SURVEILLANCE-ORIENTED TRAJECTORY-BASED ANOMALY DETECTION: SUPERVISED VS UNSUPERVISED LEARNING APPROACHES

by

Ng Lih Lin

A thesis submitted to the Swinburne University of Technology in support of an application for the degree of Master in Engineering

School of Engineering, Computing and Science
SWINBURNE UNIVERSITY OF TECHNOLOGY
(Sarawak Campus)
Jalan Simpang Tiga,
93350 Kuching,
Sarawak, MALAYSIA

JULY 2013
ABSTRACT

Transforming human understanding in visual data to electronic vision systems has been one of the aims in the field of artificial intelligence. The advancements in the digital cameras and computational capabilities have triggered the global demand in automated video understanding and anomaly detection especially in surveillance videos. Nevertheless, activities recognition and anomaly detection especially in public scenes remained a huge challenge for researchers due to countless factors such as poor quality footage, illumination, shadows, occlusion, noise etc. The state-of-the-art on behaviour analysis has built a bunch of sophisticated techniques for trajectory analysis. This thesis presents two trajectory learning methods; supervised and unsupervised trajectory learning and analysis in object’s motion pattern. Investigation on object’s motion trajectory for event recognition and anomaly detection frameworks is carried out for a video surveillance camera in an outdoor car park environment.

The main challenge for outdoor environment videos is the variation in illumination. For a better outdoor object tracking process, an adaptive GMM background modelling is implemented for object trajectory extraction. Subsequently, for a better understanding of mobility data, a hybrid spatiotemporal trajectory modelling is implemented to abstract object’s mobility data into different level; spatial location, velocity and acceleration. However, different type of object will have different repetition of motion pattern. Therefore, type of the object is classified before the motion pattern clustering.

A scene modeling is done before the event learning system to analyse the point of interest of the motion trajectory. For supervised activity learning method, the scene is decomposed into regions and the trajectory pattern for each activity is grouped and clustered. An offline trajectory learning method is done by using Principal Component Analysis (PCA). The experiment result shows the accurate event recognition and anomaly detection. However, substantial amount of labelled training dataset has to be collected and event learning cannot be performed adaptively.

Therefore, the work is extended to a second approach, unsupervised trajectory learning method to solve the addressing problem. The scene modeling is performed by clustering the entry and exit points of the object trajectories by GMM. In order to achieve automation in clustering trajectory pattern, a K-mean clustering algorithm is used to cluster the entire trajectories training dataset. A HMM framework is implemented in the system to devise parametric activity models for each class of activity. This system can perform the activities learning adaptively and fully automation. The comparison for both approaches is demonstrated in detail in this thesis.
ACKNOWLEDGEMENT

Firstly, I would like to thank my supervisor Dr. Chua Hong Siang for the continuous support of my Master research, for his patience, guidance, and encouragement. Besides, I am deeply grateful for invaluable advice and consistent support from my co-supervisors Associate Professor Dennis Wong and Dr. Kho Yau Hee throughout the project.

I would like to thank the Faculty of Engineering, Computing and Science, Swinburne University of Technology Sarawak Campus for giving me the opportunity to pursue my postgraduate degree, excellent library and, providing me comprehensive computer and research facilities.

I am indebted to my parents for their enduring love, support and understanding and encouragement throughout my years as postgraduate student.
I declare that the work in this thesis was carried out in accordance with the Regulations of the Swinburne University of Technology. This work is original except where indicated by special reference in the text and no part of the thesis has been submitted for any other degree.

Any views expressed in the thesis are those of the author and in no way represent those of the Swinburne University of Technology.

The thesis has not been presented to any other university for examination.

Signature : _____________________
Name : Ng Lih Lin
Date : 4th July 2013
MEMORANDUM

The accompanying thesis “SURVEILLANCE-ORIENTED TRAJECTORY-BASED ANOMALY DETECTION: SUPERVISED VS UNSUPERVISED LEARNING APPROACHES” is based on the work carried out by the author in the Faculty of Engineering, Computing and Science of Swinburne University of Technology (Sarawak campus).

Publications resulting from this thesis:


# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DHS</td>
<td>Department of Homeland Security</td>
</tr>
<tr>
<td>ISR</td>
<td>Intelligent, Surveillance and Reconnaissance</td>
</tr>
<tr>
<td>UAV</td>
<td>Unmanned Aerial Vehicle</td>
</tr>
<tr>
<td>CCTV</td>
<td>Closed Circuit TeleVision</td>
</tr>
<tr>
<td>VDM</td>
<td>Video Motion Detectors</td>
</tr>
<tr>
<td>GMM</td>
<td>Gaussian Mixture Model</td>
</tr>
<tr>
<td>BN</td>
<td>Bayesian Network</td>
</tr>
<tr>
<td>SOM</td>
<td>Self-Organizing Map</td>
</tr>
<tr>
<td>HMM</td>
<td>Hidden Markov Model</td>
</tr>
<tr>
<td>CHMM</td>
<td>Coupled Hidden Markov Model</td>
</tr>
<tr>
<td>DML-HMM</td>
<td>Dynamically Multi-linked Hidden Markov Model</td>
</tr>
<tr>
<td>S-HSMM</td>
<td>Switching Hidden Semi-Markov Model</td>
</tr>
<tr>
<td>LDA</td>
<td>Latent Dirichlet Allocation</td>
</tr>
<tr>
<td>FPS</td>
<td>Frame rate per second</td>
</tr>
<tr>
<td>POI</td>
<td>Point of Interest</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>EM</td>
<td>Expectation Maximization</td>
</tr>
</tbody>
</table>
### LIST OF SYMBOLS

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha )</td>
<td>Learning rate of the GMM</td>
</tr>
<tr>
<td>( \alpha_t )</td>
<td>Forward variable for Forward-backward algorithm</td>
</tr>
<tr>
<td>( A )</td>
<td>HMM transition probability matrix</td>
</tr>
<tr>
<td>( A_i )</td>
<td>Object area</td>
</tr>
<tr>
<td>( \beta_t )</td>
<td>Backward variable for Forward-backward algorithm</td>
</tr>
<tr>
<td>( B )</td>
<td>Distribution matrix of observation from hidden states, ( O_t )</td>
</tr>
<tr>
<td>( B_E )</td>
<td>Morphology structuring element</td>
</tr>
<tr>
<td>( B_i )</td>
<td>Bounding box for foreground detected</td>
</tr>
<tr>
<td>( \bar{Z}_i )</td>
<td>Centroid of the tracked object</td>
</tr>
<tr>
<td>( \Delta d )</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>( d_t )</td>
<td>A preset threshold value based on the specific scene</td>
</tr>
<tr>
<td>( \epsilon_{ij} )</td>
<td>The probability of being in state ( i ) at time ( t ) and state ( j ) at time ( t + 1 )</td>
</tr>
<tr>
<td>( e )</td>
<td>Event vector for a moving object</td>
</tr>
<tr>
<td>( E )</td>
<td>Entry zone dataset</td>
</tr>
<tr>
<td>( \Sigma_{i,t} )</td>
<td>Covariance matrices of ( t )th mixture coefficient at time ( t )</td>
</tr>
<tr>
<td>( f_i )</td>
<td>Filling ratio of object bounding box</td>
</tr>
<tr>
<td>( f_t )</td>
<td>Feature vector for an event representation</td>
</tr>
<tr>
<td>( G_B )</td>
<td>Background distribution</td>
</tr>
<tr>
<td>( h_i )</td>
<td>Object height</td>
</tr>
<tr>
<td>( K )</td>
<td>Number of Gaussian distribution</td>
</tr>
<tr>
<td>( O_i )</td>
<td>Tracked object that visible in the monitoring scene</td>
</tr>
<tr>
<td>( O_t )</td>
<td>Observable emission state from hidden state ( S ) for HMM model</td>
</tr>
<tr>
<td>( P(X_t) )</td>
<td>Probability density function of the pixel value for each color space</td>
</tr>
<tr>
<td>( \theta_t )</td>
<td>Absolute angle vector of the sampled trajectory with reference to the horizontal axis at ( t )th point</td>
</tr>
<tr>
<td>( R_i )</td>
<td>Object aspect ratio</td>
</tr>
<tr>
<td>( R )</td>
<td>Region in the monitoring scene</td>
</tr>
<tr>
<td>( S_i )</td>
<td>Object bounding box area</td>
</tr>
<tr>
<td>( S )</td>
<td>Hidden states of HMM model</td>
</tr>
<tr>
<td>( T_b )</td>
<td>Threshold for background modeling</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>$T$</td>
<td>Motion trajectory</td>
</tr>
<tr>
<td>$T_t$</td>
<td>A preset threshold value based on the nature of the event</td>
</tr>
<tr>
<td>$\mathcal{T}$</td>
<td>Trajectories set for training video set</td>
</tr>
<tr>
<td>$\mu_{k,t}$</td>
<td>Mean of the $k^{th}$ Gaussian distribution at time $t$</td>
</tr>
<tr>
<td>$\Gamma_i$</td>
<td>GMM for entry zones</td>
</tr>
<tr>
<td>$v_n$</td>
<td>Velocity of the moving object</td>
</tr>
<tr>
<td>$w_i$</td>
<td>Object width</td>
</tr>
<tr>
<td>$W_{i,t}$</td>
<td>The prior weight of the $i^{th}$ distribution at time $t$</td>
</tr>
<tr>
<td>$X_s$</td>
<td>The first state of geometry point in the trajectory</td>
</tr>
<tr>
<td>$X_f$</td>
<td>The last state of the geometry point in the trajectory</td>
</tr>
<tr>
<td>$\gamma_t$</td>
<td>The optimal sequence state estimated from Viterbi Algorithm</td>
</tr>
<tr>
<td>$\pi_o$</td>
<td>Initial state probability distribution for HMM model</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>HMM model notation which comprises of the parameters of { $\pi_o$, $A$, $B$ }</td>
</tr>
<tr>
<td>$\eta$</td>
<td>The normal distribution</td>
</tr>
<tr>
<td>$\sigma_{k,t}^2$</td>
<td>Variance of the $k^{th}$ Gaussian distribution at time $t$</td>
</tr>
<tr>
<td>$\circ$</td>
<td>Morphology opening</td>
</tr>
<tr>
<td>$\Theta$</td>
<td>Morphology erosion</td>
</tr>
<tr>
<td>$\oplus$</td>
<td>Morphology dilation</td>
</tr>
<tr>
<td>$\sim$</td>
<td>Distributed according to</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Figure 1.1</td>
<td>Typical CCTV control rooms monitoring multiple camera views. Security officers are often required to oversee multiple scenes at one time.</td>
</tr>
<tr>
<td>Figure 1.2</td>
<td>The AISight system: all cameras are connected to a server. An alert will be sent out to a human supervisor if the system detected any anomalous behavior [13].</td>
</tr>
<tr>
<td>Figure 1.3</td>
<td>screen capture of the AISight system that monitors mass transit station [14].</td>
</tr>
<tr>
<td>Figure 1.4</td>
<td>A typical framework for an object-based automated anomaly detection system.</td>
</tr>
<tr>
<td>Figure 1.5</td>
<td>Examples of abnormal events. (a) Illegal U-turn [15] and (b) A person loitering among cars in car parks [16].</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>Object detection and tracking framework proposed in this work.</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>(a) Scene with illumination changing condition. (b) The histogram of the pixel value at point + [138,96] showing multivariate distribution.</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Complete algorithm of the background modeling and object detection framework.</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>(a) – (c) snapshots of a video sequence showing a vehicle left the parking slot and drove away. The pixel value in the marked point is recorded throughout the video sequence. (d) A plot of the changes in pixel value (red channel) at the marked point. The graph shows a significant change in pixel value at approximately frame 4200 is observed which is the instance where the car left the parking slot. (e) Distribution of the pixel values at the marked point over a period of time. The histogram shows two distinct Gaussian distributions. The one on the left of the graph (centered at about 0.12) represents the distribution of the pixel value before the car left the parking slot, and the other one (centered at about 0.52) represents the distribution of pixel value after the car exits the parking slot.</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>(a) Initial scene of the outdoor car park where all pixels are static and (b) the corresponding background image. (c) A car enters the scene and triggers the object detector and resulted in (d) foreground pixels at the points where the car presents.</td>
</tr>
<tr>
<td>Figure 3.6</td>
<td>Results of the noise removal processes.</td>
</tr>
<tr>
<td>Figure 3.7</td>
<td>Blob analysis Processes. Object features are extracted by applying the blob detection and connected component analysis.</td>
</tr>
<tr>
<td>Figure 3.8</td>
<td>Results of connected component analysis. (a) The bounding box (green box) computed by the connected component analysis for one object entering the scene and (b) its corresponding foreground image. (c) The bounding boxes computed by the connected component analysis for multiple objects entering the scene and (d) its corresponding foreground image. The number of object detected is shown in the top left corner of the original images.</td>
</tr>
</tbody>
</table>
Figure 3.9 Object features extracted from the object detection and connected component analysis.

Figure 3.10 Results of the object tracking process. (a) – (c) consecutive video frames for an object passing through the scene and (d) the extracted trajectory of the event.

Figure 4.1 System Framework for event recognition system by supervised learning approach. The system can be subdivided into four main phases; namely object detection and tracking phase, event representation phase, training phase, and event recognition phase.

Figure 4.2 (a) Feature decomposition from motion trajectory and object appearance data collected from the object detection and tracking process. (b) Graphical representations of each decomposed feature.

Figure 4.3 Graphical representations of the different features extracted from the motion trajectory. (a) The POI of the trajectory shown in red circles (start point and end point) and the route taken by the moving object as shown in a series of black dots. (b) The instantaneous velocity of the moving object from the origin to the destination.

Figure 4.4 The distributions of the three appearance parameters for human and vehicle objects. (a) Blob area, \( A_i \), (b) aspect ratio, \( R_i \), and (c) filling ratio, \( f_i \).

Figure 4.5 Object recognition algorithm.

Figure 4.6 Scene decomposition based on the region of interest. Different scene decompositions for (a) vehicle events and (b) human events.

Figure 4.7 Feature vector normalization. Original and resampled components of the feature vector; (a) x axis projection, (b) y axis projection, (c) velocity profile and (d) blob area profile.

Figure 4.8 Feature vector standardization for event ‘A vehicle enters the monitored area and parks at an empty parking space’. Original components of feature vectors before standardization; (a) x axis projection, (b) y axis projection, (c) velocity profile and (d) blob area profile. Standardized components of feature vectors; (e) x axis projection, (f) y axis projection, (g) velocity profile and (h) blob area profile.

Figure 4.9 Graphical representation of the (a) centering and (b) scaling processes for standardizing feature vectors.

Figure 4.10 Principal component analysis of feature vectors of vehicle events. Top row: “A vehicle passes by the monitored parking area”. Middle row: “A vehicle enters the monitored area and parks at an empty parking space” and bottom row: “A vehicle gets out of parking space and exits the scene”. From left to right, the x axis projection, y axis projection, velocity and area profiles components of the feature vectors. The graphs show a collection of feature vectors for different events (shown in multiple colored lines) and the respective principal component representations of each component (shown in the thick black lines).
Figure 4.11 Principal component analysis of feature vectors of human events. Top row: “A person walks through the monitored parking area”. Middle row: “A person enters the scene and picks up a vehicle” and bottom row: “A person drops off from vehicle and exits the scene”. From left to right, the x axis projection, y axis projection, velocity and area profiles components of the feature vectors. The graphs show a collection of feature vectors for different events (shown in multiple colored lines) and the respective principal component representations of each component (shown in the thick black lines).

Figure 4.12 Decision tree of the event recognition process.

Figure 4.13 Decision tree of the composite event recognition process.

Figure 4.14 (a) Snapshots of the scene. (b) Topology of the monitored area and the line of sight of the camera.

Figure 4.15 Experimental results of the object detection and tracking process for vehicle events. From left to right column, the tracked object with the computed bounding box, the foreground image and the tracked trajectory of the moving object. And from top to bottom, the event of “A vehicle passes by the monitored parking area”, “A vehicle enters the monitored area and parks at an empty parking space” and “A vehicle gets out of parking space and exits the scene”.

Figure 4.16 Experimental results of the object detection and tracking process for human events. From left to right column, the tracked object with the computed bounding box, the foreground image and the tracked trajectory of the moving object. And from top to bottom, the event of “A person walks through the monitored parking area”, “A person enters the scene and picks up a vehicle” and “A person drops off from vehicle and exits the scene”.

Figure 4.17 A test video of about 1 minute which contains 4 separate events was tested on the proposed system. (a) A car passed by the scene, (b) people walked by the scene, (c) people picked up a car, and (d) car moved out of the parking space.

Figure 4.18 (a) Extracted object trajectories and (b) their sequence of events detected from the test video stream. A composite event, a person picks up a vehicle and leaves the parking lot is analysed from the low-level events and temporal information.

Figure 4.19 Imperfect object detection, (a) the shadow was detected as a part of the walking person, (b) a part of the body was occluded by the stationary car, (c) a part of the moving object was categorized as background due to similar color spectrum and lastly, (d) the new background took time before it was incorporated into the background model.

Figure 5.1 System architecture of the proposed event recognition and anomaly detection.

Figure 5.2 Hierarchical Trajectory Learning Model.

Figure 5.3 Regions of interest computed from the scene modeling technique proposed in this work. (a) The observed scene. Regions of interest for (b) vehicle events and (c) human events.

Figure 5.4 Trajectory resampling based on (a) time interval and (b) distance.

Figure 5.5 Distance and angle vectors construction.
Figure 5.6 Feature transformation framework for (a) distance vector and (b) angle vector.

Figure 5.7 K-mean clustering algorithm.

Figure 5.8 Feature vector classification strategy.

Figure 5.9 Results of the proposed feature vector classification approach. (a) Three classes of vehicle events based on the behavioral pattern of direction feature and (b) Three classes of human events based on the behavioral pattern of speed feature.

Figure 5.10 Hidden Markov Model (HMM).

Figure 5.11 Log-likelihood of the probabilistic parameters of events during the iterative EM algorithm. (a) Vehicle events and (b) human events.

Figure 5.12 Event recognition and anomaly detection algorithm.

Figure 5.13 Sample test videos. (a) A people walked through the observed scene, (b) people picked up a car, (c) A vehicle enters the observed scene and occupies an empty parking lot, and (d) car passed by the observed scene.

Figure 5.14 Experiment result of the POI analysis for sample events as listed in Table 5.3. (a) Vehicle events and (b) Human events.
### LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page no.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 4.1</td>
<td>Logical POI and object type combinations for different known events.</td>
<td>55</td>
</tr>
<tr>
<td>Table 4.2</td>
<td>Detailed conditions of the ‘pick up’ and ‘drive in’ composite events.</td>
<td>64</td>
</tr>
<tr>
<td>Table 4.3</td>
<td>Distribution of events across the training and testing datasets.</td>
<td>66</td>
</tr>
<tr>
<td>Table 4.4</td>
<td>Statistical result of the proposed event recognition system.</td>
<td>69</td>
</tr>
<tr>
<td>Table 4.5</td>
<td>A process of mapping tracks onto events in the sample video sequences. The events are labeled accurately with a high correlation coefficient.</td>
<td>71</td>
</tr>
<tr>
<td>Table 5.1</td>
<td>Logical POI and object type combinations for different known events.</td>
<td>99</td>
</tr>
<tr>
<td>Table 5.2</td>
<td>Statistical result of the proposed event recognition system.</td>
<td>100</td>
</tr>
<tr>
<td>Table 5.3</td>
<td>Sample event recognition result of the test video with the point level analysis and likelihood test results.</td>
<td>101</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

ABSTRACT ii
ACKNOWLEDGMENTS iii
AUTHOR’S DECLARATION iv
MEMORANDUM v
LIST OF ABBREVIATIONS vi
LIST OF SYMBOLS vii
LIST OF FIGURES ix
LIST OF TABLES xii

1. INTRODUCTION 1
   1.1 RESEARCH BACKGROUND 1
      1.1.1 The prevalence of surveillance camera 1
      1.1.2 Intelligent Vision Systems 3
      1.1.3 Abnormal Behavior Detection 6
         1.1.3.1 Activity Learning 8
      1.1.4 Terminology 9
   1.2 AIMS AND OBJECTIVES 12
   1.3 THESIS OUTLINES 13

2. LITERATURE REVIEW 14
   2.1 OBJECT-BASED EVENT ANALYSIS 14
      2.1.1 Object Detection and Tracking 16
         2.1.1.1 Background Modeling and Object Detection 16
         2.1.1.2 Object Tracking 17
      2.1.2 Event Representation 17
      2.1.3 Activity Modeling 18
      2.1.4 Event Learning Strategies 19
         2.1.4.1 Supervised Learning 19
         2.1.4.2 Semi-supervised Learning 20
         2.1.4.3 Unsupervised Learning 21

3. FEATURE EXTRACTION 23
   3.1 OBJECT DETECTION 25
      3.1.1 Adaptive Mixture of Gaussians (GMM) Background Modeling 26
   3.2 NOISE REMOVAL 33
      3.2.1 Median Filtering 34
      3.2.2 Morphological Operations 34
   3.3 BLOB ANALYSIS 37
      3.3.1 Connected Component Analysis 37
      3.3.2 Objects Feature Extraction 38
   3.4 OBJECT TRACKING 40
CHAPTER 1

INTRODUCTION

1.1 RESEARCH BACKGROUND

1.1.1 The Prevalence of Surveillance Camera

The use of digital video cameras in surveillance network has proliferated universally since the September 11, 2001 terrorist attack in United States. In Response to the increasing threat to the national security and safety, the Department of Homeland Security (DHS) was established by the federal government of United States to secure their homeland against fatal terrorist attack and to protect critical infrastructure and cyber networks. Since the establishment of the DHS, substantial funds have been injected to municipalities for installing a wide range of modern security equipment such as surveillance cameras and the centralised monitoring facilities [11]. Since then, surveillance cameras have become one of the essential tools for preventing crime and in response to domestic emergencies.

Today, substantial numbers of surveillance cameras have been installed in most public spaces such as railway and subway stations, parking lots, airports, highways, shopping malls, residential streets and etc. These surveillance camera networks are not only to prevent terrorism, but also for crime detection, monitoring live traffic condition and military operations. A review on “public area Closed-Circuit TeleVision (CCTV) and crime prevention” revealed that the deployment of CCTV surveillance cameras in car parks have caused a 51% reduction in crimes and CCTVs in public transport settings have recorded a 23% decrease in crimes [1]. In addition, 6,000 camera-related arrests have been documented in Chicago since 2006 [2]. In London, the largest metropolitan area of United Kingdom, 300 caught on camera per day are reported according to some estimates [3]. Apparently, the surveillance camera has become indispensable equipment for security and safety management as crime deterrent as well as conviction of offenders.
Apart from the public security tool, surveillance camera is ubiquitous for military operations as well. According to a report [4], the global military video surveillance system market is estimated to reach about USD 8.81 billion in year 2012. The rapid development of unmanned system with intelligent, surveillance and reconnaissance (ISR) has led to the increasing demand of video surveillance system. Most of these video surveillance payloads are mounted on unmanned aerial vehicles (UAVs). Moreover, the UAVs market is predicted to reach 2.7 billion USD in year 2012 and will increase to 6 billion USD in year 2021.

Enormous video footage data have been collected since the prevalence of the surveillance camera. More than 30 million of surveillance cameras are deployed in the United States shooting averagely 4 billion hours of footage a week [5]. This deluge of video data is obviously far outrun the capability of security officers to interpret the footages round-the-clock. This has been proven in [6] which stated that the human; including trained observers tend to become inattentive and notably less efficient after 20 minutes of monitoring TV screens. Examples of CCTV control room are shown in Figure 1.1. A security officer is often required to monitor multiple camera scenes continuously over long period of time.

During the International Carnahan Conference on Security Technology in 1999, the British CCTV security expert Geoff Thiel pointed out that the CCTV cameras installed do not provide effective real time surveillance unless they are watched actively [7]. In most cases, the CCTVs remain unwatched while most of the incidents are occurring. In this case, CCTVs will be relegated to a forensic role which serves as
evidences for post-mortem investigation, rather than an active alerting system to detect potential threat such as suicide bombers or other domestic emergencies/disasters.

1.1.2 Intelligent Vision Systems

This section discusses the rise of intelligent vision systems which can automatically interpret the context of video sequences and detect suspicious behavior/activity almost instantly to alert the security personnel. The unprecedented advancement in hardware of vision system which used to be erratic and monochrome feasibly enables complex video processing to be computed effectively. The latest generation of digital surveillance cameras which evolved from analogue technology provides clear and high quality video images with a relatively cheaper price. Moreover, the rapid developments of the digital signal processors, encoders, sensing modalities that increase the storage and processing capacity expedite the viability of automated video surveillance system by computer software.

Intelligent vision system was first initiated in 1980s, where the system designer sought to overcome the bored-guard effect by developing Video Motion Detectors (VDMs). Though the approach was ground-breaking at that time, it has not turned out to be a security panacea. The system raised frequent false alarms by the innocent sources of image changes. In the aftermath of the Madrid bombings in 2003, London Underground tested intelligent vision at Liverpool street station. However, the system was found error prone and triggered false alarms constantly [3].

Due to the poor reliability and effectiveness of the intelligence vision system at that time, many researches have been initiated aiming to develop a reliable automated intelligent video surveillance in order to catch up the increasing demand in the automated security and safety management system. The Defense Advanced Research Project Agency (DARPA) based in the United States has funded many research projects to develop the machine-based visual intelligence for unmanned surveillance system [17]. For instance, The Video Surveillance and Monitoring (VSAM) project was funded
in 1997 to develop battlefield surveillance and monitoring application which is capable of retreating troops from dangerous situation.

Apart from that, twelve research teams have been contracted for Mind’s Eyes project in 2011 which aims to develop a novel software sub-system that provides capabilities to learn events and generates representation of action between objects from visual inputs to deepen the understanding of the event by adding action description to the recognised objects in the scene [17]. In 2009, the European Union ploughed USD 3.1 million into the SAMURAI project to develop a new program that monitors suspicious and abnormal behavior in real time using a network of surveillance cameras [18].

After many years of research, a new generation of computerized “Big Brother” cameras will be installed in San Francisco’s MUNI transit induces the upsurge in the recent intelligent surveillance [14]. The Municipal Transit Authority spends about USD 2.2 million to install 22 BRS Labs AISight Cameras at 12 MTA train stations recently. The system will be able to detect suspicious and abnormal activities without human supervision. It is capable of tracking up to 150 objects simultaneously in real time and build memories of the behavior pattern of the observed scene. As the system mature with time, it will recognize the activity detected and alert the security personnel by sending clips of anomalous activity through the internet along with SMS text message if a suspicious behavior is detected. The Greater Lafourche Port Commission has also implemented a video surveillance system that integrates this BRS’s AISight video analytics software into Port Fourchon’s Video Management System to improve the security in Louisiana’s port [8].

This milestone redefines the role of surveillance camera in crimes and terrorists detection, traffic monitoring, military operations and commercial market. With the autonomous behavior recognition and anomaly detection, surveillance camera is now migrating from reactive to proactive security equipment. It is difficult to identify malicious and salient activity before its occurrence for a traditional video surveillance monitoring. However, intelligent surveillance camera has the capability of
independently identifying potential risks occur in the field of view, for instance, detecting an abandoned package which may be ignored by the jaded human eyes.

**Figure 1.2** - The AISight system: all cameras are connected to a server. An alert will be sent out to a human supervisor if the system detected any anomalous behavior [13].

Moreover, the intelligent camera makes the human monitoring completely redundant as the intelligent surveillance system will alert the security personnel if any salient activity occurred in a real-time basis. Thus, offering the benefits of expanded monitoring capabilities and reduction in the number of security personnel needed. Therefore, an increasing demand for the intelligent surveillance system is predicted over the next decade. The CCTV with biometrics and suspicious behavior detection market is predicted to experience growth from 750 million USD in year 2011 to 3.2 billion USD by year 2016 [9]. Figure 1.2 illustrates the system architecture of the AISight surveillance system where a centralised processing server is used to process the live video stream data from the surveillance cameras deployed all over the places and alert the security personnel when anomaly is detected. Figure 1.3 illustrates the actual graphical user interface of the AISight system installed in one of the transit station.
1.1.3 Abnormal Behavior Detection

One of the main reasons of ineffective attempts in intelligent surveillance in the past decades is because of the insufficient computational capability and inadequate understanding of how human brain models and manipulates data, particularly, computer vision [10]. Computer vision, a branch of artificial intelligence is intended to perform human’s cognition, perceiving and understanding to imaging tools by computers, in other words, attaching “brains” to these cameras. Most of the computer vision technologies use the method of mathematics, pattern recognition/classification, artificial intelligence, psycho-physiology, computer science, electronics and others scientific disciplines. Automated anomaly detection or abnormal behavior detection is one of the most active research topics in the field of computer vision.
Figure 1.4 depicts the general framework of automated anomaly detection system by employing computer vision and machine learning algorithms. Some of the common methodologies used for each process are listed as well. The first level of computation concerns of extraction of relevant features for activity understanding. This process normally comprises of low-level digital image and video processing algorithms which provides very little information about the video content. The output of the feature extraction process normally constitutes of feature vectors which may include the shape, colour histograms, motion trajectory and velocity of the moving objects.

Figure 1.4 – A typical framework for an object-based automated anomaly detection system.
Higher level of image understanding is used for semantic interpretation of the activities occur in the monitored scene. Event analysis exploits the output of the low-level feature extraction modules and uses more domain-specific knowledge to understand and interpret the video content. Most of the proposed approaches for the video surveillance-oriented event analysis can be broadly categorized into explicit event recognition and anomaly detection.

An explicit event recognition system has a prior knowledge of all the identified events. An activity is defined as a sequence of atomic events and it uses a database to store all the recognizable events descriptions. The system acts like a “parser” by matching the incoming data with the predefined events description in the database. Once an event is detected, the semantic scene description of event will be labelled. The higher level activity is subsequently classified based on the sequence of detectable atomic events happened within an acceptable time frame. For example, for a parking lots surveillance video, the activity of “a car is parked” can be recognized by detecting the sequence of atomic events “a car enters the scene”, “car stops”, “a person exits from the car”, and “the person leaves the scene”.

On the other hand, anomaly detection system employs the statistics approach to discriminate among the normal and abnormal events. The system identifies the most frequent event (normal events) patterns by a learning process based on the data acquired in the monitoring scene. Events that deviate too much from the norms constitute anomalies. Therefore, an anomaly detection system does not require prior knowledge of the events and also, it can be completely unsupervised. However, it cannot explicitly define what is the activity occurring in the scene.

1.1.3.1 Activity Learning

In the broadest sense, most of the methods for event analyses employ activity learning during the training phase. Activity learning refers to some form of algorithm that finds the normal and atypical activities pattern by using training data available. There are three categories of learning approaches which are characterised based on the nature of
the training dataset; namely supervised, semi-supervised and unsupervised learning approaches.

**Supervised Learning:** In supervised learning paradigm, each type of known activity is learned by a separated model. The training dataset encompasses labelled data for each class of activity. In other words, a priori knowledge is needed for such learning paradigm.

**Unsupervised Learning:** An unsupervised learning approach however, refers to finding the hidden pattern from a set of unlabeled data. Therefore, no prior knowledge about the dataset is required for an unsupervised learning paradigm.

**Semi-supervised Learning:** Semi-supervised learning technique falls between unsupervised learning and supervised learning methods. The learning model is initialised with handful of labelled data during the training phase and subsequently trained with a large amount of unlabeled data.

### 1.1.4 Terminology

Some of the terms that are frequently used throughout the thesis are described below. These terms will be defined and explained clearly to avoid the ambiguity.

**Activity**

Activity refers as a sequence of atomic action performed by a subject. An action, on the other hand, implies a sequence of fundamental movements performed by an object. Therefore, trajectory is used to embody an activity in this thesis context since the trajectory is the solely cue of the motion of an object which obtained from the result of motion detection algorithms. In this thesis, an activity is characterised by three components. These components make the activity analysis feasible.
**Spatial:** Different activities undergo distinct paths and routes. This can be demonstrated by the activities of “vehicle turning left”, “vehicle turning right”, “vehicle U-turning”, and “vehicle moving straight”. There is a significant difference among the patterns of the route in the sense of direction, angle and location. Moreover, this spatial information facilitates scene modeling as well. For instance, the starting and ending point of an activity enable the task of defining the location of entrance and exit in a particular scene.

**Dynamics:** The dynamic of the activity indicates the manner in which a route is being travelled. One of the dominant parameters that describe the dynamic of an activity is the velocity and acceleration profiles of the moving object.

**Temporal:** the temporal information characterises the timing of an activity, for example, the start time and end time of an activity. This information is essential for identifying composite event that involves interaction of multiple simple events.

**Event**

Most of the researches defined event as the occurrence of an activity in a particular time and place. Normally, there are two types of event that are concerned in the works for autonomous anomaly detection in video surveillance:

**Simple Events:** An event that is characterized by a single homogenous action performed by a single object which the low-level spatiotemporal entity cannot be further decomposed. Taking the case of parking lot surveillance, a simple event may include:

- A person enters the monitoring scene
- A person is walking
- A person exits the monitoring scene
- A car enters the monitoring scene
- A car stops
- A car exits the monitoring scene
Composite/Complex Event: A composite event is constituted of consecutive simple events. It can be characterized by the actions of multiple objects. Thus, a composite event may show the interactions among multiple objects. For example, in the context of a parking lot area, the complex events can be:

- A group of person entering the scene
- A queue of cars
- Loitering

Abnormal Event/Activity

Abnormal event or activity refers as unusual, odd, out of the ordinary, peculiar, unexpected type of event. In technical sense, an abnormal event/activity means the patterns of action that do not conform the expected/normal behaviours that have been learned from the previous records. Most of the intelligent video surveillance systems are aiming to detect abnormal event/activity in a scene to prevent malicious, risky and dangerous actions/activities or it may against the law or regulation.

![Figure 1.5 – Examples of abnormal events. (a) Illegal U-turn [15] and (b) A person loitering among cars in car parks [16].](image)
1.2 AIMS AND OBJECTIVES

The aim of this thesis is to develop an autonomous anomaly detection and event classification system for car park lot surveillance that yields the best performance with minimal constraint by employing the machine learning and computer vision algorithm in manipulating trajectory data.

Specifically, the following objectives will be focused to address the aim:

1. Review the existing strategies for trajectory-based event recognition and anomaly detection for surveillance application.

2. Implement a robust object tracking system that capture the dynamic changes in the spatiotemporal of a moving object and present the trajectory data accurately.

3. Accomplish a compact representation of the motion trajectories that provides meaningful information of the event in the intended application.

4. Devise an automatically learning of semantic scene model that labels regions of interest according to identifiable activity such as entry and exit zones.

5. Establish an event learning system that defines explicitly a semantically meaningful high-level description of the activities, actions and events based of trajectory data.

6. Perform events recognition and anomaly detection in real world car park surveillance video.

7. Empirically investigation on the performance and efficiency of proposed algorithms.
1.3 THESIS OUTLINES

This thesis will be presented in six chapters. The following paragraphs introduce the scope of individual chapters.

**Chapter 2** covers a literature review of existing approaches in each component in event classification and anomaly detection system. The related works in object tracking, event representation, event learning, scene modeling and event recognition will be outlined in this chapter.

**Chapter 3** presents the object detection strategy from video sequences and object tracking algorithms utilised in this work. In particular, this chapter describes the adaptive Gaussian mixture background modelling and Kalman filtering algorithm. The detected objects are classified to distinguish among human and vehicle.

**Chapter 4** presents the explicit event recognition by training the event classifier with a manual labelled dataset. Experiments are conducted to evaluate the performance of the proposed algorithm.

**Chapter 5** presents the mechanism of long term adaptive unsupervised trajectory learning for events understanding approach. Empirical investigation on the experiments result is carried out to measure the proposed anomaly detection system. The event classification performance among two approaches is compared.

**Chapter 6** provides conclusions drawn from the thesis and suggests a number of possible directions as future work.
CHAPTER 2

LITERATURE REVIEW

Vision-based event recognition and anomaly detection system can generally be categorized into two main strategies: object-based and pixel-based event analyses. Object-based event analysis takes into consideration of the spatial, dynamic, shape and temporal information of the object of interest detected from the foreground segmentation and object tracking process. On the other hand, unlike object-based event representation, the pixel-based event analysis does not require grouping of connected pixel to blobs or objects, rather, it analyses event by exploiting the features in pixel-level such as color, intensity and gradient from the image sequences.

Due to the broad research area in video processing and understanding, this review focuses only on specific techniques and algorithm used in object-based activity analysis that are widely adopted in visual surveillance systems. Specifically, this review is structured into four subsections: Object detection and tracking (Section 2.1.1), event representation (Section 2.1.2), activity modeling (Section 2.1.3) and event learning strategies (Section 2.1.4).

2.1 OBJECT-BASED EVENT ANALYSIS

In general, a typical object-based event recognition and anomaly detection system for surveillance videos consists of several important processing phases:

1. Object detection and tracking is a low-level video processing algorithm which detects the object of interest from video sequences captured by the visual sensors (typically a digital camera). The key components in this low-level video processing phase are background modeling, object detection, object tracking and feature extraction. Background modeling and foreground segmentation of the
video footage extract the target objects from the raw image sequences which provide fundamental input of the activities. The sequential locations of the moving object form a trajectory of motion that describes the spatial-temporal behavior of the object over a finite interval of time. On the other hand, the chronological changes of the blob/object describe the body-level action of the object itself over time.

2. Event Representation process which describes the behavior of the detected event by constructing a set of attributes extracted from the motion trajectory such as location, velocity, acceleration, direction and shape of the tracked object within the finite time interval. This process performs the extraction, selection, and transformation of low-level visual properties extracted from the object detection and tracking processes to construct event descriptors to activity models.

3. Event learning for activity understanding and anomaly detection is a process that learns the behavior of an event from a set of event descriptors with similar nature. There are several event learning strategies developed in the past decades based on the amount of human interpretation involved. Namely, the supervised learning, semi-supervised learning and unsupervised learning. All of these event learning approaches are aiming to devise a general description/pattern for each type of events.

4. Event Recognition and anomaly detection is an algorithm that semantically interprets the video content based on the knowledge learnt from the training data and detects unusual activities which do not belong to any class of patterns learnt from the training data provided.
2.1.1 OBJECT DETECTION AND TRACKING

2.1.1.1 Background Modeling and object detection

One of the main aims of background modeling and object detection is to identify the regions within the image where motion is present. All object detection strategies were established based on the fact that motion changes the pixels value over time. However, in the real life scene, a reliable object detection algorithm has to overcome noises such as change in pixel values regardless of motion, which is caused by environmental factors (illumination changes, waving vegetation, etc.) and recognize motions that do not change the pixel value due to occlusion.

Optical flow was one of the earliest motion detection strategies that estimate motion as either instantaneous image velocities or discrete image displacements [19, 20]. Assuming each pixel value in a video sequence as a 3-dimensional signal (2-dimensional spatial coordinates and time), the motion can be detected by local Taylor series approximation of the video signal with respect to the spatial and temporal coordinates. The main disadvantage of optical flow based motion detection is that the solution to the differential equation of video signals is computationally expensive and thus may not be suitable for real-time visual surveillance systems.

Background subtraction is another common object detection technique that uses a static background image as reference to identify foreground objects by subtracting the new image frame with the background image. The background image is continuously updated to adapt to the environment changes [62, 63]. The background subtraction strategy is relatively simple in computation but very sensitive to noise such as sudden change in illumination, shadows, and repetitive motion such as waving trees. Thus, background subtraction may not be practical in outdoor scenes where large amount of undesirable environmental factors take place.

Gaussian Mixture Model (GMM) proposed by Stauffer and Grimson has been one of the most reliable background modeling methods which models each pixel as a mixture of Gaussian distributions [21]. GMM background modeling technique has been
widely used in activity understanding and unusual event detection systems in surveillance video especially outdoor scene [21, 22, 23, 24, 25].

2.1.1.2 Object Tracking

The main aim of motion tracking is to record the motion of objects from video sequences for each individual activity. The chronological spatial locations of the moving object over a finite interval of time can be visualized as a motion trajectory. The spatial information of objects computed by the motion detection algorithm in consecutive image frames form a track in the 2-Dimensional image plane.

A trajectory is normally obtained by collecting a set of features of the detected object such as location, velocity and shape over consecutive frames using motion tracking algorithms [26]. In a real surveillance system, simultaneous tracking of multiple objects is often required especially in public scenes. In this case, the tracking process is much more complicated due to the interaction of the objects such as dynamic occlusion, merging or splitting from a group of objects. Due to the highly dynamic nature of the detected object, estimation and filtering algorithm such as Kalman Filter [27], Particle Filter [20] and moving average smoothing [28] are commonly adopted.

2.1.2 EVENT REPRESENTATION

Event representation in the context of video understanding links the low-level visual information and the high-level semantic interpretation by extracting, selecting, and transforming a set of features from the low-level video processing to construct an event model. A well-defined event representation can describe a variety of activities in a given scene, but discriminatively distinguishable in between different activities [29]. Event representation in surveillance video can be generally clustered into two categories: object-based representation and pixel-based representation. In this review, the object-based event representation approaches are presented as they are closely related to the nature of this work.
Object-based representation constructs a set of features for the moving object detected in the video streams. There are two types of features in object-based representation: spatial-temporal descriptors (trajectory) and object descriptors (size and shape). Trajectory-based feature is commonly used in representing event in video content understanding [28, 30, 31, 32]. In a recent study, Morris et. al. presented a trajectory analysis framework that characterizes an activity by decomposing the motion trajectory into multiple levels of resolution and then trains the descriptors of different levels separately with different approaches [61].

2.1.3 ACTIVITY MODELING

There is a wide variety of activity modeling approaches in surveillance video such as Bayesian networks and Probabilistic topic models. Bayesian network (BN) has been widely used for activity modeling due to its computational viability and capabilities in modeling variations in visual properties. For example, Intille and Bobick apply BNs for modeling multi agent interactions [34]. Buxton and Gong use BNs in a traffic surveillance application to model activity from low-level features extracted from object detection and tracking [33].

The Hidden Markov model (HMM) framework has been extensively used for activity modeling and recognition. For example, Brand et. al. proposed a coupled hidden Markov model (CHMM) to model interactions between temporal processes [35]. Gong and Xiang presented a dynamically multi-linked hidden Markov model (DML-HMM) for modeling group activities [36]. Apart from that, hierarchical hidden Markov model (HHMM) in [37] and switching hidden semi-Markov model (S-HSMM) in [38] are all derived from the basic HMM framework.

Another extension of HMM, The cascade HMM has been proposed for activity analysis. For instance, Zhang et al. presented a similar framework based on LHMM to learn different levels of actions [40]. Similarly, Oliver et al. use the same model to capture different levels of temporal details in recognizing human activity [39].
Other than Bayesian networks, Probabilistic topic models (PTMs) such as probabilistic latent semantic analysis (pLSA) and latent Dirichlet allocation (LDA) [43] have recently been adopted in activity modeling. Li et. al. use pLSA to model events at traffic intersection scene [41]. Mehran et. al. employ LDA to model human interactions for unusual event detection in a crowded scene [42]. Wang et. al. proposed two hierarchical PTMs to model higher-level human interactions [44].

2.1.4 EVENT LEARNING STRATEGIES

Event learning is an important process in activity understanding and anomaly detection framework. It teaches the surveillance system the behavioural pattern of potential events in the monitored area. Different event learning strategies are discussed in the following sub-sections.

2.1.4.1 Supervised Learning

A supervised event learning approach can be described as a teacher teaches a student what is right and wrong with a number of samples. Specifically, the video surveillance system is trained with both labelled normal and abnormal behavioural patterns during the learning phase. Supervised learning approach was one of the earlier endeavours on activity recognition and anomaly detection. The drawback of this learning strategy is that huge amount of labelled video samples are required at the training stage and thus, this learning approach is not feasible especially for surveillance cameras that monitor the public scene due to the broad type of events happened in the scene. Huge amount of video samples are necessary in order to capture all possible instances of normal and abnormal events. In addition, human interpretation in labelling all the positive and negative events in the training dataset is no longer practical for large volume of video inputs.
Supervised learning strategy was adopted in earlier efforts on activity recognition and anomaly detection for visual data with assumption that both normal and abnormal activities are well-defined and available during training phase. Oliver et al. propose a human interactions recognition system that learns from a set of known behavior classes by using a CHMM framework [45]. Morris and Hogg train a statistical model with both normal and unusual trajectories to detect anomaly activities in their car park surveillance system [46]. Similarly, Sacchi et al. train a neural network with both normal and unusual patterns to detect crime acts [47]. Gong and Xiang classify activities by employing a DML-HMM model with known event classes [36]. In the efforts on unusual event detection [48, 49], a set of well-defined rules on what is normal/abnormal behavior are embedded into the training system.

The above mentioned video understanding systems that employ supervised learning approach have shown to offer effective solutions for activity understanding and anomaly detection in simple scenes. However, supervised learning approach is no longer feasible given a public scene with complex activities which are varying over time. In this case, a supervised approach requires a massive amount of labelled samples to model the complex behaviours to prevent false detection on rare instances.

2.1.4.2 Semi-Supervised Learning

In vision-based surveillance system, unlabeled video samples are normally easier to collect. Unlike labelled video samples, unlabeled data do not require human analysis and interpretation which is tedious and time consuming. Due to the difficulty in collecting labelled dataset, researchers have developed a more practical way to train the system by providing a smaller amount of labelled data to construct the event models and a relatively larger amount of unlabeled data are then clustered by the predictive model trained by the handful of labelled data [50]. The unlabeled data are assigned to the prior knowledge if it fulfils a preset confidence level. Normally, the event models are retrained after classification of the unlabeled data during the event learning phase.
Zhang et. al. proposed an unusual event detection framework based on semi-supervised learning approach by training a HMM using labelled data, and then derive unusual behavior models by performing likelihood test on unlabelled video sequences [51]. Recognition of events is accomplished based on the usual and unusual models. The accuracy of event recognition systems that adopt semi-supervised learning strategy rely strongly on the correctness of classifying the unlabeled data. Poor classification of unlabeled data may impair the event models that are trained by the handful labelled data [52].

### 2.1.4.3 Unsupervised Learning

Unlike the supervised and semi-supervised learning strategies where labelled dataset are provided, unsupervised event learning approach does not required any form of human annotation on the training dataset. A set of unlabeled video samples captured from the surveillance camera are the only input to the system. Classifications of events from the unlabeled data are often performed by clustering process, where small clusters of events are assumed as outliers or anomaly. Though this learning approach requires no human involvement at all during the training phase, the number of event types (clusters) in a particular scene is often a critical parameter to be determined. Given a new behavioural pattern in the video scene, for example a re-routing of road direction in the monitored area will cause the system to classify the new event as being unusual. In this case, a re-learning of events is necessary.

Zhong et. al. implemented a bipartite co-clustering technique on video segments and identify isolated clusters as unusual events [53]. Similarly, Lee et. al. proposed an N-cut clustering technique to differentiate unusual behaviours from normal behaviours based on distance measurement [54]. Boiman and Irani construct a database of spatiotemporal data using only normal behavior and detect unusual patterns which cannot be composed of the database [55].
Another class of unsupervised learning strategy is to train a model using only normal events and anomaly are detected as outlier. This approach stands upon the observations that surveillance data are generally dominated by large amount of normal events and unusual events are typically rare. Thus, the definitions of normal events are generally more expressive as compared to rare instances [56, 53]. Duong et. al. employ only the labelled normal patterns to train a Switching Hidden semi-Markov Model (S-HSMM) to detect unusual activities [57].

Most of the methods that adopt unsupervised learning mentioned above do not update the event model over time. In reality, the definition of normal and abnormal behaviours may change over time or new unseen events may arise from time to time [58]. As a result, the static event models that are once trained may not fit well to the scene in the long run. In order to accommodate the visual context changes, Xiang and Gong proposed an unsupervised learning approach that continuously updates the model over time [59, 58]. By exploiting the classification output of the newly detected events, the model is then updated. Another adaptive unsupervised learning approach was proposed by Kim and Grauman in their abnormal event detection system [60].
CHAPTER 3

FEAUTRE EXTRACTION

A video contains sequence of images and an image, as the Chinese proverb said, is worth a thousand words. A single image provides snapshot of a scene that conveys the elementary information such as the shape, color, texture, location and motion of the objects. In conjunction with these information, the objects elements such as their semantic interpretation (objects’ meaning, for example, people, animal or vehicles), action (object’s pose such as standing or sitting) and the syntax (the way of the objects related for instance, a kids holding a woman’s hand) in the scene can be known. A video stream however is a much richer source of visual information than a still image since it registers the dynamic behavior of the scene. It captures the motion that carries a lot of information about the spatiotemporal relationships between images. Along with this information, the activity of the objects can be exhibited.

Feature extraction serves as the cardinal rule in most of the image understanding, pattern recognition/classification and machine learning methods. It extracts the useful information from the input data and transforms it into set of features that will most efficiently or meaningfully represent the information that is important for analysis and classification. This step is indispensable since much of the information in the still image data is little or no value for certain analysis problem, such as the activity recognition and anomaly detection in video surveillance applications. In other words, feature extraction reduce the dimensionality of dataset by removing irrelevant and redundant data, thus it reduces the computational cost, avoids data over-fitting and improves the efficient data-intensive processing task.

Figure 3.1 depicts the system modules to extract the object’s features and their spatiotemporal and dynamics features from video sequences. The implemented system is able to distinguish moving objects from a static background in a video sequence captured from a static CCTV camera for outdoor surveillance circumstance. The
The proposed system is also able to filter the unwanted or noisy object and classify the detected objects into different groups such as human and vehicle and finally capture the object’s trajectory information for video understanding and interpretation.

**Figure 3.1** - Object detection and tracking framework proposed in this work.
There are three main modules in the implemented system: *object detection, object tracking* and *object classification*. Algorithms for detecting motion and tracking the moving object in video sequences are undergoing tremendous development in recent years. Object detection and tracking technique is initiated by the widespread needs for surveillance such as tracking pedestrians or vehicles and detecting intruder or abandoned objects in the monitoring area. Nowadays, many other applications of such algorithms are motivated such as identifies lameness in horses [64], analysis of the motion of athlete for improving sport performance [65, 66], robot vision for navigation [67] and etc. A robust motion detection and tracking system is the cornerstone of most automated video surveillance system since it provides elementary information of the object’s behavior which serves as the key for event understanding. In this work, an object-oriented tracking system which based on statistical background modeling and object detection system is proposed. The tracked object is subsequently classified to human or vehicle by employing shape-based object classification algorithm.

### 3.1 OBJECT DETECTION

Object detection refers as the dynamic image analysis which identifies which image pixels, or more accurately which regions of the image are moving or changing over time. It aims to registers the detected motion and indicates the object’s position and orientation in a video sequence. A common approach for real time image segmentation in video sequence is through *background subtraction*. This method is based on the assumption that the background image from a static camera usually constitutes of stationary objects. As a consequence, the stationary background scene/reference image can be modeled or estimated statistically and objects which are aside from this background object simply the foreground/moving objects. Most reasonably, the background image in a real world scene is not constant as the foreground objects may incorporate into background after it is immobilized at the same location for a certain period of time or the background object may become foreground when it begins to move due to certain external factors such as wind and illumination changes in an outdoor environment. For this reason, maintaining the background frame dynamically is necessary especially for outdoor scenes. An adaptive background modeling approach
which is proposed by the Stauffer and Grimson [68] is adopted in our motion tracking system.

3.1.1 ADAPTIVE MIXTURE OF GAUSSIANS (GMM) BACKGROUND MODELING

An outdoor parking lot surveillance system normally encounters a deceptively complex scene owing to the illumination and climate changes. In such manner, a reliably real time tracker which is able to conquer these problems is essential since the performance of the event analysis is relied strongly on the quality of motion tracking outcomes. A background modeling which models each pixel independently as a mixture of Gaussians (GMM) is popularly used as a real-time outdoor tracker. The conventional background subtraction and unimodal background model is insufficient to model each pixel of the image under a variant lighting condition and repetitive motion such as waving vegetation.

The effect of illumination changes on the distribution of pixel value is demonstrated in Figure 3.2. The pixel value at point (138, 96) in a video sequence (see Figure 3.2(a)) is recorded under an illumination changing condition. The histogram of the pixel values reveals a multivariate Gaussian distribution as shown in Figure 3.2(b). The pixel value at point (138, 96) as depicted in Figure 3.2(a) was collected from a 9308 framed video sequence under a circumstance of illumination changing without any moving object. The distribution of the specific point exhibited a multivariate Gaussian distribution as shown in Figure 3.2(b). Over and above the robustness to handle illumination changes, this background modeling method can manipulate the unstructured and high frequency motion such as foliage and vegetation waving.
Figure 3.2 – (a) Scene with illumination changing condition. (b) The histogram of the pixel value at point $[138, 96]$ showing multivariate distribution.

The pixel density $X = \{X_1, X_2, \ldots, X_t\}$ for each RGB color space is modeled as $K$ multivariate Gaussians distribution where it can be written in the notation:

$$X \sim \eta(\mu_{k,t}, \sigma^2_{k,t})$$

where $\mu_{k,t}$ and $\sigma^2_{k,t}$ are the mean and variance respectively for the $K^{th}$ mixture at time $t$. These Gaussians along with their parameters (mean and variance) will be updated each frame as the environment in the video scene evolves over time. The number of $K$ distributions normally ranging from three to seven depends on the complexity of the environment. The selection of the value $K$ is also governed by the issues of available memory space and computation capability. In the conjunction with a time evolving weight parameter $\omega_{i,t}$ which defines how dense the underlying Gaussians in the pixels history, the probability of observing the current pixel value for each color space is:

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} \times \eta(X_t, \mu_{i,t}, \Sigma_{i,t})$$

where the pixel density function,

$$\eta(X_t, \mu_{i,t}, \Sigma_{i,t}) = \frac{1}{\sqrt{(2\pi)^2|\Sigma|}} e^{-\frac{1}{2}(X_t - \mu_i)^T \Sigma^{-1} (X_t - \mu_i)}$$
where $X_t$ is the current pixel value and $\Sigma_{i,t}$ is the covariance matrices of $i^{th}$ mixture coefficient at time $t$.

Thus, a new incoming pixel should be fallen into one of the major component of the Gaussian distribution. The current pixel is checked for any matches within 2.5-sigma (standard deviation) for each distribution. If there is more than one match, the best will be taken; which is the “winner takes all” theory. When the match is found, the parameters of the matched distribution, let says $k=i$, are updated with the following equation:

$$w_{i,t} = (1 - \alpha)w_{k,t-1} + \alpha$$  \hspace{1cm} (3.4)

$$\sigma_{i,t}^2 = (1 - \rho)\sigma_{i,t-1}^2 + \rho(X_t - \mu_{i,t})^T(X_t - \mu_{i,t})$$  \hspace{1cm} (3.5)

$$\mu_{i,t} = (1 - \rho)\mu_{i,t-1} + \rho X_t$$  \hspace{1cm} (3.6)

where $\rho = \alpha \eta(X_t | \mu_{i,t}, \sigma_k)$ which $\alpha$ is the learning rate, normally values in the range of 0.01 to 0.1 will be used depending on the application. The parameters for the remaining models remain the same.

If none of the distributions matches the current pixel value, the least weighted background distribution (with the lowest $w_t$ value) will be replaced with a new distribution with the current pixel value $X_t$ as the mean $\mu_t$ and a high variance value associates with low weight prior. This occurs when there is a new object occludes the current background pixel. The new pixel is given an opportunity to become a part of the background. However, the original background distributions in the $K$ mixture still remain. Hence, if a car is parked for some period and drove away subsequently, the previous background still exists with the same mean and variance however with a lower weight can be quickly reincorporated into the background again.

A background object can be reckoned as an apparently static and persistent object. Consequently, background pixel density distributions normally will have a high $w$ and low variance variable. A foreground or moving object as aforementioned will create a new distribution with a high $w$ and low variance value or fall into one of the $k$ mixture distribution however increasing the variance variable. Based on this concept,
the $k$ mixture of distributions is sorted by the ratio of $w/\sigma$. The order lists the most probable background model from top to bottom. As a result, the first $G_B$ background distributions can be extrapolated by

$$G_B = \text{argmin}_b \left( \frac{\sum_{k=1}^{b} w_k}{\sum_{k=1}^{B} w_k} > T_b \right)$$

(3.7)

where $T_b$ is a measure of the minimum threshold that portion of data should be accounted as the background model. The pixel can eventually labeled as foreground pixel if there was no match at all or the matched Gaussian distribution does not belong to the $G_B$ distribution(s) of the background. The complete algorithm of the background modeling and object detection framework used in this work is shown in Figure 3.3.

Figure 3.4 illustrates an example of the GMM background modeling and object detection on a video sequence of about 9000 frames. The video contents only one event where there is a car leaving the parking slot and then exits the scene. For the purpose of visualizing how the pixel values changes and its distribution, a point of the initial location of the car is considered (coordinated at (127,18) which marked with the white cross in Figure 3.4(a)-(c)). The pixel value of the marked point (red channel only) was recorded for the entire video frames and plotted in Figure 3.4(d). The graph shows a significant change in pixel value at approximately frame 4200 is observed which is the instance where the car left the parking slot (see Figure 3.4(b)). The distribution of pixel values over the entire video sequence is depicted in Figure 3.4(d). The distribution on the left of the figure (centered at about 0.12) shows the distribution of pixel values before the car left the parking space, and the one on the right (centered at about 0.52) depicts the distribution of pixel values after the car exits the spot.
**Input**: Set parameter for $K, T$ and $\alpha$

**Initialization**: Calculate the initial $\mu_{k,t}, \sigma_{k,t}, w_{k,t}$ for each color space and pixel by using the initial training video frames.

**For each time frame $t$ do**

**For each color space $(R, G, B)$ do**

**For each pixel $(x, y)$ do**

**For each Gaussian component $i=1$ to $K$ do**

If $\frac{|x - \mu_{i,t}|}{\sigma_{i,t}} \leq 2.5$ then

\[
\begin{align*}
    w_{i,t} &= (1-\alpha)w_{i,t-1} + \alpha \\
    \rho &= \frac{\alpha}{w_{i,t}} \\
    \mu_{i,t} &= (1-\rho)\mu_{i,t-1} + \rho(X_t) \\
    \sigma_{i,t}^2 &= (1-\rho)\sigma_{i,t-1}^2 + \rho((X_t - \mu_{i,t})^T(X_t - \mu_{i,t}))
\end{align*}
\]

else

\[
    w_{k,t} = (1-\alpha)w_{k,t-1}
\]

end

**normalize weight $w_{k,t}$ so that $\sum_{i=1}^{K} w_{k,t} = 1$**

if $\frac{|x - \mu_{i,t}|}{\sigma_{i,t}} > 2.5$ of all the distributions

\[
    l = \arg\min_k w_k
\]

\[
    \mu_{l,t} = x_t \\
    \sigma_{l,t}^2 = 30^2 (\text{uint8 format}) \\
    w_{l,t} = 0.5 \min_k w_{k,(t-1)}
\]

end

**normalize weight $w_{k,t}$**

**Classify the background and foreground distribution:**

**Sort Gaussians according $w/\sigma$**

**First $B$ distributions are chosen as the background model**

\[
    B = \arg\min_b \sum_{k=1}^{B} w_k > T
\]

**Foreground Detection:**

If the matched Gaussian is within $B$ then

\[
    X = 0
\]

else

\[
    X = 1
\]

end

end

end

**Figure 3.3** – Complete algorithm of the background modelling and object detection framework.
Figure 3.4 - (a) – (c) snapshots of a video sequence showing a vehicle left the parking slot and drove away. The pixel value in the marked point is recorded throughout the video sequence. (d) A plot of the changes in pixel value (red channel) at the marked point. The graph shows a significant change in pixel value at approximately frame 4200 is observed which is the instance where the car left the parking slot. (e) Distribution of the pixel values at the marked point over a period of time. The histogram shows two distinct Gaussian distributions. The one on the left of the graph (centred at about 0.12) represents the distribution of the pixel value before the car left the parking slot, and the other one (centred at about 0.52) represents the distribution of pixel value after the car exits the parking slot.

Figure 3.5 illustrates the resulted foreground pixels computed from the abovementioned background modeling and foreground segmentation algorithm. The scene was initially static (see Figure 3.5(a)) and therefore all pixels remain as background as shown in Figure 3.5(b). After that, a car moves into the scene (see Figure 3.5(c)) and the model detected significant changes at the locations where the car presents and hence categorizes the sudden pixel values changes as foreground pixels as shown in Figure 3.5(d).

The outcome of the object detection shown in Figure 3.5(d) may reveal imperfection of the foreground object detection as there are some holes exist in the foreground pixels (white pixels). There are several factors that cause the imperfection in the object detection algorithm such as similar color and lighting condition on certain
spot of the object compared to the background pixel, and therefore the algorithm could not recognize the changes. Also, the reflection of light caused by the moving object will result in noise where the non-foreground pixels being treated as foreground due to the sudden change in illumination. Another critical factor is the occlusion where the object of interest is blocked by other static objects relative to the sight of the static camera.

**Figure 3.5** - (a) Initial scene of the outdoor car park where all pixels are static and (b) the corresponding background image. (c) A car enters the scene and triggers the object detector and resulted in (d) foreground pixels at the points where the car presents.

Therefore, in order to optimize the object detection outcome, a series of post-processing techniques have to be applied to the foreground images to improve the accuracy of the object detection result. In this work, a noise removal technique is applied to filter off the unwanted noise and recover the foreground pixels as much as possible to the real object which will be discussed in Section 3.2. This is important as the improvement of the object detection may influence the following analysis such as object recognition (one of the important parameter for object recognition is object size which is determined by the foreground pixels detected).
3.2 NOISE REMOVAL

Following the segmentation of image frames into foreground pixels and background pixels, cleaning noises in the binary output image is essential to provide the best possible circumstance for blob analysis. These noises are caused by several factors:

- **Camera noise** which is mainly caused by the electrical components of imaging system (e.g. sensor noise).

- **Poor contrast** between background and moving objects. This happened in a manner that parts of the moving objects may have the same or similar colours and intensity with the background object.

- **Sudden variation in illumination** (e.g. cloud covers or switching off/on light) that causes the inaccurate foreground segmentation. This phenomenon happened when the rate of the changes is beyond the adaption/learning rate of the foreground detection model.

- **Shadows** of the moving objects cast on the ground surface normally cause an open and difficult problem in foreground segmentation. It is hard to discriminate shadows from foreground since they undergo dynamic changes as well.

The region-based tracking system employs blob analysis algorithm to identify the connected foreground pixels in the sequential image frames. Therefore, the goals of pixel-based noise removal should include the elimination of undesirable features and improvement of the visibility, perceptibility and delectability of the image features. In other words, noisy foreground pixels that do not correspond to the real foreground objects and noisy background pixels near and inside object region that should be foreground pixels will be removed in this noise removal operations.
3.2.1 Median Filtering

Median filtering is a nonlinear filter that can efficiently reduce salt-and-pepper and impulse noise in an image without blurring the features sharp edges. A standard median filtering operation is implemented by sliding a window over the input image. At each window position, the pixels present in the neighbourhood within the window template and the central pixel are ranked according to their intensity and subsequently, the pixel value (which is the central pixel) will be replaced with the median value. Specifically, it can be expressed as:

\[ y[i, j] = \text{median}\{x[n, m], (n, m) \in w\} \]  

where \( y[i, j] \) is the output pixel and \( w \) is neighbourhood pixels in the \( n \)-by-\( m \) dimension window centred at \( (i, j) \) point. The window can be any symmetric shape such as square, rectangle or round disc. The moving window template used in this work is a symmetrical 3 X 3 squared neighbourhood.

3.2.2 Morphological Operations

Morphology is a word of Greek origin which literally means analysis of shapes. Mathematic morphologic however, is a nonlinear mathematic tools initiated by Matheron and Serra that is used to analyse and process the geometrical structures based on set and lattice theory. It was developed for binary images for pre/post processing such as noise filtering, region filling, thinning, pruning, and segmentation of the shape of objects (boundaries, skeletons, convex hulls and connected components). In this thesis, morphological image processing is employed for removing the imperfections of the binary images after the foreground segmentation and median filtering processes. The morphological filter is adopted due to its preservation and enhancement of geometric structure of image objects characteristic. This property is important for object classification which classifies object based on their shape and area detected in an image. Further, morphological filtering offers an efficient solution for non-Gaussian noise suppression.
The morphological opening operator is adopted in this work which is derived of the fundamental operations of erosion and dilation. The morphological opening of an image $X$ by the structuring element $B_E$ can be defined as:

$$X \circ B_E = (X \ominus B_E) \oplus B_E$$  \hspace{1cm} (3.9)

Hence, an opening of image $X$ is an operation of erosion followed by dilation with the same structuring element $B_E$. An erosion or Minkowski subtraction, by using Haralick/Sternberg notation is defined as:

$$X \ominus B_E = \bigcap_{b \in B} X_{-b}$$  \hspace{1cm} (3.10)

which is the intersection of $X$ when translating $X$ by the vector $-b \in B_E$. Erosion can effectively remove the small clumps of undesirable foreground pixels or 1-valued regions in the binary image, for example the salt noises. However, it shrinks the size of the foreground object and widens the area of the holes at the same time due to the nature of the operation. A dilation or Minkowski addition however is defined as:

$$X \oplus B = \bigcup_{b \in B} X_b$$  \hspace{1cm} (3.11)

which normally used to expands the size of foreground object or 1-valued regions in the binary image while holes within those regions become smaller. It is also used to fill in the small spurious holes (pepper noise) in an image. Therefore, by cascading these two operations, an opening operation does an excellent job of elimination the extraneous 1-values or foreground pixels while preserving the size of the object. The number of pixels removed from an image depends on the size and shape of the structuring element used to process in the image.

Figure 3.6 illustrates an example of noise removal sequence and result. The binary foreground image computed by the object detection algorithm as discussed in section 3.1 (top right of Figure 3.6) has shown imperfection due to the noises in the background and holes within the object region. After applying the median filter (bottom
right of Figure 3.6), noises in the background were successfully removed and the holes in object region were recovered significantly. However, the outline of the binary image was still having distortions at the top edge of the object and holes are still presented at some spots. The filtered binary image was then processed by using the morphological opening operation discussed in Section 3.2.2. As can be seen from the final output binary image (bottom left of Figure 3.6), the remaining holes in the object region were filled and the edges have shown to be smoother and closer to the outline of the real object.

**Figure 3.6** – Results of the noise removal processes.
3.3 BLOB ANALYSIS

In image processing, a blob (Binary, Large Object) is defined as a region of connected pixels (an area of touching pixels with same logical state); also it is referred as object in the following discussion. Therefore, the function of blob analysis is to detect and analysis the connected regions (blobs) in the binary image. The blob analysis operation includes the modules as depicts in Figure 3.7, the blob detection process which identifies valid object from the foreground image, and then followed by blob analysis process which analyses the detected blobs and extracts useful information for each of the detected object.

Figure 3.7 – Blob analysis Processes. Object features are extracted by applying the blob detection and connected component analysis.

3.3.1 Connected Component Analysis

After the noise removal processing, the filtered foreground pixels are segmented into different group of regions by using connected component analysis. In this algorithm, set of pixels that are within a boundary are identified as connected and these connected foreground pixels are then assigned with a unique label according to their belonging grouping. An 8-connectivity, two-pass connected component analysis is adopted for the
region labelling in this thesis. Consider a seed point with 1-valued and coordinate \((x, y)\), any foreground pixels in the neighbouring set of coordinates

\[ \{(x-1, y-1), (x-1, y), (x-1, y+1), (x, y-1), (x, y+1), (x+1, y-1), (x+1, y), (x+1, y+1)\} \]

will belong to the same component. The seed point and the set of foreground pixels in its neighbourhood will be assigned a same label. It then moves on with the next unlabelled pixel as the seed point. For a two pass algorithm, the image is scanned pixel by pixel for two times. The first scan attempts to give temporary labels for the foreground pixels. Subsequently, the equivalent labels are sorted into equivalence classes with a unique label assigned for each class during the second scan. The blobs detected are subsequently represented by bounding boxes with the minimum and maximum coordinates of rows and columns of the identified regions. Figure 3.8 shows the result of the connected component analysis with the bounding box containing the region of foreground pixels which each side touching the foreground object.

Figure 3.8 – Results of connected component analysis. (a) The bounding box (green box) computed by the connected component analysis for one object entering the scene and (b) its corresponding foreground image. (c) The bounding boxes (green boxes) computed by the connected component analysis for multiple objects entering the scene and (d) its corresponding foreground image. The number of object detected is shown in the top left corner of the original images.

3.3.2 Objects Feature Extraction

From the previous section, objects are presented as group of 1-valued pixels or blob. However, for the purpose of object recognition and activity analysis, a set of descriptions about the characteristics and properties for a given object is needed. In this section, region or shape descriptors which characterize the geometric property of the object are concerned. Consider there are \(n\) numbers of object \(O\) detected in an image,
and for $i^{th}$ object $O_i$, the descriptors of the object are categorised into 3 main features which contains the size, shape and location information respectively.

- **Size features:**
  
  o The **object area** ($A_i$), which can be computed by counting the number of connected foreground pixels contained in the bounding box for object $O_i$.
  
  o The **bounding box area** ($S_i$) for object $O_i$ is obtained by

$$S_i = w_i \times h_i$$  \hspace{1cm} (3.12)

where $w_i$ and $h_i$ are the width and height of the bounding box and can be calculated by

$$w_i = x_{max} - x_{min}$$  \hspace{1cm} (3.13)

$$h_i = y_{max} - y_{min}$$  \hspace{1cm} (3.14)

where $x_{max}$, $x_{min}$, $y_{max}$ and $y_{min}$ are the maximum and minimum spatial coordinates of the bounding box.

- **Shape features:**

  o The **bounding box** ($B_i$) is a smallest isothetic rectangle that containing the object ($O_i \in B_i$).
  
  o **Aspect ratio of the width over height of the bounding box** ($R_i$) for object $O_i$ can be obtained by

$$R_i = \frac{w_i}{h_i}$$  \hspace{1cm} (3.15)

  o **Filling ratio of the bounding box** ($f_i$) is one of the oft-expressed of shape analysis parameter. The filling ratio of the bounding box for object $O_i$ is defined as the ratio of the object area to the bounding box area as

$$f_i = \frac{A_i}{S_i}$$  \hspace{1cm} (3.16)
• Location features:

  o **Centroid** ($\bar{c}_i$) of the object $O_i$ can be simply estimated by the centre of the bounding box as

  $$
  \bar{c}_i = \left( \frac{x_{\text{min}} + \frac{w_i}{2}}, \frac{y_{\text{min}} + \frac{h_i}{2}} \right)
  $$ (3.17)

![Image showing object features](image)

Figure 3.9 – Object features extracted from the object detection and connected component analysis.

### 3.4 OBJECT TRACKING

The aim of object tracking is to derive the trajectory over time of the moving object. The output of the object tracking generally will be, for instance, the centroid coordinates of the moving object, the 3D position of the tracked object, the contour or silhouette of the moving object. It normally depends on the representation used to describe the tracked object for different application domains.

The blob tracking algorithm which is one of the real time region-based tracking approaches is implemented. Some region pre-processing is done before the implementation of blob tracking. This is due to the imperfection of object segmentation that cannot be filtered by the median filtering and morphological operation. The blobs
that are having an area which is lower than a threshold value will be identified as noises and ignored. The centroids of the blobs detected in consecutive frames are collected until the event ends (object exits the scene, incorporates into the background or occlusion). Consequently, the high level features that are extracted from the object tracking process are constructed for activity understanding.

- **Motion features:**

  - The **trajectory** for a moving object $O_i$ is estimated by centroid ($\bar{c}$) correspondence to consecutive frame from time sequence $t = \{t_s, \ldots, t_f\}$ where $t_s$ is the time when the object $O_i$ entering the scene and $t_f$ is the time when the object leaves the scene (lost tracking). Consequently, trajectory is a set of tuple $\{(x_n, y_n): n = 1 \ldots N\}$ where $(x_n, y_n)$ is the centroid of $O_i$ at the $n^{th}$ image frame in a video sequence and $N$ is the total number of frames from $t_s$ to $t_f$.

  - Object **velocity** is one of the measures for dynamic changes of the moving object. It describes how fast of the object moves projected in both horizontal and vertical axes. The velocity of $O_i$ is a 1-by-$N$ vector of $\{v_1, \ldots, v_t\}$ where $t = t_s; t_f$. The value of $v$ can be estimated by $\frac{\Delta d}{\Delta t}$ where $\Delta d$ is the Euclidean distance between the two consecutive centroids in a two-dimensional plane and is defined by

$$\Delta d = \sqrt{(x_t - x_{t-1})^2 + (y_t - y_{t-1})^2} \quad (3.18)$$

and $\Delta t$ is the time taken between frames. In a typical CCTV camera, $\Delta t$ is a constant which can be obtained by the frame rate of the visual input device.
3.5 OBJECT CLASSIFICATION

After the object segmentation and features extraction, identification of the moving objects is necessary for the event analysis. The ultimate aim of the object classification is to recognize the moving foreground object from some predefined classes of objects. Typical video scene may contain diversity of objects such as people, animals, vehicles, shadows, plants and etc. However, the main focuses of interest in parking lot surveillance applications are generally to identify humans and vehicles.

The proposed event recognition is based on the information gathered from the object feature extraction in section 3.3.2. The feature-based object classification approaches can be broadly categorized into two groups: shape-based classification and motion-based classification [69, 70]. For motion-based object classification, the motion features such as the direction of motion, velocity, recurrent motion image [71, 72], optical flow vector and etc are exploited to classify the objects into humans or vehicles based on their nature of movement. This approaches is based on the cue that human motion generally exhibit non-rigid and repetitive motion [72].

On the other hand, Shape-based object classification is a widely used object classification method due to its simplicity and efficiency. Shape-based classification is implemented for the object classification in parking lot surveillance system which is the main focus of this work. Object classification is achieved by exploiting the shape features of the moving object such as dispersedness or compactness, blob area, aspect ratio of the blob and the filling ratio of the blob area to the bounding box.
Three parameters (blob area $A_i$, the filling ratio of the bounding box $f_i$ and the aspect ratio of the width over height $R_i$) are used to classify the detected moving object. The blob area $A_i$ serves as the critical parameter in the proposed object recognition technique. This method is based on the knowledge that vehicles are, in generally, bigger than humans. Moreover, the filling ratio $f_i$ of vehicle is typically larger than humans due to the fact that the shape of vehicles is generally rectangle and hence having higher filling ratio. Lastly, the aspect ratio $R_i$ of human objects is normally smaller than the vehicles. These three parameters of the foreground objects are extracted from the 144 training video samples during the training phase and the threshold values used to differentiate between each class of objects are determined from the collected training data.

3.6 CONCLUSION

An object detection algorithm is presented for the outdoor car park automated surveillance system concerned in this work. The detection of object(s) presents in the monitored area was detected successfully by using the proposed framework. An improved foreground image was achieved by applying noise removal strategies to enhance the reliability and accuracy of the event understanding which will be relied strongly on the results of object detection. The object features such as shape, size and location were extracted from the processed foreground image in the quest to classify the type of object that initiating the event. The motion trajectory of the moving object was captured by applying the object tracking algorithm which will serve as a critical representation of event for the learning and understanding of video behavior.
CHAPTER 4

SUPERVISED TRAJECTORY LEARNING APPROACH

4.1 SYSTEM OVERVIEW

This chapter presents an event recognition framework for video surveillance system, particularly in the outdoor car park environment. A supervised event learning strategy is utilized in this framework for event understanding and interpretation. The spatial, dynamic and object appearance features of the moving object are considered in the representation of event. The motion trajectory collected from the object detection and tracking process (refer to Chapter 3) will be used as the primary source in analyzing an event. The appearance-based information of the moving object such as object’s size, aspect ratio and filling ratio over time will be extracted from the blob analysis during the object tracking phase.

The system architecture of the proposed video surveillance system is illustrated in Figure 4.1. The system can be generally divided into four phases: object tracking, event representation, explicit data training and event recognition phases. The object tracking phases detects the moving object from the input video stream via background modeling and foreground segmentation processes as discussed in Chapter 3. In event representation phase, the spatial-temporal and appearance-based information collected from the object tracking phase are analysed and the feature vector representation of an event is constructed. The explicit data training phase collects the feature vectors of the known events into classes of similar motion pattern. A compact representation of the known events is extracted from the collected measurement data by employing principal component analysis (PCA) to serve as a map for event recognition and anomaly detection purposes.
Figure 4.1 – System Framework for event recognition system by supervised learning approach. The system can be subdivided into four main phases; namely object detection and tracking phase, event representation phase, training phase, and event recognition phase.

Lastly, the event recognition phase identifies the possible type of event by computing the similarity of the detected event with the explicit definition of events obtained from the training phase. The scene information of the tracked object path is also taken into consideration to improve the accuracy of the system. The sub-sections below present the detailed analysis and implementation of the proposed video surveillance system.

4.2 EVENT REPRESENTATION
4.2.1 TRAJECTORY ANALYSIS

The outcome (motion trajectories and object appearances) of the low-level object detection and tracking processes provides primary behavioral information of an activity. The motion trajectory defines how an object moves within a finite temporal interval
which can be decomposed into three features of different domain as shown in Figure 4.2(a). The point of interest (POI) of a trajectory such as start point $X_s$ and end point $X_f$ describes the compact location information of an activity. This simple location description of a motion trajectory is essential especially in the car park environment as it provides the origin and destination information of the moving vehicle or walking people.

The collections of point (object’s centroid) in a consecutive video frames forms the trajectory of the motion. This collection of location information describes the path taken by the moving object from the origin to the destination location. This route information is important in representing event in the application domain where the object’s movement is limited by certain constraint such as traffic management at the road junction or outdoor car park facility. It serves as an important measurement to detect anomaly behavior where vehicle violates the traffic law and regulation such as illegal U-turn or car moving in the opposite direction. The path information is also useful in identifying strange pedestrian movement such as the one shown in Figure 1.5b where a human loitering among the stationary cars in a car park area.

Apart from the location and path information that can be extracted from the motion trajectory, the dynamic behavior of the moving object is also indirectly implied in the object track. In most cases, the video sequence is captured with a fixed frame rate; therefore the absolute velocity of the moving object can be simply obtained by computing the relative distance between the two consecutive trajectory points as,

$$v_n = \sqrt{(x_n - x_{n-1})^2 + (y_n - y_{n-1})^2} \times FPS$$

where $v_n$ is the instantaneous velocity of the object at $n^{th}$ frame, $x_n$ and $y_n$ are the object’s centroid location projected on the horizontal and vertical axes respectively at $n^{th}$ frame, and $FPS$ is the frame per second.

Unlike the location and path information, the velocity profile of the tracked object contents no spatial knowledge about the object; however it describes in what manner the object moves within the monitored area. For example, object moves with constant velocity, stops, accelerates or decelerates towards the destination. This dynamic
information is useful in the application where velocity profile of the target object is dominant such as detecting car speeding on a highway. Additionally, the dynamic information of a trajectory is critical in the situation where the spatial feature of the trajectory could not specifically describe the behavior of an event. For instance, it is often difficult to extract a common pattern for a normal event of ‘pedestrian passing through an open area’ by only considering the location and path information, due to the randomness of the path taken by each and every pedestrian. Therefore, in order to interpret an event precisely, selecting the appropriate features of the motion trajectory in representing an event is essential.

**Figure 4.2** – (a) Feature decomposition from motion trajectory and object appearance data collected from the object detection and tracking process. (b) Graphical representations of each decomposed feature.
Figure 4.3 illustrates clearly the three features that can be extracted from the motion trajectory as a result of the object detection and tracking process. The red circles as illustrated in Figure 4.3(a) represents the POI of the track where $X_s$ is the start point and $X_f$ is the end point of the track. The series of black dots represents the frame by frame centroid location of the object moving from the origin towards the destination. On the other hands, the velocity profile of the trajectory is depicted in Figure 4.3(b). It can be observed that the object produced a constant velocity at about 2 pixels per frame from frame 25 until frame 80.

The deviation in velocity at the beginning and ending of the event was because of the edge effect where the tracked target is only partially the real object. This causes an inaccurate calculation of the object’s centroids. As a consequence, it causes the faulty computation of actual velocity of the moving target. However, the imprecision in the measurement of actual object’s velocity will not have significant effect on the proposed event learning recognition system since the system is to extract and learn common pattern of the track rather than measuring the real physical quantity of the event.

*Figure 4.3 – Graphical representations of the different features extracted from the motion trajectory. (a) The POI of the trajectory shown in red circles (start point and end point) and the route taken by the moving object as shown in a series of black dots. (b) The instantaneous velocity of the moving object from the origin to the destination.*
4.2.2 OBJECT APPEARANCE ANALYSIS

Apart from the spatial-temporal information brought by the motion trajectory, the appearance of the object as a result of the foreground segmentation process is also playing an important role in the context of vision-based event recognition. Unlike the motion trajectory, the object appearance information does not content any spatial quantity of the activity; conversely, it describes the dimensionless course body action of the tracked object in the scene. Furthermore, it provides a primary source of input that enables the recognition of object type that initiates the activity. For example, in an environment where multiple types of object are expected such as the surveillance camera that monitors an outdoor scene, the event recognition process will rely strongly on the classification of the object type in order to precisely interpreting the type of activity.

In this work, a simple object recognition method is used to enhance the accuracy of the proposed event recognition system. During the training phase, the object appearance parameters (blob area, $A_i$, aspect ratio, $R_i$ and filling ratio, $f_i$) were extracted from the object detection algorithm. The distribution patterns of these parameters were analyzed in order to set reasonable threshold values in classifying the type of object. The distributions of the three appearance parameters (blob area $A_i$, aspect ratio $R_i$ and filling ratio $f_i$) for human and vehicle objects are illustrated in Figure 4.4. As shown clearly in Figure 4.4(a), the size of human objects is far smaller than the vehicles. Therefore, threshold values can be easily set. However, using the size parameter of detected blob alone to classify the type of object is not reliable, for example to differentiate a vehicle and a group of people moving in the scene. Thus, examining the shape of the detected object is essential in object recognition.
Figure 4.4 – The distributions of the three appearance parameters for human and vehicle objects. (a) Blob area, $A_i$, (b) aspect ratio, $R_i$ and (c) filling ratio, $f_i$.

The shape features of the object such as the aspect ratio $R_i$ and filling ratio $f_i$ were taken into account also for the proposed object recognition strategy. Similarly, these shape parameters were extracted during the training phase and their distribution were plotted in Figure 4.4(b) and 4.4(c) respectively. The aspect ratio as discussed in Chapter 3.3.2 is a measurement of the shape of the bounding box. A bounding box having a smaller value of width with a larger value of height will result a small value of $R_i$. Thus, a human object will typically have a smaller $R_i$ compared to the vehicle. This can be observed in Figure 4.4(b) where the $R_i$ of the human objects are distributed below 1.2 and above 1.2 for vehicle objects. This is mainly due to the shape and
orientation of the respective objects (human and vehicle) while moving in this particular scene. Hence, a threshold value at 1.2 is set to determine the type of object and it will be used to strengthen the decision made by using the size feature.

In Figure 4.4(c), the distribution of the filling ratio, \( f_i \) for human and vehicle objects are plotted. As expected, the filling ratio of human object will be slightly lower than the vehicle objects because of the complexity in shape. However, the difference in filling ratio of the human and vehicle objects are not as clear as the blob area and aspect ratio. Therefore, this parameter will not be used as an indicator in classifying the type of object; instead, it will be used to filter off noises when the filling ratio falls below the threshold value set. As shown in Figure 4.4(c), valid objects detected from the training videos have a filling ratio of more than 0.4. Thus, any detected object that is having filling ratio of less than this threshold will be treated as noise such as sudden change in illumination, shadow or flickering of vegetation.

The complete flow of the object recognition algorithm is shown in Figure 4.5. Firstly the algorithm examines the filling ratio \( f_i \) of the detected object and rejects any blob that has a \( f_i \) that falls below the 0.4 threshold. Secondly, the algorithm checks the blob size \( A_i \) of the object and categorizes object based on the threshold value (300). If the blob area is larger than the threshold, the system will proceed to the checking of its aspect ratio \( R_i \) to confirm the decision that the object is a vehicle if \( R_i > 1.2 \). For the case of human object, the decision will only be made when \( A_i < 300 \) and \( R_i < 1.2 \). Once the type of object has been confirmed, the system will proceed to the event recognition process which will be discussed in detail in section 4.4.
4.2.3 FEATURE VECTORS CONSTRUCTION

An event $e$ is defined over a temporal interval $[t_s, t_f]$ where $t_s$ is the instance when the moving object is detected and $t_f$ is the time when the object exits the monitored area or incorporates into background, or as a result of occlusion. The centroids of the moving object within a finite temporal interval forms the trajectory of movement and it provides elementary spatial and dynamic information for event analysis. The motion trajectory $T$ can be represented by a flow vector of

$$T = \{(x_t, y_t), t = 1 \ldots N\} \quad (4.4)$$

which corresponds to the $x$ and $y$ axis projections of the object’s centroid location at every instance of time $t$.

Nevertheless, the spatial information of the object movement is insufficient to represent an event and therefore, the POI information and dynamic behavior of the moving object as well as the interaction of objects which can be identified from the change in the object appearance are also taken into account when constructing the
feature vector representation of the event $e$. Hence, the feature vector $f_t$ can be represented by:

$$f_t = \{T, v_t, A_o(t), t = 1 \ldots N\}$$  \hspace{1cm} (4.5)$$

Where $v_t$ and $A_o(t)$ indicate the velocity and area profiles of the moving object within the finite temporal interval $[t_s, t_f]$. Conclusively, an event vector $e$ for object observed over a sequence of $N$ consecutive frames can be written as:

$$e(t_s, t_f) = \{f_t | f_t \in O_o, t \in [1 \ldots N]\}$$  \hspace{1cm} (1.6)$$

As a consequence of time-varying tracks of different event, trajectories are often unequal in term of dimension. Thus, several preprocessing steps such as normalization and scaling operations are required to obtain a standardized representation of the event $e$ which will be discussed later in section 4.3.1.

**4.2.4 SEMANTIC DESCRIPTION OF POI**

In a car park environment, activities are normally initiated by a vehicle or a pedestrian moving in the monitored area. Simple events that are commonly detected in the parking lot environment can be summarized as follows:

1. *A vehicle passes by the monitored parking area*
2. *A vehicle enters the monitored area and parks at an empty parking space*
3. *A vehicle gets out of parking space and exits the scene*
4. *A person walks through the monitored parking area*
5. *A person enters the scene and picks up a vehicle*
6. *A person drops off from vehicle and exits the scene*
The nature of these events suggests that the scene information (region of interest) can be estimated by the POI of the motion trajectory generated. For example, a vehicle passes by the monitored area will produce a trajectory with a starting point $X_s$ at the entrance zone and at the same time, an ending point $X_f$ at the exit region of the specific parking lot. Therefore, partitioning the entire scene into different regions of interest is one of the strategies to semantically interpret the POI of a motion trajectory. As in the environment where more than one type of object exists in the scene, different scene partitioning strategies is not uncommon.

There are two types of objects exist in the car park environment considered in this work, and thus, two sets of semantic scene models are constructed as illustrated in Figure 4.6. Figure 4.6(a) illustrates the region of interest (ROI) for vehicle events and Figure 4.6(b) shows the ROI for events that are initiated by human. These regions of interest are set manually based on the human understanding of the environment. The designated zones for vehicle and human activities differ in a way that the entry and exit zones for vehicle (zone V1 and V2) are fixed at the entrance and exit location of the car park facility, whereas a person can walk in to the scene at any point along the edge of the monitored area as labelled by P1. The parking bays zone (V3 and P2) remains the same for both vehicle and human activities. Therefore, by investigating the location information of POI with the assistance of the object recognition algorithm, the simple events as mentioned above can be estimated by analysing the combination of the POI and object type of the detected motion trajectory.

Based on the nature of different activities, a simple semantic interpretation of the tracked motion trajectory can be estimated from its POI and object type. For example, the event ‘A vehicle enters the monitored area and parks at an empty parking space’ will have a high possibility to produce a starting location $X_s$ within region V1 and at the same time, a starting location $X_f$ within the region V3. Conversely, an event ‘A person drops off from vehicle and exits the scene’ will probably generate a motion trajectory with $X_s$ in region P2 and $X_f$ in P1. Therefore, by knowing these logical combinations of POI and object type, a logical table of these known events can be constructed as shown in Table 4.1.
Figure 4.6 – Scene decomposition based on the region of interest. Different scene decompositions for (a) vehicle events and (b) human events.

<table>
<thead>
<tr>
<th>Event</th>
<th>Starting pointzone</th>
<th>Ending pointzone</th>
<th>Type of object</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>V1</td>
<td>V2</td>
<td>Vehicle</td>
</tr>
<tr>
<td>2</td>
<td>V1</td>
<td>V3</td>
<td>Vehicle</td>
</tr>
<tr>
<td>3</td>
<td>V3</td>
<td>V2</td>
<td>Vehicle</td>
</tr>
<tr>
<td>4</td>
<td>P1</td>
<td>P1</td>
<td>People</td>
</tr>
<tr>
<td>5</td>
<td>P1</td>
<td>P2</td>
<td>People</td>
</tr>
<tr>
<td>6</td>
<td>P2</td>
<td>P1</td>
<td>People</td>
</tr>
</tbody>
</table>

Table 4.1- Logical POI and object type combinations for different known events.

By combining the trajectory-based (location, path and velocity) and object-based information from the object tracking detection and tracking process, an event can be comprehensively represented by these information. Thus, an event tag $T_e$ can be compactly formulated as a cell of features in different aspect as:

$$T_e = \{t_s, t_f, X_s, X_f, O, e\}$$

Where $t_s$ and $t_f$ are the starting time and ending time of the activity, $X_s$ and $X_f$ are the starting and ending centroid locations, $O$ is the outcome of the object recognition algorithm, and $e$ is the event vector as formulated in Equation 4.6. This event tag $T_e$ will be used as a parameter used in the event recognition process to determine the possible event detected from the camera.
4.3 EVENT LEARNING

4.3.1 FEATURE VECTORS NORMALIZATION

Due to the fact that each and every event occurs with different time interval, the motion trajectory tracked will also have different dimension. In order to equalize the dimension of the feature vector for all events, a normalization process is necessary. The feature vector normalization approach used in this work is to convert the arbitrary number of sample tracked in a particular event to a user-specified number of samples, while maintaining the information as much as possible. A common resampling technique together with an anti-aliasing FIR filter is used. An example of the feature vector normalization result is illustrated in Figure 4.7. Figure 4.7(a) and 4.7(b) show the outcomes of the resampling process for the \( x \) and \( y \) projections of the motion trajectory. Whereas Figure 4.6(c) and 4.6(d) show the resampled velocity and blob size profiles of the feature vector.

Notice that the last few points of the resampled \( x \) axis projection and points at both ends of the \( y \) axis projections are inaccurate. This negative side effect is because the resampling technique used in this work assumes the samples before the first point and after the last point of the given sequence are equal to zero. Thus, large deviations from zero at the start and end points of the given sequence can cause inaccuracies in its outcomes. However, since the focus of this work is to look for common pattern of events rather than measuring the accuracy of the data, the inaccuracies at both ends of the vector will not deteriorate much on the accuracy of the overall event learning and recognition process.

Based on the actual data collected from the abovementioned scene, the targeted number of sample of the feature vector was set to 500 points for all type of events. The resampled feature vectors of four component vectors with 500 in length will be used throughout the event learning and recognition processes.
Figure 4.7 – Feature vector normalization. Original and resampled components of the feature vector; (a) $x$ axis projection, (b) $y$ axis projection, (c) velocity profile and (d) blob area profile.

After normalizing the feature vector of the detected events, similar pattern in each of the components in the feature vector can be observed for events that have similar in nature. Figure 4.8(a) to 4.8(d) shows each component in feature vector for a group of events ‘A vehicle enters the monitored area and parks at an empty parking space’. Note that in Figure 4.8(a), the $x$ axis projection vectors were clustered into different groups due to different location of each individual packing space. In the quest to overcome this inconsistence in magnitude, a vector standardization algorithm is applied to the normalized feature vector.
Figure 4.8 - Feature vector standardization for event ‘A vehicle enters the monitored area and parks at an empty parking space’. Original components of feature vectors before standardization; (a) $x$ axis projection, (b) $y$ axis projection, (c) velocity profile and (d) blob area profile. Standardized components of feature vectors; (e) $x$ axis projection, (f) $y$ axis projection, (g) velocity profile and (h) blob area profile.

The standardization process basically performs centering and scaling operations to the feature vector that has uninform distribution. The centering process subtracts the value at every instance to the mean score of the specific component. The end result of this centering process is to offset the mean of the distribution of points to zero as depicted in Figure 4.9(a). After centering the distribution, a scaling process is applied to standardize the ununiformed distribution to a fixed scale. The scaling process divides the points in the distribution with its standard deviation as shown in Figure 4.9(b). Figure 4.8(e) – (f) shows the components of feature vector after the standardization process. Note that the uniformed distribution of the $x$ axis projection (Figure 4.8(a)) due to different location of parking spaces is solved by applying the standardization process as shown in Figure 4.8(e).
4.3.2 PRINCIPAL REPRESENTATION OF EVENTS

Explicit definitions of the ‘normal’ events at parking lot environment are trained with a collection of labelled video samples. The event feature vectors $e$ of these ‘normal’ events are extracted during training phase. To obtain a compact and low-dimensional representation of these events, principal component analysis (PCA) is performed on the labelled training data set. The principal components of each known event are computed from the normalised and standardized feature vectors in order to construct compact models that best describe the pattern of each know events. These principal components

---

**Figure 4.9** – Graphical representation of the (a) centering and (b) scaling processes for standardizing feature vectors.
will be used as the representative models for identifying events in the event recognition phase.

Figure 4.10 and Figure 4.11 illustrate the trained feature vectors for the known vehicle and human events that are collected from the sample videos respectively. The thick lines in the figures show the principal component representation of the feature vector for the $x$ and $y$ axis projections, velocity and area profiles within the temporal interval of the event. Note that the principal components have effectively revealed the common pattern for each individual component from the multi-variated data.

**Figure 4.10** – Principal component analysis of feature vectors of vehicle events. Top row: “A vehicle passes by the monitored parking area”. Middle row: “A vehicle enters the monitored area and parks at an empty parking space” and bottom row: “A vehicle gets out of parking space and exits the scene”. From left to right, the $x$ axis projection, $y$ axis projection, velocity and area profiles components of the feature vectors. The graphs show a collection of feature vectors for different events (shown in multiple colored lines) and the respective principal component representations of each component (shown in the thick black lines).
Figure 4.11 - Principal component analysis of feature vectors of human events. Top row: “A person walks through the monitored parking area”. Middle row: “A person enters the scene and picks up a vehicle” and bottom row: “A person drops off from vehicle and exits the scene”. From left to right, the x axis projection, y axis projection, velocity and area profiles components of the feature vectors. The graphs show a collection of feature vectors for different events (shown in multiple colored lines) and the respective principal component representations of each component (shown in the thick black lines).

### 4.4 EVENT RECOGNITION ALGORITHM

Event recognition of the proposed parking lot surveillance system is accomplished by evaluating the event tag $T_e$ extracted from the object tracking process as shown in Equation 4.7. This event recognition approach incorporates the feature matching and contextual information analysis to ensure accurate classification of the detected event. The explicit definitions of the known events computed at the training phase are used as a map to identify the incoming event detected from the live video stream. Figure 4.12 illustrates the decision tree of the proposed event recognition process.
Firstly, the type of object (vehicle or human) that initiates the event is identified through the object recognition algorithm as discussed in section 4.2.2. Once the object is determined, different set of POI mapping is carried out to perform a low-level interpretation of the event. Different scene model decomposition (refer to Figure 4.6) is used for different object due to the nature of the event. For example, if the entry-point $X_s$ of a vehicle event falls in the entry zone $V_1$ and the exit-point $X_f$ detected in exit zone $V_2$, the event will be tagged with ‘A vehicle passes by the monitored parking area’ event. On the other hand, if a vehicle initiates the event by entering the scene in the exit zone $V_2$, the event will be tagged as anomaly since it does not recorded in the POI map for normal events. This is true in human understanding as the vehicle violates the regulation of the traffic flow in the parking facility.

The final decision is made by evaluating the similarity of the feature vector $f_i$ between the input event and the explicit definition of the same event computed from the training phase. If the correlation coefficient of the two feature vectors falls within an acceptable range, the event is classified as the one tagged by the event tag $T_e$. Else, the detected event will be classified as abnormal event.
4.4.1 HIGH LEVEL ACTIVITY CLASSIFICATION

In addition, a higher level composite activity can be identified by analyzing the contextual information of the consecutive lower level of simple events. A specific combination of events such as ‘A person enters the scene and picks up a vehicle’ followed by ‘A vehicle gets out of parking lot and exits the scene’ can be classified as high-level composite activity of ‘pick up’, provided that the contextual information of the successive event fulfills certain condition. For the above example, the entry-point $X_s$ of the second event must appear within an acceptable region of the exit-point $X_f$ of the previous event. Also, the timing of the consecutive events needs to be analyzed in order to improve the accuracy of the classification algorithm. The two events will be recognized as a result of high level activity if the existence of the two events occurs within an acceptable temporal interval. As will be shown in the test result, a high level activity of ‘pick up’ was identified as a result of two low level events.

Based on the 6 defined simple events as listed in section 4.2.4, the potential composite events are ‘pick up’ and ‘drive in’. In the case of ‘pick up’, the order of simple events must be ‘A person enters the scene and picks up a vehicle’ followed by ‘A vehicle gets out of parking lot and exits the scene’. Other conditions that must be fulfilled are:

- The exit location $X_f$ of the human object in the first event must be within certain range relative to the initial location $X_s$ of the car object of the consecutive event (see Table 4.2).

- The human object must be on the right side of the car object (see Table 4.2).
- The starting time $T_s$ of the second event must occur within certain time interval after the first event ended at $T_f$. 
<table>
<thead>
<tr>
<th>Condition</th>
<th>Pick up</th>
<th>Drive in</th>
</tr>
</thead>
<tbody>
<tr>
<td>First event</td>
<td><em>A person</em> enters the scene and picks up a vehicle</td>
<td><em>A vehicle</em> enters the monitored area and parks at an empty parking space</td>
</tr>
<tr>
<td>Second event</td>
<td><em>A vehicle</em> gets out of parking space and exits the scene</td>
<td><em>A person</em> drops off from vehicle and exits the scene</td>
</tr>
<tr>
<td>Distance between Exit Location of the first event and Starting Location of the second event</td>
<td><img src="image" alt="Distance Diagram" /></td>
<td><em>d &lt; d_i</em></td>
</tr>
<tr>
<td></td>
<td><em>d</em> is a preset threshold value based on the specific scene</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Location of human object relative to the car object</td>
<td>The human object must be on the right side of the car object</td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Location Diagram" /></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Timing of the two consecutive events</td>
<td><em>T_i &lt; T_t</em></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Timing Diagram" /></td>
<td><em>T_t</em> is a preset threshold value based on the nature of the event</td>
</tr>
<tr>
<td></td>
<td>Time interval, <em>T_i</em></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2 - Detailed conditions of the ‘pick up’ and ‘drive in’ composite events.

Table 4.2 shows the details of conditions for determining the ‘pick up’ and ‘drive in’ composite events. The composite events (‘pick up’ or ‘drive in’) recognition process relies upon the decision tree as shown in Figure 4.13. As shown in the figure, the composite event recognition process will only be initiated when the system detected either event #3 or #6 (refer to Section 4.2.4). If event #3 is detected by the system, the Euclidean distance of the car and human objects during the transition will be computed and compared to the preset threshold. If the distance is within the threshold value, the system will then proceed to check the relative position of the two interactive objects. The composite event recognition will only proceed if the human object is on the right side of the car object. If the abovementioned conditions are fulfilled, the time interval of the two consecutive events will be examined. The decision will only be made if the timing of the two events falls within an acceptable time interval.
4.5 EXPERIMENTS

4.5.1 DATASETS AND SETTINGS

For evaluating the validity of the proposed supervised event recognition approach, test video sequences were captured by using a static camera that is monitoring the outdoor parking lot area at campus of Swinburne University of Technology (Sarawak campus). The test videos were captured at a frame rate of 15 frames per second. Each video frame has a size of 160 (width) X 120 (height) pixels. The snapshot and topological setting of the camera of the scene considered in this work are depicted in Figure 4.14. The camera is monitoring a partial of the entire outdoor parking facility. The camera scene consists of a one-way parking aisle across the middle of the scene, a row of parking spaces on top of the scene and an access to the building at the bottom of the scene.
Figure 4.14 – (a) Snapshots of the scene. (b) Topology of the monitored area and the line of sight of the camera.

A collection of test video sequences were captured from the abovementioned camera during normal working hour (8:30am – 5:30pm) for 5 working days continuously. The raw video streams collected from the camera were segmented into shorter video clips that contain only single event. There are in total 245 test video clips collected from the camera which comprise of the six simple events listed in section 4.2.4 and all of the video clips in training dataset were interpreted manually and labeled accordingly. Out of the 245 test video clips, 144 of the clips were used as the training dataset, and 101 of the video clips were used as the testing dataset. Table 4.3 shows the distribution of events across the training and testing datasets.

<table>
<thead>
<tr>
<th>Event</th>
<th>Training dataset</th>
<th>Testing dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>65</td>
<td>47</td>
</tr>
<tr>
<td>2</td>
<td>18</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>23</td>
</tr>
<tr>
<td>5</td>
<td>12</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>Anomaly</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Total</td>
<td>144</td>
<td>101</td>
</tr>
</tbody>
</table>

Table 4.3 - Distribution of events across the training and testing datasets.
4.5.2 EXPERIMENTAL RESULTS

The proposed object tracking and event recognition algorithm was implemented in MATLAB for the entire work. A collection of labelled training sample videos were used to extract the feature vector representation of each known events (refer to Chapter 4.2.4). The principal component representations of each vector elements computed using PCA (as discussed in Chapter 4.3.2) were used as the reference model in the event recognition process. The proposed event recognition algorithm as shown in Figure 4.12 was implemented in MATLAB during the working phase of the system. The samples of object tracking results are illustrated in Figure 4.15 and 4.16 for vehicle and human events respectively. The tracked motion trajectories of the corresponding events are shown in the rightmost images. These tracked raw trajectories serve as the key parameter in the event recognition phase. The details of the algorithm used in this work are shown in Appendix A1.

The tracked object is considered valid by computing the filling ration as discussed in Chapter 3.3.2. Object that has a filling ratio of less than 0.2 is categorized as noise due to the sudden change in illumination. Once the object is valid, the motion trajectory is tracked by collecting the centroid coordinates of the subsequent frames until the object lost its track. Once the event is completed, the type of object was determined by the object features (blob size and aspect ratio) based on the algorithm as discussed in Chapter 3.3.2. After the type of object is confirmed, the POIs (first and last centroids) of the trajectory are extracted and compared with the areas of interest as depicted in Figure 4.5.

Once the regions of POIs are confirmed, the detected event will be tagged with the potential event based on the logical combination as shown in Table 4.1. The tracked trajectory will then be processed to obtain the feature vector representation (X and Y axis projections, velocity profile and area profile of the motion trajectory) for similarity measurement with the trained event representation. Several pre-processing (RESAMPLING and ZSCORE) steps are applied to the feature vector to standardize and normalise the data received.
Figure 4.15 – Experimental results of the object detection and tracking process for vehicle events. From left to right column, the tracked object with the computed bounding box, the foreground image and the tracked trajectory of the moving object. And from top to bottom, the event of “A vehicle passes by the monitored parking area”, “A vehicle enters the monitored area and parks at an empty parking space” and “A vehicle gets out of parking space and exits the scene”.

In order to validate the effectiveness of the proposed event recognition algorithm, a number of test video sequences taken from the aforementioned scene were inputted to the system and the recognition results were tabulated in Table 4.4. As shown in the table, the test video sequences comprise of 63% of car initiated events, 33% human initiated events and 4% of anomalous events such as car moving in the opposite direction. Out of the total human based events, 73% of the events were recognised correctly. While a 100% recognition accuracy has been achieved for car events. Meanwhile, the 4% of anomalous events were also perfectly recognised by the proposed system. Taken as a whole, the proposed event recognition algorithm has recorded a 91% of precision for the random test sample videos taken from the same scene.
Figure 4.16 – Experimental results of the object detection and tracking process for human events. From left to right column, the tracked object with the computed bounding box, the foreground image and the tracked trajectory of the moving object. And from top to bottom, the event of “A person walks through the monitored parking area”, “A person enters the scene and picks up a vehicle” and “A person drops off from vehicle and exits the scene”.

<table>
<thead>
<tr>
<th>Event</th>
<th>Test Sample</th>
<th>Correctly Recognised</th>
<th>Precision (%)</th>
<th>Object-based Precision (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAR_PASS</td>
<td>47</td>
<td>47</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>CAR_IN</td>
<td>8</td>
<td>8</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>CAR_OUT</td>
<td>9</td>
<td>9</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>PPL_PASS</td>
<td>23</td>
<td>18</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td>PPL_IN</td>
<td>6</td>
<td>2</td>
<td>33</td>
<td>73</td>
</tr>
<tr>
<td>PPL_OUT</td>
<td>4</td>
<td>4</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>ANOMALY</td>
<td>4</td>
<td>4</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>TOTAL</td>
<td>101</td>
<td>92</td>
<td>91</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.4 - Statistical result of the proposed event recognition system.
The statistics of the proposed event recognition system’s accuracy in Table 4.4 shows excellent correct detection rate especially in car initiated events, these experimental results were obtained by separately feeding individual videos (contains only 1 event) to the system. Thus, it is also important to examine the reliability of the proposed system when a sequence of events is presented in a single input video source. In order to examine the accuracy of the proposed system when a continuous video stream is inputted, a video of about 1 minute (contains 4 different events) was tested as shown in Figure 4.17.

![Figure 4.17](image)

**Figure 4.17** – A test video of about 1 minute which contains 4 separate events was tested on the proposed system. (a) A car passed by the scene, (b) people picked up a car, (c) people walked by the scene, and (d) car moved out of the parking space.

The output of the abovementioned test is tabulated in Table 4.5. The first event was detected at frame number 53 and ended at frame number 82. The feature vector \( f_i \) of the test video has shown a nearly perfect match to the trained definition of the event ‘Car passes through the scene’ with a correlation of 0.9894. Second event was detected at frame number 124 and ended at frame number 246. The scene information of the trajectory suggests that this is ‘a person walked through the scene’ event, and the feature vector computed from the trajectory strengthens the decision with a 0.9038 correlation to the trained definition. The event tag \( T_e \) of the third detected event matched with the computed event definition of ‘a person enters the scene and picks up a vehicle’ event with feature vector correlation of 0.9616 from frame number 372 to 563. Lastly, the fourth event was detected at frame number 679 to 824 that matches the ‘car leaves the scene from rest’ event with a feature vector correlation of 0.7650.
<table>
<thead>
<tr>
<th>Event sequence</th>
<th>Start Frame (Ts)</th>
<th>End Frame (Tf)</th>
<th>Start Position (Xs)</th>
<th>End Position (Xf)</th>
<th>Object Type</th>
<th>Type of Event</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>53</td>
<td>82</td>
<td>entrance</td>
<td>exit</td>
<td>car</td>
<td>1</td>
<td>0.9894</td>
</tr>
<tr>
<td>#2</td>
<td>124</td>
<td>246</td>
<td>entrance</td>
<td>exit</td>
<td>people</td>
<td>4</td>
<td>0.9038</td>
</tr>
<tr>
<td>#3</td>
<td>372</td>
<td>563</td>
<td>entrance</td>
<td>internal</td>
<td>people</td>
<td>5</td>
<td>0.9616</td>
</tr>
<tr>
<td>#4</td>
<td>679</td>
<td>824</td>
<td>internal</td>
<td>exit</td>
<td>car</td>
<td>3</td>
<td>0.7650</td>
</tr>
</tbody>
</table>

Table 4.5 - A process of mapping tracks onto events in the sample video sequences. The events are labeled accurately with a high correlation coefficient.

Figure 4.18(a) shows the object trajectories of the four detected events in the test video stream and Figure 4.18(b) illustrates the graphical representation of the sequence of events detected from the test video. A composite event ‘Pick up’ resulted from low-level events ‘a person enters the scene and picks up a vehicle’ and ‘car leaves the scene from rest’ is identified by analysing the spatial and temporal descriptions (as described in Section 4.4.1) of the consecutive events.

Figure 4.18 - (a) Extracted object trajectories and (b) their sequence of events detected from the test video stream. A composite event, a person picks up a vehicle and leaves the parking lot is analysed from the low-level events and temporal information.

In this experiment, the threshold distance $d_t$ was set to the longest distance from the centroid location of the car object to the corner point of the bounding box. The bounding box of the car object (the first detected object of second event) in this specific case was measured at 37 pixels height $\times$ 24 pixels width. Therefore, the distance threshold $d_t$ was calculated at 22 pixels. The measured Euclidean distance $d$ of the two objects (from A to B) was found at 19 pixels and thus fulfilled the condition $d < d_t$. Also, the human object (A) was found at the right side relative to the car object (B) in this
particular case. As for the timing, the time interval threshold $T_t$ was set to 30 seconds (450 frames) between the two consecutive events. The event #4 (refer to Table 4.5) in the experiment was initiated 116 frames after the event #3 ended. Hence, the timing condition was also fulfilled in this case. After the composite event algorithm, the event #3 and #4 was recognised successfully as a composite event of ‘pick up’.

4.5.3 DISCUSSIONS

The proposed autonomous event recognition system comprises four main phases as depicted in Figure 4.1 for both training and running processes. The main challenges in this system were encountered in the object detection and tracking phase, where large amount of external factors such as irregular changes in lighting condition, unpredictable environmental factors (wind, rain fall, etc...), randomness of the human behavior and many more unforeseen interactions from the external sources. In terms of computation/algorithm limitation, the difficulty in determining the exact shape of foreground objects under certain special conditions such as occlusion, shadow, similar color spectrum between foreground and background objects, and transition from background scene to foreground object when car moves from static have limited the accuracy of the proposed system.

The adaptive GMM background modeling employed in this work has shown to effectively handle the variation in lighting condition and repetitive motion caused by the wind by modeling each background pixels with a multimodal Gaussian distributions and learning the changes in pixel value over time (refer to Chapter 3.1.1). Therefore, the main limitations of this system is the accuracy of estimating the precise contour of the moving object and thus affecting the correctness of the feature extraction and event recognition processes. Figure 4.19 shows examples of the abovementioned conditions faced during the experiment which affected the accurate detection of the foreground object. From left to right, the shadow was detected as a part of the walking person, a part of the body was occluded by the stationary car, a part of the moving object was categorized as background due to similar color spectrum and lastly, the new background took time before it was incorporated into the background model. All these imperfections
in object detection may cause inaccuracy in the computation of object features such as centroid location of the object (and thus the tracked trajectory), shape and size information and consequently leading to deviation in the feature vector.

![Figure 4.19 – Imperfect object detection](image)

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

**Figure 4.19** – Imperfect object detection, (a) the shadow was detected as a part of the walking person, (b) a part of the body was occluded by the stationary car, (c) a part of the moving object was categorized as background due to similar color spectrum and lastly, (d) the new background took time before it was incorporated into the background model.

It is undoubtedly that improvement in the object detection can significantly enhance the accuracy of the event recognition system; however the main focus of this works is to study and evaluate different feature vector representations and event learning approaches (supervised and unsupervised learning) rather than implementation of an accurate event recognition system. Nevertheless, improvement in the object detection algorithm will be one of the future works proposed after this thesis.

In the training (event learning) phase, the event representations computed by PCA method have shown to effectively distinguish different events by extracting the position, velocity and blob size profiles throughout the entire activity. As illustrated in Figure 4.10, the principal components of respective car events have shown to behave uniquely for all the $x$ and $y$ projections, velocity and blob size profiles thus providing a very discriminative representation of car events. On the other hand, the principal components representation of feature vectors extracted from the human events (as shown in Figure 4.11) have shown to deviate significantly from some training data notably the $x$ and $y$ projections. This is mainly due to the difference in the nature of object (car or human) that initiates the event.
In a car park facility, the direction of the traffic flows are normally fixed and organized. For example the topology setting of this work (as shown in Figure 4.14), vehicle will normally enter the scene from the left edge of the scene and leave the scene at the right edge of the scene. Also, the car will normally move within the aisle where it is located at the center of the entire scene. Therefore, one can expect the $x$ and $y$ projections feature of the car events will exhibit a more consistent pattern. Unlike the nature of the car events, the human initiated events will generate a relatively more random and inconsistence features compared to the car events. This is because in an open space such as the outdoor car park environment studied in this work, human objects will be expected to enter and leave the scene at any location along the edges.

In addition, the path that a human object takes is not limited by the topology settings of the scene. Thus, the trajectory generated by the human initiated event will somehow distributed randomly across the scene and the manner the object moves will vary even for similar class of event. As can be in Figure 4.11, the horizontal projection of the motion trajectories (subfigure a, e and i) for a set of training data of similar class have shown discrepancy due to different path taken by the human object, either from left to right (increasing lines) or vice versa. Thus it is difficult to compute a single vector representation that describes all the variations in the training data, especially for the path information of the motion trajectory.

However, the velocity and blob size profiles of the human events behave in a more consistent pattern as can be seen in the last two column of Figure 4.11. Thus the computed principal components for these two features have a more reliable description of the general behavior of each event. Due to the randomness of the path information in human events, a selective feature representation approach is proposed where the spatial information of the human events is removed from the feature vector since it does not fully define the behavior of the human events. This selective feature representation between human and car events will be implemented in the next event recognition system which will be discussed in Chapter 5 and then a comprehensive comparison between the two approaches will be discussed.
In the event recognition phase, the manually defined region of interest (ROI) works perfectly on the scene studied in this work. However, this scene modeling approach is specific for one scene and therefore, the model used in one scene may not be applicable to another. The relatively simple topology settings of the outdoor car park scenario made the scene model straightforward and reliable. As can be seen in Figure 4.5, there is only one entrance zone and one exit zone for the vehicle on both sides of the image, and fixed parking spaces located on top of the image. On the other hand, the entrance and exit zone of the human events are located along the four edges of the image where the human object can enter/exit the scene at any direction outside the camera’s sight. Other than the entrance and exit zones, the parking space area where the interaction of human and car activity occurs will also be one of the critical zones in the recognition algorithm.

Undoubtedly, the ROI specifically set in this work might not perform equally well even in an equivalent scene that consists of similar amount of zones but located differently across the image. Therefore, in order to enhance the system to be as generic as possible, an unsupervised scene modeling technique is proposed and will be applied to the unsupervised event recognition system in Chapter 5. As can be seen in Chapter 5, the proposed automatic scene modeling approach is capable of identifying critical ROIs in a given scene of equivalent topological settings by analyzing the POIs of the training data.

The event recognition algorithm (refer to Figure 4.12) works effectively on the car event in which a 100% accuracy was recoded. This outstanding event recognition accuracy was mainly because of the consistent in the appearance of car object and the much simpler manner of motion compared to human activities. The fault detections in human initiated event were generally due to the unmatched feature vectors that caused by the inconsistence in path information (x and y projections of motion trajectory). In addition, the high level activity classification algorithm implemented in this work has shown to sufficiently recognize the two common activities that occur in outdoor parking facility. Certainly, there are more activities which can be found in the car park environment other than the abovementioned ones, similar approach can be extended to include other high level activities for example, the human-to-human interactions (i.e.
talking, passing object from one to another, fighting, etc.) and human-to-static object 
interactions (i.e. putting and object to the scene, remove static object from the scene, etc.).

In general, the proposed event recognition system with supervised learning 
approach has demonstrated to be able to detect and recognize simple events in an 
outdoor car park environment. Though the overall recognition accuracy of 91% is not as 
good as other system in the literature, the car initiated events have been detected and 
recognized with outstanding results. The explicitly defined event representations 
computed from the motion trajectory have shown to reliably and discriminatively 
describe the common events in the car park scene especially for the car initiated events.

4.6 CONCLUSIONS

An event recognition and anomaly detection system is presented in this chapter. The 
proposed system consists of four main sub-systems that perform specific task essential 
to effectively recognise event, namely object tracking phase, event representation phase, 
training phase and lastly event recognition phase. The outcomes of the object tracking 
process (refer to Chapter 3) will be analysed and trained to construct the representations 
of common events and then classify the detected event in the car park scene while the 
system is in the operation mode.

In the event representation phase, the feature vectors extracted from the object 
tracking phase will be decomposed into four features, namely the x and y projections of 
the moving object, the absolute velocity of the object, and the change in object size over 
time. The normal events are defined by its object type (car or vehicle) and the 
abovementioned features (x and y projections, velocity and object size profiles). While 
in the training phase, the event representations of the known classes are trained 
separately to form an explicit definition of normal events. The PCA is employed in this 
system to compute the principal component of each data in the feature vectors of the 
same class.
In order to reliably recognise the detected event, the proposed event recognition algorithm combines the object type, POIs mapping and matching of the feature vector in the process of recognising event. A composite activity recognition algorithm is also implemented in this system to identify the possible higher level activity such as “pick up” and “drive in”.

The experimental results have demonstrated an overall accuracy of 91% for both human and vehicle initiated events including abnormal events. The proposed system has shown an exceptionally good recognition accuracy for the car initiated events (100%) compared to the human events of only 73%. The low recognition accuracy in human events was mainly due to the difficulties in extracting the common pattern from the trajectory data, especially the path information of the moving object due to the randomness of path taken by each individual human, and their unique manner of motion. In general, the works in this chapter have demonstrated the feasibility of training explicit definition of the normal events by using PCA approach. The integration of object appearance, POIs and trajectory data have shown to effectively recognise the common events that can be found in an outdoor car park environment.
CHAPTER 5

UNSUPERVISED TRAJECTORY LEARNING APPROACH

5.1 OVERVIEW

In this chapter, an unsupervised trajectory learning approach for car park surveillance video is presented. The main objective of this work is to develop a hierarchical trajectory learning system that is capable of automatically learn from observations without any prior knowledge (unlabeled data) and with little human intervention. As mentioned in Chapter 3, an object tracking process creates motion trajectory of a given event which is a record of the evolution of object positions within a finite time interval from $t_s$ to $t_f$. Therefore, a trajectory can be perceived as a function of temporal domain to a range of spatial/geometry values, for instance:

$$\text{trajectory}, T : \{t_s, t_f\} \rightarrow \text{space}$$

(5.1)

where space are represented by coordinates of horizontal axis $x$ and vertical axis $y$ in 2-dimensional plane. The key of a learning framework is to learn the spatial and dynamic information of the sample trajectory data. Specifically, the dynamic/velocity and the direction/orientation of the object movement are concerned in this work.

The following section presents the implementation of a hierarchical trajectory learning system by exploiting the abstraction of the trajectory data at different level: spatial, dynamics, direction and spatiotemporal. In a typical scenario, most of the activities in a parking lot surveillance video are induced by either vehicle or human. Therefore, the trajectory learning framework proposed in this work focuses on analyzing and learning the behavior performed by these two agents.
Resulting from the hierarchical trajectory learning framework is the semantic annotation of a detected trajectory that describes the activity performed by the moving object. Abnormal event is detected if the new trajectory did not pass the confidence level in the likelihood test against the learnt patterns from the training dataset. Figure 5.1 illustrates the overall system architecture of the event recognition and anomaly detection framework. The object detection and tracking processes presented in Chapter 3 provides the low-level visual properties (object type and motion trajectory) for the higher level activity analysis and learning process (feature extraction, learning and reasoning) which are the main focuses of this chapter.

This chapter first presents the hierarchical trajectory learning approach which includes the autonomous scene modeling technique and activity modeling by exploiting multilevel information of the motion trajectory. Secondly, an event recognition and anomaly detection algorithm is presented based on the topographical information of the track and likelihood assessment of the feature vector to the trained event model.

Figure 5.1 – System architecture of the proposed event recognition and anomaly detection.
5.2 HIERARCHICAL LEARNING OF MOTION TRAJECTORY

An offline trajectory computing and learning model is presented in this section. During the observation and learning phase, trajectories are collected to create a training dataset. The hybrid trajectory learning model abstracts different level of data for mobility understanding. The first kind of information extracted with the proposed approach is static information (point level) that provides knowledge about the structure of the scene. The second level of data is the direction of the motion trajectory (spatial level) that describes the pattern of the motion. Lastly, the dynamic level of the motion trajectory is extracted that defines the manner of the object movement. Figure 5.2 depicts the overall architecture of the proposed hierarchical trajectory learning framework.

![Hierarchical Trajectory Learning Model](image)

**Figure 5.2** – Hierarchical Trajectory Learning Model.
5.2.1 SEMANTIC SCENE MODELING

To apprehend semantics of the trajectories data, the geometric spatial points (x and y) have to be transformed into a meaningful description of the scene region [73 - 76]. This is because human usually interprets activity relative to the scene features, for example, “A person walked to the bus stop” or “a car parked at car lot”. Moreover, the structure of the scene normally influences the dynamics and the direction of the activity of the moving targets. For instances, traffic lanes indicate where vehicles move, vehicle U-turns at a U-turn lane, bus station indicates where the vehicles and the pedestrians will stop, and pedestrians will walk along the pathways. Therefore, categorizing image view into semantic regions is crucial for the activity modeling in the subsequent sections.

As mentioned earlier, scene features affect the activities of the moving objects. Therefore, a reverse engineering technique is used for learning the scene elements. The scene models can be defined manually as mentioned in Chapter 4, however, this chapter suggests that scene features can be learnt from observations in an unsupervised way. This approach provides the advantages that it minimizes the human effort in labeling the scene feature manually, and most importantly, it allows the scene model to detect and respond adaptively to the changes of the scene environment.

The goal of the scene modeling is to provide topographical scene description (entrance, exit, parking zone, etc.) by unsupervised inference of trajectory patterns in the static camera scene. We aim to provide the surveillance systems to automatically build a knowledge base of their environment based on the given dataset. Unsupervised learning is done by the probabilistic labeling. A probabilistic description which allows multiple labels to co-exist in objects is desirable in this system. Labels are probabilistically weighted with a confidence level assigned to each object label. Besides, probabilistic labeling can deal with uncertainties and allows the adaption of the model. Therefore, probabilistic labeling is more suitable for a long-term prediction and atypical detection in this surveillance system.
5.2.1.1 Entry and Exit Zones Modeling

Considering a car park scene environment, a vehicle enter the scene is either parking in the car park lot or passing by the observed scene. On the other hand, a person either enters the scene to find a specific car, or exits the scene after parked the car, or passes by the observed scene. This situation sets the level of goals which regions to be modeled: entry and exit zones (which are also called sources and sinks). The entry and exit zones are the places where the targets enter or exit the camera of view or where the targets appear and disappear from the scene. In other words, entry and exit zones are either the view-based features (borders of the image) or the scene-based features (doors, gates, car and etc.). In some circumstances, entry zone and exit zone may be coincident. This occurred mostly for the pedestrian environments. However, for a road traffic environments or vehicle circumstances, entry and exit zone mostly fall into distinct regions. This is due to the constraints of the road traffic regulations.

As mentioned earlier, since the entry zone is the region where the target enters the scene, the entry zone can be learnt by using a dataset that consists of the first points of the trajectories of the moving target. Similarly, the exit zones can be located by examining the end points of the trajectories. These spatial-temporal points are collected from a given set of training video sequence and will be used to train the semantic scene model by using a probabilistic framework. The scene modeling aims to decompose the camera views into $i$ regions of $R$ by learning the geometrical distribution of the spatial points collected from the training phase. Therefore, the semantic region $R$ is a set of locations which extent is comprised of spatial points $p$:

$$R_i = \{p_1, p_2, p_3, \ldots, p_n\}$$  \hspace{1cm} (5.2)

5.2.1.2 ROI Learning

The input of the scene analysis modeling is the set of motion trajectories $T$ which are the result of the object tracking module from the training video sequences. Thus, $T$ can be represented by:
where \( m \) is the total number of training dataset. Each motion trajectory \( T_i \) for a given time interval \([t_s, t_f]\) consists of sequence of geometry points \( \{X_s, \ldots, X_f\} \). Thus, an entry-point dataset \( E \) which comprises of \( m \) number of \( X_s \) (which are the first point of each trajectory in \( T \)) is collected. The same approach is applied to the training set for exit point dataset \( S \) that consists of \( m \) number of \( X_f \). The entry and exit points’ dataset \((E \text{ and } S)\) are then modeled by a 2D Gaussian mixture model (GMM).

Taking the entry point as example, the number of Gaussians that best represent dataset \( E \) need to be determined in order to find number of entry zones available in the scene (lets denote it as \( k \)). Therefore, there are \( k \) number of Gaussian distributions \( \Gamma_i \), where \( i = 1, 2 \ldots k \), to represent each of the entry zones. Each of these \( \Gamma_i \) can be written in the notation:

\[
\Gamma_i \sim N(\mu_i, \Sigma_i) \quad \text{(5.4)}
\]

The probability of an \( j^{th} \) observation data point \( X_s \), which denoted as \( x_j \) from a \( K^{th} \) GMM is given by

\[
p(x_j) = \sum_{i=1}^{k} \pi_i \ p(x_j | \Gamma_i) \quad \text{(5.5)}
\]

where \( \pi_i \) is the distribution contribution weight and \( \sum_{i=1}^{k} \pi_i = 1 \) and \( p(x_j | \Gamma_i) \) is the Gaussian distribution

\[
p(x_j | \Gamma_i) = - \frac{1}{(2\pi)^{d/2} |\Sigma_k|^{1/2}} \ e^{-1/2(x_j - \mu_k)^T \Sigma_k^{-1}(x_j - \mu_k)} \quad \text{(5.6)}
\]

where \( d \) is the dimensionality of the model (in this case \( d = 2 \) for a trajectory data input).

The Expectation Maximization (EM) algorithm is used to estimate the parameters of these GMM distributions [77]. During the expectation step, the probability of the \( K^{th} \) Gaussian that accountable to an \( j^{th} \) observation data is computed as:

\[
p_{jk} = \frac{\pi_k p(x_j | \Gamma_k)}{\sum_{i=1}^{k} \pi_i p(x_j | \Gamma_i)} \quad \text{(5.7)}
\]
During the maximization step, the parameters of the Gaussians are recomputed as:

\[
\begin{align*}
\pi_K &= \frac{1}{k} \sum_{i=1}^{k} p_{jK} \\
\mu_K &= \frac{\sum_{i=1}^{k} p_{jK} x_j}{\sum_{i=1}^{k} p_{jK}} \\
\Sigma_K &= \frac{\sum_{i=1}^{k} p_{jK} (x_j - \mu_K)^T (x_j - \mu_K)}{\sum_{i=1}^{k} p_{jK}}
\end{align*}
\]

The EM iterates the algorithm until the parameters converge.

The main idea of this algorithm is that the observations from the training trajectory through the Point of Interest (POI) will contribute to the probability distribution function at that location. Although k-means can cluster the trajectory data on areas of interest, it only returns a single point to represent a cluster. However, we wish to know the spatially extension of the geographical area of the region of interest.

The region of interest in the outdoor car park scene considered in this work is relatively straightforward compared to other public scenes such as subway station and plaza. The planned traffic route suggests that there is only one entrance and one exit zone for vehicle as shown in Figure 5.3(a). Another region of interest for vehicle events is the parking bay area. Therefore, the main aim of the scene modeling described above is to identify these regions of interest automatically, without any human interpretation. The entry point and exit point dataset \(E\) and \(S\) were clustered by the abovementioned GMM clustering technique and the entry and exit regions were identified as depicted in Figure 5.3(b).

As shown in the figure, there are two clusters (regions) for the entry point dataset \(E\) (shown in red), and similarly two distinct regions were identified from exit point dataset \(S\). In this work, an algorithm is proposed to analyze the scene structure and meaningfully label the regions of interest (entrance, exit, parking lot) by exploiting the nature of the activities. Specifically for vehicle events, there are three types of common events:
1. A vehicle passes by the monitored parking area
2. A vehicle enters the monitored area and parks at an empty parking space
3. A vehicle gets out of parking space and exits the scene

Based on the nature of the vehicle events listed above, we can expect overlapping of the entry point and exit point clusters within the parking lot area. This is because in event type #2, the exit point of the event will be located at the parking lot area, and similarly the event type #3 will be initiated at the same region. Thus, the parking lot region can be identified by analyzing the topology of the clustered regions. The overlapping area of entry and exit cluster will be labeled as parking lot region, whereas the isolated entry cluster will be labeled as entrance zone and similarly, the isolated exit cluster will be classified as exit zone.

As shown in Figure 5.3(b), the overlapping entry and exit cluster (shaded ellipses) is located at the top of the scene, which is exactly the parking lot area of the observed scene. Also, the entrance (yellow ellipse) and exit (green ellipse) zones are both accurately reflecting the real entrance and exit area of the observed scene. This region of interest identification algorithm is valid for scenes that have equivalent scene structures (one entrance and exit) regardless of the topological setting of the scene (location of the entrance, exit and parking lot zones).

As for the human event, since pedestrian can enter/exit the observed scene at any location along the edges, the entrance and exit zones for human events are set stationary along the four edges of the observed scene as shown in Figure 5.3(c). These scene models for vehicle and human events respectively will be used as part of the event recognition process. A preliminary event classification will be performed by analyzing the point level description (entry and exit points) of the motion trajectory with the two scene models computed by the proposed approach.
Figure 5.3 – Regions of interest computed from the scene modeling technique proposed in this work. (a) The observed scene. Regions of interest for (b) vehicle events and (c) human events.

5.2.2 TRAJECTORY ANALYSIS

This section presents a detailed analysis of the motion trajectory by exploiting the higher level spatiotemporal information of the tracks. Specifically, the direction and velocity features of the motion trajectories extracted from the object tracking process are analyzed and transformed into a low dimensional but descriptive feature vectors. An unsupervised activity learning framework is implemented by adopting k-means and HMM to equip the system with a knowledge base for event reasoning and detection of unusual events.
5.2.2.1 Trajectory Resampling

The centroids of the moving object forms the trajectory of movement and it provides important information of the event’s behavior such as path, direction and velocity of the object. In many cases, the dimension of the motion trajectory is relatively higher than the information that it holds. Therefore, taking into account the raw trajectory data recorded by the object tracking process in the event learning and recognition processes is ineffective and required unnecessary computational power.

In order to reduce the dimension of the input trajectory data, a low dimensional feature vector representation of an event is constructed based on a resampled trajectory points as shown in Figure 5.4. The sampled motion trajectory can be represented by a flow vector of

$$\{(x_t, y_t), t = 1, 2, \cdots, N\}$$

(5.9)

where \(x_t\) and \(y_t\) are the projections of the object’s centroid location corresponding to the \(x\) and \(y\) axes at instance of time \(t\). \(N\) is the number of sample. By resampling the raw trajectory, the dimension of the data can be reduced significantly while retaining the important information of the event. Admittedly, selecting an appropriate number of samples \(N\) is crucial as it will directly affect the accuracy of the system. Too few samples will lose the information brought by the data, while too much samples decreases the efficiency of the system.

In this work, the raw trajectory points are divided into 7 sections which the number of points from one sample point to the next point is equal. In other words, the time interval between adjacent points is the same. Sampling the raw trajectory by its time interval rather than distance has the advantage of retaining more information brought by the raw data. This advantage can be observed clearly when the object changes its direction as depicted in Figure 5.4. The sampled trajectory based on time interval has shown to fit better to the original trajectory data compared to the sampled trajectory based on distance. This is because of the nature of the moving object where the velocity
of the object tends to slow down when changing direction and thus shorter distance travelled within a fixed amount of time interval. Therefore when sampling the raw trajectory based on time interval, more points will be sampled near the curvy path compared to the straight path. In addition, sampling based on time interval also maintains the velocity information of the moving object which is critical in characterizing an event. The sampled trajectory will be analyzed and transformed into a digitized feature vector parameters that will be used in the event learning and recognition processes.

![Figure 5.4](image)

**Figure 5.4** – Trajectory resampling based on (a) time interval and (b) distance.

### 5.2.2.2 Feature Vector Transformation

The feature vector of event can be represented by $e = \{v_t, \theta_t\}, t = 1 \ldots N$, where $v_t$ is the distance vector of the sampled trajectory and $\theta_t$ is the absolute angle vector of the sampled trajectory with reference to the horizontal axis at $t^{th}$ point as depicted in Figure 5.5. $N$ is the total number of sample points. The distance vector $v_t$ describes the Euclidean distance of two consecutive points. Given a fixed frame rate, the distance vector also indirectly describes the velocity information of the object movement. On the other hand, the angle vector $\theta_t$ defines the direction of the object movement between two consecutive points.
Once the distance and angle vectors extracted from the sampled trajectory, a discretization transformation is applied to the feature vector to reduce the dimension of the data. The discretization technique employed in this work is illustrated in Figure 5.6. As for the distance vector, the distance values are categorized into nine classes as shown in Figure 5.6(a) with an interval of 5 pixels. On the other hand, the angle values are grouped into nine classes in which each class are 40° apart as depicted in Figure 5.6(b). The quantized feature vector will then be used in the event learning and recognition processes which will be discuss in the later section.

Figure 5.5 – Distance and angle vectors construction.

Figure 5.6 – Feature transformation framework for (a) distance vector and (b) angle vector.
5.2.2.3 Feature Selection

As discussed in Chapter 4.2, event that initiated by different type of object (car or human) has different dominant features, some features such as the path and direction information of human initiated events are meaningless as the human object can enter/exit the scene at any location and take different path in each and every instances. Thus, selecting appropriate features to represent events of different nature is essential in enhancing the recognition accuracy as well as reducing the computational power required.

Selecting appropriate features to represent an event is often dependent on the applications. Different nature of events or type of objects that initiate the event have their own dominant feature such as position, absolute and relative directions, velocity, and acceleration information. For example, a system that intended to detect illegal speeding on a federal highway requires the dynamic information of the moving object to be analyzed. On the other hand, detecting illegal turning at intersection will only require the relative direction of the motion vector as the feature. One of the most essential parts in constructing feature vector representation of an event is not to gather all the information from a trajectory, rather, to extract the dominant feature that best describes the event.

In this work, a surveillance camera that monitors the outdoor parking facility is considered. Particularly for the parking lot scenario, there are two types of object that initiates events; the vehicle and the human. Due to the limited path available for the vehicle moving in the parking lot, a feature that describes the spatial information of the movement was selected. Specifically, the direction information (angle vector) of the trajectory is selected as the dominant feature for vehicle events. Based on the selected scene, there is only one entrance and one exit region in the area and therefore, vehicle can only move in certain pattern. Also, due to the nature of the movement of vehicle, the trajectory of certain event is somehow predictable. For example, a vehicle enters the scene and occupies a parking space will generally generate a track of left turn after entering the scene. Thus, the absolute angle vector of the trajectory is sufficient to describe the type of events initiated by vehicle.
Unlike the path spatial information of vehicle events, the paths of human events are typically random and unpredictable because of the open space setting in the parking lot environment. For instance, a person can enter the monitored area at any point along the edge of the scene, and most essentially, the path taken by a person is generally random in every single event. Therefore, analysing the angle vector is not feasible in this scenario. The classification algorithm for human events takes into consideration the dynamic information (distance/velocity vector) of the motion trajectory. For example, a person enters the scene and pick up a car will generate a decreasing velocity vector due to the person stops in front of the car temporarily before entering the car. Therefore, the car and human events will be trained and recognized separately with different feature in this work.

5.2.2.4 Clustering of Feature Vectors

After the transformation of trajectory data in the previous section, the feature vectors are clustered into different classes according to their dynamics and orientation pattern. There are many algorithms for spatiotemporal clustering available in the literatures. Comprehensive reviews and comparisons on the algorithms for trajectory clustering can be found in [77-79].

K-mean clustering method which is a partition clustering based on the distance among cluster centroid is adopted to classify the feature vectors in this system. The quality of the clustering method is depending on the distance function of the clustering algorithm. A comprehensive review on different clustering distance function can be found in [80]. This algorithm minimizes the intra-cluster over all clusters and inter-cluster distances iteratively by using Squared Euclidean distance. The algorithm converges once it has reached the minimum intra-cluster and inter-cluster distance. Each features vectors are mapped into one of the $K$ cluster indices. Each of these cluster indices represents each class of events in the surveillance video. The detailed K-mean algorithm is shown in Figure 5.7.
The feature vectors of the unlabeled trajectory data are first classified into two major classes: the vehicle and human classes by analyzing the object type with the object recognition algorithm presented in Chapter 3. Once the type of object is identified, the dominant feature (direction or velocity) is selected based on the nature of the object movement. Specifically, the direction feature will be selected to describe vehicle events. In contrast, the velocity feature will be chosen to represent human events. After the dominant feature is selected, the K-mean clustering algorithm is applied to further classify the feature vectors into different classes of similar behavioral pattern. The number of classes is set to three for vehicle and human events respectively. The graphical representation of the entire feature vector clustering process is illustrated in Figure 5.8.
The abovementioned feature vector clustering strategy was applied to the trajectories collected from the training dataset and the outcomes of the clustering process are shown in Figure 5.9. Figure 5.9(a) shows the clusters of feature vectors (direction feature) for vehicle event and Figure 5.9(b) shows the clusters of feature vectors (velocity feature) of the human events. The selected dominant features for vehicle and human events have shown to discriminatively describe the behavior of each class of events.

Figure 5.9 – Results of the proposed feature vector classification approach. (a) Three classes of vehicle events based on the behavioral pattern of direction feature and (b) Three classes of human events based on the behavioral pattern of velocity feature.
For example, the vehicle event classes, the Class 1(V) event does not change the direction throughout the entire activity which implies the activity of vehicle passes through the observed scene. Class 2(V) event changes its orientation in the counter clockwise direction which describes the events of vehicle occupies the parking lot on top of the observed scene. Lastly, Class 3(V) events have similar pattern where the direction vectors have orientation around the third quarter initially and then change it direction towards the exit location of the observed scene. This class of events has shown a similar behavior compared to the activity of vehicle reverses from the parking lot and exits the observed scene.

As for the velocity information of the human events, the stable velocity observed in Class 1(H) describes a pedestrian walking through the monitored scene. While the increasing velocity from nearly stationary (Class 2(H)) and decreasing velocity to motionless (Class 3(H)) express clearly the activities of human entering and leaving a car.

5.2.2.5 Activity Training by HMM

HMM model is employed in the training and recognition system as the result of its capability in handling temporal variations in time series data which satisfies the Markovian property. Hidden Markov Model has been widely used in most of the works in vision-based anomaly detection due to the capability of capturing spatial-temporal dynamic changes characteristic [81-84]. A succinct overview of the HMM concept will be depicted before the application of the algorithm into the system. A detailed explanation on HMM is documented in [85]. HMM is a stochastic process characterized by the several parameters as illustrated in Figure 5.10.

The $N$ number of hidden states $S = \{S_1, S_2, ..., S_N\}$ and state at each time step $t$ is denoted by $q_t \in S$. A set of initial state probability distribution $\pi_0 = \{\pi_i\}$ where $\pi_i = P(q_1 = S_i), 1 \leq i \leq N$. The $N \times N$ transition probability matrix, $A = a_{ij}$ where $a_{ij} = P(q_{t} = S_j | q_{t-1} = S_i)$ and $i \geq 1, j \leq N$. The $N \times M$ distribution matrix of observation from hidden states $O_t = \{O_1, O_2, ..., O_M\}$, $B = \{b_j(O_t)\}$ where $b_j(O_t) =$
$P(O_t | q_t = S_i)$. Generally, the observation probability density function (PDF) is modeled as a Gaussian Mixture distribution. Therefore, a HMM model is commonly denoted as a compact notation of $\lambda = \{ \pi_0, A, B \}$.

![Figure 5.10 - Hidden Markov Model (HMM).](image)

An event is a sequence of actions generated by the human or vehicle in the surveillance video. The trajectories points that extracted from the tracking modules are samples drawn from each atomic action for an event. Figure 5.10 shows the HMM model that is employed to model trajectory in this work. The initial input is the $k$ clusters of trajectory segment points which form the HMM observation $O_k$. The sequences of atomic actions are the hidden states $S$. To learn the HMM model for trajectory sequence, three parameters, $\lambda = \{ \pi_0, A, B \}$ need to be learnt for each clusters of trajectories. The parameters of HMM model for each class of events can be estimated iteratively using Baum-Welch algorithm [86] (extended forward-backward algorithm). Basically, the parameters of the HMM are trained to maximizes the $P(O | \lambda)$ by employing the forward-backward algorithm iteratively.

The probability of the observation sequence given an HMM parameter $\lambda^m$ is computed by employing Bayes rule

$$P(O | \lambda) = \sum_{i=1}^{N} \alpha_t(i)|_{t=m}$$ (5.10)
where a forward variable is defined as

$$\alpha_t(i) = P(o_1o_2 \ldots o_M|q_t = S_i, \lambda)$$  \hspace{1cm} (5.11)

The backward variable is defined as

$$\beta_t(i) = P(o_t o_{t+1} \ldots o_T|q_t = S_i, \lambda)$$  \hspace{1cm} (5.12)

The optimal sequence state can be found by utilizing Viterbi algorithm where

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)}$$  \hspace{1cm} (5.13)

The probability of being in state $i$ at time $t$ and state $j$ at time $t + 1$ is defined as

$$\varepsilon_t(i, j) = P(q_t = O_1 \ldots O_m, \lambda) = \frac{\alpha_t(i)\beta_{t+1}(j)}{P(O|\lambda)}$$  \hspace{1cm} (5.14)

The parameters $\{\pi_0, A, B\}$ are learned by using an EM algorithm iteratively until the $\gamma$ and $\varepsilon$ convergence.

The classes of feature vectors recognized from the clustering process were trained by using the HMM framework described above. Each class of events was trained iteratively by employing the EM algorithm with 30 iterations. Figure 5.11 shows the log-likelihood of the parameter $\lambda$ against the time series data at each iteration. From the figure, the parameter converges as the number of iteration increases. In other words, the probabilistic parameter is getting closer in defining the variations in the data. The parameter $\lambda$ of each class of events computed from the training process will be used in the event recognition and anomaly detection process.
Figure 5.11 – Log-likelihood of the probabilistic parameters of events during the iterative EM algorithm. (a) Vehicle events and (b) human events.

5.3 EVENT RECOGNITION AND ANOMALY DETECTION

The event recognition and anomaly detection algorithm in the proposed video surveillance system relies upon three processes: object recognition, likelihood test of feature vector and matching of point of interest as shown in Figure 5.12. When a new event is detected, the low-level visual information of the motion is analyzed in multiple levels. First, the object that initiates the activity is identified by employing the object recognition algorithm as presented in Chapter 3.5. The new event will be categorized to one of the two major classes of event according to the type of object (vehicle or human event). Once the type of object is determined, a likelihood test with the activity models devised from the training process will be carried out on the new feature vector. Different set of feature and activity models will be used in the likelihood test based on the type of object. For example, if the new event is classified as a vehicle event, the direction feature will be selected to perform likelihood test against the vehicle event models. Conversely, instead of performing likelihood test on the direction feature, the velocity feature will be used in the case of human event.
Figure 5.12 - Event recognition and anomaly detection algorithm.

The new event will be recognized and labeled as anomaly if the likelihood test did not pass certain confidence level. In short, the new feature vector does not match any of the known classes of behavioral patterns learnt from the training data. On the other hand, if the new feature vector passed the likelihood test to one or more known classes, the new event will undergo the point-level analysis to further analyze the nature of the newly detected event. The scene model computed from the POI analysis (refer to Chapter 5.2.1) will be used. If the new event was recognized as vehicle event, the vehicle scene model (Figure 5.3(b)) will be used. Alternatively, the human scene model (Figure 5.3(c)) will be employed for the case of human event.

The POI (entry and exit points) extracted from the motion trajectory will be compared to the logical table as shown in Table 5.1. There are six combinations of object type and POI of known events that are common in an outdoor parking facility. For example, if a vehicle event entered the scene at the vehicle entrance zone and terminated at the parking lot zone will be interpreted as “a vehicle enters the observed scene and occupies an empty parking lot”. Otherwise, if no match is found for the newly detected event, the event will be categorized as unusual event. In addition, the high level activity recognition approach presented in Chapter 4.4.1 can be adopted in this system to provide higher level of activity understanding in surveillance videos.
<table>
<thead>
<tr>
<th>Event</th>
<th>Type of object</th>
<th>Start point, $X_i$</th>
<th>End point, $X_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>Vehicle</td>
<td>Entrance (Vehicle)</td>
<td>Exit (Vehicle)</td>
</tr>
<tr>
<td>V2</td>
<td>Vehicle</td>
<td>Entrance (Vehicle)</td>
<td>Parking Lot</td>
</tr>
<tr>
<td>V3</td>
<td>Vehicle</td>
<td>Parking Lot</td>
<td>Exit (Vehicle)</td>
</tr>
<tr>
<td>H1</td>
<td>Human</td>
<td>Entrance/Exit (Human)</td>
<td>Entrance/Exit (Human)</td>
</tr>
<tr>
<td>H2</td>
<td>Human</td>
<td>Entrance/Exit (Human)</td>
<td>Parking Lot</td>
</tr>
<tr>
<td>H3</td>
<td>Human</td>
<td>Parking Lot</td>
<td>Entrance/Exit (Human)</td>
</tr>
</tbody>
</table>

**Table 5.1-** Logical POI and object type combinations for different known events.

### 5.4 EXPERIMENTS

In order to have a fair comparison of the effectiveness of the proposed event recognition and anomaly detection system presented in this chapter against the one presented in Chapter 4, the training and testing video data were exactly the same as the one used in Chapter 4 (refer to Table 4.3). However, since the unsupervised activity learning approach is implemented, the labelled training data used in the previous approach (Chapter 4) were made unlabeled before the training process begins. A similar set of testing video sequences were applied to the system.

To extract the low level visual properties of the tracked target, similar object detection and tracking technique as presented in Chapter 3 were employed. Specifically, the adaptive GMM background modelling and foreground segmentation approach was adopted to detect the motion. The spatial and appearance properties of the foreground object were collected during the entire activity for object analysis and motion trajectory formulation. The object recognition framework as presented in Chapter 3.5 was employed to classify the type of event. The motion trajectory of the detected event will then be analysed by using the hierarchical trajectory analysis technique presented in Chapter 5.2.2. The feature vector of the newly detected event will be undergoing the likelihood test and the POI of the motion trajectory will be analysed with the scene models developed earlier in Chapter 5.2.1.
5.4.1 EXPERIMENTAL RESULTS

In order to validate the effectiveness of the proposed unsupervised event learning and recognition framework, the test video sequences taken from the aforementioned scene were inputted to the system and the recognition results were tabulated in Table 5.2. As shown in the table, the test video sequences comprise of 63% of car events, 33% human events and 4% of unusual events (car moving in the opposite direction). Out of the total human based events, 97% of the events were recognised correctly. While a 100% recognition accuracy has been achieved for car events. Meanwhile, three out of four unusual activities were correctly detected with the proposed anomaly detection algorithm. In general, the proposed event recognition algorithm has recorded a 98% of accuracy for the test videos taken from the same scene.

The proposed autonomous event recognition and anomaly detection system has demonstrated a great accuracy in both car and human events by inputting one video sequence with single event. In order to visualise the detailed analysis of the proposed system, four sample events of different nature were presented as shown in Figure 5.13. The first sample event (Figure 5.13(a)) was “a people walked through the observed scene”, followed by “people picked up a car”, “A vehicle enters the observed scene and occupies an empty parking lot”, and lastly “car passed by the observed scene”. The detailed analysis of the event recognition outcomes of these sample videos are tabulated in Table 5.3.

<table>
<thead>
<tr>
<th>Event</th>
<th>Test videos</th>
<th>Successfully recognized</th>
<th>Precision (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class 1</td>
<td>23</td>
<td>22</td>
<td>96</td>
</tr>
<tr>
<td>Class 2</td>
<td>6</td>
<td>6</td>
<td>100</td>
</tr>
<tr>
<td>Class 3</td>
<td>4</td>
<td>4</td>
<td>100</td>
</tr>
<tr>
<td>Vehicle</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class 4</td>
<td>47</td>
<td>47</td>
<td>100</td>
</tr>
<tr>
<td>Class 5</td>
<td>9</td>
<td>9</td>
<td>100</td>
</tr>
<tr>
<td>Class 6</td>
<td>8</td>
<td>8</td>
<td>100</td>
</tr>
<tr>
<td>Anomaly</td>
<td>4</td>
<td>3</td>
<td>75</td>
</tr>
<tr>
<td>Total</td>
<td>101</td>
<td>99</td>
<td>98</td>
</tr>
</tbody>
</table>

Table 5.2 - Statistical result of the proposed event recognition system.
Figure 5.13 — Sample test videos. (a) A people walked through the observed scene, (b) people picked up a car, (c) A vehicle enters the observed scene and occupies an empty parking lot, and (d) car passed by the observed scene.

<table>
<thead>
<tr>
<th>Event</th>
<th>Ground Truth</th>
<th>Object type</th>
<th>Start Point, $X_s$</th>
<th>End Point, $X_f$</th>
<th>Log-likelihood</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>H1</td>
<td>Human</td>
<td>Entrance/Exit (Human)</td>
<td>Entrance/Exit (Human)</td>
<td>-5.83</td>
<td>H1</td>
</tr>
<tr>
<td>b</td>
<td>H2</td>
<td>Human</td>
<td>Entrance/Exit (Human)</td>
<td>Parking Lot</td>
<td>-10.45</td>
<td>H2</td>
</tr>
<tr>
<td>c</td>
<td>V2</td>
<td>Vehicle</td>
<td>Entrance (Vehicle)</td>
<td>Parking lot</td>
<td>-6.49</td>
<td>V2</td>
</tr>
<tr>
<td>d</td>
<td>V1</td>
<td>Vehicle</td>
<td>Entrance (Vehicle)</td>
<td>Exit (Vehicle)</td>
<td>-0.02</td>
<td>V1</td>
</tr>
</tbody>
</table>

Table 5.3 — Sample event recognition result of the test video with the point level analysis and likelihood test results.

The first event was classified as a human event by the object recognition process based on the appearance features of the detected blob. Thus, a likelihood test of its feature vector against human event classes was applied. The maximum log-likelihood of the feature vector was found at -5.83 which is within the set confidence level at -17. Hence, the detected event was classified as a normal human event. In order to further interpret the event, a point of interest analysis was performed by using the scene model as shown in Figure 5.3(c). The POI matching results are shown in Figure 5.14(a) where the entry point $X_s$ and exit point $X_f$ of the track fall under the entry/exit region. Based on the logical table shown in Table 5.1, the event in Figure 5.13(a) was classified as event “a people walked through the observed scene” by the system as a final decision.

The second event as shown in Figure 5.13(b) was also recognized as human event by the object recognition process and the maximum log-likelihood of its feature vector was computed as -10.45 against the human classes. Since the log-likelihood was within the confidence level, further analysis of the POI was performed. The start point $X_s$ was found at entry/exit zone while the end point $X_f$ was located within parking lot zone. Therefore, the event (b) was interpreted as “people picked up a car” which
matches the ground truth.

![Diagram](image.png)

**Figure 5.14** – Experiment result of the POI analysis for sample events as listed in Table 5.3. (a) Vehicle events and (b) Human events.

For event (c) and (d), both events were classified as vehicle event by the object recognition system and both events passed the likelihood test against vehicle event classes. In the POI analysis, event (c) was eventually labeled as “A vehicle enters the observed scene and occupies an empty parking lot” because the $X_s$ located within the entry zone, while $X_f$ located within the parking lot zone. On the other hand, the event (d) was labeled as “car passed by the observed scene” as the $X_s$ located within the entry zone and $X_f$ located within the exit zone.

In addition, the high level activity recognition as presented in Chapter 4.4.1 can be directly applied to the system to further understand the high level activities such as “pick up” and “drive in” which are common in an outdoor car park facility.

### 5.4.2 DISCUSSIONS

The proposed event learning and recognition approach in this chapter has shown significant improvement in the reasoning of the human initiated events as compared to the method demonstrated in Chapter 4. Again, the system encountered similar problems in the object detection and tracking process due to the undesirable environmental factors (detailed discussions refer to Chapter 4.5.3). However, the feature extraction and transformation technique implemented in this system has shown to significantly
mitigate the negative effects brought by the imperfection in object detection and tracking process. For example, the high frequency noises such as sudden change in the foreground image/object as a result of occlusion or misinterpretation of object as background due to the same colour scheme (refer to Figure 4.19(b) and 4.19(c)) will not have much effect on the feature vector. This is because the feature extraction and transformation technique only takes into consideration the general behavioural pattern of the track rather to analyse the entire data points of the track which are typically noisy.

The automatic scene modelling and decomposition technique proposed in this chapter has shown to work very efficiently without any manual human annotation in the modelling and learning of event at point level. As compared to the manual scene decomposition technique utilized in the previous chapter, the probabilistic scene modelling is more flexible as it is applicable to other scene as long as the scene structure is the same regardless of the topological setting. For example, the system can adapt to any car park scene that has one vehicle entrance and exit zone even though the locations of these entrance and exit zones are different with the one under consideration. Therefore, this system minimizes the effort of human interpretation and supervision.

The HMM activity modelling method proposed in this chapter handles the variations in consecutive data points efficiently by calculating the probability of each state (data point) with the previous state. The HMM modelling technique improves significantly the reasoning of human events as compared to the activity modelling approach presented in Chapter 4. The randomness of human motion was efficiently modelled by using the HMM compared to the PCA based modelling methods which is dominated by the majority patterns in the same class of events. Thus, a wider range of behavioural variations are taken into consideration in HMM. As a result, the accuracy of recognizing human events increases from 73% to 97% with a similar set of testing video data.
5.6 CONCLUSIONS

An event recognition and anomaly detection system based on an unsupervised, hierarchical learning architecture is presented in this chapter. The low level visual properties extracted from the video sequences by applying the proposed object detection and tracking algorithm (refer to Chapter 3) were analysed and learnt by the system for recognising activities in car park surveillance video and detecting unusual event autonomously. The tracked motion trajectory is analysed in different levels to devise a highly descriptive model for all classes of common events. The proposed hierarchical event analysis and learning approach utilises fully the information brought about by the motion trajectory. Specifically, the point level data (entry and exit points) describes the location information of the activity which is highly dependent to the structure of the observed scene. Secondly, the spatial information of the motion trajectory, such as direction/orientation, indicates the pattern of the movement. And lastly, the dynamic information such as velocity and acceleration of the track describes the manner of the object movement.

The system proposed an automated scene modelling technique by utilising the entry and exit points of the motion trajectory extracted from the training data. The method clusters the entry and exit points into logical groups based on the probabilistic measurement. A novel region of interest identification is implemented based on the nature of the events. Two distinct scene decompositions were constructed for human and vehicle targets which are the most common objects observed in an outdoor car park environment. The proposed scene modelling technique is flexible and applicable to the scenes of similar structure regardless of the scene topography.

A multilevel, low dimensional and selective trajectory analysis technique is implemented in this work. A low-dimensional feature vector is constructed by extracting and transforming the raw trajectory points into discretised descriptions in the direction and velocity behavioural patterns of the activity. A selective feature vector approach is applied to events of different object type. As shown in the experiment results, selection of most dominant feature in representing events of different object type has demonstrated higher accuracy due to the removal of higher variations in the less
dominant feature which may deteriorate the representation of events.

The HMM based activity modeling approach provides an efficient way to tackle temporal variations in the feature vector by computing the probability of each state against the neighbouring states. This allows the activity model to capture variations in the time series data due to the randomness of behavioural pattern especially the human motion. The experimental results have shown that the statistical activity modeling technique significantly improves recognition accuracy in human activities. In addition, the event learning framework adopts the unsupervised learning paradigm where k-means clustering technique is employed to classify the feature vectors collected from a set of unlabelled data into finite classes of behavioural pattern. Each class of feature vector will then be trained by the HMM framework to devise the statistical parameters that best describe the event class.

The experimental results have demonstrated an overall accuracy of 98% for both human and vehicle initiated events including normal and abnormal activities. The most significant improvement of the proposed hierarchical event analysis and learning method is reflected in the recognition accuracy of the human events. The accuracy has increased from 73% in the non-statistical activity modeling technique as presented in Chapter 4, to an outstanding accuracy of 98% where similar set of testing surveillance videos were applied. In general, the hierarchical event analysis and learning framework presented in this chapter has shown to effectively model activities in an outdoor car park surveillance video by analysing multilevel information extracted from the raw motion trajectories.
CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 CONCLUSIONS

This thesis presents two distinct object-based event recognition and anomaly detection systems that employ different approaches in the activity modeling and learning strategies. Specifically, this thesis focuses on the surveillance videos in an outdoor car park scene. Detection and recognition of simple events which are composed of single object is considered in this work. Both of the activity understanding and unusual event detection approaches model an event with multi-level descriptors ranging from the object level, spatial level and dynamic level.

In Chapter 3, an object detection and tracking algorithm is implemented to collect the low-level visual properties of an event when there is motion present in the monitored scene. The state-of-the-art adaptive GMM background modeling technique is employed to detect foreground region of interest which provides fundamental input to the activity model. Several noise removal and image enhancement methods are utilized to improve the quality of the detected blob. An object classification framework is presented to classify the detected event into two major classes: (1) the vehicle event and (2) human event. The spatial-temporal information of the moving object in consecutive frames are collected and visualized in the form of motion trajectory.

In Chapter 4, an event representation technique is proposed by exploiting the spatial temporal information extracted from the motion trajectory tracked by the object detection algorithm presented in Chapter 3. Firstly, a scene decomposition process is performed manually to divide the monitored area into several important zones of interest such as entrance, exit and parking bay zones based on the entry and exit points of the motion trajectory collected from the training dataset. Two distinct scene decompositions are established specifically for vehicle and human events due to their
different nature of motion pattern.

Secondly, a feature vector is constructed based on the normalized and scaled x and y axis projections, instantaneous velocity and blob size profiles computed from the motion trajectory of the event. A supervised learning approach is exploited to train the system where a set of labeled data are trained separately by using the Principal Component Analysis (PCA) method to devise the activity models for different classes of activities.

Lastly an event recognition and anomaly detection algorithm is implemented to classify the newly detected events based on the knowledge about the object type based on the proposed object recognition technique, entry and exit points analysis and similarity test of feature vector against the learnt explicit definition of events. In addition, a composite event recognition algorithm is presented to identify high-level activity that involves combination of low-level events by analyzing spatial and temporal relationship between detected simple events.

The second event recognition and anomaly detection framework considered in this thesis takes an alternative approach in both activity modeling and event learning strategy. Specifically, an automatic scene decomposition method is introduced by analyzing the statistical distribution of the entry and exit points extracted from the unlabeled training dataset. A GMM clustering technique is employed to group the entry and exit points into logical regions and the important zones in the observed scene are indentified based on the common behavioral pattern of the known events. The proposed scene decomposition method is applicable for scenes that have similar topological setting, for instance, any outdoor car park facility with only one entrance and exit area.

A feature vector transformation is used to construct low-dimensional and discredited event representation that takes into consideration the absolute direction and instantaneous velocity of the motion pattern from the tracked trajectory. Unsupervised event learning approach is adopted in this framework. The transformed feature vectors of a set of unlabeled training videos are classified into finite classes by using the k-means clustering algorithm. Each class of feature vectors are then trained by HMM
model to compute the probabilistic parameters. The recognition and anomaly detection of new event is accomplished based on the object recognition, POI matching and likelihood test of feature vector.

The effectiveness of the two event recognition and anomaly detection frameworks are carried out with a set of testing videos recorded from the scene. A comprehensive comparison of the two approaches in different perspectives is addressed in this thesis.

6.2 FUTURE WORK

In general, the two event recognition and anomaly detection frameworks presented in this thesis achieved the initial aim where a relatively straightforward scene in outdoor car park is considered. However, the surveillance videos in reality may be much more complex in term of content and topological settings. Furthermore, multiple cameras are normally deployed to fully monitor the entire car park facility. Thus, further enhancement and refinement of the proposed event recognition and anomaly detection systems is necessary.

Extension of the current system to accommodate more complex scene is essential for better applicability of the system to the real surveillance footage. In order to cater for a more sophisticated scene, simultaneous tracking of multiple objects is indispensable. For a complex scene, complex event that involves interactions of multiple objects must be carefully investigated to provide better understanding of the activities. Thus, the future extension of this work will investigate into the tracking of multiple objects and modeling of interactive activities, for example, human-human interactions such as talking, fighting, passing object from one to another, etc., and human-vehicle interaction such as loading/unloading and drop off, and lastly human-external object interaction such as removing/adding object to the scene permanently.
In order to maximize the semantic understanding of the activities, modeling of the body-level actions such as running, jumping, vandalizing, waving, etc. are critical for the interpretation of higher level of activities. To achieve this, a body-level model has to be constructed on top of the motion-level model established in this work.

Furthermore, the adaptability of the model in the long run has not been explicitly investigated in this work. Specifically, the activity model presented in this work does not take into consideration the changes of the behavioral pattern over time, for example rerouting of the current layout or introduction of new type of events may result in false negative detection for normal events. Also, the definitions of normality/abnormality may drift over time and thus significantly affect the accuracy of the event recognition and anomaly detection system in practical settings. Therefore, in future work, an adaptive learning approach will be embedded into the current system to improve the adaptability of the surveillance system.

Though the current work in automated video surveillance system focuses only in a single camera setting, similar framework can be extended to multi camera settings by transforming individual scene of different camera to a global coordinate space. Lastly, in order to evaluate the reliability and robustness of the system, extensive testing of the proposed surveillance system with diverse dataset taken from different car park scenes will be recommended in the future work.
REFERENCES


APPENDIXES

%% get event descriptors
load('ppl_pass','ppl_pass');
load('car_pass','car_pass');
load('ppl_in','ppl_in');
load('ppl_out','ppl_out');
load('car_in','car_in');
load('car_out','car_out');

[ppl_pass_prin] = getprincomp1(ppl_pass);
[car_pass_prin] = getprincomp1(car_pass);
[ppl_in_prin] = getprincomp1(ppl_in);
[ppl_out_prin] = getprincomp1(ppl_out);
[car_in_prin] = getprincomp1(car_in);
[car_out_prin] = getprincomp1(car_out);

eventtype =
['ppl_pass';'car_pass';'ppl_in';'ppl_out';'car_in';'car_out';'anomaly';'noise '];
event_rep(:,:,1) = ppl_pass_prin;
event_rep(:,:,2) = car_pass_prin;
event_rep(:,:,3) = ppl_in_prin;
event_rep(:,:,4) = ppl_out_prin;
event_rep(:,:,5) = car_in_prin;
event_rep(:,:,6) = car_out_prin;

%% Introduction
% This program uses the gaussian mixture models to detect foreground in a video
% After foreground detection, the program performs blob analysis and object tracking

%% Initialization
% Use these next sections of code to initialize the required variables and
% System objects.

hbfr = vision.VideoFileReader('Filename', 'car_out10.avi');

%% % Create color space converter System objects to convert the image from
%% % YCbCr to RGB format.
%% hcssc = vision.ColorSpaceConverter('Conversion', 'YCbCr to RGB');

%% Create a System object to detect foreground using gaussian mixture models.
hof = vision.ForegroundDetector( ... 
    'NumTrainingFrames', 10, ... % only 10 because of short video
    'LearningRate', 0.005, ... 
    'NumGaussians', 3, ... 
    'InitialVariances', (30/255)^2); % initial standard deviation of 30/255

%% Create a blob analysis System object to segment moving objects.
hblob = vision.BlobAnalysis( ... 
    'CentroidOutputPort', true, ... 
    'AreaOutputPort', true, ... 
    'BoundingBoxOutputPort', true, ... 
    'OutputDataType', 'single', ... 
    'NumBlobsOutputPort', false, ... 
    'MinimumBlobAreaSource', 'Property', ... 
    'MinimumBlobArea', 70, ... 
    'MaximumBlobAreaSource', 'Property', ... 
    'MaximumBlobArea', 6000, ... 
    'FillValues', 0, ... 
    'MaximumCount', 2);

%% Create and configure two System objects that insert shapes, one for
%% drawing the bounding box around the cars and the other for drawing the
%% motion vector lines.
hshapeins1 = vision.ShapeInserter( ... 
    'BorderColor', 'Custom', ... 
    'CustomBorderColor', [0 255 0]);
% hshapeins2 = vision.ShapeInserter( ... 
%     'Shape', 'Lines', ... 
%     'BorderColor', 'Custom', ... 
%     'CustomBorderColor', [255 255 0]);

%%
% Create and configure a System object to write the number of objects being
% tracked.
htextins = vision.TextInserter( ... 
    'Text', '%4d', ... 
    'Location', [0 0], ... 
    'Color', [255 255 255], ... 
    'FontSize', 12);

%%
% Create System objects to display the results.

hVideoFg=vision.VideoPlayer('Name','Foreground');
hVideoFg.Position=[350 250 240 180];
hVideoRes=vision.VideoPlayer('Name','Result');
hVideoRes.Position=[50 250 240 180];
figure('Position',[650 250 240 180]);

line_row = 0; % Define region of interest (ROI)

%% Stream Processing Loop
% Create a processing loop to track the cars in the input video. This
% loop uses the previously instantiated System objects.
% When the BinaryFileReader object detects the end of the input file, the loop
% stops.

output = [];
trajectory=[];
velocity=[0 0];
blobarea=[];
event = 0;
frame = 1;
while ~isDone(hbfr)    
    image=step(hbfr);
    y = im2single(image);
    y = y-mean(y(:));

    fg_image = step(hof, y); % Foreground
    fg_image = medfilt2(fg_image);

    %fg_image = bwmorph(fg_image,'close');
    fg_image = bwareaopen(fg_image,50);
    fg_image = bwmorph(fg_image,'close');

    % Estimate the area and bounding box of the blobs in the foreground
    % image.
    [area,centroid, bbox] = step(hblob, fg_image);
    Idx = bbox(1,:) >= line_row; % Select boxes which are in the ROI.
    % Based on dimensions, exclude objects which are not cars. When the
    % ratio between the area of the blob and the area of the bounding box
    % is above 0.4 (40%) classify it as a car.
    ratio = zeros(1, length(Idx));
    ratio(Idx) = single(area(1,Idx))./single(bbox(3,Idx).*bbox(4,Idx));
    ratiob = ratio > 0.2;
    count = int32(sum(ratiob));    % Number of cars
    bbox(:, ~ratiob) = int32(-1);
    if count > 0
        if event == 0
            startingframe = frame;
        end
        event = 1;
        blobarea=[blobarea;area]; %collect object size
        newcentroid=[centroid(2,1),centroid(1,1),centroid(2,2),centroid(1,2)];
        s=size(trajecotory);
        trajectory=[trajectory;newcentroid]; %collect trajectory data
    end
    if s > 0
        % calculate velocity

    end
end

end
x1 = [trajectory(s(1),1),
     trajectory(s(1),2);newcentroid(1,1),newcentroid(1,2)];
distance1 = pdist(x1);
if distance1 > 3 + velocity(s(1),1)
    distance1 = velocity(s(1),1);
end

x2 = [trajectory(s(1),3),
     trajectory(s(1),4);newcentroid(1,3),newcentroid(1,4)];
distance2 = pdist(x2);
if distance2 > 3 + velocity(s(1),2)
    distance2 = velocity(s(1),2);
end
velocity = [velocity; distance1, distance2]; %calculate & collect object's velocity
end
else
    if event > 0
        event = 0;
        endingframe = frame;
        vectorlength = size(velocity);
        normalizedvelocity = resample(double(velocity), 500, vectorlength(1,1));
        normalizedblobarea = resample(double(blobarea), 500, vectorlength(1,1));
        event_descriptor = [filter(0.05, [1 -0.95], normalizedvelocity),
                           filter(0.05, [1 -0.95], single(normalizedblobarea))];
        event_descriptor_nor = zscore(event_descriptor);
        correlation = getcorrcoef(event_descriptor_nor, event_rep);
        startingpoint = trajectory(1,1:2);
        endpoint = trajectory(s(1) + 1,1:2);
        average_area = sum(blobarea)/(s(1)+1);
        event_tag(1,1) = startingframe;
        event_tag(1,2) = endingframe;
        if startingpoint(1) < 10 || startingpoint(1) > 150 || startingpoint(2) < 10 || startingpoint(2) > 110
            event_tag(1,3) = 1;
        else
            event_tag(1,3) = 0;
        end
        if endpoint(1) < 10 || endpoint(1) > 150 || endpoint(2) < 10 || endpoint(2) > 110
            event_tag(1,4) = 1;
        else
            event_tag(1,4) = 0;
        end
        if average_area(1) > 600
            event_tag(1,5) = 1;
        else
            event_tag(1,5) = 0;
        end
        [Y, I] = max(correlation);
        event_tag(1,6) = I;
        event_tag(1,7) = Y;
        [out_event] = getevent(event_tag);
        display(eventtype(out_event,:));
        output = [output;event_tag];
        trajectory=[];
        velocity=[0 0];
        blobarea=[];
% Draw bounding rectangles around the detected cars.
y2 = step(hshapeins1, image, bbox);
% Display the number of cars tracked.
y2(1:15,1:30,:) = 0; % Black background for displaying count
image_out = step(htextins, y2, count);
frame = frame + 1;
for j=1:count
    axis([0 159 0 119], 'xy');
    plot(centroid(2,j),(120-centroid(1,j)),'r.');
    hold on;
    j=j+1;
end
step(hVideoRes, image_out); % Bounding boxes around cars
step(hVideoFg, fg_image); % Foreground
i=i+1;
end

Appendix A.1 – MATLAB program for event recognition and anomaly detection.

function [principal_features] = getprincomp(data)
    [row col sample] = size(data);
    velocity_1 = [];
    velocity_2 = [];
    trajectory_x1 = [];
    trajectory_y1 = [];
    trajectory_x2 = [];
    trajectory_y2 = [];
    area_1 = [];
    area_2 = [];

    for i = 1:sample
        trajectory_x1 = [trajectory_x1 data(:,1,i)];
        trajectory_y1 = [trajectory_y1 data(:,2,i)];
        trajectory_x2 = [trajectory_x2 data(:,3,i)];
        trajectory_y2 = [trajectory_y2 data(:,4,i)];
        velocity_1 = [velocity_1 data(:,5,i)];
        velocity_2 = [velocity_2 data(:,6,i)];
        area_1 = [area_1 data(:,7,i)];
        area_2 = [area_2 data(:,8,i)];
    end

    [c trajectory_x1] = princomp(trajectory_x1);
    [c trajectory_y1] = princomp(trajectory_y1);
    [c trajectory_x2] = princomp(trajectory_x2);
    [c trajectory_y2] = princomp(trajectory_y2);
    [c velocity_1] = princomp(velocity_1);
    [c velocity_2] = princomp(velocity_2);
    [c area_1] = princomp(area_1);
    [c area_2] = princomp(area_2);

    principal_features = [trajectory_x1(:,1) trajectory_y1(:,1) trajectory_x2(:,1)
                          trajectory_y2(:,1) velocity_1(:,1) velocity_2(:,1) area_1(:,1) area_2(:,1)];

    principal_features = zscore(principal_features);

Appendix A.2 – MATLAB function for computing the principal component representations of event.
function [correlation] = getcorrcoef(input, event_rep)
    correlation = [];
    for i=1:6
        temp = corrcoef(input,event_rep(:,:,i));
        correlation = [correlation temp(2,1)];
    end

Appendix A.3 – MATLAB function for evaluating the similarity of feature vector by obtaining the correlation coefficient.

function [event] = getevent(event_tag)
    event_mask = [1 1 0 1; 1 1 2; 1 0 0 3; 0 1 0 4; 1 0 1 5; 0 1 1 6];
    for i = 1:6
        if event_tag(3:6) == event_mask(i,:)
            event = i;
            break;
        else
            event = 0;
        end
    end
    if event == 0 || event_tag(7) < 0.6
        event = 7;
    end
    if event_tag(2) - event_tag(1) <= 10
        event = 8;
    end

Appendix A.4 – MATLAB function to determine event tag by matching the POI distribution.

%clear

%% Introduction
% This program uses the gaussian mixture models to detect foreground in a video
% After foreground detection, the program performs blob analysis and object tracking

%% Initialization
% Use these next sections of code to initialize the required variables and
% System objects.
hbfr = vision.VideoFileReader('Filename', 'car_out10.avi');

%%
% Create color space converter System objects to convert the image from
% YCbCr to RGB format.
hcsc = vision.ColorSpaceConverter('Conversion', 'YCbCr to RGB');

%%
% Create a System object to detect foreground using gaussian mixture models.
hof = vision.ForegroundDetector(...
    'NumTrainingFrames', 10, ... % only 5 because of short video
    'LearningRate', 0.006, ...
    'NumGaussians', 3, ...
    'InitialVariance', (30/255)^2); % initial standard deviation of 30/255

%%
% Create a blob analysis System object to segment moving objects.
hblob = vision.BlobAnalysis( ...
    'CentroidOutputPort', true, ...
    'AreaOutputPort', true, ...
'BoundingBoxOutputPort', true, ...
'OutputDataType', 'single', ...
'NumBlobsOutputPort', false, ...
'MinimumBlobAreaSource', 'Property', ...
'MinimumBlobArea', 80, ...
'MaximumBlobAreaSource', 'Property', ...
'MaximumBlobArea', 6000, ...
'FillValues', 0, ...
'MaximumCount', 2);

%%
% Create and configure two System objects that insert shapes, one for
% drawing the bounding box around the cars and the other for drawing the
% motion vector lines.
htshapeins1 = vision.ShapeInserter( ...
    'BorderColor', 'Custom', ...
    'CustomBorderColor', [0 255 0]);
htshapeins2 = vision.ShapeInserter( ...
    'Shape', 'Lines', ...
    'BorderColor', 'Custom', ...
    'CustomBorderColor', [255 255 0]);

%%
% Create and configure a System object to write the number of objects being
% tracked.
htextins = vision.TextInserter( ...
    'Text', '%4d', ...
    'Location', [0 0], ...
    'Color', [255 255 255], ...
    'FontSize', 12);

%%
% Create System objects to display the results.
hVideoFg = vision.VideoPlayer('Name', 'Foreground');
hVideoFg.Position = [350 250 240 180];
hVideoRes = vision.VideoPlayer('Name', 'Result');
hVideoRes.Position = [50 250 240 180];

figure('Position', [650 250 240 180]);
line_row = 0; % Define region of interest (ROI)

%% Stream Processing Loop
% Create a processing loop to track the cars in the input video. This
% loop uses the previously instantiated System objects.
% When the BinaryFileReader object detects the end of the input file, the loop
% stops.
trajectory = []; velocity = [0 0]; Blobarea = []; event = 0;
while ~isDone(hbfr)
    image = step(hbfr);
    y = im2single(image);
    y = y - mean(y(:));

    fg_image = step(hof, y); % Foreground
    fg_image = medfilt2(fg_image);
    %fg_image = bwmorph(fg_image,'close');
    fg_image = bwareaopen(fg_image, 50);

    % Estimate the area and bounding box of the blobs in the foreground
    % image.
    [area, centroid, bbox] = step(hblob, fg_image);
    Idx = bbox(1,:) >= line_row; % Select boxes which are in the ROI.
    % Based on dimensions, exclude objects which are not cars. When the
    % ratio between the area of the blob and the area of the bounding box
    % is above 0.4 (40%) classify it as a car.
    ratio = zeros(1, length(Idx));
    ratio(Idx) = single(area(1,Idx))./single(bbox(3,Idx).*bbox(4,Idx));
    ratiob = ratio > 0.2;
    count = int32(sum(ratiob)); % Number of cars
    bbox(:, ~ratiob) = int32(-1);
if count > 0
    event = 1;
    blobarea = [blobarea; area]; % collect object size
newcentroid = [centroid(2,1), centroid(1,1), centroid(2,2), centroid(1,2)];
s = size(trajectory);
trajectory = [trajectory; newcentroid]; % collect trajectory data
if s > 0
    % calculate velocity
    x1 = [trajectory(s(1,:), 1),
    trajectory(s(1,:), 2); newcentroid(1,1), newcentroid(1,2)];
    distance1 = pdist(x1);
    if distance1 > 3 + velocity(s(1,:), 1)
        distance1 = velocity(s(1,:), 1);
    end
    x2 = [trajectory(s(1,:), 3),
    trajectory(s(1,:), 4); newcentroid(1,3), newcentroid(1,4)];
    distance2 = pdist(x2);
    if distance2 > 3 + velocity(s(1,:), 2)
        distance2 = velocity(s(1,:), 2);
    end
    velocity = [velocity; distance1, distance2]; % calculate & collect object's velocity
end
else
    if event > 0
        event = 0;
        vectorlength = size(velocity);
        normalizedvelocity = resample(double(velocity), 500, vectorlength(1,1));
        normalizedtrajectory = resample(double(trajectory), 500, vectorlength(1,1));
        normalizedblobarea = resample(double(blobarea), 500, vectorlength(1,1));
        event_descriptor = [normalizedtrajectory, filter(0.05, [1 -0.95], normalizedvelocity), filter(0.05, [1 -0.95], single(normalizedblobarea))];
        event_descriptor_nor = zscore(event_descriptor);
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        % perform event recognition algorithm here
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    else
        % idle
    end
end
% Draw bounding rectangles around the detected cars.
y2 = step(hshapeins1, image, bbox);
% Display the number of cars tracked.
y2(1:15,1:30,:) = 0; % Black background for displaying count
image_out = step(htextins, y2, count);
for j = 1:count
    AXIS([0 159 0 119], 'xy');
    plot(centroid(2,j), (120-centroid(1,j)),'r.'); % hold on;
end
step(hVideoRes, image_out); % Bounding boxes around cars
step(hVideoFg, fg_image); % Foreground
i = i+1;
end
Appendix A.5 – MATLAB program for training explicit definition of events.
function [feature, polar, segment_points] = getFeatureVector(trajectory)

    [vector_length, column] = size(trajectory);
    segment_size = fix(vector_length / 8);

    segment_points(:,1) = [trajectory(1,1) trajectory(1,2);
                         trajectory(segment_size,1) trajectory(segment_size,2);
                         trajectory(segment_size*2,1) trajectory(segment_size*2,2);
                         trajectory(segment_size*3,1) trajectory(segment_size*3,2);
                         trajectory(segment_size*4,1) trajectory(segment_size*4,2);
                         trajectory(segment_size*5,1) trajectory(segment_size*5,2);
                         trajectory(segment_size*6,1) trajectory(segment_size*6,2);
                         trajectory(segment_size*7,1) trajectory(segment_size*7,2)];

    polar(:,1) = [sqrt((segment_points(2,1) - segment_points(1,1)).^2 +
                     (segment_points(2,2) - segment_points(1,2)).^2);
                  sqrt((segment_points(3,1) - segment_points(2,1)).^2 +
                     (segment_points(3,2) - segment_points(2,2)).^2);
                  sqrt((segment_points(4,1) - segment_points(3,1)).^2 +
                     (segment_points(4,2) - segment_points(3,2)).^2);
                  sqrt((segment_points(5,1) - segment_points(4,1)).^2 +
                     (segment_points(5,2) - segment_points(4,2)).^2);
                  sqrt((segment_points(6,1) - segment_points(5,1)).^2 +
                     (segment_points(6,2) - segment_points(5,2)).^2);
                  sqrt((segment_points(7,1) - segment_points(6,1)).^2 +
                     (segment_points(7,2) - segment_points(6,2)).^2)];

    feature(:,1) = polar(:,1)./5;
    feature(:,1) = ceil(feature(:,1));

    for j=1:7
        dy = segment_points(j+1,2) - segment_points(j,2);
        dx = segment_points(j+1,1) - segment_points(j,1);

        if dy > 0 && dx > 0
            angle = (180/pi) * atan(dy/dx);
        elseif dy > 0 && dx < 0
            angle = 180 - ((-180/pi) * atan(dy/dx));
        elseif dy < 0 && dx > 0
            angle = (180/pi) * atan(dy/dx);
        else
            angle = -180 + (180/pi) * atan(dy/dx);
        end

        polar(j,2) = angle;

        if angle <= 20 && angle >= -20
            feature(j,2) = 1;
        elseif angle <= 60 && angle > 20
            feature(j,2) = 2;
        elseif angle <= 100 && angle > 60
            feature(j,2) = 3;
        elseif angle <= 140 && angle > 100
            feature(j,2) = 4;
        elseif angle <= 180 && angle > 140
            feature(j,2) = 5;
        elseif angle < -20 && angle > -60
            feature(j,2) = 9;
        elseif angle <= -60 && angle > -100
            feature(j,2) = 8;
        elseif angle <= -100 && angle > -140
            feature(j,2) = 7;
        elseif angle <= -140 && angle > -180
            feature(j,2) = 6;
        else
        end
    end
end

Appendix A.6 – MATLAB program for feature vector extraction and transformation.
% Scene Modelling by using Gaussian Mixture Distribution
%% Initial Configuration
entryvector=[];
exitvector=[];

%% Get input vector
for i=1:102
    filename=['car',int2str(i),'.mat'];
    x=importdata(filename);
    entryvector=[entryvector; x.trajectory(1,1:2)];
    exitvector=[exitvector;x.trajectory(size(x.trajectory,1),1:2)];
end

entryvector=double(entryvector);
exitvector=double(exitvector);
entrydis=gmdistribution.fit(entryvector,3);
exitdis=gmdistribution.fit(exitvector,3);
idx = cluster(entrydis,entryvector);
idx1= cluster(exitdis,exitvector);
cluster1 = entryvector(idx == 1,:);
cluster2 = entryvector(idx == 2,:);
cluster3 = entryvector(idx == 3,:);
h1 = scatter(cluster1(:,1),cluster1(:,2),10,'r.');
n1= scatter(cluster1(:,1),cluster1(:,2),10,'r.');
h2 = scatter(cluster2(:,1),cluster2(:,2),10,'g.');
h3= scatter(cluster3(:,1),cluster3(:,2),10,'b.');

exitvector=double(exitvector);
exitvector=double(exitvector);
cluster4 = exitvector(idx1 == 1,:);
cluster5 = exitvector(idx1 == 2,:);
cluster6 = exitvector(idx1 == 3,:);
h4 = scatter(cluster4(:,1),cluster4(:,2),10,'b+');
h5 = scatter(cluster5(:,1),cluster5(:,2),10,'k+');
h6 = scatter(cluster6(:,1),cluster6(:,2),10,'g+');
axis([0 159 0 119])

Appendix A.7 – MATLAB function for automatic scene modeling.

clear
load('carHMMmodel');
load('pplHMMmodel');

%% Introduction
% This program uses the gaussian mixture models to detect foreground in a video
% After foreground detection, the program performs blob analysis and object tracking

%% Initialization
% Use these next sections of code to initialize the required variables and
% System objects.
filename = 't101.avi';
hbfr = vision.VideoFileReader('Filename', filename);

% Create color space converter System objects to convert the image from
% YCbCr to RGB format.
hcsc = vision.ColorSpaceConverter('Conversion', 'YCbCr to RGB');

% Create a System object to detect foreground using gaussian mixture models.
hof = vision.ForegroundDetector(...
    'NumTrainingFrames', 10, ...
    'NumGaussians', 3, ...
    'InitialVariance', (20/255)^2); % initial standard deviation of 30/255

% Create a blob analysis System object to segment moving objects.
hblob = vision.BlobAnalysis(...
    'CentroidOutputPort', true, ...
    'AreaOutputPort', true, ...
    'BoundingBoxOutputPort', true, ...
    'OutputDataType', 'single', ...
'MinimumBlobAreaSource', 'Property', ...
'MinimumBlobArea', 40, ...
'MaximumBlobAreaSource', 'Property', ...
'MaximumBlobArea', 6000, ...
'FillValues', 0, ...
'MaximumCount', 1);

%%
%% Create and configure two System objects that insert shapes, one for
%% drawing the bounding box around the cars and the other for drawing the
%% motion vector lines.
$hshapeins1 = vision.ShapeInserter(...
    'BorderColor', 'Custom', ...
    'CustomBorderColor', [0 255 0]);
$hshapeins2 = vision.ShapeInserter(...
    'Shape', 'Lines', ...
    'BorderColor', 'Custom', ...
    'CustomBorderColor', [255 255 0]);

%%
%% Create and configure a System object to write the number of objects being
%% tracked.
$htextins = vision.TextInserter(...
    'Text', '%4d', ...
    'Location', [0 0], ...
    'Color', [255 255 255], ...
    'FontSize', 12);

%%
%% Create System objects to display the results.
%nVideoFg=vision.VideoPlayer('Name','Foreground');
%nVideoFg.Position=[350 250 240 180];
%nVideoRes=vision.VideoPlayer('Name','Result');
%nVideoRes.Position=[50 250 240 180];

figure('Position',[650 250 240 180]);

line_row = 0; % Define region of interest (ROI)

%% Stream Processing Loop
%% Create a processing loop to track the cars in the input video. This
%% loop uses the previously instantiated System objects.
%% When the BinaryFileReader object detects the end of the input file, the loop
%% stops.
trajectory=[]; % Video frame
blobarea=[]; % Blob area
event = 0; % Event

while ~isDone(hbfr) % Image
    image = step(hbfr);
    y = im2single(image);
    y = y-mean(y(:));
    fg_image = step(hof, y); % Foreground
    fg_image = medfilt2(fg_image);
    fg_image = bwmorph(fg_image,'close');
    %fg_image = bwareaopen(fg_image,8);
    % Estimate the area and bounding box of the blobs in the foreground
    % image.
    [area, centroid, bbox] = step(hblob, fg_image);
    %Idx = bbox(1,:) >= line_row; % Select boxes which are above the ROI.
    ratio = zeros(1, length(Idx));
    ratio(Idx) = single(area(1,Idx))./single(bbox(3,Idx).*bbox(4,Idx));
    ratio = single(area(1,1))./single(bbox(3,1).*bbox(4,1));
    ratio = ratio > 0.2;
    count = int32(sum(ratio));
    bbox(:, ~ratio) = int32(-1);
    if count > 0
        event = 1;
blobarea=[blobareas;area]; %collect object size
newcentroid=[centroid(2,1),120 - centroid(1,1)];

s=size(trajectory);
trajectory=[trajectory;newcentroid]; %collect trajectory data
end

% Draw bounding rectangles around the detected cars.
y2 = step(hshapeins1, image, bbox);
% Display the number of cars tracked.
y2(1:15,1:30,:) = 0; % Black background for displaying count
image_out = step(htextins, y2, count);

for j=1:count
    AXIS([0 159 0 119], 'xy');
    plot(centroid(2,j),(120-centroid(1,j)),'r.);
    hold on;
    j=j+1;
end

step(hVideoRes, image_out); % Bounding boxes around cars
step(hVideoFg, fg_image); % Foreground
i=i+1;
end

(vector_length,column) = size(trajectory);
meanArea = mean(blobarea);
entrypoints = trajectory(1,:);
exitpoints = trajectory(vector_length,:);
[event_vector,polar,sampled] = getFeatureVector(trajectory);

if meanArea > 300
    loglik(1) = dhmm_logprob(event_vector(:,2)', car_prior1, car_trans1, car_obs1);
    loglik(2) = dhmm_logprob(event_vector(:,2)', car_prior2, car_trans2, car_obs2);
    loglik(3) = dhmm_logprob(event_vector(:,2)', car_prior3, car_trans3, car_obs3);
else
    loglik(4) = dhmm_logprob(event_vector(:,1)', ppl_prior1, ppl_trans1, ppl_obs1);
    loglik(5) = dhmm_logprob(event_vector(:,1)', ppl_prior2, ppl_trans2, ppl_obs2);
    loglik(6) = dhmm_logprob(event_vector(:,1)', ppl_prior3, ppl_trans3, ppl_obs3);
end

close all
delete(hVideoRes);
delete(hVideoFg);

Appendix A.8 – MATLAB program for event recognition and anomaly detection system using the hierarchical trajectory analysis and learning approach.