observations. Initially, we set the chunk size to 10. If our system detects a suspicious pattern within a chunk, we record it and compute the false positive rate for the chunk. If the method does not detect any suspicious pattern, on the other hand, we increase the chunk's size by 1 until a true positive is given, then we reset the next chunk's size back to 10. This resulted in 22 chunks for our test set. Cumulative false and true positive rate data are shown in Figure 6.2. Initially, the incremental method has the same false alarm rate as the batch method. However, the incremental method's false alarm rate drops quickly to achieve the eventual 2.2% false alarm rate at a 91% hit rate.

For comparison, we also tested incremental versions of the three alternative approaches previously described in Section 6.6.2: local method with LRT, global method with z-scoring, and global method with LRT. In every condition, we used the same model configuration and detection threshold found

Table 6.3: Anomaly detection results on average for incremental HMM learning for our local method and the global method over 5 trials. TP, FP, TN, and FN are the number of true positive, false positive, true negative, and false negative, respectively. TPR and FPR stand for true positive rate and false positive rate, respectively.

Incremental method	TP	FP	TN	FN	TPR	FPR
Local (z-scoring)	21.8	9.8	476.2	2.2	0.91	0.022
Local (LRT)	16	243.4	212.6	8	0.6664	0.564
Global (z-scoring)	3	13	214	21	0.125	0.057
Global (LRT)	18	184.4	42.6	6	0.75	0.813

best in the batch experiment. For our incremental approach, both typical and atypical behavior models get updated during incremental learning. The results are shown in Table 6.3. Similar to the incremental method results, none of the three alternative methods performed as well as the proposed method. Our experimental results also demonstrate that over 510 new patterns, our approach requires security personnel to correct the system a mere 9–10 times.

The incremental version of the global method has lower false alarm rates and lower detection rates than the batch version of the global method. To an extent, the fixed z-score threshold of 2.0 places the global method at an inopportune position on the ROC curve. However, generally, we find that the global method with z-scoring performs poorly regardless of the threshold. With the global representation, the abnormal models tend to be quite specific whereas the normal models tend to be a little more general, so that we often see new anomalous patterns receiving relatively good scores from some normal models but no good scores from abnormal models. The anomalies thus tend to get incorrectly classified as normal more often.

Besides improved detection, another advantage of our incremental approach is that it is more efficient than the batch method over time. To demonstrate this, we performed a brief experiment on the runtime requirements on a Core 2 Duo processor at 2.2 GHz. We first trained a model with 1,000 synthetic sequences then re-trained that model with 1,000 sequences plus 10 new sequences using batch learning or incremental learning. The batch method required 33816 ms, whereas our incremental method with only the 10 new sequences required only 1056 ms. Obviously, the incremental approach uses far less time than the batch method to integrate new data into the training set, because it does not require retraining on old data.

As a final manipulation, we performed a brief experiment to determine whether incremental learning alone without a bootstrap set is effective on these data. We found that test set performance improves with the size of the bootstrap set. With more bootstrap data we tend to discover more compact behavior classes, leading to better performance on incremental updates. Conversely, incremental updates without a good bootstrap model lead to poor detection rates throughout the experiment. So while a no-bootstrap method could lessen the user's up-front workload, we conclude that the risk of missed anomalies is too high.

6.7 Discussion

We have proposed and evaluated an efficient method for bootstrapping scene-specific anomalous human behavior detection systems that incrementally learns behavior models without requiring storage of large databases of training data. The method requires minimal involvement of a human operator; the only required action is to label the patterns in a small bootstrap set as normal or anomalous and then to label false positive alarms as normal when they occur.

On a testbed data set acquired in a real-world video surveillance situation, with a bootstrap set of 150 sequences, the incremental method achieves a false positive rate of 2.2% at a 91% hit rate. With threshold tuning, the method can yield an equal error rate of 95.8% or a false positive rate of 3.7% at a 100% hit rate. The experiments show that it is possible to learn a complex set of varied behaviors occurring in a specific scene with a collection of simple HMMs while allowing evolution of the learned typical behavior model over time. This can lead in turn to more effective anomaly detection.

Chapter 7

Conclusion and Recommendations

We summarize the contributions of the dissertation and provide recommendations for further work.

7.1 Conclusion

In this dissertation, we have proposed and evaluated a new method for bootstrapping scene-specific anomalous human behavior detection systems that incrementally learns behavior models without requiring storage of large databases of training data. The method requires minimal involvement of a human operator; the only required action is to label the patterns in a small bootstrap set as normal or anomalous and then to label false positive alarms as normal when they occur.

Our contributions are five-fold. First, we have developed a blob extraction and appearance-based blob tracking method (explained in Chapter 2) that segments blobs and generate blobs' trajectories as input to the behavior modeling module. Although the method works well with typical surveillance data, the blob tracking method is not always robust for complex events involving multiple people.

Second, we propose a new method for detecting shadows using a simple maximum likelihood approach based on color information (explained in Chapter 3). We extend the deterministic nonmodel-based approach, designing a parametric statistical model-based approach. In our experiments, we find that our method works well in many cases but misclassifies shadow pixels in regions that have similar color to the background.

Third, we propose and evaluate a new method for clustering human behaviors (explained in Chapter 4). The method can be used to bootstrap an anomaly detection module for intelligent video surveillance systems. The combination of DTW with the type of linear HMMs we use in this work is extremely effective in separating anomalous from typical behaviors on real-world testbed video surveillance data. One main limitation of this work is that the method creates a large number of single-sequence clusters that would have to be manually labeled as normal or abnormal by security personnel.

Fourth, we propose a *semi-supervised* method for automatic identification of suspicious behavior from a small bootstrap set (explained in Chapter 5). The method partitions the bootstrap set into clusters then assigns new observation sequences to clusters based on statistical tests of HMM log likelihood scores. Our experiments find that the size of the bootstrap set could affect the results of the current system. With more bootstrap data we tend to discover more compact behavior classes, leading to better anomaly detection rates in practice. The main limitation of the method proposed in this chapter is that it never learns any new typical behavior over time. This requires a more adaptive approach to learn new behaviors over time. We address this issue in the next contribution.

Fifth, we propose an incremental behavior modeling and suspicious activity detection method (explained in Chapter 6) that incrementally learns scene-specific statistical models of human behavior

without requiring storage of large databases of training data.

The experimental results presented in this dissertation are extremely promising, demonstrating that our approach is a practical and effective solution to the problem of inducing scene-specific statistical models useful for bringing suspicious behavior to the attention of human security personnel. Deploying our system on a large video sensor network would potentially lead to substantial increases in the productivity and proactivity of human monitors.

That said, as mentioned above, the work has several limitations that could be addressed with further research. Next, I provide recommendations for future research.

7.2 Recommendations

Our approach can be extended to larger-scale situations as long as typical behavior can be modeled in terms of the spatio-temporal motion of foreground blobs. It could also be extended to recognize more complex events involving multiple persons — we believe that we can handle interactions between pedestrians, for example in pickpocket and assault events, within the current framework of temporal statistical models for individual humans by including observation features that characterize a person's interaction with others while in the scene. Integrating with pedestrian detection and tracking methods that rely on body part detection and tracking rather than motion blob tracking might help capture a larger proportion of the kinds of unusual behaviors we would observe in building entrances or office hallways.

Although the current method would work well for most building entrances, office building hallways, and similar environments, the blob tracking process would not be robust for scenes with dense crowds. Integrating with pedestrian tracking methods for crowds (Ali & Dailey, 2012) would be a potential solution to the problem.

In some cases, our shadow detection method misdetects shadow pixels as object pixels due to similar colors between the object and the background and unclear background texture in shadow regions. Incorporating geometric or shadow region shape priors would potentially improve detection and discrimination rates.

Next, constructing a fixed codebook to quantize the feature space may be inappropriate for incremental approaches since the codebook would need to be revised over time to account for changing "typical" behaviors. It may be better to take a probabilistic generative approach to the assignment of feature vectors to discrete categories rather than making hard assignments.

We found in the experiments that our clustering method tends to construct unbalanced dendrogram structures that leads our method to create a large number of single-sequence clusters. An approach to maintain the balanced dendrogram structure could address this issue.

Also, our current system can add new HMMs but cannot remove them. Over time, the number of HMMs would grow without bound. It would be better to periodically merge similar models or remove old models that no longer represent typical behavior. Since each HMM has the same structure, it would be very straightforward to check for pairs of similar models and merge them.

The clustering results suggest it is likely that the patterns DTW groups together are perfectly suited for modeling by linear HMMs with bypass transitions. We may further explore this idea in future work.

In the experimental results for our incremental method, we have shown that local event processing is more effective than global event processing on our data set. It would be interesting to compare to a hybrid approach in which we apply the global method to blocks in an image rather than the whole image.

Lastly, we will explore integrating the behavior understanding and anomaly detection algorithms into a complete video surveillance system such as ZoneMinder (Coombes, 2007).

References

Aggarwal, J., & Cai, Q. (1999). Human motion analysis: A review. *Computer Vision and Image Understanding (CVIU)*, *73*(3), 428–440.

Ali, I., & Dailey, M. N. (2012). Multiple Human Tracking in High-Density Crowds. *Image and Vision Computing (IVC)*, *30*(12), 966–977.

Alon, J., Sclaroff, S., Kollios, G., & Pavlovic, V. (2003). Discovering clusters in motion time-series data. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 375–381).

Andrade, E. L., Blunsden, S., & Fisher, R. B. (2006). Modelling Crowd Scenes for Event Detection. In *IEEE International Conference on Pattern Recognition (ICPR)* (pp. 175–178).

Anh, D. T. (2008). *Abnormal Behavior Detection using Star Skeleton*. Unpublished master's thesis, Computer Science and Information Management (CSIM), Asian Institute of Technology (AIT).

Arsić, D., Schuller, B., & Rigoll, G. (2007). Suspicious Behavior Detection in Public Transport by Fusion of Low-Level Video Descriptors. In *IEEE International Conference on Multimedia and Expo (ICME)* (pp. 2018–2021).

Bahl, L., Brown, P., Souza, P. de, & Mercer, R. (1986). Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 49–52).

Bedecs, V. (2009). *SecureCam.* (Open source software available at `http://sourceforge.net/projects/securecam`)

Blauensteiner, P., Wildenauer, H., Hanbury, A., & Kampel, M. (2006). Motion and Shadow Detection with an Improved Colour Model. In *IEEE International Conference on Signal and Image Processing (ICSIP).*

Bobick, A. F., & Davis, J. W. (2001). The recognition of human movement using temporal templates. *IEEE Transsactions on Pattern Analysis and Machine Intelligence (PAMI)*, *23*(3), 257–267.

Bobick, A. F., & Wilson, A. D. (1997). A State-Based Approach to the Representation and Recognition of Gesture. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *19*(12), 1325–1337.

Bodor, R., Jackson, B., & Papanikolopoulos, N. (2003). Vision-based human tracking and activity recognition. In *Mediterranean Conference on Control and Automation (MED)* (pp. 18–20).

Burbeck, K., & Nadjm-Tehrani, S. (2004). ADWICE - anomaly detection with real-time incremental clustering. In *International Conference on Information Security and Cryptology (ICISC).* Springer-Verlag.

Burbeck, K., & Nadjm-Tehrani, S. (2007). Adaptive real-time anomaly detection with incremental clustering. *Inf. Secur. Tech. Rep.*, *12*, 56–67.

Campbell, L., & Bobick, A. (1995). Recognition of human body motion using phase space constraints. In *IEEE International Conference on Computer Vision (ICCV)* (pp. 624–630).

Cao, D., Masoud, O. T., Boley, D., & Papanikkolopoulos, N. (2004). Online motion classification

using support vector machines. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2291–2296).

Cavalin, P. R., Sabourin, R., Suen, C. Y., & Jr., A. S. B. (2008). Evaluation of incremental learning algorithms for HMM in the recognition of alphanumeric characters. *Pattern Recognition (PR)*, *42*, 3241–3253.

Cedras, C., & Shah, M. (1995). Motion-based recognition: A survey. *Image and Vision Computing (IVC)*, *13*, 129–155.

Chan, M. T., Hoogs, A., Schmiederer, J., & Petersen, M. (2004). Detecting Rare Events in Video Using Semantic Primitives with HMM. In *IEEE International Conference on Pattern Recognition (ICPR)* (Vol. 4, pp. 150–154).

Chang, C.-C., & Lin, C.-J. (2001). *LIBSVM: a library for support vector machines.* (Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`)

Chen, Y., Yu, S., Sun, W., & Li, H. (2008). Objects detecting based on adaptive background models and multiple cues. In *ISECS International Colloquium on Computing, Communication, Control, and Management (CCCM)* (pp. 285–289). IEEE Computer Society.

Coombes, P. (2007). *ZoneMinder: A Linux-based camera monitoring and analysis tool.* (Open source software available at `http://www.zoneminder.com`)

Cucchiara, R., Grana, C., Piccardi, M., & Prati, A. (2001). Detecting objects, shadows and ghosts in video streams by exploiting color and motion information. In *International Conference on Image Analysis and Processing (ICIAP)* (pp. 360–365).

Cupillard, F., Bremond, F., & Thonnat, M. (2002). Group behavior recognition with multiple cameras. In *IEEE Workshop on Applications of Computer Vision (WACV)* (p. 177-183).

Darrell, T. J., & Pentland, A. P. (1993). *Recognition of Space-Time Gestures using a Distributed Representation* (Tech. Rep.). MIT Media Laboratory Vision and Modeling.

Davis, J. W., & Bobick, A. F. (1997). The representation and recognition of action using temporal templates. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 928–934).

Davis, L. S., Fejes, S., Harwood, D., Yacoob, Y., Haratoglu, I., & Black, M. J. (1998). Visual Surveillance of Human Activity. In *Asian Conference on Computer Vision (ACCV)* (pp. 267–274).

Du, Y., Chen, F., Xu, W., & Li, Y. (2006). Recognizing Interaction Activities using Dynamic Bayesian Network. In *IEEE International Conference on Pattern Recognition (ICPR)* (pp. 618–621).

DVTel, I. (2009). *Video Analytics.* (Available at `http://www.ioimage.com`)

Ekman, J., & Holst., A. (2008). *Incremental stream clustering and anomaly detection* (Tech. Rep.). SICS Technical Report T2008:01.

Elgammal, A., Duraiswami, R., Harwood, D., & Davis, L. (2002). Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, *90*(7), 1151–1163.

Finlayson, G. D., Schiele, B., & Crowley, J. L. (1998). Comprehensive Colour Image Normalization. In *European Conference on Computer Vision (ECCV)* (pp. 475–490).

Florez-Larrahondo, G., Bridges, S., & Hansen, E. A. (2005). Incremental Estimation of Discrete Hidden Markov Models Based on a New Backward Procedure. In *National Conference on Artificial Intelligence (AAAI)* (Vol. 2, pp. 758–763). AAAI Press.

Fujiyoshi, H., & Lipton, A. (1998). Real-time human motion analysis by image skeletonization. In *IEEE Workshop on Applications of Computer Vision (WACV)* (pp. 15–21).

Fusier, F., Valentin, V., Brémond, F., Thonnat, M., Borg, M., Thirde, D., et al. (2007). Video understanding for complex activity recognition. *Machine Vision and Applications (MVA)*, *18*(3), 167–188.

Gao, J., Hauptmann, A. G., Bharucha, A., & Wactlar, H. (2004). Dining Activity Analysis Using a Hidden Markov Model. In *IEEE International Conference on Pattern Recognition (ICPR)* (Vol. 2, pp. 915–918).

Gavrila, D. (1999). The visual analysis of human movement: A survey. *Computer Vision and Image Understanding (CVIU)*, *73*, 82–98.

Gavrila, D. M., & Davis, L. S. (1995). Towards 3-D model-based tracking and recognition of human movement: a multi-view approach. In *IEEE International Workshop on Automatic Face- and Gesture-Recognition* (pp. 272–277).

Georis, B., Brmond, F., & Thonnat, M. (2007). Real-time control of video surveillance systems with program supervision techniques. *Machine Vision and Applications (MVA)*, *18*(3–4), 189–205.

Georis, B., Mazire, M., Brmond, F., & Thonnat, M. (2004). A Video Interpretation Platform Applied to Bank Agency Monitoring. In *Intelligent Distributed Surveillance Systems Workshop.*

Gharti, S. (2010). *Left Luggage Detection and Tracking using Appearance Model*. Unpublished master's thesis, Computer Science and Information Management (CSIM), Asian Institue of Technology (AIT).

Girondel, V., Caplier, A., & Bonnaud, L. (2004). Real time tracking of multiple persons by Kalman filtering and face pursuit for multimedia applications. In *IEEE Southwest Symposium on Image Analysis and Interpretation (IAI)* (pp. 201–205).

Gotoh, Y., & Silverman, H. F. (1996). Incremental ML estimation of HMM parameters for efficient training. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 585–588). IEEE Computer Society.

Hautamäki, V., Nykänen, P., & Fränti, P. (2008). Time-series clustering by approximate prototypes. In *IEEE International Conference on Pattern Recognition (ICPR)* (pp. 1–4).

Havasi, L., Sziranyi, T., & Rudzky, M. (2006). Adding Geometrical Terms to Shadow Detection Process. In *European Signal Processing Conference (EUSIPCO).*

Hawkins, J. (2007). *Hierarchical temporal memory: How a new theory of neocortex may lead to truly intelligent machines.* (Information available at `http://nanohub.org/resources/3555`)

Hong, D., & Woo, W. (2003). A background subtraction for a vision-based user interface. In *Pacific Rim Conference on Multimedia (PCM)* (pp. 263–267).

Jacques Jr., J. C. S., Jung, C. R., & Musse, S. R. (2005). Background Subtraction and Shadow Detection in Grayscale Video Sequences. In *Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI)* (p. 189). IEEE Computer Society.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME-Journal of Basic Engineering*, *82*, 35–45.

Ke, Y., Sukthankar, R., & Hebert, M. (2007). Spatio-temporal shape and flow correlation for action recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–8).

Kender, J. R. (1976). *Saturation, hue, and normalized color: calculation, digitization effects and use* (Tech. Rep.). Tech. Rep., Department of Computer Science, Carnegie-Mellon University.

Krüger, V., Kragic, D., Ude, A., & Geib, C. (2007). The Meaning of Action: A Review on action recognition and mapping. *Advanced Robotics*, *21*(13), 1473–1501.

Kulić and, D., & Nakamura, Y. (2010). Incremental learning of human behaviors using hierarchical hidden Markov models. In *International Conference on Intelligent Robots and Systems (IROS)* (pp. 4649–4655).

Lee, K. K., Yu, M., & Xu, Y. (2003). Modeling of human walking trajectories for surveillance. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 1554–1559).

Li, C., & Biswas, G. (1999). Temporal pattern generation using hidden Markov model based unsupervised classification. In *Advances Intelligent Data Analysis (IDA)* (pp. 245–256).

Li, H., Hu, Z., Wu, Y., & Wu, F. (2006). Behavior modeling and recognition based on space-time image features. In *IEEE International Conference on Pattern Recognition (ICPR)* (pp. 243–246).

Lv, F., Song, X., Wu, B., Kumar, V., & Nevatia, S. R. (2006). Left luggage detection using bayesian inference. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)* (pp. 83–90).

MacKay, D. J. C. (1997). *Ensemble learning for hidden Markov models.*

Masoud, O., & Papanikolopoulos, N. (2003). A method for human action recognition. *Image and Vision Computing (IVC)*, *21*, 729–743.

Mikić, I., Cosman, P. C., Kogut, G. T., & Trivedi, M. M. (2000). Moving shadow and object detection in traffic scenes. In *IEEE International Conference on Pattern Recognition (ICPR)* (p. 1321). IEEE Computer Society.

Nair, V., & Clark, J. J. (2002). Automated visual surveillance using hidden Markov models. In *Vision Interface Conference (VIC)* (pp. 88–92).

Neal, R. M., & Hinton, G. E. (1993). A new view of the EM algorithm that justifies incremental and other variants. In *Learning in graphical models* (pp. 355–368).

Ng, J., & Gong, S. (2003). Learning pixel-wise signal energy for understanding semantics. *Image and Vision Computing (IVC)*, 1183–1189.

Niu, W., Jiao, L., Han, D., & Wang, Y.-F. (2003). Real-time multi-person tracking in video surveillance. In *Pacific Rim Conference on Multimedia (PCM)* (pp. 1–5).

Nowlan, S. J. (1991). *Soft Competitive Adaptation: Neural Network Learning Algorithms based on Fitting Statistical Mixtures*. Unpublished doctoral dissertation.

Oates, T., Firoiu, L., & Cohen, P. R. (2001). Using dynamic time warping to bootstrap HMM-based

clustering of time series. In *Sequence learning: Paradigms, algorithms, and applications* (pp. 35–52).

Oliver, N. M., Rosario, B., & Pentland, A. P. (2000). A Bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *22*(8), 831–843.

Ouivirach, K. (2008). *Human behavior profiling for a video surveillance system*. Unpublished master's thesis, Computer Science and Information Management (CSIM), Asian Institute of Technology (AIT).

Ouivirach, K., & Dailey, M. N. (2010). Clustering Human Behaviors with Dynamic Time Warping and Hidden Markov Models for a Video Surveillance System. In *IEEE Conference on Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON)* (pp. 884–888).

Ouivirach, K., Gharti, S., & Dailey, M. N. (2012). Automatic Suspicious Behavior Detection from a Small Bootstrap Set. In *International Conference on Computer Vision Theory and Applications (VISAPP)* (pp. 655–658).

Ouivirach, K., Gharti, S., & Dailey, M. N. (2013). Incremental Behavior Modeling and Suspicious Activity Detection. *Pattern Recognition (PR)*, *46*(3), 671–680.

Park, S., & Aggarwal, J. K. (2004). A hierarchical bayesian network for event recognition of human actions and interactions. In *Multimedia systems* (pp. 164–179).

Park, S., & Trivedi, M. (2007). Multi-person interaction and activity analysis: a synergistic track- and body-level analysis framework. *Machine Vision and Applications (MVA)*, *18*(3–4), 151–166.

Pass, G., Zabih, R., & Miller, J. (1996). Comparing images using color coherence vectors. In *ACM International Conference on Multimedia (MULTIMEDIA)* (pp. 65–73).

Poppe, C., Martens, G., Lambert, P., & Van de Walle, R. (2007). Improved background mixture models for video surveillance applications. In *Asian Conference on Computer Vision (ACCV)* (Vol. 4843, pp. 251–260). Springer.

Prati, A., Mikic, I., Trivedi, M., & Cucchiara, R. (2003). Detecting moving shadows: algorithms and evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *25*(7), 918–923.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 257–286.

Rasoulifard, A., Ghaemi Bafghi, A., & Kahani, M. (2008). Incremental Hybrid Intrusion Detection Using Ensemble of Weak Classifiers. In *International CSI Computer Conference (CSICC).*

Remagnino, P., Velastin, S., Foresti, G., & Trivedi, M. (2007). Novel concepts and challenges for the next generation of video surveillance systems. *Machine Vision and Applications (MVA)*, *18*(3–4), 135–137.

Ren, F., Hu, L., Liang, H., Liu, X., & Ren, W. (2008). Using Density-Based Incremental Clustering for Anomaly Detection. *International Conference on Computer Science and Software Engineering (CSSE)*, *3*, 986–989.

Rissanen, J. (1998). Hypothesis selection and testing by the MDL principle. *The Computer Journal*, 260–269.

Rosales, R., & Sclaroff, S. (1999). 3D trajectory recovery for tracking multiple objects and trajectory guided recognition of actions. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (p. 123).

Sakoe, H. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing (ICASSP)*, 43–49.

Salvador, E., Cavallaro, A., & Ebrahimi, T. (2004). Cast shadow segmentation using invariant colour features. *Computer Vision and Image Understanding (CVIU)*, *95*(2), 238–259.

Sanin, A., Sanderson, C., & Lovell, B. C. (2012). Shadow detection: A survey and comparative evaluation of recent methods. *Pattern Recognition (PR)*, *45*(4), 1684–1695.

Schreer, O., Feldmann, I., Golz, U., & Kauff, P. (2002). Fast and robust shadow detection in video-conference applications. In *IEEE International Symposium on Video/Image Processing and Multimedia Communications (VIPromCom)* (pp. 371–375).

Senior, A., Hampapur, A., Tian, Y.-L., Brown, L., Pankanti, S., & Bolle, R. (2006). Appearance models for occlusion handling. *Image and Vision Computing (IVC)*, *24*, 1233–1243.

Shechtman, E., & Irani, M. (2005). Space-time behavior based correlation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (Vol. 1, pp. 405–412).

Sminchisescu, C., Kanaujia, A., Li, Z., & Metaxas, D. (2005). Conditional Random Fields for Contextual Human Motion Recognition. In *IEEE International Conference on Computer Vision (ICCV)* (pp. 1808–1815).

Snoek, J., Hoey, J., Stewart, L., & Zemel, R. S. (2006). Automated detection of unusual events on stairs. In *IEEE Canadian Conference on Computer and Robot Vision (CRV)* (p. 5).

Stauffer, C., & Grimson, W. (1999). Adaptive background mixture models for real-time tracking. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (p. 252).

Stenger, B., Ramesh, V., & Paragios, N. (2001). Topology free hidden Markov models: Application to background modeling. In *IEEE International Conference on Computer Vision (ICCV)* (pp. 294–301).

Stratech. (2009). *SuperTrack: intelligent video surveillance and analysis system.* (Available at `http://www.stratechsystems.com/iv_supertrack.asp`)

Swears, E., Hoogs, A., & Perera, A. (2008). Learning motion patterns in surveillance video using HMM clustering. In *IEEE Workshop on Motion and Video Computing (WMVC)* (pp. 1–8).

Tian, Y.-L., Lu, M., & Hampapur, A. (2005). Robust and efficient foreground analysis for real-time video surveillance. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1182–1187). IEEE Computer Society.

Turaga, P., Chellappa, R., Subrahmanian, V. S., & Udrea, O. (2008). Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, *18*(11), 1473–1488.

Turner, R., Ghahramani, Z., & Bottone, S. (2010). Fast online anomaly detection using scan statistics. In *Machine Learning for Signal Processing (MLSP)* (pp. 385–390).

Umeda, M. (1982). Recognition of multi-font printed Chinese characters. In *International Conference on Pattern Recognition (ICPR)* (pp. 793–796). IEEE Computer Society.

Vail, D. L., & Guestrin, C. (2007). Conditional random fields for activity recognition. In *International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)* (pp. 1–8).

Vasquez, D., Fraichard, T., & Laugier, C. (2009). Growing Hidden Markov Models: An Incremental Tool for Learning and Predicting Human and Vehicle Motion. *International Journal of Robotics Research (IJRR)*, *28*(11–12), 1486–1506.

Vaswani, N., Chowdhury, A., & Chellappa, R. (2003). Activity recognition using the dynamics of the configuration of interacting objects. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (Vol. 2, pp. II - 633–40).

Vaswani, N., Roy-Chowdhury, A., & Chellappa, R. (2005). "shape activity": a continuous-state HMM for moving/deforming shapes with application to abnormal activity detection. *IEEE Transactions on Image Processing*, *14*(10), 1603–1616.

Vitamin D, I. (2010). *Vitamin D Video.* (Available at `http://www.vitamindinc.com`)

Vu, V.-T., Brmond, F., & Thonnat, M. (2003). Automatic video interpretation: A recognition algorithm for temporal scenarios based on pre-compiled scenario models. *2626*, 523–533.

Wang, S. B., Quattoni, A., Morency, L.-P., & Demirdjian, D. (2006). Hidden conditional random fields for gesture recognition. In *International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1521–1527).

Wilson, A., & Bobick, A. (2000). Realtime online adaptive gesture recognition. In *IEEE International Conference on Pattern Recognition (ICPR)* (Vol. 1, pp. 270–275).

Wren, C. R., Azarbayejani, A., Darrell, T., & Pentland, A. (1997). Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *19*, 780–785.

Wu, X., Ou, Y., Qian, H., & Xu, Y. (2005). A detection system for human abnormal behavior. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 1204–1208).

Xiang, T., & Gong, S. (2005). Video Behaviour Profiling and Abnormality Detection without Manual Labelling. In *IEEE International Conference on Computer Vision (ICCV)* (Vol. 2, pp. 1238–1245).

Xiang, T., & Gong, S. (2008a). Activity based surveillance video content modelling. *Pattern Recognition (PR)*, *41*(7), 2309–2326.

Xiang, T., & Gong, S. (2008b). Incremental and adaptive abnormal behaviour detection. *Computer Vision and Image Understanding (CVIU)*, *111*(1), 59–73.

Xiang, T., & Gong, S. (2008c). Spectral clustering with eigenvector selection. *Pattern Recognition (PR)*, *41*(3), 1012–1029.

Xiang, T., Gong, S., & Parkinson, D. (2002). Autonomous visual events detection and classification without explicit object-centred segmentation and tracking. In *British Machine Vision Conference (BMVC).*

Xu, L.-Q., Landabaso, J. L., & Pardàs, M. (2005). Shadow removal with blob-based morphological reconstruction for error correction. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (p. 729-732). IEEE Computer Society.

Yacoob, Y., & Black, M. J. (1998). Parameterized Modeling and Recognition of Activities. In *IEEE*

International Conference on Computer Vision (ICCV) (p. 120). IEEE Computer Society.

Yamato, J., Ohya, J., & Ishii, K. (1992). Recognizing human action in time-sequential images using hidden Markov model. In *International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 379–385).

Yang, W., Wang, Y., & Mori, G. (2010). Recognizing human actions from still images with latent poses. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2030–2037).

Yao, Y., Sharma, A., Golubchik, L., & Govindan, R. (2010). Online anomaly detection for sensor systems: a simple and efficient approach. *Performance Evaluation (PE)*, *67*(11), 1059–1075.

Yu, Y., Harwood, D., Yoon, K., & Davis, L. (2007). Human appearance modeling for matching across video sequences. *Machine Vision and Applications (MVA)*, *18*(3–4), 139–149.

Zhang, D., Gatica-Perez, D., Bengio, S., & McCowan, I. (2005). Semi-supervised adapted hmms for unusual event detection. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 611–618).

Zhong, C., & Li, N. (2008). Incremental Clustering Algorithm for Intrusion Detection Using Clonal Selection. In *Pacific-Asia Workshop on Computational Intelligence and Industrial Application (PACIIA)* (pp. 326–331).

Zhong, H., Visontai, M., & Shi, J. (2004). Detecting unusual activity in video. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 819–826).