

Department of Computing

**Video Foreground Extraction for Mobile Camera Platforms**

Wilson Suryajaya Leoputra

**This thesis is presented for the Degree of  
Doctor of Philosophy  
of  
Curtin University of Technology**

December 2009

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university. To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgement has been made.

---

Wilson Suryajaya Leoputra

---

Date

# Video Foreground Extraction for Mobile Camera Platforms

by

Wilson Suryajaya Leoputra

Submitted to the Department of Computing  
in December, 2009 in partial fulfillment of the  
requirements for the degree of  
Doctor of Philosophy

## Abstract

Foreground object detection is a fundamental task in computer vision with many applications in areas such as object tracking, event identification, and behavior analysis. Most conventional foreground object detection methods work only in a stable illumination environments using fixed cameras. In real-world applications, however, it is often the case that the algorithm needs to operate under the following challenging conditions: drastic lighting changes, object shape complexity, moving cameras, low frame capture rates, and low resolution images. This thesis presents four novel approaches for foreground object detection on real-world datasets using cameras deployed on moving vehicles.

The first problem addresses passenger detection and tracking tasks for public transport buses investigating the problem of changing illumination conditions and low frame capture rates. Our approach integrates a stable SIFT (Scale Invariant Feature Transform) background seat modelling method with a human shape model into a weighted Bayesian framework to detect passengers. To deal with the problem of tracking multiple targets, we employ the Reversible Jump Monte Carlo Markov Chain tracking algorithm. Using the SVM classifier, the appearance transformation models capture changes in the appearance of the foreground objects across two consecutive frames under low frame rate conditions. In the second problem, we present a system for pedestrian detection involving scenes captured by a mobile bus surveillance system. It integrates scene localization, foreground-background separation, and pedestrian detection modules into a unified detection framework. The scene localization module performs a two stage clustering of the video data. In the first stage, SIFT Homography is applied to cluster frames in terms of their structural similarity, and the second stage further clusters these aligned frames according to consistency in illumination. This produces clusters of images that are differential in viewpoint and lighting. A kernel density estimation (KDE) technique for colour and gradient is then used to construct background models for each image cluster, which is further used to detect candidate foreground pixels. Finally, using a hierarchical template matching approach, pedestrians can be detected.

In addition to the second problem, we present three direct pedestrian detection methods that extend the HOG (Histogram of Oriented Gradient) techniques (Dalal and Triggs,

2005) and provide a comparative evaluation of these approaches. The three approaches include: **a)** a new histogram feature, that is formed by the weighted sum of both the gradient magnitude and the filter responses from a set of elongated Gaussian filters (Leung and Malik, 2001) corresponding to the quantised orientation, which we refer to as the Histogram of Oriented Gradient Banks (HOGB) approach; **b)** the codebook based HOG feature with branch-and-bound (efficient subwindow search) algorithm (Lampert *et al.*, 2008) and; **c)** the codebook based HOGB approach.

In the third problem, a unified framework that combines 3D and 2D background modelling is proposed to detect scene changes using a camera mounted on a moving vehicle. The 3D scene is first reconstructed from a set of videos taken at different times. The 3D background modelling identifies inconsistent scene structures as foreground objects. For the 2D approach, foreground objects are detected using the spatio-temporal MRF algorithm. Finally, the 3D and 2D results are combined using morphological operations.

The significance of these research is that it provides basic frameworks for automatic large-scale mobile surveillance applications and facilitates many higher-level applications such as object tracking and behaviour analysis.

---

# ACKNOWLEDGEMENTS

---

In many ways, this thesis could not have been possible without the contributions of many individuals over the course of the PhD.

First and foremost, I wish to express my deepest gratitude to Prof. Svetha Venkatesh and Assoc. Prof. Tele Tan who gave me the opportunity to do a PhD on the exciting field of foreground object detection, who taught me how to conduct scientific research, and who patiently guided me through the process of writing my dissertation. Their constant support and enthusiasm have been the biggest encouragement through my studies.

I would also like to thank all my friends at IMPCA with whom I have shared the highs and lows, Kevin Chee, Ba Tu Troung, Thorsten Kühnapfel, Annika Kramer, and Simon Moncrieff. Thanks to Patrick Peursum for the stimulating discussions and inspiring opinions. Thanks to Wilson Walters for his assistance on the data collections.

Special thanks to Australian Research Council (ARC), who provided funding for the mobile surveillance project that this thesis is part of, and Curtin University for providing me with a scholarship without which I could not have continued to study.

I am deeply grateful to my parents and my sisters, Hugo Leoputra, Catheryn Tsjiam, and Novianca Leoputra, who have always been there for me. Mom, Dad and Sis, completing this degree (among many other things) would not have been possible without your love and support.

Finally, my heartfelt appreciation goes to Devi Cyintia Onggo, who makes every day better.

---

# CONTENTS

---

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Aims . . . . .	3
1.2	Approach and Contributions . . . . .	3
1.2.1	A Bayesian Framework for Detecting and Tracking Passengers in Bus Environment (Chapter 3) . . . . .	3
1.2.2	Pedestrian Detection for Mobile Bus Surveillance (Chapter 4) . . . . .	4
1.2.3	Comparative Evaluation of Pedestrian Detection Methods for Mobile Bus Surveillance (Chapter 5) . . . . .	5
1.2.4	Scene Change Detection (Chapter 6) . . . . .	5
1.3	Structure of the Thesis . . . . .	6
<b>2</b>	<b>Literature Review</b>	<b>8</b>
2.1	Low Level Features . . . . .	9
2.1.1	Filter Based Features . . . . .	9
2.1.2	Image Features . . . . .	10
2.1.3	Feature Detector (Interest Region Detector) . . . . .	10
2.1.4	Feature Descriptors (Histogram Based Feature) . . . . .	14
2.2	Feature Comparison and Classification Methods . . . . .	21
2.2.1	Histogram Feature Comparison . . . . .	22
2.2.2	Classification Methods . . . . .	26
2.2.2.1	Support Vector Machine . . . . .	26
2.2.2.2	Cascaded Adaboost . . . . .	27
2.2.2.3	Graphical Models . . . . .	27
2.3	Background Subtraction . . . . .	27
2.4	Pedestrian Detection . . . . .	30
2.5	Object Tracking . . . . .	32
2.6	Image Registration and 3D Scene Reconstruction . . . . .	33
2.7	Chapter Summary . . . . .	34
<b>3</b>	<b>Passenger Monitoring System</b>	<b>35</b>
3.1	Offline Background Patches Estimation . . . . .	36
3.2	Bayesian Formulation for Passenger Detection . . . . .	38
3.2.1	Seat Occupancy Detection . . . . .	40
3.2.1.1	SIFT Codebook Background Modelling (Constructing $\Gamma$ ) . . . . .	40

3.2.1.2	Computing Probability of Seat Occupancy . . . . .	41
3.2.1.3	Estimating Local Keypoint Probability – $P(\mathbf{S} \mathbf{F}, \Gamma)$ . . . . .	43
3.2.1.4	Estimating Seat Occupancy Probability – $P(h \mathbf{S})$ . . . . .	43
3.2.2	Human Detection . . . . .	44
3.3	Object Tracking . . . . .	46
3.3.1	Bayesian Filtering Distribution for Object Tracking . . . . .	49
3.3.2	Independent Particle Filter . . . . .	50
3.3.3	Proposal Distribution with Trajectory Model for LFR Tracking . . . . .	51
3.3.4	Particle Filter for Interacting Objects . . . . .	53
3.3.4.1	State Definitions for Multi-object Tracking . . . . .	53
3.3.4.2	Interaction Model . . . . .	53
3.3.4.3	Dynamic Model for Variable Numbers of Targets . . . . .	54
3.3.5	Reversible Jump MCMC Algorithm for Variable Numbers of Targets . . . . .	55
3.3.6	Move-Specific Proposal Distribution and Acceptance Ratio . . . . .	59
3.3.7	Observation Model (Appearance Transformation Model) . . . . .	62
3.3.7.1	Appearance Transformation Feature . . . . .	62
3.3.7.2	Observation Model using The Appearance Transformation Model . . . . .	64
3.3.8	Computing MAP Estimation . . . . .	64
3.4	Experimental Results . . . . .	65
3.4.1	Perth Central Area Transit (CAT) Bus Dataset and Camera Param- eters . . . . .	65
3.4.2	Ground truth . . . . .	67
3.4.3	System Output (Detection) . . . . .	68
3.4.4	Evaluation Criteria . . . . .	69
3.4.5	Experiments (Detection) . . . . .	69
3.4.5.1	Experiment I: Empty Bus with Lighting Changes (Video 1) . . . . .	71
3.4.5.2	Experiment II: Single Passenger with Lighting Changes (Video 2) . . . . .	73
3.4.5.3	Experiment III: Crowded Passengers at Night Time (Video 3) . . . . .	76
3.4.5.4	Experiment IV: Crowded Passengers with Lighting Changes (Video 4) . . . . .	79
3.4.5.5	Experiment V: Comparison With Other Detection Ap- proaches . . . . .	83
3.4.6	Experiments (Tracking) . . . . .	85
3.4.6.1	Experiment VI: Non-Overlapping People Tracking (Indoor Environment) . . . . .	86
3.4.6.2	Experiment: Seat-switch Detection . . . . .	88

3.4.6.3	Experiment VII: Different Lighting Scenario (video 2) . . .	90
3.4.6.4	Experiment VIII: Crowded Scenario . . . . .	92
3.4.6.5	Experiment IX: Similar Appearance Scenario (video 7) . .	95
3.4.6.6	Experiment X: Quantitative Comparison For Different Tracking Methods Against Low Frame Rate . . . . .	97
3.5	Chapter Summary . . . . .	99
<b>4</b>	<b>Pedestrian Detection For Mobile Bus Surveillance</b>	<b>101</b>
4.1	Scene Localization . . . . .	103
4.1.1	Location Based Clustering (L1) . . . . .	103
4.1.2	Intensity Based Clustering (L2) . . . . .	104
4.2	Combined Foreground Background Separation and Pedestrian Detection . .	106
4.2.1	KDE on Colour . . . . .	106
4.2.2	KDE on Gradient . . . . .	108
4.2.3	Combined KDE Colour and Gradient . . . . .	108
4.2.4	Pedestrian Detection . . . . .	108
4.3	Experimental Results . . . . .	109
4.3.1	Perth Dataset . . . . .	110
4.3.2	Scene Localization Results . . . . .	111
4.3.3	Comparison Between KDE and GMM Methods . . . . .	113
4.3.4	System Performance . . . . .	114
4.4	Chapter Summary . . . . .	115
<b>5</b>	<b>Comparative Evaluation of Pedestrian Detection Methods</b>	<b>117</b>
5.1	Different Pedestrian Detection Methods . . . . .	117
5.1.1	Method 1: Histogram of Oriented Gradient (HOG) . . . . .	117
5.1.2	Method 2: Histogram of Oriented Gradient Banks (HOGB) . . . . .	118
5.1.3	Method 3: Codebook HOG with ESS (CHOG) . . . . .	119
5.2	Datasets and Evaluation Criteria . . . . .	121
5.3	Experiments . . . . .	121
5.3.1	Implementation Details . . . . .	122
5.3.2	Detection Performance . . . . .	123
5.3.3	Speed Performance . . . . .	124
5.4	Chapter Summary . . . . .	124
<b>6</b>	<b>Automatic Video Scene Change Detection</b>	<b>125</b>
6.1	Overview of The Algorithm . . . . .	126
6.2	3D Scene Change Detection . . . . .	127
6.2.1	Time-ordered 3D Scene Construction . . . . .	127
6.2.1.1	Structure From Motion (SFM) . . . . .	129



6.2.1.2	Iterative Closest Point (ICP) . . . . .	129
6.2.2	3D Cloud Point Subtraction . . . . .	132
6.2.3	Spatial Clustering to Identify Foreground Objects and Eliminate Noise	132
6.3	Stationary/Non-stationary Object Probability . . . . .	135
6.4	2D Change Detection . . . . .	137
6.4.1	2D Data Alignment . . . . .	137
6.4.2	Spatio-Temporal MRFs for Scene Change Detection . . . . .	138
6.4.2.1	MRF Model . . . . .	139
6.4.2.2	Likelihood . . . . .	141
6.4.3	Estimating the Posterior Probability . . . . .	141
6.4.4	Observation Model ( $\theta$ ) . . . . .	142
6.4.4.1	Discarding the Uniform Regions . . . . .	144
6.4.4.2	HOG Codebook Observation Model . . . . .	144
6.5	Combining Results from 3D and 2D Scene Change Detection . . . . .	146
6.6	Experimental Results . . . . .	146
6.6.1	Scene Changes Dataset . . . . .	146
6.6.2	Pre-processing . . . . .	147
6.6.3	Parameter Settings . . . . .	150
6.6.4	Evaluation Criteria . . . . .	152
6.6.5	Experiments . . . . .	153
6.6.5.1	Experiment I: Single Video Changes . . . . .	154
6.6.5.2	Experiment II: Single Video Changes . . . . .	157
6.6.5.3	Experiment III: Multiple Video Changes . . . . .	160
6.6.6	Discussion . . . . .	162
6.7	Chapter Summary . . . . .	163
<b>7</b>	<b>Conclusion</b> . . . . .	<b>164</b>
7.1	Future Work . . . . .	165
<b>A</b>	<b>Copyright Clearance</b> . . . . .	<b>182</b>

---

# LIST OF FIGURES

---

1.1	Real-world challenges for people detection. . . . .	2
2.1	The general framework to solve computer vision problems. . . . .	9
2.2	Dictionaries of different filter banks. ©IEEE . . . . .	10
2.3	The responses from different image region detectors (Mikołajczyk <i>et al.</i> , 2005). ©2005 IEEE . . . . .	14
2.4	State of art feature descriptors. ©IEEE . . . . .	16
2.5	Different sampling strategies. . . . .	17
2.6	The SIFT algorithm (Lowe, 2004). . . . .	18
2.7	Spatioграм (Birchfield and Rangarajan, 2005). ©2005 IEEE . . . . .	21
2.8	Colour correlogram (Huang <i>et al.</i> , 1997). . . . .	21
2.9	Co-occurrence matrix (Chang and Krumm, 1999). ©1999 IEEE . . . . .	21
2.10	A general framework for ordinal-based image correspondence (Cramariuc <i>et al.</i> , 2000). ©2000 IEEE . . . . .	25
2.11	Haar features: black and white areas denote the negative and positive weights, respectively. . . . .	30
3.1	Proposed framework for passenger detection. . . . .	37
3.2	Seat alignment process. . . . .	38
3.3	A graphical model representation for the passenger detection. . . . .	38
3.4	An example of SIFT codebook background modelling on a seat inside a bus. . . . .	41
3.5	Human configuration model based on ellipse fitting. . . . .	45
3.6	Output of seat occupancy probability. . . . .	47
3.7	Output of head detection probability. . . . .	47
3.8	A graphical model representation for target tracking. . . . .	50
3.9	The passenger trajectory model. . . . .	52
3.10	A graphical model representation for multiple targets tracking. . . . .	53
3.11	Interaction model for passengers inside the bus. . . . .	54
3.12	The four different moves for RJMCMC: <i>birth</i> , <i>death</i> , <i>update</i> , and <i>swap</i> . . . . .	57
3.13	Homography results. . . . .	65
3.14	Camera setup. . . . .	66
3.15	Ground truth for video 8. . . . .	67
3.16	Comparison graph between the ground truth (dark colour bar) and the detection results (light colour bar) for video 8. . . . .	68
3.17	The detected passengers in video 8. . . . .	69

3.18	Detection evaluation criteria. . . . .	70
3.19	Snapshot for video 1 (CAT 2). . . . .	71
3.20	Snapshot comparisons of the SIFT codebook background model and the GMM background subtractions. . . . .	72
3.21	Snapshots of video 2. . . . .	73
3.22	Snapshot images of the passenger detection in video 2 (CAT 1). . . . .	74
3.23	Comparison of the detection results (light colour bar) and the ground truth (dark colour bar). . . . .	75
3.24	Detection results of video 2. . . . .	75
3.25	Snapshots for video 3 (CAT 3). . . . .	76
3.26	Snapshot images of the passenger detection in video 3. . . . .	77
3.27	Comparison of the detection results (light colour bar) and the ground truth (dark colour bar). . . . .	78
3.28	Detection results of video 3. . . . .	78
3.29	Snapshots for video 4 (CAT 2). . . . .	79
3.30	Snapshot images of the passenger detection in video 4. . . . .	80
3.31	Comparison of the detection results (light colour bar) and the ground truth (dark colour bar). . . . .	81
3.32	Detection results of video 4. . . . .	81
3.33	Detection results for video 2, 5, 6, 7, 8, 10, and 11. . . . .	82
3.34	Comparison of the passenger detection results on the selected CAT datasets. . . . .	85
3.35	Experimental environment. . . . .	86
3.36	Example of the annotated ground truth for indoor people segmentation. . . . .	87
3.37	Examples of the positive training samples used in the bus experiment. . . . .	89
3.38	Snapshots of the passenger tracking in video 2 (CAT 1). . . . .	91
3.39	Comparison between the tracking results and the ground truth. . . . .	91
3.40	Examples of the Particle Filter with HSV appearance model fails to track the passengers when the lighting changes. . . . .	92
3.41	Snapshots of passenger tracking in video 5 (CAT 1). . . . .	93
3.42	Comparison between the tracking results and the ground truth. . . . .	93
3.43	Examples of frames when the MCMC tracking technique fails to track pas- senger 2 under occlusion situations. . . . .	94
3.44	Examples of frames when Particle Filter method fails to track passenger 3 under the LFR situations. . . . .	94
3.45	Some of the tracking results of video 6 (CAT 1). . . . .	95
3.46	Comparison between the tracking results and the ground truth. . . . .	95
3.47	Some of the tracking results of video 7 (CAT 1). . . . .	96
3.48	Comparison between the tracking results and the ground truth. . . . .	97

3.49	Comparison projection errors graphs between four different tracking algorithms under low frame rate situation . . . . .	98
3.50	Examples of frames where other methods fail under the LFR situations. . .	98
4.1	A VirtualObserver system. . . . .	102
4.2	Overview of the proposed pedestrian detection algorithm. . . . .	102
4.3	Scene localization. . . . .	103
4.4	Example of misalignment problem. . . . .	107
4.5	Foot-to-head calibration using Homography method. . . . .	109
4.6	Perth dataset: (a) camera setup and (b) description of the dataset. . . . .	110
4.7	Evaluation criteria for comparing the ground truth (solid) bounding box with the detected candidate bounding box (dash). . . . .	111
4.8	Scene localization results. . . . .	112
4.9	Visualising the view-sites on GoogleEarth (2009). . . . .	113
4.10	Comparison results of the foreground detections between Stauffer-and-Grimson's based and our proposed KDE-based method. . . . .	114
4.11	System performance. . . . .	115
4.12	Pedestrian detection results on Perth datasets. . . . .	116
5.1	Foot-to-head calibration using Homography method. . . . .	119
5.2	The nine banks of elongated Gaussian filters. . . . .	119
5.3	Codebook for pedestrians. . . . .	120
5.4	(a) Examples of the INRIA dataset and (b–c) the Perth dataset that were used for training and testing in our experiments. . . . .	121
5.5	Performance of HOGB: (a) detection rate (b) false positives per image. . .	123
5.6	Evaluation of different pedestrian detection methods. . . . .	124
6.1	Transformed data: time-ordered 3D reconstructed scene, motion-field, and 2D aligned images data. . . . .	126
6.2	Overview of our proposed scene change detection framework. . . . .	128
6.3	3D cloud point subtraction based on the spatial point similarity. . . . .	133
6.4	SNN clustering algorithm to detect foreground and eliminate noise. . . . .	135
6.6	Motion-field data. . . . .	136
6.5	View frustum points. . . . .	136
6.7	Example of two scenes with objects of different velocity. . . . .	137
6.8	Spatial connections. . . . .	139
6.9	Temporal connection. . . . .	141
6.10	Discarding the uniform regions. . . . .	144
6.11	Spatio-temporal MRF segmentation results at different iterations. . . . .	145
6.12	Camera setup and map where the video is recorded. . . . .	147

6.13	Dataset 1 consists of two video. (Top) video 1.1. (Bottom) video 1.2. . . . .	148
6.14	Dataset 2 consists of five video. From top to bottom: video 2.1 to 2.5. . . . .	148
6.15	Dataset 3 consists of five video. From top to bottom: video 3.1 to 3.5. . . . .	149
6.16	Pre-processing examples. . . . .	149
6.17	Evaluation using different features. . . . .	151
6.18	ROC curve for different local features. . . . .	152
6.19	Evaluation criteria. . . . .	153
6.20	Dataset 1. . . . .	155
6.21	Scene change detection results. . . . .	156
6.22	SFM-HOG scene change detection results. . . . .	156
6.23	Time ordered 3D data and camera motion for experiment 2. . . . .	158
6.24	Comparison of different motion-field data for video set 2. (a) and (c) contain only stationary scene, whilst (b) and (d) contain moving foreground objects. 158	
6.25	3D SNN subtraction results. (Note that, for explanation purposes the SNN results in (b) are annotated. The actual results do not contains any annotations) . . . . .	159
6.26	Scene change detection results. . . . .	159
6.27	SFM-HOG scene change detection results. . . . .	160
6.28	Time-ordered 3D scene reconstruction and its camera motion of dataset 3. . . . .	161
6.29	Motion-fields for video in dataset 3. . . . .	161
6.30	3D scene change detection results. (Note that, for explanation purposes the SNN results in (a), (b), and (c) are annotated. The actual results do not contains any annotations) . . . . .	161
6.31	Scene change detection results. . . . .	162
6.32	SFM-HOG scene change detection results. . . . .	162

---

# LIST OF TABLES

---

1.1	Challenges in the three applications that form the focus of this thesis. . . .	2
2.1	Feature detectors. . . . .	11
2.2	Common feature detectors and their performance. . . . .	13
2.3	Different type of histograms. . . . .	15
2.4	The strength of SIFT object recognition algorithm (Lowe, 2004). . . . .	19
2.5	Other SIFT-like descriptors. . . . .	20
2.6	Metrics and their computation. . . . .	23
2.7	Texture measures. . . . .	26
2.8	Summary of different background subtraction methods. . . . .	29
2.9	Pedestrian datasets. . . . .	31
2.10	Different pedestrian detection approaches. . . . .	31
2.11	Image registration methods. . . . .	34
3.1	Difference between independent PF, MCMC, and RJMCMC for multiple target tracking. . . . .	59
3.2	SVM feature vectors. . . . .	63
3.3	Descriptions for the CAT dataset. . . . .	66
3.4	Comparison results for video 1. . . . .	72
3.5	Summary of different method performances on the CAT video. . . . .	83
3.6	System performance of the proposed appearance transformation model. . .	88
3.7	Summarised seat switching detection results for all test video in CAT 1. . .	97
4.1	Scene localization accuracy tested on two different datasets. . . . .	112
4.2	Comparison performance results between Stauffer-Grimson and the proposed KDE background subtraction. . . . .	113
5.1	Speed performance for the 4 approaches. . . . .	124
6.1	Alignment of multiple 3D scenes using the Iterative Closest Point method. .	131
6.2	Descriptions of the dataset acquired from vehicle. . . . .	147
6.3	Evaluation results. . . . .	163

---

## PUBLISHED WORK

---

This thesis is based upon several works that have been published over the course of the author's PhD, listed as follows in chronological order:

- Leoputra, W. S., Venkatesh, S., and Tan, T. (2008). Passenger Monitoring in Moving Bus Video. In *IEEE International Conference on Control, Automation, Robotics and Vision*. **2**, 741-744. December 17-20, Hanoi, Vietnam.
- Leoputra, W. S., Venkatesh, S. and Tan, T. (2008). Pedestrian Detection for Mobile Bus Surveillance. In *IEEE International Conference on Control, Automation, Robotics and Vision*. **2**, December 17-20, Hanoi, Vietnam.
- Leoputra, W. S., Venkatesh, S. and Tan, T. (2009). Comparative Evaluation of Pedestrian Detection Methods for Mobile Bus Surveillance. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, April 20-24, Taiwan.

---

# CHAPTER 1

## INTRODUCTION

---

Most conventional computer vision problems involve building an autonomous surveillance system with fixed cameras to detect, track, and identify object behaviours. As the scale of the environment increases, it is not feasible to have all areas monitored by cameras due to cost and efficiency (Lipton *et al.*, 2002). This changes the focus from building surveillance systems using fixed cameras to building surveillance systems with mobile cameras, enabling monitoring of events in large and complex environments.

A recent technology, the VirtualObserver (Greenhill and Venkatesh, 2006), allows collections of large quantities of video data from mobile vehicle cameras, offering efficient footage retrieval, and ease of visualisation. However, with the large amount of data, this kind of surveillance is “at most partially automated” (Bradshaw *et al.*, 1997), as it is not possible for security personnel to monitor all data simultaneously and identify important events, which may lead to vital information being overlooked.

Foreground object detection is one of the fundamental problems in this surveillance domain. It identifies objects of interest from the video data for the purposes of higher level processing. A reliable people detection system is useful for identity recognition, threat prevention, or analysis. In city streets, pedestrian detection system can provide useful information for events such as road accidents. Scene change detection in an urban city can be used to monitor environmental changes over time. Thus, foreground object detection in these real-world applications is the focus of this thesis.

What makes foreground object detection in these applications difficult is the limitless variability in the image that arises from viewpoint and global illumination changes, deformation of articulated or non-rigid objects, intra-class appearance variations, the presence of occlusion, and background clutter as shown in Figure 1.1 (Lazebnik *et al.*, 2006). This is generally caused by two factors: (1) the nature of the application, for example, fixed vs mobile cameras, outdoor vs indoor environments, high vs low frame rates, and (2) the type of the foreground objects, for example, human vs non-human objects, or rigid vs non-rigid objects. In foreground object detection, these issues directly translate to the design of





Figure 1.1: Real-world challenges for people detection.

Challenges	Applications		
	Passenger detection & tracking	Pedestrian detection	Scene change detection
Foreground object	Upper body parts of human	Full profile of the human (head, body, and limb)	General objects (cars, vegetation, buildings)
Low frame rate	✓	✓	
Dynamic scenes		✓	✓
Lighting changes	✓	✓	✓
Crowd	✓	✓	✓
Non-rigid objects	✓	✓	✓

Table 1.1: Challenges in the three applications that form the focus of this thesis.

*low level features* and *classification methods*. The low level features capture distinctive attributes from the local image regions, whilst the classification methods solve the problem using mathematical models.

In this thesis, we investigate foreground object detection frameworks to address three challenging problems shown in Table 1.1. Each framework involves designing the low level feature and mathematical model that is suitable to deal with the problems in the specific application.

## 1.1 Aims

Previous works on foreground extraction, (Stauffer and Grimson, 2000; Elgammal *et al.*, 2000; Javed *et al.*, 2002; Kim *et al.*, 2005), cannot be applied to this data due to the strict assumptions that the images have to be captured by fixed cameras and have similar background lighting. Similarly, the direct pedestrian detection methods (Papageorgiou and Poggio, 1999; Dalal and Triggs, 2005) often encounter major problems, such as high false positives. Thus, the aim of this thesis is primarily focused on:

- The design of distinctive and robust features for each problem.
- The integration of multi-model approaches to perform the foreground object detection.
- The comparative performance evaluation with other state-of-art methods to study the effectiveness of the proposed framework on real world datasets.

## 1.2 Approach and Contributions

This thesis makes two main contributions to the field of computer vision – (1) the development of robust foreground object detection frameworks to handle challenging applications and (2) the evaluation of each system with real-world datasets. The contributions and their significance are detailed as follows.

### 1.2.1 A Bayesian Framework for Detecting and Tracking Passengers in Bus Environment (Chapter 3)

The first major contribution of this thesis is the development and demonstration of a Bayesian framework for passenger detection and tracking, which operates inside a moving bus environment. The detection framework combines background-modelling (bottom-up) and direct foreground detection (top-down) in a weighted Bayesian formulation. The bottom-up approach models the background as a set of Scale Invariant Feature Transform (SIFT) codebooks capturing different illumination changes, while the passenger shape is modelled using elliptical shape templates. In addition, we propose the Reversible Jump Monte Carlo Markov Chain (RJMCMC) tracking framework with a novel appearance

transformation model to track multiple passengers inside the bus. The proposal distribution of RJMCMC is modelled using the passenger trajectory model. This framework offers several advantages:

- The SIFT codebook background model is robust to different lighting conditions and the human elliptical shape is suitable to model the upper body parts of the passengers.
- The Bayesian framework provides an efficient way of combining the background and foreground modelling.
- A proposal distribution that incorporates the *passenger trajectory model* to allow prediction of object movements according to the bus environment. The *passenger trajectory model* defines allowable seat transitions for a foreground object for two consecutive frames under the low frame rate situations. Thus, it allows the tracker to explore larger search spaces and, at the same time, eliminates impossible state transitions.
- An *appearance transformation model* that learns the similarity measure for a pair of objects under low frame rate and different lighting situations.
- The RJMCMC tracking algorithm allows tracking of dynamically moving passengers (people entering and leaving the bus) and copes with occlusions.

### 1.2.2 Pedestrian Detection for Mobile Bus Surveillance (Chapter 4)

The second contribution of the thesis is the development of a system for pedestrian detection in scenes captured by mobile bus surveillance cameras in city streets. Our approach integrates scene localization, foreground-background separation, and pedestrian detection modules into a unified detection framework. The scene localization module transforms sequences of video data into arrays of images grouped as “view sites”. A view site is a “unique region” storing images captured from different times, but having *highly similar background structure* and *lighting* (i.e. they can be aligned in fixed-camera-manner), which can then be used for foreground-background separation. The kernel density estimation (KDE) methods for colour and gradient are then used to construct the background model for each view site to detect foreground objects. Finally, using a hierarchical template matching approach on the foregrounds from KDE, pedestrians can be identified. This framework has several advantages:

- The construction of the view sites allows moving scenes to be aligned in a fixed camera manner, which can be used for background modelling purposes. With the large amount of data, we are able to generate a rich profile of spatially aligned images for any view site along the bus route.
- The KDE background subtraction can be used to eliminate false positives, improving the overall detection performance.

### 1.2.3 Comparative Evaluation of Pedestrian Detection Methods for Mobile Bus Surveillance (Chapter 5)

The third contribution of the thesis is twofold: (1) the extensions of pedestrian detection algorithms based on the HOG feature (Dalal and Triggs, 2005) and (2) the comparative performance evaluation of the proposed pedestrian detection algorithms against the HOG approach. The proposed methods and their advantages are outlined below:

- A new histogram feature that is formed by the weighted sum of both the gradient magnitude and the filter responses from a set of elongated Gaussian filters (Leung and Malik, 2001) corresponding to the quantised orientation, called Histogram of Oriented Gradient Banks (HOGB) approach. The performance evaluation results show that the HOGB approach has a comparable performance compared to the HOG.
- To overcome the issues of computational complexity in search, we integrate code-books based on the HOG and HOGB descriptions with the recent branch-and-bound (efficient sub-window search) algorithm (Lampert *et al.*, 2008), which results in faster pedestrian detections.

### 1.2.4 Scene Change Detection (Chapter 6)

The final contribution of the thesis is the use of 3D information to model the background scene and 2D spatio-temporal MRFs model for video based segmentation to detect scene changes. This has several benefits:

- The 2D spatio-temporal MRF background modelling allows robust and consistent segmentation results by combining information over the space and time. In addition,

the inference problem is efficiently solved using the Gibbs sampling algorithm.

- Using the 3D scene reconstruction method, we can recover the depth information of the scene. This has benefits over the 2D approach in clustering different classes of objects more accurately and identifying the inconsistent clusters as the scene changes.

### 1.3 Structure of the Thesis

This thesis is organised as follows. In Chapter 2, a review of the related work in the fields of object detection and multiple target tracking is presented. Existing low level features and classification methods are examined to provide motivation for the development of the detection frameworks in this thesis. This is followed by a detailed overview of previous work closely related to this thesis.

Chapter 3 presents a surveillance system for detecting and tracking onboard passengers on public transport vehicles. The detection system is formulated using the Bayesian framework, which is decomposed into seat occupancy and passenger model detection to detect these passengers. In the second part of this chapter, the RJMCMC tracking algorithm is presented along with the passenger trajectory and appearance transformation models. The tracker’s ability to handle occlusions, dynamic number of targets, low frame rates, and changing appearance is also described. Experiments on real bus footage are conducted to demonstrate the usefulness of the proposed framework.

Chapter 4 describes our proposed pedestrian detection framework for front-facing bus cameras. It consists of two main components: background modelling followed by pedestrian detection. The background modelling approach involves scene alignment and foreground segmentation. The segmentation results are then used to detect pedestrians. Experiments are conducted to demonstrate the improvement of the proposed framework over other direct pedestrian detection approaches.

Chapter 5 presents three direct pedestrian detection methods that extend the HOG approach (Dalal and Triggs, 2005) and their comparative performance evaluation in terms of detection accuracy and speed.

Chapter 6 presents the proposed scene change detection for a moving vehicle’s camera. This chapter starts by describing the 3D scene reconstruction and 3D background mod-

elling. Then, it describes the 2D spatio-temporal background modelling followed by the combination of the two approaches. Experiments using urban city street datasets are conducted and evaluated to show the feasibility of the proposed method.

Finally, Chapter 7 concludes the thesis and formulates some directions for future work.

---

## CHAPTER 2

# LITERATURE REVIEW

---

Most computer vision algorithms involve design of low level features into a high level mathematical model, as shown in Figure 2.1. The three fundamental techniques in computer vision include:

1. Low level feature extraction that transforms the raw image into computer-processable data, in the form of statistical or numerical data.
2. Feature comparison techniques to measure similarities and differences between features.
3. Classification methods (mathematical tools) to solve the problem.

In Sections 2.1 and 2.2, we provide an overview of different low level features, feature comparison, and classification methods. We then review relevant works related to object detection, multiple-target tracking, and image registration. We classify the object detection methods into two categories: background subtraction and direct foreground detection. Section 2.3 covers different ways to construct background models from sequences of images and discusses their performance in terms of memory, speed, and accuracy. Section 2.4 provides an overview of various direct foreground detection methods based on features and classification methods, followed by Section 2.5 which discusses the background of multi-target tracking methods. Section 2.6 covers several image registration and 3D scene reconstruction methods. Finally, a summary of the chapter is presented in Section 2.7.

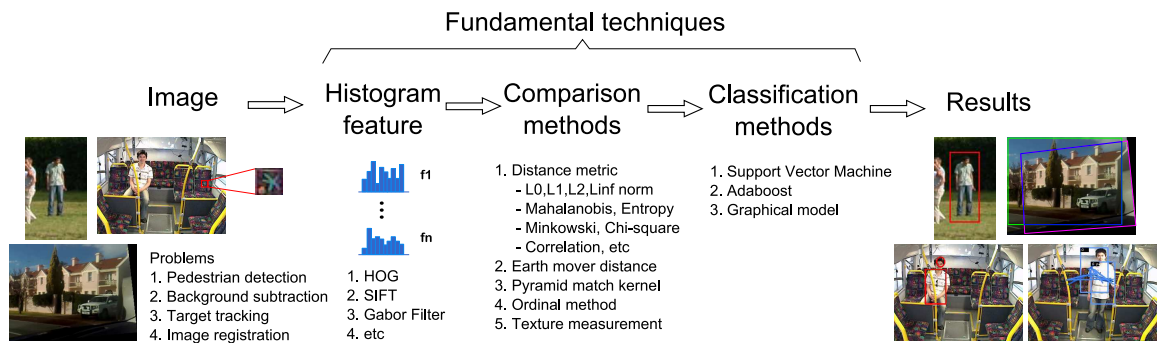


Figure 2.1: The general framework to solve computer vision problems.

## 2.1 Low Level Features

Low level features can be classified into two categories: filter based features and image features. Filter based features are designed to find certain patterns in the image using convolution. Image features, on the other hand, take an image and transform it into a set of interest regions descriptors. It normally has two components: a detector that extracts the interest regions and a descriptor that stores distinctive information around the regions.

### 2.1.1 Filter Based Features

Local regions in an image can often be explained by shape patterns, ranging from simple to complex. In literature, researchers have proposed different dictionaries of complex filters to respond to these patterns. Schmid (2001) introduced rotationally symmetrical filter-banks suitable for detecting circle or riddled texture patterns. Leung and Malik (2001) propose a set of elongated Gaussian filters that respond well to bar shapes of different orientations. Freeman and Adelson (1991) propose steerable filter banks to capture directional derivative responses. Similarly, Gabor filters (Jain and Farrokhnia, 1991) are often used to detect repetitive directional texture patterns. Oliva and Torralba (2001) introduce Gist filters that are suitable to describe scenery images. Simoncelli and Farid (1996) design steerable wedge filters for local orientation and junction analysis. Schafalitzky and Zisserman (2001); Baumberg (2000) propose other complex filters to detect various “ring type” patterns. Recently, Reisert *et al.* (2007) propose non-linear filters called holomorphic filters, which have rotational equivariant properties. Figure 2.2 shows the dictionaries of different filter banks.



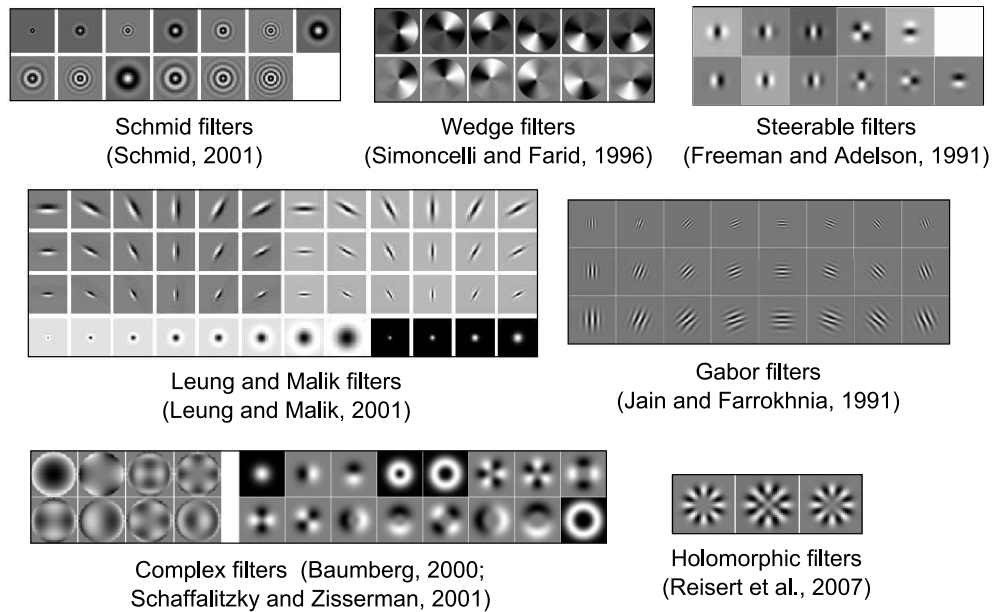


Figure 2.2: Dictionaries of different filter banks. ©IEEE

### 2.1.2 Image Features

Another way to describe local image regions is to use image features, also known as region descriptors. The basic idea is to first detect a collection of interest regions, which is also called the *feature detector*. Then, for each interest region, an invariant *descriptor* is constructed. Sections 2.1.3 and 2.1.4 review various feature detectors and descriptors along with their strengths and weaknesses. According to Tuytelaars and Mikolajczyk (2008), the desired properties of image features include: (a) local invariance (rotation, scaling, cropping, shift); (b) robustness (distinctive, quantity, accurate, repeatability); (c) subjective relevance (illumination, object pose, background clutter, occlusion, intra-class appearance, view point); and (d) effective representation (low dimensionality, well-defined distance metric, and computational efficiency).

### 2.1.3 Feature Detector (Interest Region Detector)

In literature, feature detectors can be classified into three different classes: corner, blob, and edgel detectors. Table 2.1 summarises some of the detectors. The Hessian corner detector (Beaudet, 1978) searches for strong derivatives in two orthogonal directions. The corners are identified by the determinant of the Hessian matrix. The Harris corner detec-

Corner detector	Region detector	Edgelet detector
Hessian (Beaudet, 1978)	Harris-Laplace, Hessian-Laplace (Mikolajczyk and Schmid, 2002)	Edge based region (EBR) (Tuytelaars and Gool, 2004)
Harris (Harris and Stephens, 1988)	Harris-Affine, Hessian-Affine (Mikolajczyk and Schmid, 2004)	Pairs of adjacent segment (PAS) (Ferrari <i>et al.</i> , 2008)
Laplacian-of-Gaussian (LoG) (Lindeberg, 1998)	Maximally Stable Extremal Regions (Matas <i>et al.</i> , 2002)	Contour fragment (Shotton <i>et al.</i> , 2005)
Difference-of-Gaussian (DoG) (Lowe, 2004)	MSER colour (Forssén, 2007)	Boundary fragment model (Opelt <i>et al.</i> , 2006)
	Entropy based region (salient regions) (Kadir <i>et al.</i> , 2004)	Shape model (Leordeanu <i>et al.</i> , 2007)
	Intensity based region (IBR) (Tuytelaars and Gool, 2004)	Edgelet feature (Wu and Nevatia, 2005)
		Gestalt descriptor (Bileschi and Wolf, 2007)

Table 2.1: Feature detectors.

tor (Harris and Stephens, 1988) algorithm is based on computing the second moment or autocorrelation matrix and can be used to determine whether a pixel/point in an image belongs to edge, corner, or is located in a uniform intensity region. It defines a pixel as a corner when the local neighbourhoods have two main directions (eigenvectors). The Laplacian-of-Gaussian (Lindeberg, 1998) computes interest points, by searching for local maxima in the scale space of Laplacian-of-Gaussian. Lowe (2004) detects interest points using the Difference of Gaussian (DoG). The algorithm searches for stable corner features over a multi-scale image pyramid. It defines a pixel as a corner if it belongs to extrema points.

In blob based detectors, Mikolajczyk and Schmid (2002) propose the Harris-Laplace detectors, which are an extension to the Harris corner detector. The algorithm detects Harris corners at multiple scales and selects points at which the Laplacian measure attains local extrema (maxima and minima). Using a similar process and replacing the Harris detector with Hessian detector, we obtain the Hessian-Laplace features. Mikolajczyk and Schmid (2004) further extend the Harris and Hessian detectors to include affine invariant properties to obtain the Hessian/Harris Affine feature. The algorithm can be summarised in four iterative steps: (1) detect multi-scale Harris corner points; (2) select the scale automatically with the Laplacian; (3) adapt an affine neighbourhood shape using the second order moment matrix; (4) repeat the estimation until convergence. Matas *et al.* (2002) propose a Maximally Stable Extremal Regions (MSER) feature that has similar intuition to the watershed segmentation algorithm (Beucher and Meyer, 1993). The algorithm finds extremal regions that stay stable over large threshold ranges. Forssén (2007) extend the

MSER to work on colour images. Tuytelaars and Gool (2004) propose two features, namely edge-based regions (EBR) and intensity-based regions (IBR). The edge based region constructs the parallelogram formed by selecting local extrema of interest points and edges. The intensity based region fits an ellipse to the intensity extrema regions along predefined rays. The salient regions detector (Kadir *et al.*, 2004) identifies local region maxima based on the entropy method.

Table 2.2 summarises the strengths and weaknesses of each detector. According to the performance evaluation carried out by Mikolajczyk and Schmid (2005); Mikolajczyk *et al.* (2005), all detectors give relatively good performance because most of them are invariant to intensity, rotation, scale, and affine, except the Hessian and Harris corner detectors. The evaluation criteria is defined by the repeatability rate (percentage of corresponding points) against the overlap error, i.e. two points/regions are corresponding if the location error is small and the intersection area is large. In the experiment, they show that MSER gives the best performance in many cases, while Hessian-Affine obtains the second best performance. MSER and EBR are well adapted to structured scenes, whilst Harris-Affine and Hessian-Affine are well adapted to textured scenes. On the other hand, Kadir's salient region detectors generally give low performance in terms of the repeatability. The other disadvantage is related to the slow computation. Finally, Hessian and Harris Affine return more corner features than other detectors. Figure 2.2 shows the different responses obtained from different feature detectors.

Detectors	Intuition	Detect	Invariant				Advantages	Disadvantages
			Intensity	Rotation	Scale	Affine		
Hessian	Strong derivatives in two orthogonal directions	corner	✓	✓			detect corners	not scale and affine invariant
Hessian-Laplace	Stable Hessian corners in multiple scales.	blob	✓	✓	✓		affine invariant	not scale invariant
Hessian-Affine	Affine Hessian-Laplace	blob	✓	✓	✓	✓	respond well to textured scenes	
Harris	Local neighbourhoods that has two main directions (eigenvectors).	corner	✓	✓			detect corners	not scale and affine invariant
Harris-Laplace	Stable Harris corners in multiple scales.	blob	✓	✓	✓		affine invariant	not scale invariant
Harris-Affine	Affine Harris Laplace	blob	✓	✓	✓	✓	stable under image blurring	
LoG	Laplacian	corner	✓	✓	✓	✓	respond more on edges than the determinant of Hessian	
DoG	Stable corners obtained from difference of Gaussian in pyramid.	corner	✓	✓	✓		scale and affine invariant	structured flat images.
MSER	Maximal region	blob	✓	✓	✓	✓	efficient, high repeatability	sensitive to image blur.
EBR	Edge based region	edge	✓	✓	✓	✓	respond well in structured scenes.	depends on presence of edges
IBR	Search maximum using image ray	blob	✓	✓	✓	✓	accurate regions	
Salient Region	Maximum entropy combined with interscale saliency	blob	✓	✓	✓	✓	suitable for object recognition	slow to extract, limited number of regions.

Table 2.2: Common feature detectors and their performance.

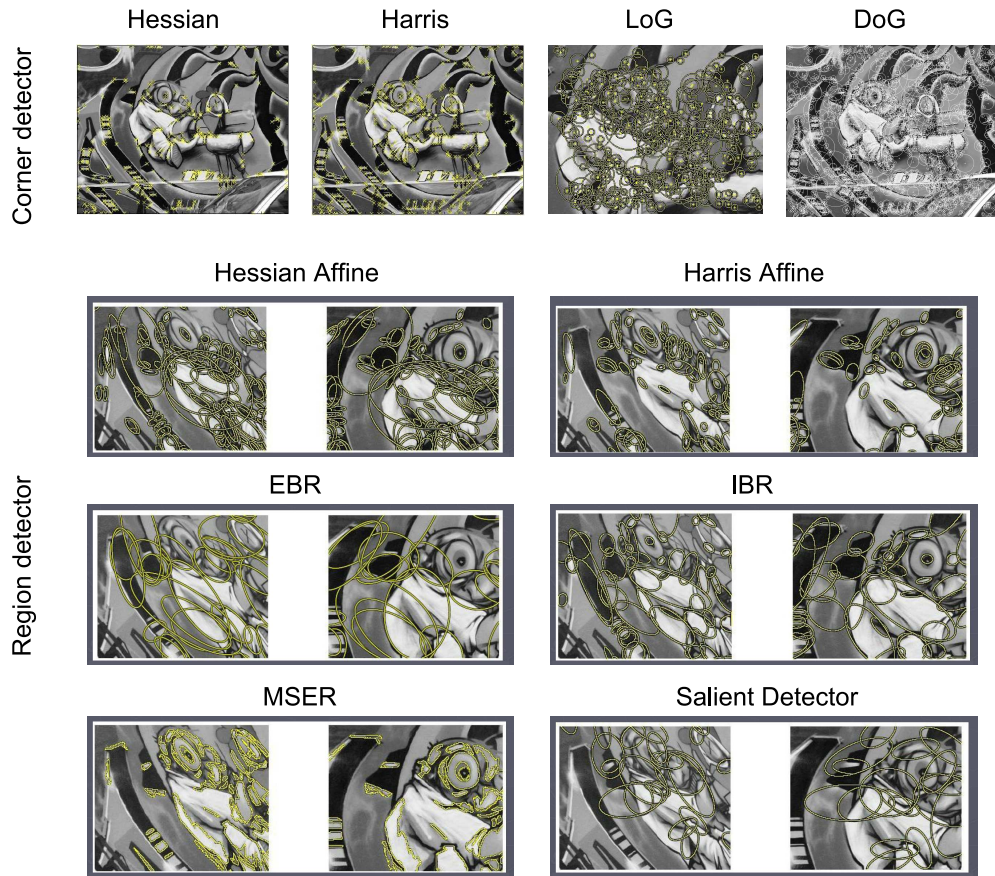


Figure 2.3: The responses from different image region detectors (Mikolajczyk *et al.*, 2005). ©2005 IEEE

#### 2.1.4 Feature Descriptors (Histogram Based Feature)

To capture the characteristics of local image regions, feature descriptors are often designed to be distribution based, i.e. the local image regions are usually transformed into a histogram. Table 2.3 summarises various existing histogram features in the literature along with their applications. The spin image (Lazebnik *et al.*, 2005) is a 2D histogram of distances from the center vs the intensity value. Similarly, the RIFT descriptor (Lazebnik *et al.*, 2005) is a 2D histogram of the distance from the center point vs the orientation in the log-polar axis. The SIFT descriptor (Lowe, 2004) is a 3D histogram of oriented gradients, where the contribution to the orientation bins is weighted by the gradient magnitude and a Gaussian window overlaid over the region. The shape context (Belongie *et al.*, 2000) operates on a set of points sampled from the object shape contours. Again using a log-polar grid, the algorithm creates a 2D histogram descriptor counting the number of points that fall into each orientation bin. The Geometric blur operator (Berg and Malik, 2001)

Authors	Histogram feature	Application
<i>Descriptor based histogram</i>		
Lowe (2004)	Localized orientation histogram (SIFT)	Object matching, etc.
Dalal and Triggs (2005)	Dense orientation histogram	Pedestrian detection
Grauman and Darrell (2005)	Pyramid Histogram (Pyramid Match Kernel)	Object detection
Bosch <i>et al.</i> (2007)	Pyramid HOG	Object detection
Belongie <i>et al.</i> (2000)	Polar bin histogram (Shape context)	Shape correspondence
Lazebnik <i>et al.</i> (2005)	Spin Image and RIFT descriptor	Scene recognition
Ling and Jacobs (2005)	Geodesic-intensity histogram (GIH)	Deformable objects
Berg and Malik (2001)	Geometric Blur	Template matching
Mikolajczyk and Schmid (2005)	GLOH	Object detection
Burghouts and Geusebroek (2009)	Local colour invariant histogram	Object detection
Levi and Weiss (2004)	Edge orientation histogram (EOH)	Face detection
<i>Filter based histogram</i>		
Schneiderman (2000)	Localized histograms of wavelet coefficient	3D object detection
Leung and Malik (2001)	Gaussian elongated histogram	Texture recognition
Winn <i>et al.</i> (2005)	Texton histogram	Object categorization
Ekvall and Kragic (2005)	Receptive field co-occurrence histogram	Object detection
Schiele and Crowley (1996)	Receptive field histogram	Object recognition
<i>Histogram with spatial information</i>		
Birchfield and Rangarajan (2005)	Spatioqram	Head tracking
Huang <i>et al.</i> (1997)	Correlogram	Object matching
Chang and Krumm (1999)	Colour co-occurrence histogram	Object recognition
Pass and Zabih (1996)	Colour coherent vector (CCV) histogram	Image retrieval
Nilsson <i>et al.</i> (2008)	Mapogram (a special form of spatioqram)	Tracking
Wang and Makedon (2003)	R-Histogram	Image retrieval
Miyajima and Ralescu (1994)	Angle histograms	Object detection
Chalechale <i>et al.</i> (2004)	Angular radial histogram	Image retrieval
Qin and Gao (2005)	Angular radial partitioning intensity histogram	Image retrieval
<i>Others</i>		
Swain and Ballard (1991)	Colour histogram	Tracking
Laptev (2009)	Boosted histogram	Object detection
Deselaers <i>et al.</i> (2006)	Sparse histogram	Tracking
Javed <i>et al.</i> (2005)	Cumulative histogram	Object correspondence

Table 2.3: Different type of histograms.

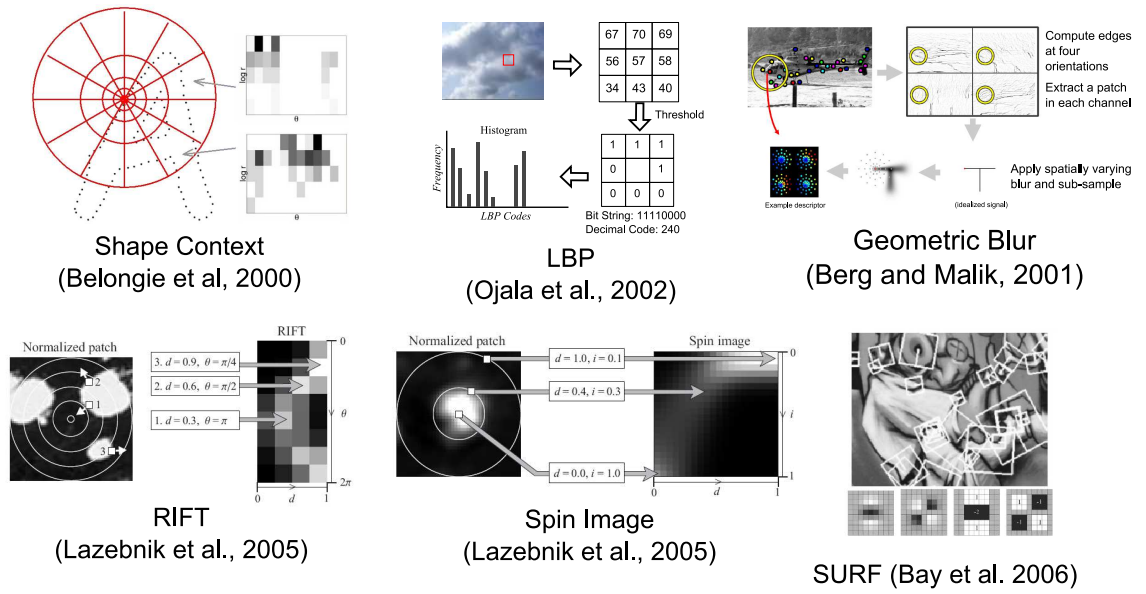


Figure 2.4: State of art feature descriptors. ©IEEE

creates a histogram descriptor of spatially non-uniform blurs of the image region. The local binary pattern (Ojala *et al.*, 2002) operator creates a 1D histogram that counts the pixel changes around the neighbourhood, and is invariant to monotonic gray level changes. Figure 2.4 shows construction of different feature descriptors.

### Different sampling strategies

Apart from the choice of the histogram feature, the sampling strategy is another important factor that determines the system performance. Existing sampling strategies include dense sampling (every pixel at different scale), rectangle or block sampling (SIFT), log polar sampling (shape context), circular sampling (GLOH), and multi-scale model (pyramid match kernel). Figure 2.5 shows the different sampling strategies. According to Dalal and Triggs (2005), dense histogram of oriented gradients created using overlapped rectangle sampling gives the best performance to detect pedestrians, as compared to using circular HOG. For multi-scale methods, the minimum image scale at which patches can be sampled has considerable influence on the results because the vast majority of patches or interest points typically occur at the finest scales (Nowak *et al.*, 2006).

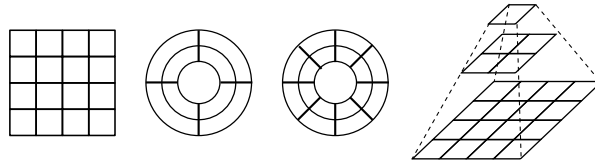


Figure 2.5: Different sampling strategies for: (a) SIFT descriptor (b) the Shape context (log polar grid), (c) GLOH descriptor, and (d) multi-scale or pyramid model.

### Scale Invariant Feature Transform (SIFT)

SIFT algorithm (Lowe, 2004) transforms an image into a set of invariant keypoints, which can be used for image matching, object recognition, tracking, and other applications. It is inspired by biological vision systems. The algorithm consists of 4 major stages:

**Scale-space extrema detection** detects interest keypoints. For this, the image is convolved with Gaussian filters at different pyramidal scales and the Difference of Gaussian is computed. The extrema points across the different scales are selected as the interest keypoints. Due to the blurring step, it eliminates the geometric distortion problem (affine invariant).

**Keypoint localization** is the process of eliminating noisy keypoints. It first performs interpolation of nearby data using the quadratic Taylor expansion of the Difference-of-Gaussian scale-space function. Next, it discards low contrast keypoints by computing the second-order Taylor expansion. Lastly, it eliminates edge responses by finding the eigenvalues of the second-order Hessian matrix. The results of this step are a set of stable corner keypoints that are invariant to image scale.

**Orientation assignment.** Dominant orientations are assigned to each corner keypoint. For each keypoint, a local image patch of size  $16 \times 16$  pixels is extracted and rotated against the dominant orientation to achieve invariance in rotation.

**Keypoint descriptor.** The  $16 \times 16$  pixels image patch is divided into  $4 \times 4$  sub-regions. For each region, a histogram of oriented gradients of 8 bins is constructed where the contribution to the orientation bins is weighted by the gradient magnitude and a Gaussian window overlaid over the region. This results in 128 dimension vectors ( $4 \times 4 \times 8$ ). Finally, this vector is normalised to achieve invariance to illumination changes.

Figure 2.6 shows an overview of the SIFT algorithm. From the nature of its design, SIFT



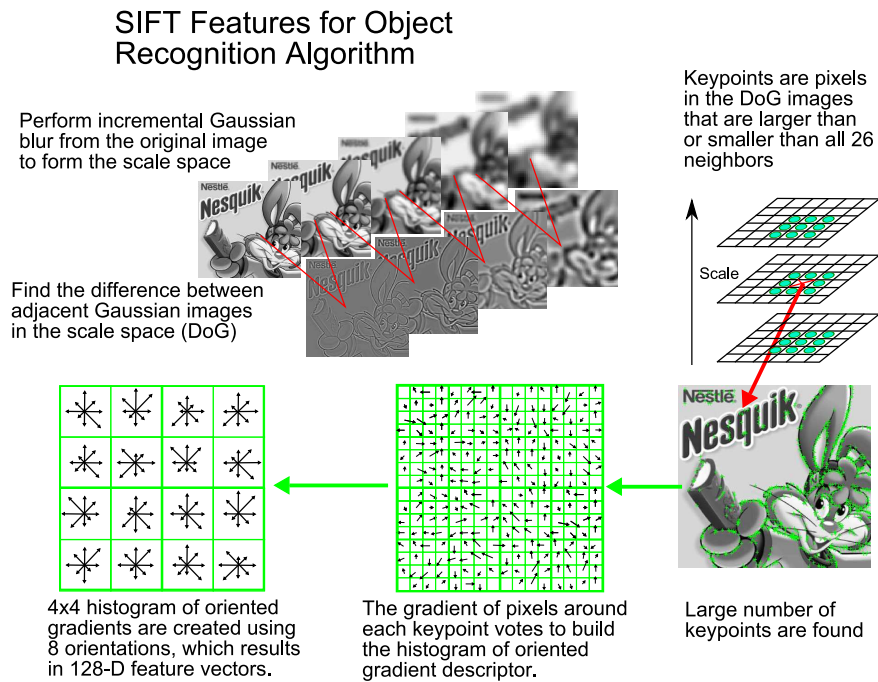


Figure 2.6: The SIFT algorithm (Lowe, 2004).

keypoints are invariant to scale, orientation, affine distortion and is partially invariant to illumination changes (Lowe, 2004), which makes it robust for feature matching and object recognition. Lowe (2004) further proposed an object recognition framework using SIFT features, which consists of four components: feature matching, cluster identification, model verification, and hypothesis acceptance. To achieve fast feature matching, Beis and Lowe (1997) propose the Best Bin First algorithm that efficiently finds the approximate nearest neighbour matches between the image and the model SIFT keypoints. Once the matches are found, (Lowe, 2004) use the Hough transform (Ballard, 1981) and linear least square algorithms (Hartley and Zisserman, 2004) to discard outliers and find the reliable pose models. The final decision to accept or reject the model hypothesis is formulated using the Bayesian analysis (Lowe, 2001). Table 2.4 shows the strength of the SIFT object recognition algorithm.

#### Other SIFT-like descriptors

Many methods have been proposed that either extend the functionalities or improve the efficiency of SIFT descriptors. Table 2.5 shows the summary of the extension to SIFT descriptors. Ke and Sukthankar (2004) reduce SIFT dimensionality by projecting the 128 local histogram gradients using PCA. (Mikolajczyk and Schmid, 2005) propose the

Problems	Techniques	Advantages
Key localization	Difference-of-Gaussian	Accuracy, stability
Scale	Scale-space pyramid	Scale invariance
Orientation	Orientation assignment	Rotational invariance
Lighting changes	Descriptor normalization	Partial lighting invariance
Geometric distortion	Blurring and resampling of local image orientation planes	Affine invariance
Indexing and matching	Nearest neighbour, best bin first search	Efficiency / speed
Cluster identification	Hough transform voting	Reliable pose models
Model verification / outlier detection	Linear least squares	Better error tolerance with fewer matches
Hypothesis acceptance	Bayesian probability analysis	Reliability

Table 2.4: The strength of SIFT object recognition algorithm (Lowe, 2004).

GLOH descriptor that replaces the Cartesian location grid used by the SIFT with a log-polar coordinate system, and apply PCA to reduce the size of the descriptors. Stein and Hebert (2005) propose the alternative weighting functions using isotropic blurring when constructing the SIFT descriptor to separate boundaries of foreground and background objects. Abdel-Hakim and Farag (2006) and Burghouts and Geusebroek (2009) extend the original SIFT to incorporate colour invariance features. Mortensen *et al.* (2005) combine SIFT with shape context using a log-polar grid to detect symmetric objects, the application being targeted for detecting insects. Methods that include different spatial strategies include: Lodha and Xiao (2006), Franz *et al.* (2006), and (Cui *et al.*, 2009). Tamimi and Zell (2005); Sinha *et al.* (2006); Ledwich and Williams (2005); Grabner *et al.* (2006) are some extensions to achieve faster SIFT computation. Stanski and Hellwich (2005) propose an algorithm to cluster SIFT features based on region growing and the matching is done using a group of features. Cheung and Hamarneh (2007) propose N-Dimension SIFT that can be applied to medical applications. Ancuti and Bekaert (2007) incorporate the colour co-occurrence features during the descriptor construction. Recently, Yu and Morel (2009) extend the SIFT to affine invariant SIFT (ASIFT) by incorporating roll and pitch model during the construction of the descriptor. ASIFT demonstrates that it is able to obtain feature matching even under large affine transformation. With similar intuition to SIFT, the SURF feature (Bay *et al.*, 2006) replaces the DoG detector with the Hessian interest point detector, and the descriptor is efficiently computed using 2D Haar rectangle features and the integral image (Viola and Jones, 2002). By relying on the integral image for image convolutions, the computation time is significantly reduced. DAISY (Tola *et al.*, 2009) use Gaussian convolution on predefined locations to approximate the SIFT computation. Liu *et al.* (2008) propose SIFT-flow that is a combination of SIFT features in the optical flow

Variant of SIFT	Advantage over original SIFT
ASIFT (Affine invariant SIFT) (Yu and Morel, 2009)	Affine invariant, can achieve affine up to 40 percent without
BSIFT (Background invariant SIFT) (Stein and Hebert, 2005)	Background invariant
SIFT CCH (Ancuti and Bekaert, 2007)	Increase the distinctness using the colour co-occurrence
Colour SIFT (Abdel-Hakim and Farag, 2006)	Colour invariant
Colour SIFT (Burghouts and Geusebroek, 2009)	Colour invariant
Fast approximation SIFT (Grabner <i>et al.</i> , 2006)	Efficiency
SIFT with global context (Mortensen <i>et al.</i> , 2005)	Has global and local information
SIFT with LBP (CS-LBP) (Heikkilä <i>et al.</i> , 2009)	Combination of SIFT and LBP
GPU SIFT (Sinha <i>et al.</i> , 2006)	Fast computation
Geometric SIFT (GSIFT) (Lodha and Xiao, 2006)	Preserve some geometry information
N-Dimensional SIFT (Cheung and Hamarneh, 2007)	Medical imaging
PCA SIFT (Ke and Sukthankar, 2004)	Less computation
Reduced SIFT (Ledwich and Williams, 2005)	Less computation
Watermark SIFT (Lee <i>et al.</i> , 2006)	Watershed effect
DAISY (Tola <i>et al.</i> , 2009)	Faster computation
GLOH (Mikolajczyk and Schmid, 2005)	Faster computation
Adaptive Grid SIFT for medical (Franz <i>et al.</i> , 2006)	Has geometry information
SURF (Bay <i>et al.</i> , 2006)	Faster computation
Bags of word (codebook SIFT) (Li and Perona, 2005)	Resistant against cluttered scene
SIFT flow (Liu <i>et al.</i> , 2008)	Combination of SIFT and optical flow
SIFT with irregular grid binning (Cui <i>et al.</i> , 2009)	Reduces the negative effect of scale error and significantly increases the matching precision for image features.

Table 2.5: Other SIFT-like descriptors.

formulation.

### Histogram with some spatial information

Apart from distribution based histograms, some researchers also propose histograms that preserve some local spatial information, including spatiogram (Birchfield and Rangarajan, 2005), co-occurrence matrix (Chang and Krumm, 1999), and colour correlogram (Huang *et al.*, 1997). The spatiogram (Birchfield and Rangarajan, 2005) keeps an additional set of mean vectors and covariance matrices of the pixel coordinates that fall into each bin. Figure 2.7 shows an example of a spatiogram that is used for head tracking. Colour

correlogram (Huang *et al.*, 1997) is a 3D histogram where the dimensions are the intensities of two adjacent pixels and their distance, as shown in Figure 2.8. Gray level co-occurrence matrix (Chang and Krumm, 1999) is a 2D histogram that computes the frequency of the adjacent pixel patterns, as shown in Figure 2.9.

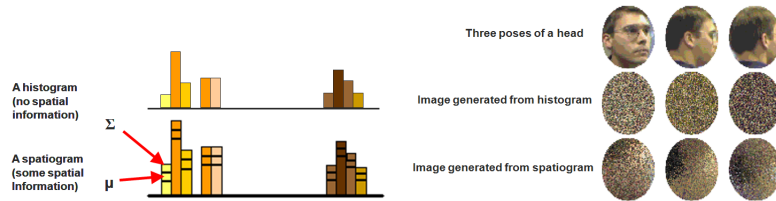


Figure 2.7: Spatiogram (Birchfield and Rangarajan, 2005). ©2005 IEEE

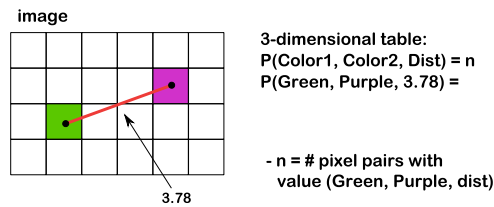


Figure 2.8: Colour correlogram (Huang *et al.*, 1997).

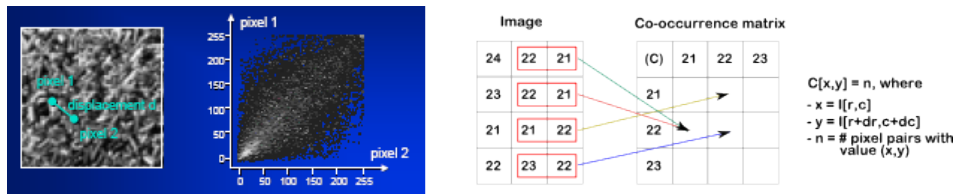


Figure 2.9: Co-occurrence matrix (Chang and Krumm, 1999). ©1999 IEEE

## 2.2 Feature Comparison and Classification Methods

One of the key questions in object detection and tracking involves comparing two objects to measure their similarities or differences. In this section, we review methods to compare these features. In a simple way, the histogram feature can be directly compared using different comparison metrics. In a more sophisticated way, it involves classification methods to learn the structure of the feature groups.

### 2.2.1 Histogram Feature Comparison

Existing methods to compare histogram features can be summarised as: distance metrics (Perlibakas, 2004), earth mover distance (Rubner *et al.*, 1998), pyramid match kernel (Grauman and Darrell, 2005), ordinal method (Cramariuc *et al.*, 2000), and texture measurements (Chen *et al.*, 2000).

#### Distance metrics

Let  $\mathbf{x}$  and  $\mathbf{y}$  be two histograms having  $n$  number of bins, where the index of each bin is represented by  $i = 1, \dots, n$ , then different distance measures can be summarised as in Table 2.6.

#### Earth mover distance (EMD)

The earth mover distance (Rubner *et al.*, 1998) is a method to compute the distance between two distributions (signatures) by finding the minimum amount of work needed to move one distribution into another. For the discrete case, the EMD problem is equivalent to the well-known transportation problem, which can be solved by the Hungarian algorithm (Kuhn, 1955). Let  $\mathbf{x} = \{(x_i, w_{p_i}), i = 1, \dots, m\}$  and  $\mathbf{y} = \{(y_j, w_{q_j}), j = 1, \dots, m\}$  be two signatures, where each element of the signatures are a set of clusters with the weights, then they can be compared using the following equation:

$$EMD(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^m \sum_{j=1}^n f_{ij} d_{ij}}{\sum_{i=1}^m \sum_{j=1}^n f_{ij}}$$

where  $f_{ij}$  is the flow, and  $d_{ij}$  is the ground distance between cluster  $x_i$  and  $y_j$ .

#### Pyramid match kernel

Grauman and Darrell (2005) propose the spatial pyramid match kernel that is able to efficiently compare histograms in high dimensions. The main idea is to map the unordered feature sets to multi-resolution histograms. The advantage of this method is that the computation is linear and the kernel function is positive-definite, which means it is efficient, yet satisfies the Mercer's condition and can be used by standard classification methods

Distance measure	Formula
Manhattan metric ( $L_1$ norm)	$ D _1 = D_{Mn} = \sum_{i=1}^n  x_i - y_i $
Euclidean metric ( $L_2$ norm)	$ D _2 = D_E = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$
Chebyshev metric ( $L_\infty$ norm):	$ D _\infty = \sum_{i=1}^n \max(x_i, y_i)$
Mahalanobis metric	$D_{Mh}(\mathbf{x}, \mathbf{y}) =  \det \Sigma ^{-1} \sqrt{(\mathbf{x} - \mathbf{y}) \Sigma^{-1} (\mathbf{x} - \mathbf{y})^T}$ , where $\Sigma$ is the covariance matrix. If $\Sigma$ is the identity matrix Mahalanobis distance is reduced to the Euclidean distance.
Minkowski metric	$D_{Mk} = (\sum_{i=1}^n  x_i - y_i ^q)^{\frac{1}{q}}$
Weighted Minkowski metric	$D_{Mk} = (\sum_{i=1}^n w_i  x_i - y_i ^q)^{\frac{1}{q}}$
Chi-square metric	$\chi^2 = \sum_{i=1}^n \frac{(x_i - y_i)^2}{x_i + y_i}$ , when $\sum_{i=1}^n x_i = \sum_{i=1}^n y_i = 1$
Canberra metric	$D_C = \sum_{i=1}^n \frac{ x_i - y_i }{(x_i + y_i)}$
Generalised Euclidean metric (quadratic)	$D_{GE} = (\mathbf{x} - \mathbf{y})^T \Sigma (\mathbf{x} - \mathbf{y})$
Pearson correlation metric	$corr(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^n (x_i - \bar{x}_i)(y_i - \bar{y}_i)}{\sqrt{\sum_{i=1}^n (x_i - \bar{x}_i)^2 \sum_{i=1}^n (y_i - \bar{y}_i)^2}}$ , where $\bar{x}_i = \frac{1}{n} \sum_i x_i$
Spearman correlation coefficient	$\rho(\mathbf{x}, \mathbf{y}) = 1 - \frac{6 \sum_{i=1}^n (x_i - y_i)^2}{n(n^2 - 1)}$
Relative entropy (Kullback-Leibler divergence)	$D(\mathbf{x}  \mathbf{y}) = \sum_{i=1}^n x_i \log \frac{x_i}{y_i}$ when $\sum_{i=1}^n x_i = \sum_{i=1}^n y_i = 1$
Bhattacharyya coefficient	$D_{Bh}(x_b, y_b) = \frac{\sqrt{x_b y_b}}{\sqrt{(\sum_{i=1}^n x_i)(\sum_{i=1}^n y_i)}}$
Cosine similarity	$\cos \theta = \frac{(\mathbf{x}, \mathbf{y})}{\ \mathbf{x}\  \ \mathbf{y}\ }$ , where $(\cdot)$ is the dot product.
Angular separation	$D = \frac{\sum_i x_i \cdot y_i}{\sqrt{\sum_i x_i^2 \cdot \sum_i y_i^2}}$
Histogram intersection	$D = \frac{\sum_{i=1}^n \min(x_i, y_i)}{\sum_{i=1}^n y_i}$

Table 2.6: Metrics and their computation.

such as SVM. Given two feature vectors,  $\mathbf{x}$  and  $\mathbf{y}$ , the pyramid match can be computed as:

$$K(\Psi(\mathbf{x}), \Psi(\mathbf{y})) = \sum_{i=0}^L \frac{1}{2} (I(H_i(\mathbf{x}), H_i(\mathbf{y})) - I(H_{i-1}(\mathbf{x}), H_{i-1}(\mathbf{y})))$$

where  $I(H(\mathbf{x}), H(\mathbf{y})) = \sum_{j=1}^r \min(H(\mathbf{x})_j, H(\mathbf{y})_j)$  defines the histogram intersection between histogram  $\mathbf{x}$  and  $\mathbf{y}$ , and  $H_i$  defines the current histogram and  $H_{i-1}$  defines the histogram from previous level in the pyramid.

### Ordinal correlation method

Cramariuc *et al.* (2000) propose a new mechanism to measure the similarity of two images, called ‘‘Ordinal Correlation’’. The traditional method of Euclidean distance compares values pixel by pixel. On the other hand, the ordinal method operates on the rank of pixels rather than directly on the pixel values. Hence, only relative ordering between data values will determine the distance or correlation between two images. Let  $X_k$  and  $Y_k$  represent two images to be compared, with  $k = 1$  to  $n$  as the image pixels. The goal is to review the pixels inside one image and to find the correlation in the other image.

From the given images, the ordinal method computes a set of ordinal relations between each pixel to the whole images as:

$$\begin{aligned} S_k^X &= \{S_{k,l}^X : l = 1 \dots n\} \\ \text{where} & \\ S_{k,l}^X &= \begin{cases} 1 & \text{if } X_k < X_l \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{2.1}$$

In other words, to obtain the first ordinal set,  $S_1$ , the ordinal method finds the relation between pixel 1 to the whole image (i.e. from 1 to  $n$ ), using equation 2.1. The same process is repeated until the  $n^{\text{th}}$  pixel. Hence, we end up with a set of ordinal relation,  $S_{1:k}$ . Next,  $S_{1:k}$  are used to create a metaslice  $M$ , expressed as:

$$\begin{aligned} M_j^X &= OP(\{S_k^X : X_k \in R_j^X\}) j = 1 \dots m \\ \text{where} & \\ M_j^X &= \sum_{k: X_k \in R_j^X} (S_k^X) \end{aligned} \tag{2.2}$$

where  $R$  is the size of one region (for example,  $4 \times 4$  or  $8 \times 8$ ), and  $j$  is the region number, ranging from 1 to  $m$  ( $m$  is the number of regions in the whole image). The option of choosing  $R$  depends on how high the resolution is, i.e. the smaller the  $R$ , the higher the resolution, and vice versa. In the end, we get two sets of metaslices,  $M_j^X = \{\text{set of summation of } S_k \text{ for object } X\}$  and  $M_j^Y = \{\text{set of summation of } S_k \text{ for object } Y\}$ ,  $j = 1$  to  $m$  that corresponds to the two images  $X$  and  $Y$ . Each metaslice describes the object appearance model. To measure the level of dissimilarity, a simple Euclidean distance is applied, i.e.  $D_j = \sum_{k=1}^m \|M_j^X - M_j^Y\|^2$  (Cramariuc *et al.*, 2000). A clear illustration of ordinal method for image correspondence is shown in Figure 2.10. Recently, the ordinal method was extended to compare SIFT features (Toews and Wells, 2009).

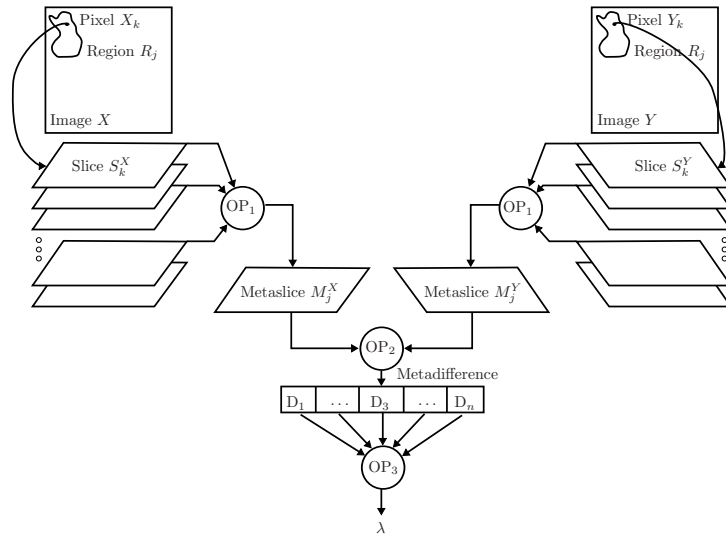


Figure 2.10: A general framework for ordinal-based image correspondence (Cramariuc *et al.*, 2000). ©2000 IEEE

### Texture measurement

Existing methods to measure texture features include entropy, energy, contrast, homogeneity, first, second, and third order moments (Chen *et al.*, 2000). Table 2.7 shows a summary on the computation of the texture measures.



	Texture measures	Formula	Measure
(a)	Energy	$\sum_i \sum_j p_{ij}^2$	intensity pair concentration
(b)	Entropy	$-\sum_i \sum_j p_{ij} \times \log p_{ij}$	disordely of intensity distribution
(c)	Contrast	$\sum_i \sum_j (i - j)^2 p_{ij}$	intensity strength difference
(d)	Homogeneity	$\sum_i \sum_j \frac{p_{ij}}{1+ i-j }$	homogeneity of intensity variation
(e)	First order moment	$\mu = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N p_{ij}$	mean
(f)	Second order moment	$\sigma = \left[ \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N (p_{ij} - \mu) \right]^{\frac{1}{2}}$	standard deviation
(g)	Third order moment	$\theta = \left[ \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N (p_{ij} - \mu)^3 \right]^{\frac{1}{3}}$	skewness
(h)	Correlation	$\frac{\sum_i \sum_j (i - \mu_x)(j - \mu_y) p_{ij}}{\sigma_x \sigma_y}$	intensity correlation

where,  $p_{ij}$  is the element value in the co-occurrence matrix  $P$ ,  $i$  and  $j$  are the nearby intensity pair in the co-occurrence matrix.

Table 2.7: Texture measures.

## 2.2.2 Classification Methods

Classification methods for object detection and tracking can be divided into two categories: the discriminative approach, including SVM and Adaboost, and the generative approach, such as Graphical Models.

### 2.2.2.1 Support Vector Machine

One of the promising methods for learning separating functions in pattern recognition tasks is the Support Vector Machine (Cortes and Vapnik, 1995). It is a binary classification algorithm that finds the optimal decision surface based on the concept of structural risk minimization. Given training data, SVM maps the input vectors into a high dimensional space through nonlinear mapping. The optimal separating hyperplanes are then constructed in the high dimensional space and the classification solution is obtained in terms of generalization error (that has the maximum margin) among the number of hyperplanes.

In (Dalal and Triggs, 2005), a two-class linear SVM classifier is used to trained the dense HOG features to study the shape of the pedestrian. Serre *et al.* (2005) use multiple-class linear SVM and visual cortex features to recognise different classes of objects in the Caltech

datasets<sup>1</sup>. Grauman and Darrell (2005) introduce the pyramid match kernel which can be used in SVM classifier to efficiently detect objects and recognise scenes. Tang *et al.* (2007) propose the co-tracking algorithm that uses an online semi-supervised SVM classifier to update target appearance at each frame. This results in a discriminative solution that is able to locate and track the foreground objects at the same time.

#### 2.2.2.2 Cascaded Adaboost

Adaboost (Freund and Schapire, 1996) is a machine learning algorithm that constructs a “strong” classifier using a set of “weak” classifiers. It builds a cascaded model, where each stage involves the process of accepting the most relevant classifiers and discarding the unimportant ones. Although the training requires longer time as compared to SVM training, the resultant Adaboost classifier generally gives faster decision during run time.

Viola and Jones (2002) propose a real time face detection implementation by utilising Haar features and the integral image concepts trained using Adaboost. Similarly, Levi and Weiss (2004) replace the Haar feature with Edge Orientation Histogram features and train using Adaboost to achieve better accuracy in face detection results than Viola and Jones (2002). In (Zhu *et al.*, 2006), HOG features are trained using cascaded Adaboost to achieve fast and accurate pedestrian detection. Li *et al.* (2007) propose a cascaded particle filter using different Adaboost trained feature sets to track human face at low frame rate conditions.

#### 2.2.2.3 Graphical Models

Graphical models formulate the problem in a probabilistic framework and solve it through posterior probability estimation. Bayesian statistics, Markov Random Field, Conditional Random Field, Hidden Markov Models are methods belonging to this group. Graphical models have been applied in many applications including object detection (Zhe *et al.*, 2007; Leibe *et al.*, 2005), background subtraction (Stauffer and Grimson, 2000; Elgammal *et al.*, 2000), tracking (Khan *et al.*, 2005; Smith *et al.*, 2005), and activity recognition (İkizler and Duygulu, 2007). Descriptions of these methods are discussed in the following sections.

---

<sup>1</sup>[http://www.vision.caltech.edu/Image\\_Datasets/Caltech101/](http://www.vision.caltech.edu/Image_Datasets/Caltech101/)

## 2.3 Background Subtraction

Background subtraction is a method to detect foreground objects by finding the difference between the current frame and the background model. The background model is generally created by *fusing* a sequence of images with no foreground object present. An ideal background model should adapt to illumination changes (gradual or sudden changes), motion changes (camera oscillations, noisy background objects such as tree branches, sea waves, and similar), and changes in the background geometry (Piccardi, 2004).

Stauffer and Grimson (2000) use  $k$ -Gaussian mixture models to express the distribution of the background pixels. Despite its simplicity, the algorithm allows real-time processing with reasonable accuracy when it does not deal with lighting changes in the environment. Javed *et al.* (2002) propose a hierarchical background subtraction algorithm that consists of pixel, region, and frame level processing to address the problem of quick illumination changes, relocation of the background object, and initialization with moving objects. In this framework, both colour and gradients are used as the observation features. Elgammal *et al.* (2000) propose a kernel density approach to model the background distribution. This model is able to deal with cluttered scenes, scenes that are not completely static, and small motion noise such as tree branches and bushes. Oliver *et al.* (2000) apply Principle Component Analysis (PCA) to a sequence of frames to compute the *eigenbackground*, storing only the important eigenvectors that are used to detect the foreground objects. The advantage of this approach is that the background model is compact, easy to construct, update, and is able to run in real-time. Kim *et al.* (2005) model each background pixel as a set of codewords that holds the following information: the minimum and maximum codeword brightness, the frequency, first and last access time in which the codeword has occurred, and the longest interval that the codeword has not recurred. This results in an adaptive and compact background model that captures the background motion over a long period of time with limited memory. It is also resistant to artifacts, has the ability to cope with illumination changes, and allows initialization with moving objects. Xu *et al.* (2005) propose a spatio-temporal multi-resolution MRF background subtraction that has three MRF connections: pixels and their neighbourhood, the corresponding pixels at two adjacent pyramid level, and pixels across time. The inference problem is solved using Gibbs sampling. Huang *et al.* (2008) propose MRF based video segmentation and efficiently solve the inference problem using the Loopy Belief Propagation (LBP) algorithm. Due to the spatial and temporal connections, the MRF based approaches are robust to small movements in the scene and generally produce more accurate segmentation results.

Paper	Method	Speed	Memory
Stauffer and Grimson (1999)	Mixture of Gaussians	$mF$	$mF$
Elgammal <i>et al.</i> (2000)	Kernel density estimation	$NF$	$NF$
Oliver <i>et al.</i> (2000)	Eigenbackground	$MF$	$NF$
Javed <i>et al.</i> (2002)	Hierarchical colour and edge	$F(m+1) + R + F$	$F(m+1)$
Kim <i>et al.</i> (2005)	Codebook approach	$CF$	$CF$
Xu <i>et al.</i> (2005)	Spatio-temporal MRF	$FT(n+l+t)$	$\left(\frac{1-\frac{1}{2}^l}{1-\frac{1}{2}}\right)F(n+t)$

Table 2.8: Summary of different background subtraction methods.

We summarise the strengths and weaknesses of each method in terms of its speed and memory complexity in Table 2.8.

### Speed

The complexity for the GMM method (Stauffer and Grimson, 2000) is  $O(mF)$ , where  $m$  is the number of Gaussians used, typically between 3 to 5,  $F$  is the image dimension ( $F = \text{width} \times \text{height}$ ). The KDE approach computes the pixel differences based on the Gaussian kernels centered on the past  $N$  frames, thus the complexity is  $O(NF)$ . The complexity of the eigenbackground method can be estimated as  $O(MF)$ , where  $M$  is the number of the best eigenvectors. The complexity for the codebook approach is  $O(C)$ , where  $C$  is the size of the codewords. The complexity of the spatio-temporal MRF based approach is twofold: (1) the construction of spatial and temporal MRF connections, and (2) the time for Gibbs sampling algorithm to compute the posterior probability. Thus, it is estimated as  $O(FT(n+L+t))$ , where  $T$  is the number of Gibbs sampling iterations to solve the inference problem,  $n$  is the number of neighbourhood pixels typically 4 or 8,  $L$  is the number of pyramid levels (it is set to 3 in the paper), and  $t$  is the number of temporal connection, typically set to 1. The complexity for the hierarchical approach is estimated as  $O(F(m+1)+R+F)$ , where  $I(m+1)$  is the complexity for pixel level processing,  $(m+1)$  is the number of Gaussians plus the gradient features,  $R$  is the complexity for region level processing, and since the frame level processing involves checking for the global thresholds, it has complexity of at most  $F$ .

### Memory

The memory requirement for GMM, KDE, and codebook approaches are  $O(mF)$ ,  $O(NF)$ , and  $O(C)$  respectively. During training, the eigenbackground requires all  $N$  training images, with complexity of  $O(NF)$ . However, in classification, it only requires memory

requirement of  $O(MF)$ . For the spatio-temporal MRF approach, because the size of a pyramid follows the geometric series, therefore the memory complexity can be written as:  $O\left(\left(\frac{1-\frac{1}{2}^L}{1-\frac{1}{2}}\right)F(n+t)\right)$ . The memory for the hierarchical background subtraction approach that uses both mixture of Gaussians and gradient feature is  $O(F(m+1))$ .

## 2.4 Pedestrian Detection

Another way to detect the foreground object is through object learning using training samples. In this dissertation, we only consider pedestrian detectors as this is closely related our work. The performance of the detector usually depends on the choice of the features, the learning algorithm, and the training samples. Tables 2.9 and 2.10 show the summary of different pedestrian datasets and algorithms respectively. Papageorgiou and Poggio (1999) use Haar wavelets of pedestrian foregrounds as input features to an SVM classifier. Jones *et al.* (2003) propose a fast pedestrian detection method through extended Haar features, trained using Adaboost. Figure 2.11 shows example of Haar dictionaries. Gavrilu and Philomin (1999); Zhe *et al.* (2007) implement a hierarchical template matching using chamfer distances to detect pedestrians. Wu and Nevatia (2005); Sabzmeydani and Mori (2007) study the shape of the pedestrians using edgelet and shapelet features trained using Adaboost. Tuzel *et al.* (2007) build the covariance of multi-cues with the Riemannian manifolds learning to detect pedestrians. Dalal and Triggs (2005) employ the dense 3D orientation gradient histogram approach trained with linear SVM. Leibe *et al.* (2005) present an Implicit Shape Model in a probabilistic framework to detect pedestrian crowds. The algorithm requires creation of codebook appearance (Jurie and Triggs, 2005) followed by Bayesian rules to obtain the final detection results. Recently, Felzenszwalb *et al.* (2008) propose a multi-scale or pyramid HOG trained using latent SVM. Maji *et al.* (2008) propose a multilevel HOG like feature with fast hierarchical (histogram intersection) and efficient SVM kernel. The main contributions lie in the novel features based on pyramid of oriented energy and the introduction of intersection kernel SVM.



Figure 2.11: Haar features: black and white areas denote the negative and positive weights, respectively.

Previous pedestrian detection evaluations (Enzweiler and Gavrilu, 2009),(Munder and Gavrilu, 2006), (Dollar *et al.*, 2009), and (Leibe *et al.*, 2005) have demonstrated that:

Datasets	Training samples			Test samples		
	# pedestrians	# negative	resolution	# images	resolution	# pedestrians
INRIA (Dalal and Triggs, 2005)	2416	1218	$96 \times 160$	1132	-	1132
NICTA (Overett <i>et al.</i> , 2008)	25551	5207	$32 \times 80$	-	-	-
DaimlerChrysler (Munder and Gavrila, 2006)	4000	5000	$18 \times 36$	-	-	-
Daimler (Enzweiler and Gavrila, 2009)	15560	6744	$18 \times 36$	21790	-	56492
MIT-CBCL (Papageorgiou and Poggio, 1999)	924	0	$64 \times 128$	-	-	-
Caltech (Dollar <i>et al.</i> , 2009)	-	-	-	250000	$640 \times 480$	350000
USC-A (Wu and Nevatia, 2005)	-	-	-	205	-	313
USC-B (Wu and Nevatia, 2005)	-	-	-	54	-	271
USC-C (Wu and Nevatia, 2007)	-	-	-	100	-	232
CVC (Geronimo <i>et al.</i> , 2007)	1000	6175	variable	-	-	-
TUD-Brussels (Wojek <i>et al.</i> , 2009)	3552	192	$64 \times 128$	-	$640 \times 480$	1326

Table 2.9: Pedestrian datasets.

Full body detector	Feature	Classifier	Search method
Papageorgiou and Poggio (1999)	Haar Wavelets	SVM (quadratic)	Sliding windows
Felzenszwalb <i>et al.</i> (2008)	Multi-scale or pyramid HOG	Latent SVM	
Maji <i>et al.</i> (2008)	Multilevel HOG	Hierarchical SVM kernel	
Tuzel <i>et al.</i> (2007)	Covariance feature	Riemannian manifolds	
Dalal and Triggs (2005)	HOG	Linear SVM	
Our (Chapter 5)	HOGB	Linear SVM	Sliding windows, plane estimation

Part body detector	Feature	Classifier	Search method
Jones <i>et al.</i> (2003)	Haar wavelets	Adaboost	Sliding windows
Leibe <i>et al.</i> (2005)	ISM (intensity codebook)	MDL verification	Probability search
Wu and Nevatia (2005)	Edgelet	Adaboost	Sliding windows
Sabzmeydani and Mori (2007)	Shapelet	Adaboost	Sliding windows
Zhu <i>et al.</i> (2006)	Integral HOG	Cascaded Adaboost	Sliding windows
Our (Chapter 5)	Codebook HOG & HOGB	K-mean clustering	Efficient Subwindow Search

Table 2.10: Different pedestrian detection approaches.

(i) using INRIA dataset, HOG (Dalal and Triggs, 2005) performs the best to detect fully visible upright persons; (ii) in crowded scenes, the method by (Leibe *et al.*, 2005) performs the best since it uses patch/codebook based approach that is not affected by the crowds.

## 2.5 Object Tracking

The problem of tracking a target using low frame rate camera can also be viewed as tracking using non-overlapping cameras for the following reason: between two consecutive frames in a single low-frame-rate camera, a target may have different appearances due to lighting changes and large pose differences. This is also true for the non-overlapping cameras situation, where a person being viewed by a camera may look different from one camera to the next in the network. Recent work in this area includes (Jeong and Jaynes, 2006; Javed *et al.*, 2005) who learn the brightness transfer function (colour mapping) across pairs of non-overlapping cameras. Using the colour mapping function, objects that move from one camera to another can be associated correctly. In the case of single low-frame-rate camera tracking, Porikli and Tuzel (2005) use multiple kernels to extend the mean-shift algorithm to track multiple targets for a 1 fps single camera video. Li *et al.* (2007) propose a cascade particle filters tracking algorithm where they break down the multi feature observation likelihoods and solve them in a coarse-to-fine manner. This allows the algorithm to focus on the relevant regions more quickly. This approach is able to track human faces at 5 fps. It is an extension over the simple Haar based face detection by Viola and Jones (2002). Porikli *et al.* (2005) propose covariance tracking where the covariance feature of the target is updated using Lie Algebra. Because the method combines multi-cue features to form the covariance matrix, it has the advantage of utilising the features to track objects under low frame rate situations.

The problem becomes more complicated when multiple targets need to be tracked due to object occlusions and similar appearances. In this area, Zhao and Nevatia (2004) use a MCMC framework to estimate the dynamics of multiple overlapping objects. The approach uses the 3D human shape, object appearance, and background models as the observation model. Their experiment demonstrates promising human tracking results in crowded environments. Khan *et al.* (2005) introduce an efficient MCMC based Particle Filter framework for multiple object tracking. Since the posterior is estimated linearly using MCMC simulation, it provides a tractable and faster solution. Smith *et al.* (2005), Khan *et al.* (2006), and Yu and Medioni (2007) extend the approach further to handle varying number of targets by using reversible jump MCMC techniques, which allows for changes in dimensions between samples within a Markov Chain. Yu and Wu (2004, 2005)

propose a decentralized tracking framework using multiple sensors to efficiently track multiple targets. They employ a set of autonomous and collaborative trackers to distribute the computation between multiple sensors. Recently, Kembhavi *et al.* (2008) propose an efficient resource allocation particle filter combined with the decentralized tracking concept. It first clusters the targets based on their spatial proximity and pairwise appearance similarity, and then efficiently assigns different numbers of particles for different groups, which leads to better tracking performance using lesser number of particles.

## 2.6 Image Registration and 3D Scene Reconstruction

Image registration is the process of geometrically aligning two or more images of the same scene taken from different viewpoints at different times by different sensors (Zitova and Flusser, 2003). Extensive research has been done in the area of image registration, (Zitova and Flusser, 2003; Brown, 1992) contain detailed survey of image registration methods. In this chapter, we cover three common image registration methods: Homography (Hartley and Zisserman, 2004), Thin Plate Spline (Bookstein, 1989), and iterative closest point algorithm (Besl and McKay, 1992).

*Homography* (Hartley and Zisserman, 2004) is a 2D projective transformation that maps points from one plane to another plane. It is defined as a  $3 \times 3$  homogeneous matrix such that for any point  $x_i$  on plane  $\pi$ , and its corresponding point  $x'_i$  on  $\pi'$ ,  $x'_i = \mathbf{H}x_i$ . The  $\mathbf{H}$  defines the translation, rotation, scale, and shear transformation from the first to the second plane.

*Thin Plate Spline* (Bookstein, 1989) is an interpolation method that finds a “minimally bended energy” of the smooth surface that passes through all given points. In image registration, the energy function is defined as  $E = \int \int \left[ \left( \frac{\partial^2 f}{\partial x^2} \right)^2 + 2 \left( \frac{\partial^2 f}{\partial xy} \right)^2 + \left( \frac{\partial^2 f}{\partial y^2} \right)^2 \right] dx dy$ , where  $f$  is the mapping function for the corresponding point-sets  $\{x_i\}$  and  $\{y_i\}$ .

*The iterative closest point algorithm* (Besl and McKay, 1992) is a method to fit points in a target model to points in a source model. The final goal of the algorithm is to find translation  $t$  and rotation  $R$  that minimizes the sums of square errors with respect to the closest source points and their corresponding target points:  $E(R, t) = \frac{1}{N_p} \sum_{i=1}^{N_p} \|x_i - Rp_i - t\|^2$ , where  $x_i$  and  $p_i$  are corresponding points.



	Camera	Number of images required	Methods
2D	single camera	2	Homography
2D	single camera	2	Thin Plate Spline
3D	single camera	$N$	Structure from Motion
3D	stereo cameras	$N$	Essential Matrix
3D	stereo cameras	$N$	Fundamental Matrix

Table 2.11: Image registration methods.

The 3D scene reconstruction is essentially the geometry of the intersection of multiple image planes (multiple points) using either single or stereo cameras. Structure From Motion (Snavely *et al.*, 2008) is a method to reconstruct the 3D scene model from a sequence of 2D scene images from a single camera. Given feature correspondences from all image sequences, the objective is to minimise the projection error of the 2D feature correspondence in 3D space. Another way to reconstruct the 3D scene model is to use stereo cameras. To calculate the depth information from stereo cameras, we need to compute the so-called *epipolar* geometry. There are two types of approaches: calibrated and uncalibrated approaches. In the calibrated approach, it is done by extracting the *essential* matrix. In the uncalibrated approach, it is captured in the fundamental matrix, which is calculated from image correspondence. Using either one of this matrix, we can compute the 3D structure of the scene. In both approaches, the underlying principle is that of *triangulation* (Hartley and Zisserman, 2004).

## 2.7 Chapter Summary

This chapter has summarised state of the art in low level features, multiple-target tracking, image registration, and more specifically foreground detection. However, many of those techniques perform poorly when applied to the challenging images acquired by mobile cameras mounted on vehicles (taken under different lighting conditions using low frame rate low resolution cameras). Thus, the focus of this thesis is to construct techniques for this new and emerging dataset.

---

## CHAPTER 3

# PASSENGER MONITORING SYSTEM

---

In this chapter we consider passenger detection and tracking scenarios in which cheap and static cameras are used to monitor the inner environments of public transport vehicles, such as trains and buses. This problem is challenging due to the following three factors: drastic lighting changes, object complexity, and low frame rate. When the bus or train moves, influences from external light sources like the sun and street/vehicle lighting have a huge impact on the scene. In addition, the lighting inside changes drastically as it travels through a series of tunnels, stations, freeways, and tree-lined streetscapes, causing traditional assumptions of near-constant background intensity to be violated. This means algorithms for foreground-background separation (Stauffer and Grimson, 2000; Toyama *et al.*, 1999; Elgammal *et al.*, 2000; Kim *et al.*, 2005; Noriega and Bernier, 2006; Javed *et al.*, 2002), which are often the first step in tracking, would fail. Crowd levels, occlusions, complex human appearances, and irregular poses are factors that cause human detection algorithms (Papageorgiou and Poggio, 1999; Viola and Jones, 2002; Gavrila and Philomin, 1999; Zhe *et al.*, 2007) to fail. CCTV surveillance systems onboard buses are made to accommodate multiple cameras onto a single recording system which impose a limited frame capture throughput, thereby resulting in low capture frame rates for each camera. Approaches for tracking (Comaniciu and Meer, 2002; Xiao and Shah, 2005; Khan *et al.*, 2005) that rely on temporal consistency and smooth tracked trajectories will fail under these low frame rate situations.

In this chapter we propose a novel passenger detection and tracking system to tackle the above problems. Our approach integrates a stable SIFT (Scale Invariant Feature Transform) background seat modelling mechanism with a human shape model into a weighted Bayesian framework to detect passengers inside buses. In addition, we propose an appearance transformation observation model and a proposal distribution with a passenger trajectory model, and show how these models can be integrated into the Reversible-Jump Markov Chain Monte Carlo (RJMCMC) framework (Smith *et al.*, 2005) to track multiple passengers under the low frame rate situations.

SIFT background modelling extracts local stable features in the pre-annotated background

seat areas and tracks these features over time to build a statistical background model for each seat. Since SIFT features are partially invariant to lighting, this background model can be used robustly to detect the seat occupancy status even under severe lighting changes. Flexible human configuration models based on ellipse fitting is then employed to confirm the existence of passengers. Figure 3.1 shows an overview of the foreground detection approach.

To deal with the problem of tracking in low frame rate video, we are motivated by previous works in tracking across multiple non-overlapping cameras using appearance models (Javed *et al.*, 2005) and Particle Filters with prior information about the environment (Leoputra *et al.*, 2006; Smith *et al.*, 2005). We invoke a learning phase to acquire the passenger appearance transformations that arises due to low frame rates in a single sensor. This training phase uses colour, texture, and spatial cues as the appearance features, and an SVM classifier to learn the transformation. The transformation models capture how foreground objects change across frames in low frame rate situations, which can be used to compute object correspondence. In addition, we introduce a new sampling method based on the allowable passenger trajectories to achieve better prediction of the object movements. This supports robust people monitoring and tracking that is resilient to sudden illumination changes and is suitable in low frame rate situations.

The layout of this chapter is as follows. Section 3.1 describes the estimation of seat locations using a Homography alignment method. Section 3.2 outlines the unified framework of background-modelling and foreground-detection algorithm to detect the passengers. The proposed tracking framework is detailed in Section 3.3. Finally, Section 3.4 shows the experimental results that demonstrate performance of the detection and tracking system in a real transport surveillance video footage, dealing with the problem of drastic lighting changes and low frame rates.

### 3.1 Offline Background Patches Estimation

For practical reasons, the design of the CCTV configurations with regard to camera placement and orientation would be common for the same model of buses. Based on this, we can safely assume that between two similar types of buses, each of the corresponding cameras are located almost in the same position, with only slight view-point variations (satisfying the 2D projective assumption), as shown in Figure 3.1(a). We first define background patches for the images taken from cameras in a bus. This process is done once and the focus is on the seat regions of the bus. For portability, we apply SIFT-Homography

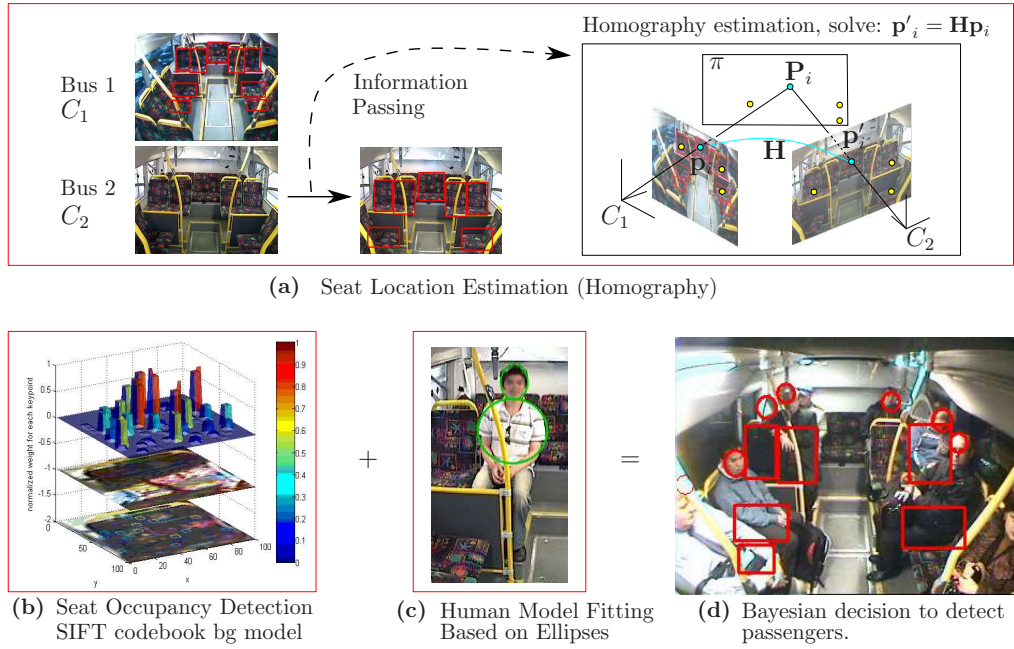


Figure 3.1: Proposed framework for passenger detection.

algorithm (Lowe, 2004; Hartley and Zisserman, 2004) to compute 2D point transformation from one camera to a corresponding camera in another bus. The SIFT algorithm is used to find point correspondences between two images, while the Homography algorithm performs the 2D mapping estimation.

The Homography algorithm is described as follows: Let two cameras  $C_1$  and  $C_2$  from two different buses be directed at a point  $\mathbf{P}_i$  on a plane  $\pi$  and let  $\mathbf{p}_i$  and  $\mathbf{p}'_i$  be the projections of  $\mathbf{P}_i$  into the image camera  $C_1$  and  $C_2$  respectively, as shown in Figure 3.1(a). Then there exists a  $3 \times 3$  matrix  $\mathbf{H}$  such that  $\mathbf{p}'_i = \mathbf{H}\mathbf{p}_i$ , where  $\mathbf{H}$  is called the Homography matrix of the plane  $\pi$  (Hartley and Zisserman, 2004). There are a number of methods to estimate  $\mathbf{H}$ , such as the Direct Linear Transform (DLT) algorithm, normalized DLT, LMedS, etc (Hartley and Zisserman, 2004). In our implementation, we perform SIFT feature correspondence, followed by Homography-RANSAC using the normalized DLT approach. RANSAC (Fischler and Bolles, 1981) is used to estimate the initial Homography. Then, we refine the Homography via a normalized DLT algorithm iteratively, until the algorithm reaches an optimal estimation of  $\mathbf{H}$ . Figure 3.2 shows an example of the seat alignment process.

The annotation of background information in a bus offers a number of advantages: (1) Knowing the seat locations enables the background model for each seat to be created

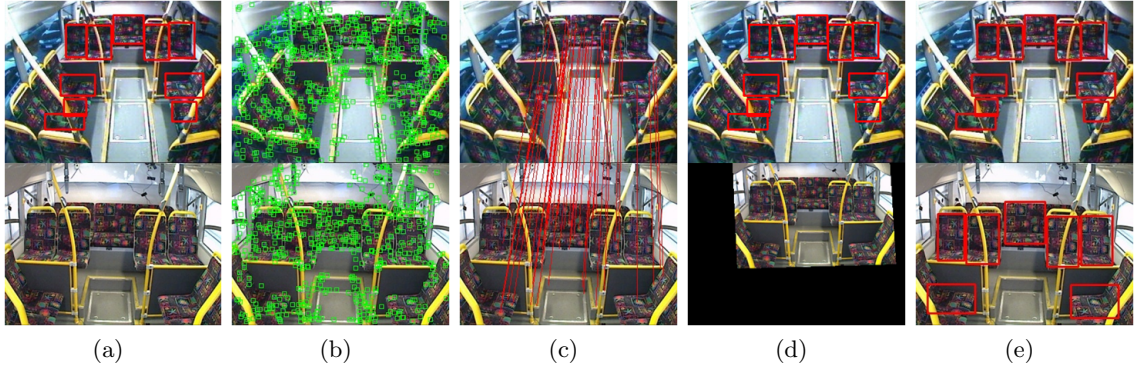


Figure 3.2: Seat alignment process. (a) The task is to “transfer” the annotated seat configurations from bus-1 (top) to bus-2 (bottom). (b) SIFT feature extraction. (c) SIFT feature correspondences. (d) Homography alignment. (e) The seat configurations in bus-2 can be estimated using the Homography matrix.

and used to probe its occupancy status. (2) Using the seat arrangement information, the camera can be calibrated and used to efficiently search for possible human presence; whilst ignoring certain areas such as windows that may cause false positives. (3) Hypotheses such as allowable tracking trajectories can be assigned to the global bus structure, for example, a passenger who sits close to the window has to appear in between seats before proceeding to the aisle. The following section discusses the proposed passenger detection algorithm for a single seat region using the seat configuration information obtained in this section.

## 3.2 Bayesian Formulation for Passenger Detection

We formulate the passenger detection problem in a Bayesian framework as shown in Figure 3.3. In this network, the passenger inference relies on two pieces of evidence: the fitted human template model and the seat occupancy status.

Let  $I$  be the observed image,  $h$  is a latent variable for a seat region that defines the state of a passenger being *detected* (1) or *not detected* (0),  $\mathbf{F} = \{f_i\}_{i=1}^N$  is the set of SIFT keypoints extracted from a seat region,  $\mathbf{S} = \{s_i\}_{i=1}^N$  is the corresponding keypoint states for being either *foreground* (1) or *background* (0),  $N$  is the number of SIFT keypoints found in a seat region at a particular time instance, and  $\Gamma$  is the SIFT codebook background model that has been learnt. Hence, its joint probability distribution can be written as:

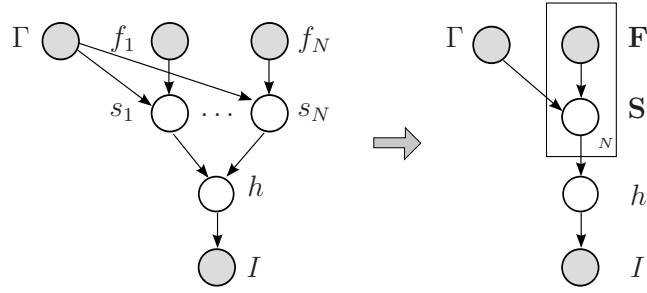


Figure 3.3: A graphical model representation for the passenger detection, which is decomposed into probability of seat occupancy ( $P(h|\mathbf{S})P(\mathbf{S}|\mathbf{F}, \Gamma)$ ) and probability of fitting the best ellipse onto the observed image to infer possible passenger presence ( $P(I|h)$ ).

$$\begin{aligned}
 P(h, I, \mathbf{S}, \mathbf{F}, \Gamma) &= P(I|h) P(h|\mathbf{S})P(\mathbf{S}|\mathbf{F}, \Gamma)P(\mathbf{F})P(\Gamma) \\
 &\propto P(I|h) P(h|s_1, \dots, s_n) \prod_{i=1}^N P(s_i|f_i, \Gamma)
 \end{aligned} \tag{3.1}$$

The first term  $P(I|h)$  computes the probability for the best fit human configuration, while the second and third terms ( $P(h|\mathbf{S})P(\mathbf{S}|\mathbf{F}, \Gamma)$ ) represent the probability of seat occupancy. Intuitively, when the probability of seat occupancy is high it also implies that a passenger is occupying the seat. From the joint probability distribution, we can compute the probability of the passenger being detected as:

$$\begin{aligned}
 P(h|I, \mathbf{S}, \mathbf{F}, \Gamma) &= \frac{P(h, I, \mathbf{S}, \mathbf{F}, \Gamma)}{\sum_h P(h, I, \mathbf{S}, \mathbf{F}, \Gamma)} \\
 &\propto P(h, I, \mathbf{S}, \mathbf{F}, \Gamma)
 \end{aligned} \tag{3.2}$$

Here, we assume that the prior is uniform, and therefore the MAP estimation turns into maximizing the joint probability.

Recently, Zhou and Huang (2006) propose a weighted Bayesian Network model and we adopt a similar idea. Thus, the joint probability from Equation 3.1 can be re-expressed as:

$$P(h|I, \mathbf{S}, \mathbf{F}, \Gamma) \propto \left(P(I|h)\right)^\beta \left(P(h|s_1, \dots, s_n) \prod_{i=1}^N P(s_i|f_i, \Gamma)\right)^{(1-\beta)} \quad (3.3)$$

where  $\beta$  and  $(1 - \beta)$  are the weight factors for the best fitted human model and the seat occupancy respectively. The computation of  $P(I|h)$  will be described in Section 3.2.2, and  $P(h|\mathbf{C})P(\mathbf{C}|\mathbf{F}, \Gamma)$  will be described in the next section.

### 3.2.1 Seat Occupancy Detection

#### 3.2.1.1 SIFT Codebook Background Modelling (Constructing $\Gamma$ )

The SIFT codebook background modelling algorithm is motivated by Kim *et al.* (2005), in which we build codebooks of SIFT features to represent a particular seat background model. Unlike the codebook background modelling (CB) approach (Kim *et al.*, 2005), we focus on extracting the statistics of SIFT features instead of intensity.

Let  $F = \{f_1, f_2, \dots, f_M\}$  be a set of SIFT keypoints extracted for a pixel region during training. Each feature  $f_j$ ,  $j = 1, \dots, M$  is an individual keypoint that consists of information about its location  $p_j$  and its descriptors are  $\varphi_j \in \mathbb{R}^{128}$ . Intuitively, we expect the keypoint descriptors in  $F$  to be the same because (1) the seat is rigid and the camera is static and (2) SIFT is partially invariant to lighting (Lowe, 2004). In real situations, however, their descriptors can still change due to severe lighting changes and the effect of shadows. Hence, we employ the codebook background model. Let  $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_K\}$  represent the codebook for a pixel region consisting of  $K$  codewords. Each codeword  $\mathbf{c}_k$ ,  $k = 1, \dots, K$ , consists of a SIFT descriptor  $\varphi_k$  and the normalised frequency of the codeword  $l_k$ . The codeword has to satisfy the following two conditions:

$$\begin{aligned} \varphi_j \cap \varphi_k &= \emptyset, \forall j \neq k \in \mathcal{C} \\ l_k &> 0, \sum_k l_k = 1 \end{aligned} \quad (3.4)$$

First, each keypoint descriptor  $\varphi_k$  has to be unique among other keypoint descriptors in  $\mathcal{C}$ , capturing a distinctive background lighting characteristic. Secondly,  $l_k$  represents the likelihood for the corresponding unique keypoint descriptor belonging to the background. Intuitively, the SIFT codebook background model consists of clusters of histogram descriptors, counting how many feature points of each codeword occur.

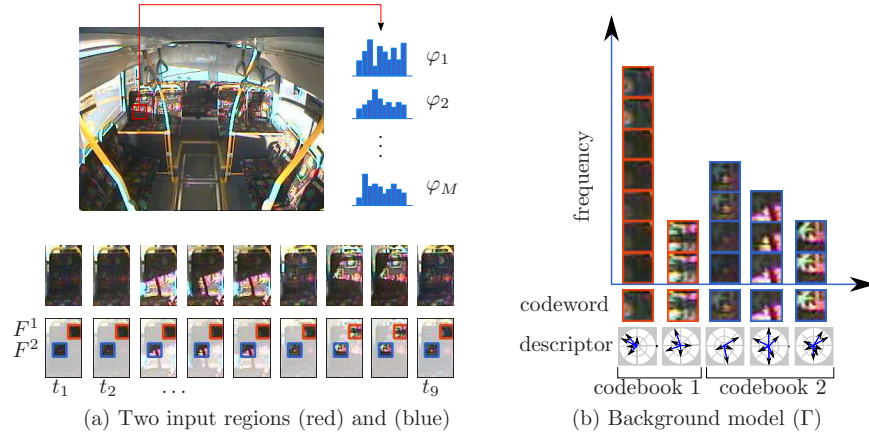


Figure 3.4: An example of SIFT codebook background modelling on a seat inside a bus.

Then, the compact background model for a particular seat  $\Gamma$  can be written as:

$$\Gamma = \{\mathcal{C}^1, \dots, \mathcal{C}^Q\} \quad (3.5)$$

where  $\mathcal{C}^r$ ,  $r = 1, \dots, Q$ , is a codebook for a particular region  $r$ , and  $Q$  is the total number of background codebooks found in a particular seat for time  $t = 1 \dots M$ . Figure 3.4 shows an example of SIFT codebook background modelling of a seat over 9 time instances under significant changes in illumination. For illustration purposes, let  $F^1 = \{f_1^1, \dots, f_9^1\}$  and  $F^2 = \{f_1^2, \dots, f_9^2\}$  be the lists of keypoint descriptors located in the red and blue regions respectively, as shown in Figure 3.4 (a). In this example,  $f_1^1, f_2^1, f_3^1, f_4^1, f_5^1, f_6^1$ , and  $f_9^1$ , have similar keypoint descriptors, hence they are clustered into the first codeword  $\mathbf{c}_1^1$  in the first region 1; while  $f_7^1$  and  $f_8^1$  belong to second codewords  $\mathbf{c}_2^1$  in region 1. Therefore, the codebook for region 1 is represented as  $\mathcal{C}^1 = \{\mathbf{c}_1^1, \mathbf{c}_2^1\}$ . Similarly, we obtain three codewords to form the codebook for region 2:  $\mathcal{C}^2 = \{\mathbf{c}_1^2, \mathbf{c}_2^2, \mathbf{c}_3^2\}$ . The final background model for the seat is shown in Figure 3.4 (b). The high-level pseudo code to construct the SIFT codebook background model is presented in Algorithm 1.

### 3.2.1.2 Computing Probability of Seat Occupancy

From Equation 3.3, the problem of inferring whether a seat is *occupied* given the current SIFT keypoints can be formulated as:  $P(\mathbf{S}|\mathbf{F}, \Gamma) \times P(h|\mathbf{S})$ .



---

**Algorithm 1:** Seat background model construction.

---

1.  $\Gamma \leftarrow \emptyset$  ( $\leftarrow$  means assignment)
  2. **for**  $t = 1$  to  $M$  **do**
    - (a) Let  $F$  be the set of SIFT keypoints extracted from the seat
    - (b) **foreach**  $f_j \in F$  **do**
      - i. Find the codebook  $\mathcal{C}^r$  in  $\Gamma = \{\mathcal{C}^1, \dots, \mathcal{C}^Q\}$  that satisfy the following condition:  $dist(p_j, r) < \epsilon_1$
      - ii. Find the codeword  $\mathbf{c}_m$  in  $\mathcal{C}^r = \{\mathbf{c}_k | 1 \leq k \leq K\}$  that best matches with  $\rho_j$  based on the following condition:  $descp(\varphi_j, \varphi_m) < \epsilon_2$ , where  $descp(., .)$  is a function that computes the difference between two SIFT descriptors.
      - iii. If  $\mathcal{C} = \emptyset$  or there is no match, then  $K \leftarrow K + 1$ . Create a new codebook list  $\mathbf{c}_K$  by setting
        - $l_K \leftarrow 1$
        - $\mathcal{C}^r \leftarrow \mathcal{C}^r \cup (\varphi_j, l_K)$ .
      - iv. Otherwise, update the matched codeword  $\mathbf{c}_m$  by setting
        - $l_m \leftarrow l_m + 1$
        - $\varphi_m[i] = \gamma_b \times \varphi_j[i] + (1 - \gamma_b) \times \varphi_m[i]$ , where  $\gamma_b$  is the background learning rate.
    - (c) **endfor**
  3. **end for**
  4. For each codeword  $c_k$ ,  $k = 1 \dots K$ , normalised the histogram to 1
-

The first term  $P(\mathbf{S}|\mathbf{F}, \Gamma)$  describes the local probability function to infer whether a keypoint belongs to either *foreground* or *background*; while the second term  $P(h|\mathbf{S})$  provides an inference for the seat occupancy based on the spatial relationship of all the keypoints.

### 3.2.1.3 Estimating Local Keypoint Probability – $P(\mathbf{S}|\mathbf{F}, \Gamma)$

Locally, we treat all keypoints independently. Hence, we can obtain the probability of each keypoint being *foreground* or *background* in Equation 3.6:

$$P(\mathbf{S}|\mathbf{F}, \Gamma) = \prod_{i=1}^N P(s_i|f_i, \Gamma) \quad (3.6)$$

where each  $P(s_i|f_i, \Gamma)$  is defined as:

$$P(s_i=0|f_i, \Gamma) = \begin{cases} l_k^* & \text{if } l_k^* < \epsilon \mid \min_{d_k} \{(l_k, d_k) = sd(f_i, \varphi_k)\} \\ 0 & \text{otherwise} \end{cases} \quad (3.7)$$

where  $sd(\cdot)$  is a function that measures the descriptor difference  $d_k$  between  $f_i$  and  $(\{\varphi_k\} : \text{for each } \varphi_k \in \Gamma)$ . Equation 3.7 assigns the probability of a keypoint being a *background* as  $l_k^*$  if  $l_k^*$  satisfies the minimum threshold constraint  $\epsilon$ , and 0 otherwise.

### 3.2.1.4 Estimating Seat Occupancy Probability – $P(h|\mathbf{S})$

A seat occupancy probability is formulated in Equation 3.8:

$$P(h|\mathbf{S}) = P(h|s_1, \dots, s_N) \quad (3.8)$$

One way to estimate the probability is by using the spatial information. We express this probability as:

$$P(h = 1|s_1, \dots, s_N) = \frac{A^{fg}}{A^{fg} + A^{bg}} \quad (3.9)$$

where  $A^{fg}$ ,  $A^{bg}$  are areas of the foreground and background respectively. To compute the area of the foreground  $A^{fg}$ , we first filter out noisy *foreground keypoints* using a morphological operation. Next, we perform K-Nearest Neighbour clustering on all remaining *foreground keypoints*. Once the clusters are formed, we perform a convex hull algorithm and use the result to calculate the foreground area. If there is more than one cluster, we add them to define the final area of the foreground. The area of the background is simply the difference of total seat area and the area of the foreground.

With the formulation of both local and global probabilities, the seat occupancy is computed as a product of Equations 3.7 and 3.9.

### 3.2.2 Human Detection

Due to the general arrangements of the seats, it is expected that only the upper parts of passengers are generally visible to the camera, and direct pedestrian detection methods as in (Zhe *et al.*, 2007) cannot be employed. We therefore model each passenger using flexible models based on two ellipses corresponding to the head and body, as shown in Figure 3.5.

Let  $\psi$  be a single human configuration model, which is decomposed into  $\psi = (e^{head}, e^{body}, d, \theta, r)$ : the head and body ellipse configuration  $(e^{head}, e^{body})$ ; the distance between the centroid of head and body  $d$ ; the angle between the head and body  $\theta$ ; and the size ratio between the head and body  $r$ . Hence, for a candidate head pixel  $\mathbf{x}$  in the image, the likelihood of a human being detected in Equation 3.3 can be expression in Equation 3.10:

$$P(I|h) = P(I|\mathbf{x}, \psi) = \prod_E P(I|\mathbf{x}, E)P(I|\mathbf{x}, d, \theta, r) \quad (3.10)$$

where  $E \in \{e^{head}, e^{body}\}$ . The first term  $P(I|\mathbf{x}, E)$  is goodness of fit for the head and body ellipses being fitted onto the image, while the second term  $P(I|\mathbf{x}, d, \theta, r)$  penalizes large changes in terms of the relative distance, angle, and size ratio between the head and body.

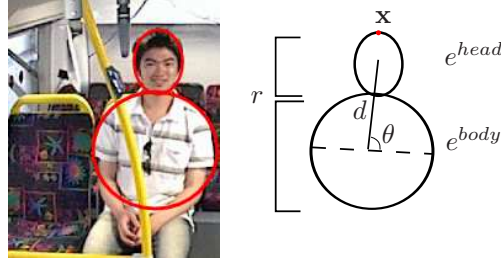


Figure 3.5: Human configuration model based on ellipse fitting.

The ellipse fitting score is computed as the normalized sum of the dot product between the gradient direction image and the unit normal ellipse image similar to Birchfield (1998):

$$P(I|\mathbf{x}, E) = D_{ellipse}(\mathbf{x}, I_s, E)$$

$$D_{ellipse}(\mathbf{x}, I_s, E) = \frac{1}{N_E} \sum_{\mathbf{y} \in E} |E(\mathbf{y}) \cdot I_s(\mathbf{x} + \mathbf{y})| \quad (3.11)$$

where  $E(\mathbf{y})$  is the unit vector normal to the ellipse image at pixel  $\mathbf{y}$ ,  $I_s(\mathbf{x} + \mathbf{y})$  is the gradient intensity direction of image  $I$  at the corresponding pixel  $(\mathbf{x} + \mathbf{y})$ ,  $\mathbf{y}$  is the pixel that passes the perimeter of the ellipse image,  $(\cdot)$  denotes the dot product, and  $N_E$  is the number of pixels on the perimeter of an ellipse.

The second term in Equation 3.10 is defined as the sum of all three penalty functions for the distance, orientation, and the ratio between the head and the body:

$$P(I|\mathbf{x}, d, \theta, r) = \exp \left\{ \sum_q -D_q(q) \right\} \quad (3.12)$$

Let  $q \in \{d, \theta, r\}$  be the distance, orientation, and ratio between head and body ellipses, then  $D_q(\cdot)$  is defined as:

$$D_q(q) = \begin{cases} |q - \varepsilon_q| & \text{if } |q - \varepsilon_q| > \alpha_q \\ 0 & \text{otherwise} \end{cases} \quad (3.13)$$

where  $\varepsilon_q$  is the control parameter for  $q \in \{d, \theta, r\}$  relationship between head, body and  $\alpha_q$  is the corresponding threshold. Hence, the best human model fit can be defined as follows:

$$\psi^* = \arg \max_{\psi \in \Psi} P(I|\mathbf{x}, \psi) \quad (3.14)$$

where  $\Psi$  contains the human configuration space. Intuitively, this strategy searches for the best fitted ellipses for the head and body, as well as the optimal head-body distance constraints. From Equations 3.3 and 3.14, the final likelihood of a human being detected can be expressed as:

$$P(I|h) = P(I|\mathbf{x}, \psi^*) \quad (3.15)$$

Using Equations 3.7, 3.9, and 3.15, the joint probability for passenger detection can be computed.

### Seat occupancy and human detection output

The output of seat occupancy detection is a probability image, where each pixel contains a probability value (0 to 1) of it being a foreground (the higher the probability value of the pixel, the higher the likelihood that it is a foreground pixel). Figure 3.6 plots the probability of the seat occupancy detection results alone. Areas with the peaks represent higher likelihood that they are occupied by the foreground objects. As seen in Figure 3.6, when a foreground exists on a particular seat, the probabilities of the areas being occupied are also high, indicating that the seat is occupied. In Figures 3.6 (d) and (e), the seat occupancy probability shows peaks in areas where the seat is being occupied, despite the change in illuminations. In the same sequence illustrated in Figure 3.6, we also plot the head detection results in Figure 3.7 (the result is after applying a threshold to eliminate the small probability of ellipse fittings). As seen, the human detection is able to detect the heads correctly in almost all sequences. But using the head detection alone generally results in several false positives, especially in areas on the windows and the bus handles, which resemble the shape of ellipses. In Section 3.4, the unified results of seat occupancy and human detection methods will be used to represent the final output of the detection system.

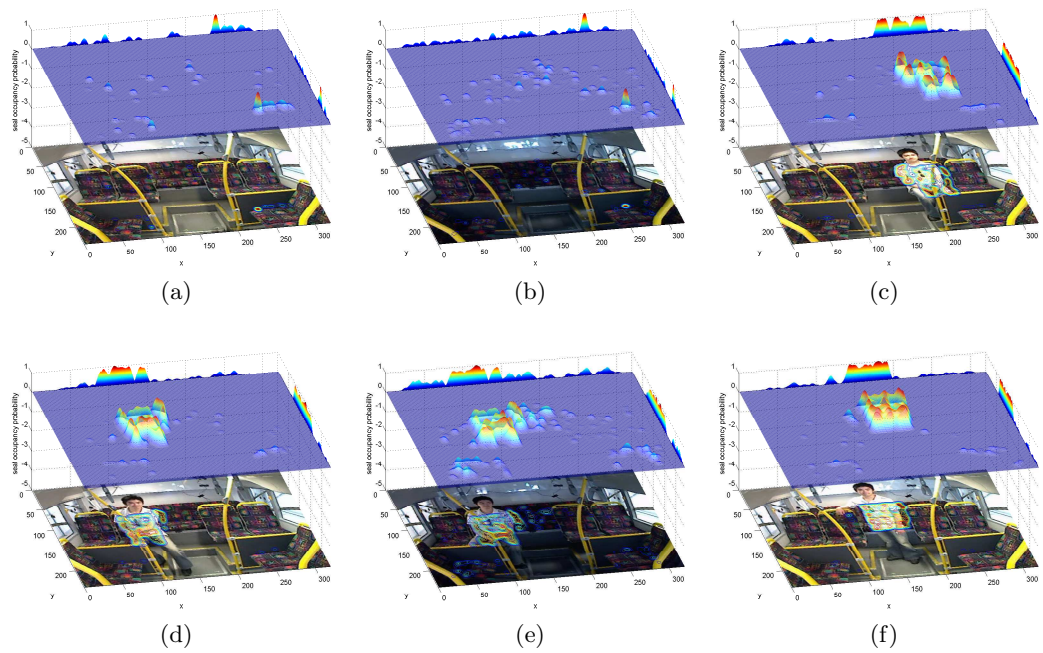


Figure 3.6: Output of seat occupancy probability.

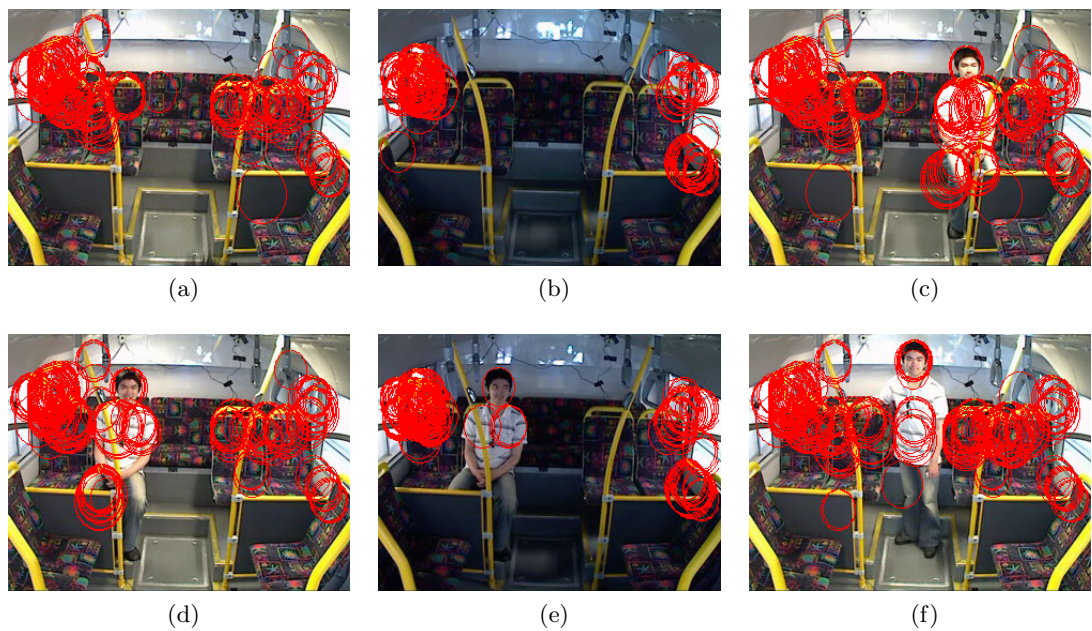


Figure 3.7: Output of head detection probability.

### 3.3 Object Tracking

In the previous section, we discussed a surveillance system to detect passengers in a moving bus environment. The output of the system is a set of passengers/foreground objects with elliptical shapes of the head and body. In this section, we aim to track these foregrounds under low frame rates (LFR) and different lighting change situations. This problem is complicated given that (1) the object trajectory is not smooth, (2) it may look different between two consecutive sparsely sampled frames, and (3) the overall process involves tracking multiple moving targets.

In the context of probabilistic tracking, this relates to the design of the proposal distribution, the observation model, and the tracking framework. The proposal distribution defines *where to look for the target*, while the observation model computes *the likelihood for a target being present at a certain location*. Intuitively, a good proposal distribution provides close positional estimation around the true target location, and a good foreground observation model should possess distinctive attributes and correct scoring function that separates different objects based on their appearances. Finally, the proposal distribution and the observation model can be combined together in different ways, for example Particle Filter (Leoputra *et al.*, 2006), MCMC (Khan *et al.*, 2005), or Reversible-Jump MCMC (Smith *et al.*, 2005), to tackle problems suited for the application.

Under the low frame rate and different lighting situations, previous algorithms for tracking (Comaniciu and Meer, 2002; Xiao and Shah, 2005; Khan *et al.*, 2005) will fail for two reasons: (1) most proposal distributions are modelled using linear kinematic models or second-order regressive models that have limited search space. This assumption often does not hold in the low frame rate situations where abrupt motions exist. (2) observation models that use colour or HSV is unable to capture distinctive attributes of target appearance under varying lighting situations. To deal with these problems, we propose a new tracking algorithm, called RJMCMC-ATMT (Reversible Jump MCMC with Appearance Transformation and Trajectory Model) tracking. Our contributions include:

1. A proposal distribution that incorporates the *passenger trajectory model* to allow prediction of the object movements according to the bus environment. The passenger trajectory model defines the allowable seat transitions for a foreground object for two consecutive frames under the low frame rate situations. It allows the tracker to explore larger search spaces, and at the same time, eliminates impossible state transitions.

2. An *appearance transformation model* that learns the similarity measure for a pair of objects under low frame rate situations. The algorithm consists of three steps: (a) it characterises the appearance feature of tracked objects by the cumulative normalised colour histogram (Javed *et al.*, 2005), histogram of oriented of gradients (HOG) (Dalal and Triggs, 2005), and its spatiogram (Birchfield and Rangarajan, 2005); (b) for each pair of the same foreground, we compute its histogram transformation as the feature vectors; (c) these feature vectors are trained using a probabilistic SVM to form the appearance transformation model.
3. Besides the low frame rate and different lighting conditions, the bus environment is generally filled with varying numbers of passengers due to occlusions, false positive detections, people entering and leaving the bus. To handle this situation, we employ a Reversible Jump MCMC tracking algorithm (Smith *et al.*, 2005) and show how the proposal distribution and observation model can be integrated into the RJMCMC tracking framework.

### 3.3.1 Bayesian Filtering Distribution for Object Tracking

Given a hidden state sequence  $\mathbf{X}_{1:t} = \{\mathbf{X}_1, \dots, \mathbf{X}_t\}$ ,  $\mathbf{X}_t \in \mathbb{R}^{\mathbf{n}_x}$  (for example position, size, orientation, velocity, etc.) and an observation sequence  $\mathbf{Z}_{1:t} = \{\mathbf{Z}_1, \dots, \mathbf{Z}_t\}$ ,  $\mathbf{Z}_t \in \mathbb{R}^{\mathbf{n}_z}$  (the actual measurement/observations: pixel intensities, low level features, etc) as shown in Figure 3.8, where  $\mathbf{n}_x$  and  $\mathbf{n}_z$  are the dimensions for the state and observation respectively, the multiple target tracking problem can be expressed in Bayesian Filtering formulation. This involves solving the *prediction* and *update* equations, which can be written as:

$$p(\mathbf{X}_t | \mathbf{Z}_{1:t-1}) = \int p(\mathbf{X}_t | \mathbf{X}_{t-1}) p(\mathbf{X}_{t-1} | \mathbf{Z}_{1:t-1}) d\mathbf{X}_{t-1} \quad (3.16)$$

and

$$p(\mathbf{X}_t | \mathbf{Z}_{1:t}) = \frac{p(\mathbf{Z}_t | \mathbf{X}_t) p(\mathbf{X}_t | \mathbf{Z}_{1:t-1})}{\int p(\mathbf{Z}_t | \mathbf{X}_t) p(\mathbf{X}_t | \mathbf{Z}_{1:t-1}) dx_t} \quad (3.17)$$

The denominator of Equation 3.17 is a normalising constant that can be defined as  $C$ . By substituting Equation 3.16 into Equation 3.17, we arrive at the recursive Bayesian Filtering distribution, which is in the form of Equation 3.18:



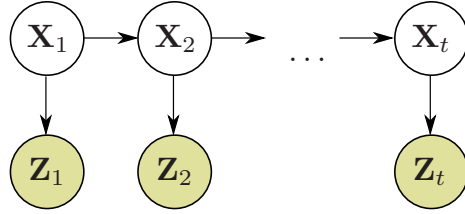


Figure 3.8: A graphical model representation for target tracking, where  $\mathbf{X}_t$  is the hidden state configuration and  $\mathbf{Z}_t$  is the observation at time  $t$ .

$$p(\mathbf{X}_t|\mathbf{Z}_{1:t}) = C^{-1}p(\mathbf{Z}_t|\mathbf{X}_t) \times \int p(\mathbf{X}_t|\mathbf{X}_{t-1})p(\mathbf{X}_{t-1}|\mathbf{Z}_{1:t-1})d\mathbf{x}_{t-1} \quad (3.18)$$

Here the *motion model*  $p(\mathbf{X}_t|\mathbf{X}_{t-1})$  predicts the current state  $\mathbf{X}_t$  given the previous state  $\mathbf{X}_{t-1}$ , the *likelihood* or *observation model*  $p(\mathbf{Z}_t|\mathbf{X}_t)$  defines the probability we observe the measurement  $\mathbf{Z}_t$  given the state  $\mathbf{X}_t$  at time  $t$ , and  $p(\mathbf{X}_t|\mathbf{Z}_{1:t})$  is the *posterior distribution* at time  $t$ .

### 3.3.2 Independent Particle Filter

Sequential Monte Carlo or Particle Filter (PF) provides an approximate solution to the recursive Bayesian Filter distribution in Equation 3.18 by using a finite set of weighted samples called particles, which is expressed as:

$$p(\mathbf{X}_t|\mathbf{Z}_{1:t}) \approx C^{-1}p(\mathbf{Z}_t|\mathbf{X}_t) \sum_{n=1}^N w_{t-1}^{(n)}p(\mathbf{X}_t|\mathbf{X}_{t-1}^{(n)}) \quad (3.19)$$

where each particle is a 2-tuple consisting of the state domains and its corresponding probability (weight), and is denoted by  $\{\mathbf{X}_t^{(n)}, w_t^{(n)}\}_{n=1}^N$ , where  $n$  is the particle number (index), and  $N$  is the number of particles. Note that the continuous posterior pdf in Equation 3.18 is now approximated by a set of discrete samples in Equation 3.19 (a change from integral into summation). In the standard Particle Filters algorithm, at each time instance,  $\mathbf{X}_t^{(n)}$  represents the random target's state that is drawn from a proposal distribution  $q(\cdot)$ :

$$\mathbf{X}_t^{(n)} \sim q(\mathbf{X}_t|\mathbf{X}_{1:t-1}^{(n)}, \mathbf{Z}_{1:t}) = p(\mathbf{X}_t|\mathbf{X}_{t-1}^{(n)}) \quad (3.20)$$

where  $q(\cdot)$  is normally defined according to the *motion model*, and the weight of each particle is defined by the *observation model*.

$$w_t^{(n)} \propto p(\mathbf{Z}_t | \mathbf{X}_t^{(n)}) \quad (3.21)$$

To avoid the degradation problem (Gordon *et al.*, 1993), a resampling algorithm is generally applied. It is a procedure to initiate the large-weight particles prediction into small-weight particles. As a result, all particles will converge to a better prediction.

In low frame rate situations, one of the challenges is poor temporal continuity, which means the standard proposal distribution that relies only on the motion model of the previous state is generally error prone. Next, we discuss a new proposal distribution which incorporates the *trajectory model* for LFR tracking in the bus.

### 3.3.3 Proposal Distribution with Trajectory Model for LFR Tracking

We generalise the proposal distribution to include prior information about the environment, similar to Leputra *et al.* (2006). This new proposal distribution can be expressed as:

$$\mathbf{X}_t^{(n)} \sim q_\Lambda(\mathbf{X}_t | \mathbf{X}_{1:t-1}^{(n)}, \mathbf{Z}_{1:t}) = p(\mathbf{X}_t | \mathbf{X}_{t-1}^{(n)}, \Lambda) \quad (3.22)$$

where  $\Lambda$  represents the environment model. Note that both the *proposal distribution* and the *motion model* in Equation 3.22 incorporate an additional parameter  $\Lambda$  as compared to the standard PF's proposal distribution in Equation 3.20. Equation 3.22 also implies that the environment information contributes to the behaviour (search space) of the particles apart from the previous state information. In practice,  $\Lambda$  can be implemented in a number of ways: (1) it can be used to ignore some areas in the environment which are impossible to reach, for example, the walls between rooms where targets cannot move through, or (2) it can define a set of allowable target trajectories in a structured environment so that the particles move according to the path. When there is no constraint about the environment, we return to the original proposal distribution.

Inside the bus, the passenger's movements generally follow certain trajectories due to the

structured nature of the environment, for example: (1) a passenger who sits close to the window should appear in between seats or at the aisle before leaving the bus; (2) it is unlikely that a passenger will move from one side of the bus to the other side of the bus in two consecutive frames. Based on these characteristics, we propose a trajectory model that defines a set of allowable passenger movements (seat transitions) inside the bus. Figure 3.9 shows the trajectory model that is used in our experiment, modelled as a directed graph. Each node corresponds to a seat where a passenger normally exists and the edges connecting the nodes specify the *likelihood* of a passenger movement in two consecutive frames. In this model, a passenger who is currently occupying seat A is likely to stay at seat A, or move to seat B or C on the next frame, but it is unlikely for him to reach seat D, E, F, or G between two consecutive frames. Our new proposal distribution uses this trajectory model to predict the target’s movement. Intuitively, the trajectory model allows exploring larger search spaces, whilst eliminating impossible state transitions.

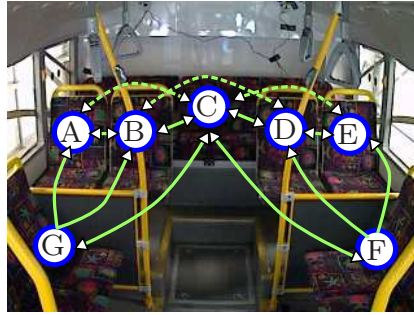


Figure 3.9: The trajectory model defines the possible passenger movements (seat transition) between two consecutive frames. Note that each node has a loop transition to itself that is not shown in the figure for simplicity purposes

In the implementation, let  $\Lambda$  be the trajectory model for the bus. Each edge (seat transition) in  $\Lambda$  is assigned with a prior probability defining the likelihood of the passenger to move there between two consecutive frames. Each time the proposal distribution method is evoked, we pick a particle  $\mathbf{X}_{t-1}^{(n)}$ , and then generate a uniform random number between 0 and 1. Based on where the random number falls, a new seat destination is chosen. We move the particle’s location relative to the new seat destination and run the sampling distribution (e.g. Gaussian, second order regressive model, etc) to obtain a new state particle at time  $t$ ,  $\mathbf{X}_t^{(n)}$ . This strategy allows the particles to move according to the seat transition defined in Figure 3.9. In the remaining subsections we will use the new proposed distribution inside the tracking framework. (Note that to refer back to the standard proposal distribution, we can simply discard the  $\Lambda$  notation).

So far, we have shown the use of Particle Filter for a single object tracking. In tracking multiple targets, the standard Particle Filter normally involves creating multiple *independent* trackers that do not communicate with each other because the algorithm does not model any form of interaction (Khan *et al.*, 2005). This leads to high failure when multiple targets with the same appearance model are close to each other. Next, we will discuss Particle Filter for multiple interacting objects.

### 3.3.4 Particle Filter for Interacting Objects

#### 3.3.4.1 State Definitions for Multi-object Tracking

Following the notations from Smith *et al.* (2005), let  $\mathbf{X}_t = \{\mathbf{X}_{i,t}, i \in \mathcal{O}_t\}$  be the multiple-object configuration at time  $t$ , where  $i$  defines the object index,  $\mathcal{O}_t$  is the set of object indexes,  $m_t = |\mathcal{O}_t|$  denotes the number of objects ( $m_t \in \mathcal{M} = \{0, \dots, M\}$ , where  $M$  be the maximum number of objects and  $|\cdot|$  indicates set cardinality).  $\mathbf{X}_{i,t}$  denotes a single-object configuration and  $\mathbf{X}_{i,t} = \emptyset$  denotes the zero object configuration. This way, the notation allows varying numbers of targets, as in Smith *et al.* (2005). Figure 3.10 shows the graphical model for multi-object tracking.

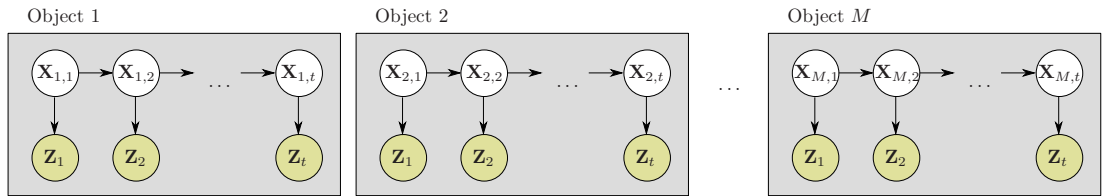


Figure 3.10: A graphical model representation for multiple targets tracking, where  $\mathbf{X}_t$  is the multi-object configuration and  $\mathbf{Z}_t$  is the observation.

#### 3.3.4.2 Interaction Model

To model the object interactions, Khan *et al.* (2005) proposed a pairwise Markov Random Field (MRF) prior in the dynamic model. The MRF is an undirected graph where nodes represent the objects, and edges between nodes define the connection of a pair of objects based on their proximity (Smith *et al.*, 2005). The joint probability of the MRF is factored as a product of local potential functions at each node, and interactions are defined on neighbourhood cliques (Khan *et al.*, 2005). The dynamic model can be expressed as:

$$p(\mathbf{X}_t | \mathbf{X}_{t-1}, \Lambda) \propto \prod_i p(\mathbf{X}_{i,t} | \mathbf{X}_{i,t-1}, \Lambda) p_0(\mathbf{X}_t) \quad (3.23)$$

where  $p(\mathbf{X}_{i,t} | \mathbf{X}_{i,t-1}, \Lambda)$  is the dynamics of the  $i$ -th object and the prior  $p_0(\mathbf{X}_t) = \prod_{ij \in \mathcal{C}} \psi_\Lambda(\mathbf{X}_{i,t}, \mathbf{X}_{j,t})$ , is a product of pairwise interaction potentials over  $\mathcal{C}$ , the set of cliques (i.e. pairs of connected nodes) in the graph (Smith *et al.*, 2005). Figure 3.11 shows an example of the MRF configuration, where the edges between the passengers represent the MRF connection between the targets. The absence of an edge in the MRF implies that the two targets do not influence each other's behaviour (Khan *et al.*, 2005).  $\psi_\Lambda(\mathbf{X}_{i,t}, \mathbf{X}_{j,t})$  is the *interaction model* and is defined as

$$\psi_\Lambda(\mathbf{X}_{i,t}, \mathbf{X}_{j,t}) \propto \exp(-g(\mathbf{X}_{i,t}, \mathbf{X}_{j,t})) \quad (3.24)$$

where  $g(\mathbf{X}_{i,t}, \mathbf{X}_{j,t})$  is a penalty function. In our passenger tracking problem, the penalty function is defined as the number of pixel overlaps between the body bounding boxes of the two targets. Intuitively, the interaction model places constraints to avoid two or more trackers being hijacked by a dominant object that has the highest likelihood model amongst the others.

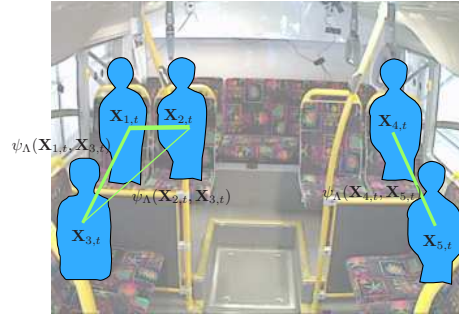


Figure 3.11: Interaction model for passengers inside the bus.

### 3.3.4.3 Dynamic Model for Variable Numbers of Targets

Similar to Smith *et al.* (2005), the dynamic model can be defined for a variable number of targets as in:

$$p(\mathbf{X}_t|\mathbf{X}_{t-1}, \Lambda) \propto \prod_{i \in \mathcal{O}_t} p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1}, \Lambda) p_0(\mathbf{X}_t) \quad (3.25)$$

$$= p_V(\mathbf{X}_t|\mathbf{X}_{t-1}, \Lambda) p_0(\mathbf{X}_t) \quad (3.26)$$

where

$$p(\mathbf{X}_{i,t}|\mathbf{X}_{t-1}, \Lambda) = \begin{cases} p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1}, \Lambda) & \text{if object } i \text{ existed in the previous frame} \\ p(\mathbf{X}_{i,t}) & \text{if object } i \text{ did not exist in the previous frame} \\ k & \text{for the zero case object (where } k \text{ is a constant)} \end{cases} \quad (3.27)$$

Based on Equations 3.23, and 3.26, the Monte Carlo approximation to the Bayesian Filtering distribution in Equation 3.19 can be written as:

$$p(\mathbf{X}_t|\mathbf{Z}_{1:t}) \approx C^{-1} p(\mathbf{Z}_t|\mathbf{X}_t) \prod_{ij \in \mathcal{C}} \psi_\Lambda(\mathbf{X}_{i,t}|\mathbf{X}_{j,t}) \sum_{n=1}^N w_{t-1}^{(n)} p_V(\mathbf{X}_t|\mathbf{X}_{t-1}^{(n)}, \Lambda) \quad (3.28)$$

Since Equation 3.28 involves computing the joint particle filter where each particle contains the joint position of *all* the targets, it suffers from exponential complexity as the number of tracked targets,  $n$ , increases (Khan *et al.*, 2004).

### 3.3.5 Reversible Jump MCMC Algorithm for Variable Numbers of Targets

To solve the inference problem in Equation 3.28, Khan *et al.* (2004) propose a method to approximate the posterior distribution for a fixed target number tracking through MCMC sampling techniques using the Metropolis Hasting (MH) algorithm. The MH algorithm improves the multi-object configuration by implementing a random walk over a long run (iterations). To ensure that it improves the configuration, the MH algorithm keeps the best state configuration at the end of each iteration. At a new iteration, it samples a random state configuration from the proposal distribution  $q(\mathbf{X}^*|\mathbf{X})$ , where  $\mathbf{X}$  and  $\mathbf{X}^*$  denote the best and proposed state configuration respectively, and computes the acceptance ratio

between the two configurations, which can be expressed as:

$$\alpha = \min \left( 1, \frac{p(\mathbf{Z}|\mathbf{X}^*)q_{\Lambda}(\mathbf{X}|\mathbf{X}^*)}{p(\mathbf{Z}|\mathbf{X})q_{\Lambda}(\mathbf{X}^*|\mathbf{X})} \right) \quad (3.29)$$

Based on the acceptance ratio, it selects either to *accept* the proposed state configuration or to *stay* at the best configuration from the previous iteration because the proposed configuration does not improve the chain. The *chosen* state configuration is then inserted into the Markov Chain and kept as the best configuration for this iteration. After several iterations, the MH algorithm reaches a *peak* configuration (when no other state configuration is better) and stays irreplaceable. This way, the Markov Chain is filled dominantly with the *peak* configurations at the end of the iterative process. Hence, the final posterior distribution for the multi-object configuration can be estimated by either finding the mean or the mode of the whole state configuration chains. Note that because the initial iterations of random walk generally return unstable measurements, they are discarded from the chain; this process is called *burn-in*. With this formulation, the inference in Equation 3.28 can be solved efficiently.

From the above explanations, we know that the two major components in MH algorithm are the proposal distribution and the acceptance ratio (observation model). It is important to design them correctly because: (1) a bad proposal distribution produces samples that do not overlap with the true location of the target, hence the tracker will fail to track the target despite having a good observation model, and (2) a good proposal distribution with a bad observation model will sample locations that are close to the targets, but the tracker is unable to accurately “recognise” the target, which results in tracking failure. Hence, the ideal tracking algorithm is to produce samples around the true target locations and to be able to recognise them. The sooner it finds the true state configurations for all the targets, the better the posterior estimation.

Smith *et al.* (2005) extended the approach further to handle varying numbers of targets by using trans-dimensional MCMC (also known as reversible jump MCMC) techniques. Following from the work of Smith *et al.* (2005), the reversible jump MCMC with Metropolis Hasting is implemented as follows: First, we define a set of moves, denoted as  $\Upsilon$  (indexed by  $v$ ) and its prior  $p_v$ , along with the proposal distribution for each move  $\{q_v(\cdot)\}$ . Secondly, the *move-specific* proposal distributions are functions of an auxiliary variable  $\mathbf{U}$ . At each MH iteration, a new move  $v^*$  is chosen with probability  $p_{v^*}$  and a new state  $\mathbf{X}^*$  is defined by a deterministic function of the current state, a new sample of the auxiliary variable, and the trajectory model drawn from  $q_{v^*}(\mathbf{U}^*)$ ,  $\mathbf{X}^* = h(\mathbf{X}, \mathbf{U}^*)$ . The reverse move  $v$  from

$\mathbf{X}^*$  to  $\mathbf{X}$  is computed by sampling  $\mathbf{U}$  from  $q_v(\mathbf{U})$  with  $\mathbf{X} = h'(\mathbf{X}^*, \mathbf{U})$ . The acceptance ratio is then defined as:

$$\alpha_v = \min \left( 1, \frac{p(\mathbf{Z}_t | \mathbf{X}_t^*)}{p(\mathbf{Z}_t | \mathbf{X}_t)} \frac{p_v}{p_{v^*}} \frac{q_v(\mathbf{U})}{q_{v^*}(\mathbf{U}^*)} \left| \frac{\partial(\mathbf{X}^*, \mathbf{U})}{\partial(\mathbf{X}, \mathbf{U}^*)} \right| \right) \quad (3.30)$$

Because one of the requirements for the reversible jump MCMC is to keep balance for each move-specific type (i.e. by defining the reverse move for each move-specific type), it can be shown that the Jacobian matrix is equal to 1 (Smith *et al.*, 2005).

Similar to Smith *et al.* (2005), we define four types of moves to handle the varying numbers of targets tracking. Two moves involve jumping across dimensions and two in a fixed dimension:

1. *Birth (b)* of a new object implies a change in the dimension from  $m_t$  to  $m_t + 1$ .
2. *Death (d)* of an existing object implies a change in the dimension from  $m_t$  to  $m_t - 1$ .
3. *Swap (s)* between two existing objects, where the dimension remains unchanged  $m_t$ .
4. *Update (u)* of the parameters of the existing objects, where the dimension is fixed  $m_t$ .

Figure 3.12 illustrates the four different types of moves.



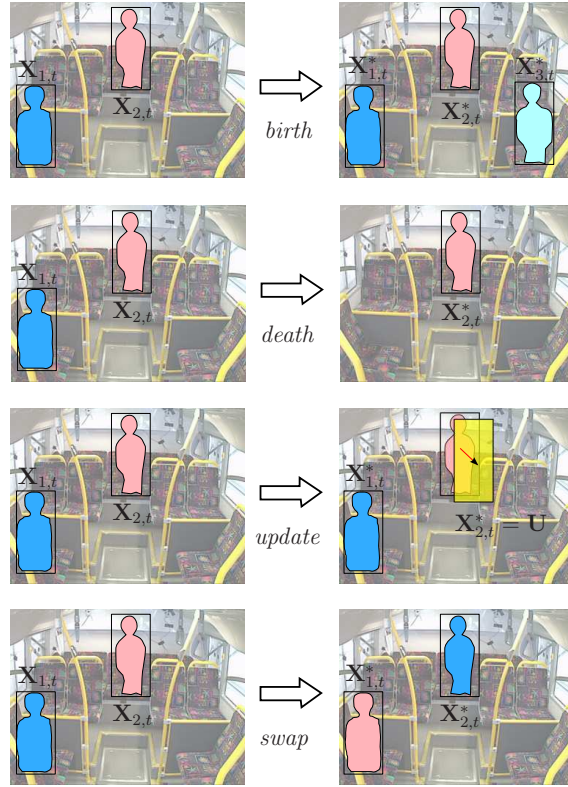


Figure 3.12: The four different moves for RJMCMC: *birth*, *death*, *update*, and *swap*.

Once the prior distribution for each move has been defined ( $\{p_b, p_d, p_s, p_u\}$ ), the RJMCMC-PF can be summarised as in Algorithm 2.

Table 3.1 summarises the differences between independent PF, MCMC, and RJMCMC tracking algorithms. Because RJMCMC allows change in the dimensions, it is able to simulate varying numbers of target configurations; and with the appropriate design of the acceptance ratio it is able to reach the best multi-object state configurations. This advantage is not found in either the independent PF or MCMC tracking. Next we will present the move-specific proposal distribution for RJMCMC followed by our proposed observation model.

### 3.3.6 Move-Specific Proposal Distribution and Acceptance Ratio

The sampling method for the *birth*, *death*, and *update* can be written as follows:

**Algorithm 2:** The Reversible Jump Markov Chain Monte Carlo Particle Filter (RJMCMC PF).

---

- 1 At each time step,  $t$ , the posterior distribution for the previous time step is represented by a set of  $N$  unweighted samples  $p(\mathbf{X}_{t-1}|\mathbf{Z}_{1:t-1}) \approx \{\mathbf{X}_{t-1}^{(n)}\}_{n=1}^N$ . The approximation of the current distribution  $p(\mathbf{X}_t|\mathbf{Z}_{1:t})$  is constructed as follows.
- 2 Initialise the Markov Chain by randomly selecting a sample from  $t - 1$  with the mode number of objects. Apply the motion model to each object  $\mathbf{X}^* = \mathbf{X}^{n-1}$  and accept as sample  $n = 0$ .
- 3 RJMCMC Sampling Step: Draw  $N + N_B$  samples according to the following schedule, where  $N_B$  is the length of the burn-in period:
  1. Choose a move type by sampling from the set of moves  $v \sim U[0, 1]$  where  $\Upsilon = \{b, d, s, u\}$ .
    - (a) if  $0 \leq u \leq p_b$ ,  $v^* = b$ .
    - (b) else if  $p_b < u \leq p_b + p_d$ ,  $v^* = d$ .
    - (c) else if  $p_b + p_d < u \leq p_b + p_d + p_s$ ,  $v^* = s$ .
    - (d) else  $v^* = u$ .
  2. Select a target  $i^*$  (or targets  $i^*, k^*$  for a swap) according to the target proposal distribution  $q_{v,\Lambda}(i)$  (or  $q_{v,\Lambda}(i, k)$ ) for the selected move type.
  3. Propose a new configuration  $\mathbf{X}_t^*$  from the move-specific proposal distribution defined for the chosen move type. For the various moves, this implies:
    - (a) Birth - add a new object  $\mathbf{X}_{i^*,t}^*$  sampled from the proposal distribution in Equation 3.33.
    - (b) Death - removing an existing  $\mathbf{X}_{i^*,t}^*$  sampled from the proposal distribution in Equation 3.35.
    - (c) Update - update the parameters of object  $\mathbf{X}_{i^*,t}^*$  sampled from the proposal distribution in Equation 3.37.
    - (d) Swap - swap the parameters of objects  $\mathbf{X}_{i^*,t}^*$  sampled from the proposal distribution in Equation 3.39.
  4. Compute the acceptance ratio  $\alpha_v$  for the chosen move type (*birth*: Equation 3.34, *death*: Equation 3.36, *update*: Equation 3.38, and *swap*: Equation 3.40).
  5. Accept/Reject. Accept the proposal  $\mathbf{X}_t^*$  if  $\alpha \geq 1$ , and add it to the Markov Chain  $\mathbf{X}_t^{(n)} = \mathbf{X}_t^{*(n)}$ , otherwise add the previous sample in the Markov Chain to the current position.

Compute MAP estimate  $\mathbf{X}_t^{MAP}$ .

---

	Proposal distribution	Computing likelihood
Independent PF (IPF)	$q(\mathbf{X}_{i,t} \mathbf{X}_{i,t-1})$ , samples the new state according to the previous state.	$p(\mathbf{Z}_t \mathbf{X}_{i,t})$ , the observation model is computed independently for each tracker $i$ .
MCMC trackers	$q(\mathbf{X}_t \mathbf{X}_{t-1})$ . Here $\mathbf{X}_t$ is the multi-object configuration as compared to IPF tracker where $\mathbf{X}_{i,t}$ is the state for a particular object $i$ .	$\min\left(1, \frac{p(\mathbf{Z}_t \mathbf{X}_t^*) p_0(\mathbf{X}_t^*) q(\mathbf{X}_t \mathbf{X}_t^*)}{p(\mathbf{Z}_t \mathbf{X}_t) p_0(\mathbf{X}_t) q(\mathbf{X}_t^* \mathbf{X}_t)}\right)$ , where the observation model is computed based on all the multi-object configuration and there is an interaction model $p_0(\mathbf{X}_t) = \prod_{i,j \in \mathcal{C}} \psi(\mathbf{X}_{i,t} \mathbf{X}_{j,t})$ as compared to IPF tracker.
RJMCMC	$q_v(\mathbf{X}_t \mathbf{X}_{t-1})$ , with additional $v \in \Upsilon$ (move types) as compared to IPF and MCMC trackers.	$\min\left(1, \frac{p(\mathbf{Z}_t \mathbf{X}_t^*) p_0(\mathbf{X}_t^*) p_v \frac{q_v(\mathbf{U})}{q_v^*(\mathbf{U}^*)}}{p(\mathbf{Z}_t \mathbf{X}_t) p_0(\mathbf{X}_t) p_{v^*} \frac{q_v^*(\mathbf{U}^*)}{q_v^*(\mathbf{U}^*)}}\right)$ , where there is a dimension matching probability that allows us to move between different dimensions, as compared to MCMC.

Table 3.1: Difference between independent PF, MCMC, and RJMCMC for multiple target tracking.

$$q_{v,\Lambda}(\mathbf{X}_t^*) = \sum_i q_{v,\Lambda}(i) q_{v,\Lambda}(\mathbf{X}_t^*|i), \quad (3.31)$$

over all of the appropriate object  $i$ . At each time, an index  $i^*$  is chosen with probability  $q_{v,\Lambda}(i^*)$ , and a move is chosen with probability  $p_v$ . The move is applied to the object while keeping the other multi-object configuration unchanged. The mixture component of the new configuration  $\mathbf{X}_t^*$  is defined so that:

$$q_{v,\Lambda}(\mathbf{X}_t^*) = \begin{cases} \frac{1}{N} \sum_n p_V(\mathbf{X}_t^*|\mathbf{X}_{t-1}^{(n)}, \Lambda) & i = i^* \\ 0 & i \neq i^* \end{cases} \quad (3.32)$$

Defining the proposal in this manner cancels out all factors involving summations over the particles in the acceptance ratio, and keeps the algorithm computationally efficient (Smith *et al.*, 2005).

### Birth move

Adding an object  $i^*$  implies that  $\mathcal{O}_t^* = \mathcal{O}_t \cup \{i^*\}$ . The mixture component for the birth move can be expressed as:

$$q_{b,\Lambda}(\mathbf{X}_t^* | \mathbf{X}_t, \mathbf{U}, i) = \frac{1}{N} \sum_n p(\mathbf{X}_t^* | \mathbf{X}_{t-1}^{(n)}, \Lambda) \prod_{l \in \mathcal{O}_t} p(\mathbf{X}_{l,t} | \mathbf{X}_{t-1}^{(n)}) \delta_{X_{l,t}}(\mathbf{X}_{l,t}^* - \mathbf{X}_{l,t}) \quad (3.33)$$

When  $i^*$  is the index of the new object, chosen from the available objects, it can be shown that the acceptance ratio is expressed as (Smith *et al.*, 2005):

$$\alpha_b = \min \left( 1, \frac{p(\mathbf{Z}_t | \mathbf{X}_t^*) \prod_{j \in \mathcal{C}_{i^*}} \psi_\Lambda(\mathbf{X}_{i^*,t}^*, \mathbf{X}_{j,t}^*) p_d q_d(i^*)}{p(\mathbf{Z}_t | \mathbf{X}_t) p_b q_b(i)} \right) \quad (3.34)$$

Note that the interaction model for the new object is designed to discourage new births that overlap with an existing object (Smith *et al.*, 2005).

### Death move

This is the reverse move of the birth. Removing an object  $i^*$  implies that  $\mathcal{O}_t^* = \mathcal{O}_t - \{i^*\}$ . The mixture component for the death move can be expressed as:

$$q_{d,\Lambda}(\mathbf{X}_t^*, \mathbf{U}^* | \mathbf{X}_t, i) = \frac{1}{N} \sum_n p(\mathbf{X}_t^* | \mathbf{X}_{t-1}^{(n)}, \Lambda) \prod_{l \in \mathcal{O}_t} p(\mathbf{X}_{l,t} | \mathbf{X}_{t-1}^{(n)}) \delta_{X_{l,t}}(\mathbf{X}_{l,t}^* - \mathbf{X}_{l,t}) \quad (3.35)$$

It can be shown that the acceptance ratio for the death move can be reduced to (Smith *et al.*, 2005):

$$\alpha_d = \min \left( 1, \frac{p(\mathbf{Z}_t | \mathbf{X}_t^*)}{p(\mathbf{Z}_t | \mathbf{X}_t) \prod_{j \in \mathcal{C}_{i^*}} \psi_\Lambda(\mathbf{X}_{i^*,t}^*, \mathbf{X}_{j,t}^*)} \frac{p_b q_b(i^*)}{p_d q_d(i)} \right) \quad (3.36)$$

### Update move

After randomly choosing an existing object index  $i^*$ , the update move samples the target movement from the previous state information and the environment  $p(\mathbf{X}_{i^*,t}^* | \mathbf{X}_{t-1}^{(n)}, \Lambda)$  keeping all other configurations unchanged. The mixture component for the update move can be expressed as:

$$q_{u,\Lambda}(\mathbf{X}_t^*, \mathbf{U}^* | \mathbf{X}_t, \mathbf{U}, i) = \frac{1}{N} \sum_n p(\mathbf{X}_t^* | \mathbf{X}_{t-1}^{(n)}, \Lambda) \prod_{i \neq j} p(\mathbf{X}_{j,t}^* | \mathbf{X}_{t-1}^{(n)}) \delta_{X_{i,t}}(\mathbf{X}_{j,t}^* - \mathbf{X}_{j,t}) \quad (3.37)$$

The acceptance ratio for the update move is then written as:

$$\alpha_u = \min \left( 1, \frac{p(\mathbf{Z}_t | \mathbf{X}_t^*) \prod_{j \in \mathcal{C}_{i^*}} \psi_{\Lambda}(\mathbf{X}_{i^*,t}^*, \mathbf{X}_{j,t}^*)}{p(\mathbf{Z}_t | \mathbf{X}_t) \prod_{j \in \mathcal{C}_{i^*}} \psi_{\Lambda}(\mathbf{X}_{i^*,t}, \mathbf{X}_{j,t})} \right) \quad (3.38)$$

### Swap move

This move involves a pair of objects without changing dimension. The proposal is defined as a mixture model over all object pairs,

$$q_{s,\Lambda}(\mathbf{X}_t^* | \mathbf{X}_t) = \sum_{i,j \in \mathcal{O}_t} q_{s,\Lambda}(i, j) q_{s,\Lambda}(\mathbf{X}_t^* | \mathbf{X}_t, i, j) \quad (3.39)$$

A candidate  $\mathbf{X}_t^*$  is chosen by selecting a pair of existing object indexes  $i^*$  and  $j^*$  with probability  $p_s(i^*, j^*)$  and swapping their configurations. The acceptance ratio for the swap can be reduced to:

$$\alpha_s = \min \left( 1, \frac{p(\mathbf{Z}_t | \mathbf{X}_t^*)}{p(\mathbf{Z}_t | \mathbf{X}_t)} \right) \quad (3.40)$$

### 3.3.7 Observation Model (Appearance Transformation Model)

The observation model is used to measure the observation likelihood of the samples drawn from the proposal distribution, and thus is an important part of object tracking. In this work, we introduce a new observation model namely the appearance transformation model that learns the foreground appearance changes. To do this, we build cumulative normalised colour histograms (Javed *et al.*, 2005), normalised Histogram of Oriented Gradient (HOG) (Dalal and Triggs, 2005), and HOG's spatiogram (Birchfield and Rangarajan, 2005) from regions inside the detected foregrounds as the appearance feature vector. Then, for each pair of the same foreground appearances, we compute its histogram transformation and

use that as the input features to the SVM classifier to form the appearance transformation model. This classifier is then used during run time to determine the likelihood that two appearances belong to the same passenger.

### 3.3.7.1 Appearance Transformation Feature

We present the appearance transformation feature of an object as  $\mathbf{H} = (H, G, S)$ , where  $H$ ,  $G$ , and  $S$  are the cumulative colour (RGB) histogram (Javed *et al.*, 2005), the histogram of oriented gradients (Dalal and Triggs, 2005), and the HOG's spatiogram (Birchfield and Rangarajan, 2005) respectively (note that we discard the time and object index in this subsection for simplicity purposes). Let  $H = \{m_b\}$ ,  $G = \{n_b\}$ , and  $S = \{\mu_b, \Sigma_b\}$  where  $b = 1, \dots, B$  is the bin index of each histogram,  $m_b$  is the cumulative response of the colour histogram,  $n_b$  be the frequency value in the  $b$ -th bin of the HOG, and  $\mu_b$  and  $\Sigma_b$  be the mean vector and covariance matrices of the image coordinates that fall into each bin of the spatiogram. For each pair of foreground appearances, we compute the histogram transformation and use that as the input samples to the SVM classifier. The positive and negative samples contain transformation histograms of the same and different foregrounds respectively. Let  $\mathbf{H}$  and  $\widehat{\mathbf{H}}$  be the appearance features of two foregrounds. We compute the histogram transformation in three different ways:

1. **Inverted cumulative histogram for colour histograms (ICH)** (Javed *et al.*, 2005). This is written as:  $Z^H(H, \widehat{H}) = \forall_{b=1}^B H^{-1}(\widehat{H}(b))$ , where  $Z^H$  is the transformation feature of two cumulative histograms,  $H$  and  $\widehat{H}$ . The histogram inverse function on  $b^{\text{th}}$  bin is defined as:

$$H^{-1}(b) = \{\widehat{a} \in 1 \dots \widehat{B} : H(\widehat{a}) = m_b\}$$

This can be worded as follows: *the inverse of the  $b^{\text{th}}$  bin in the first histogram is the  $a^{\text{th}}$  bin in the second histogram with the condition that their cumulative values are equal, i.e.  $\widehat{m}_a = m_b$ .*

2. **Histogram differences for HOGs (DHOG)** (Dalal and Triggs, 2005) using the Bhattacharyya coefficient. This is expressed as:  $Z^G(G, \widehat{G}) = \forall_{b=1}^B \rho(n_b, \widehat{n}_b)$ , where  $Z^G$  is the transformation feature of two HOGs,  $G$  and  $\widehat{G}$ . The Bhattacharyya coefficient is defined as:

$$\rho(n_b, \widehat{n}_b) = \frac{\sqrt{n_b \widehat{n}_b}}{\sqrt{(\sum_{j=1}^B n_j)(\sum_{j=1}^B \widehat{n}_j)}}$$

3. **Covariance distance measure for spatiograms (SHOG)** (Forstner and Moonen, 1999). We define  $Z^S(S, \widehat{S}) = \forall_{b=1}^B \gamma(\Sigma_b, \widehat{\Sigma}_b)$ , where  $Z^S$  is the transformation feature of two spatiograms,  $S$  and  $\widehat{S}$ . The distance measure is defined as:

$i$	Feature, $\mathbf{Z} = \{Z^H, Z^G, Z^S\}$			Label
	ICH (RGB)	DHOG	SHOG	
1	[1...32][1...32][1...32]	[1...9]	[1...9]	{-1,1}
...				
$L$	[1...32][1...32][1...32]	[1...9]	[1...9]	{-1,1}

Table 3.2: SVM feature vectors.

$$\gamma(\Sigma_b, \widehat{\Sigma}_b) = \sqrt{\sum_{i=1}^d \ln^2 \lambda_i(\Sigma_b, \widehat{\Sigma}_b)}$$

where  $\{\lambda_i(\Sigma_b, \widehat{\Sigma}_b)\}_{i=1\dots d}$  are the generalized eigenvalues of  $\Sigma_b$  and  $\widehat{\Sigma}_b$ .

Hence, the final appearance transformation vector can be written as:

$$\mathbf{Z} = \{Z^H, Z^G, Z^S\} \quad (3.41)$$

Despite the simplicity of the proposed feature, the inputs to the SVM are collections of possible foreground appearance transformations due to low sample rates and strong illumination changes, making them suitable for tracking in the moving bus environment. Intuitively, the use of inverted cumulative colour histogram captures the colour transformation of an object between two consecutive frames, and it is used by Javed *et al.* (2005) to learn the Brightness Transfer Function (BTF) of a pair of cameras that have different camera parameters. In addition, the HOG and its spatiogram transformations learn the texture and spatial changes that have lighting and rotation invariant properties. Thus, the colour, texture, and spatial descriptors are robust to lighting changes. After the training, we obtain the support vectors as the appearance transformation model. Using this model, a pair of foreground candidates can be tested to generate the tracking decisions.

In the implementation, we treat the RGB channel independently and quantise it into 32 bins per channel, resulting in a total of 96 bins. Both HOG and its spatiograms consist of 9 bins, resulting in 18 bins in total. Note that since the orientation is sensitive to object pose, we rotate the object against the dominant orientation before creating the HOG to achieve rotationally invariant descriptors, similar to the SIFT descriptor (Lowe, 2004). Finally we concatenate all histogram transformations to represent the foreground appearance descriptor. Table 3.2 shows the input SVM feature vectors, where each row consists of 114 dimensions.

### 3.3.7.2 Observation Model using The Appearance Transformation Model

If we denote  $\mathbf{H}^* = (H^*, G^*, S^*)$  as the reference feature model for an object and  $\mathbf{H}(\mathbf{X}_{i,t})$  as the candidate feature model, the likelihood that  $\mathbf{H}^*$  and  $\mathbf{H}(\mathbf{X}_{i,t})$  belong to the same object is measured by first computing their transformation,  $\mathbf{Z}_{i,t}$ , and then comparing it against the appearance transformation model. Denote the projection of the SVM vector as  $\mathbf{w}$  and the SVM feature as  $f(\mathbf{Z}_{i,t})$ , then the observation likelihood is modelled as a sigmoid function based on the classifier's output:

$$p(\mathbf{Z}_t|\mathbf{X}_t) \propto \frac{1}{1 + \exp(\mathbf{sign}(\mathbf{w}^T f(\mathbf{Z}_{i,t})))} \quad (3.42)$$

Here  $f(\mathbf{Z}_{i,t})$  can be viewed as the distance from the  $\mathbf{Z}_{i,t}$  to the support vectors for the appearance transformation model. That is, the probability outputs of SVMs are based on the distance of test and support vectors. So the output reflects the inter-class likelihood.

### 3.3.8 Computing MAP Estimation

At the end of the RJMCMC iteration process, we obtain the entire Markov Chain configurations. The final task is to compute MAP estimate  $\mathbf{X}_t^{MAP}$ , i.e. the predicted target's state at time  $t$ . In the implementation, we search for the best state configurations with the highest number of occurrences (*mode*).

## 3.4 Experimental Results

### 3.4.1 Perth Central Area Transit (CAT) Bus Dataset and Camera Parameters

**Test video.** The experimental data consists of 11 sets of video, which were recorded from cameras mounted inside Perth Central Area Transit (CAT) buses. Videos from four different CAT buses are used in this experiment. Figure 3.13 shows snapshots of the empty CAT buses along with their corresponding seat structure or seat label information. The seat structure information is transferred from one bus to other buses using the Homography estimation method described in Section 3.1. In Figure 3.13 (d), the seat structures



Test video	CAT	NOP	NOF	Description
Video 1 *	2	0	188	Empty bus with inconsistent lighting.
Video 2 *†	1	1	3558	Single passenger, lighting changes, and multiple seat switchings.
Video 3 *	3	8	156	Multiple crowded passengers, occlusions.
Video 4 *	2	9	292	Multiple crowded passengers, severe lighting changes, occlusions.
Video 5 †	1	3	159	Multiple passengers, seat switchings.
Video 6 †	1	3	224	Multiple passengers, single seat switching.
Video 7 †	1	2	400	Multiple passengers, multiple seat switchings.
Video 8	1	6	473	Multiple passengers, occlusions.
Video 9	4	5	184	Multiple passengers, lighting changes, no significant motions.
Video 10	1	5	254	Multiple passengers, lighting changes, no significant motions.
Video 11	1	2	325	Single passenger, no significant motions.

\*These video are used to illustrate our detection results in Section 3.4.5.

†These video are used to illustrate our tracking results in Section 3.4.6.2.

NOF = Number of frames, NOP = Number of passengers

Table 3.3: Descriptions for the CAT dataset.

are manually labelled. The corresponding estimation results for the other three buses are shown in Figures 3.13 (a), (b), and (c). Without this calibration step, the seat topology has to be manually defined for each bus and for each time the camera in a bus is physically adjusted, which is impractical. Table 3.3 summarises descriptions of the CAT video covering different scenarios inside the bus.

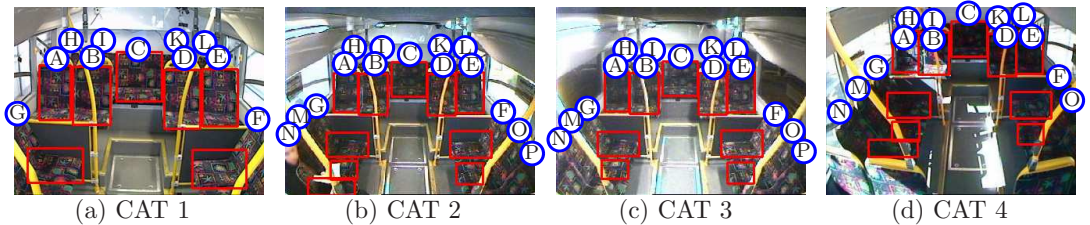


Figure 3.13: Using the Homography approach, the background seats information can be automatically transferred across similar types of buses even with differing viewpoints and lighting.

**Bus camera parameters.** Each camera was set up to record a video stream of size 320x240 at different frame rates. Figure 3.14 shows the existing bus surveillance camera (the black colour camera) that is used to capture video in CAT 2, 3, and 4 and has approximately 4–8 frames per second (fps) capture rate, while the white colour camera is used to capture CAT 1’s video at 25 fps. The video from CAT 2, CAT 3, and CAT 4 are captured in real bus situations as they operate in the urban city, covering different lighting and common passenger situations. Due to the limited scenarios in the three CAT’s footage,

additional data collection trials were carried out. These trials are captured using CAT 1’s camera, providing a broader range of passenger behaviours. Throughout the tracking experiment, we downsample CAT 1’s video frame capture rates to 5 fps to simulate the low frame rate conditions. Because the frame rates do not affect the detection performance, all detection experiments were conducted on the original frame rates.



Figure 3.14: Camera setup.

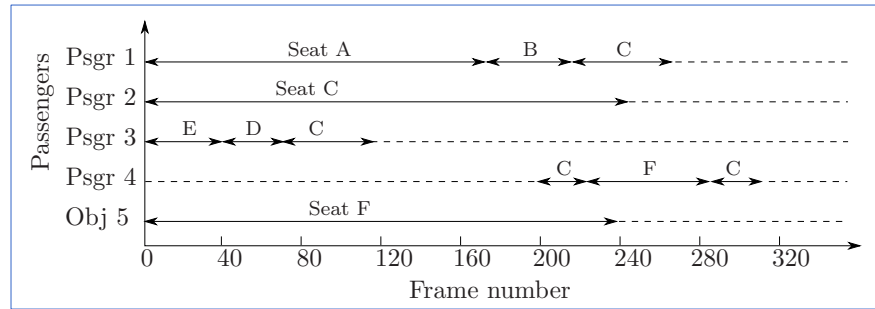
**Training data (detection).** Prior to testing, we collect training frames from the four moving buses without passengers with varying lighting conditions (comprising around 200–340 frames per bus). Training images are then used to build the SIFT codebook background model as described in Section 3.2.1.1.

### 3.4.2 Ground truth

To quantify the performance of our system, all ground truth associated with the test video are provided. For each test video, we manually annotate the ground truth at two resolutions: the high resolution ground truth (video ground truth) and the low resolution ground truth (individual frame ground truth).

**The high resolution ground truth (video ground truth)** consists of the total number of passengers and their trajectories throughout the video. It is presented as the passenger trajectory graph (PTG) as shown in Figure 3.15 (a), where the seat and passenger annotations are presented in Figure 3.15 (b) and (c) respectively. We use *alphabets* to label the seats and *numbers* to label the passengers. The x-y axes represent the frame number and passenger ID respectively. The trajectories of each passenger in the video are represented by line segments. Each line segment is labelled with a seat number and the length of the line represents durations of the passenger occupying the particular seat. By rearranging

the PTG using the seat and frame number, we obtain the seat occupancy graph (SOG) as shown in Figure 3.16, where the ground truth (dark colour bar segments) can be directly compared with the detection results (light colour bar segments). The SOG is used to assess the performance of the detection method, while the PTG will be used to evaluate the tracking methods.



(a) Passenger trajectory graph for Video 6



Figure 3.15: Ground truth for video 8.

**The low resolution ground truth (individual frame ground truth)** consists of the following information: (1) the number of passengers in a frame instance, (2) centroid of their head-body locations and the bounding boxes, (3) their seat occupancies, and (4) bounding boxes of other objects apart from passengers (e.g. suitcase, bag, or jackets, etc). Figures 3.15 (d) and (e) show snapshots ground truth of the foreground bounding boxes in video 8, comprising three passengers (denoted by green and brown bounding boxes) and a bag (purple bounding box) on seat F.

### 3.4.3 System Output (Detection)

At the end of the detection program execution, we obtain a set of foregrounds associated with the seat label and frame number. The summarised outputs of test video 8 are shown in Figures 3.16 and 3.17. In Figure 3.16, the detection results (light colour bar segments) are evaluated against the ground truth (dark colour bar segments) to measure the performance

of the detection algorithm.

Because the detection results will be used for tracking purposes, we need to extract reliable foreground objects. In some situations, due to the lack of edges, the probability of the human model detection is not able to locate the passenger correctly. Hence, we use the output of the seat occupancy results as the foreground (instead of the human detection results). In Figure 3.17, the red ellipse represents the detected head of the passenger, while the blue rectangle and blue ellipses represents the seat occupancy and the detected human body respectively.

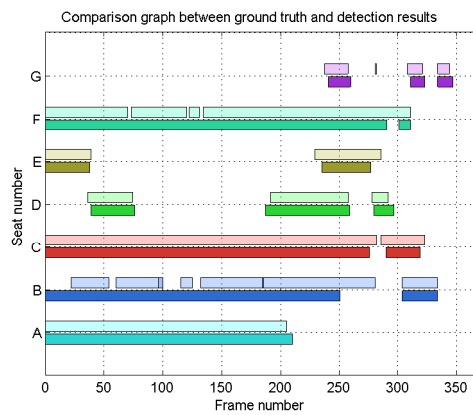


Figure 3.16: Comparison graph between the ground truth (dark colour bar) and the detection results (light colour bar) for video 8.



Figure 3.17: The detected passengers in video 8. The different thickness of the ellipses represent the likelihood of the human detection model (thin = low likelihood, thick = high likelihood).

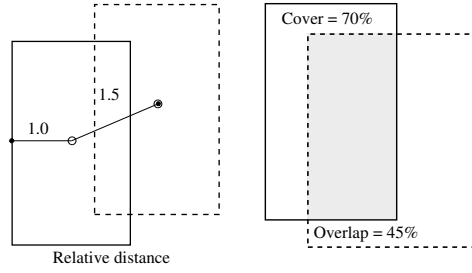


Figure 3.18: Detection evaluation criteria.

### 3.4.4 Evaluation Criteria

To evaluate the detection performance, we apply three criteria: *relative distance*, *ratio of the cover* and *overlap* (Leibe *et al.*, 2005), as shown in Figure 3.18. A detected candidate is considered to be a true positive when its *relative distance* from the object is less than 0.5 times the actual size of the ground truth’s bounding box, and the *cover* and *overlap* of the bounding box are both above 40%, and anything else is considered as a false positive. The overall detection performance for each experiment is computed as:

$$TP(\%) = \frac{\text{the total number of true positives}}{\text{the total number of ground truth}} \quad (3.43)$$

and

$$FP(\%) = \frac{\text{the total number of false positives}}{\text{the total number of detected candidate}} \quad (3.44)$$

### 3.4.5 Experiments (Detection)

The main objectives of the following experiments are to test the:

1. Performance of the SIFT codebook background model under severe lighting changes with no passengers (Experiment 3.4.5.1).
2. Performance of the proposed detection algorithm under different lighting conditions with a passenger moving and switching seats inside the bus (Experiment 3.4.5.2).

3. Performance of the proposed detection algorithm under crowd levels, including occlusion and background clutter (Experiment 3.4.5.3).
4. Performance of the proposed detection algorithm under varying crowd levels and lighting changes (Experiment 3.4.5.4).
5. Overall accuracy of the proposed detection algorithm against two standard foreground detection algorithms (Experiment 3.4.5.5).

For each experiment, snapshot images of major events are shown to illustrate the content of the video. The corresponding snapshot of the detection results is also generated along with the performance graph and the final detected foregrounds.

**Implementation details:** We set the weight factor  $\beta$  in Equation 3.3 to  $\beta = 0.3$  to perform all the testing.

### 3.4.5.1 Experiment I: Empty Bus with Lighting Changes (Video 1)

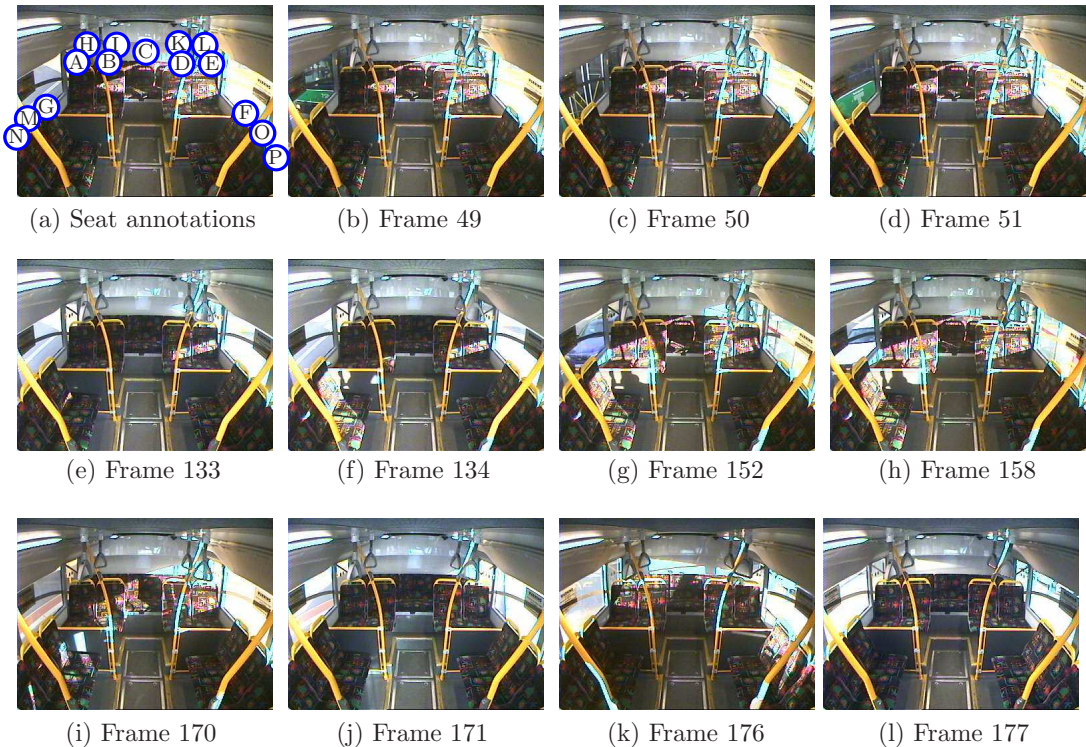


Figure 3.19: Snapshot for video 1 (CAT 2).

This experiment demonstrates the performance of the SIFT codebook background model against lighting changes.

**Description:** Figure 3.19 shows snapshots of test video 1. It contains images of an empty bus with frequent lighting changes. From Figure 3.19 we can see that between most of the images, drastic ambient lighting changes occur while the bus is moving. For example: (1) between frames 49, 50, and 51, the scenes outside the left window keep changing and seats B, C, D, and E experience erratic lighting changes, (2) between frames 133, 134, 170, and 177, seats K and L are exposed with different lighting and shadows, (3) in frames 171 and 177, even though there is a normal lighting level, the global illuminations at the rear of the bus are different.

In this experiment, we use the first 60% of the data (112 frames) for constructing the background model and the remaining 40% for testing (76 frames). The goal is to test the background model against the false positives. We compare our proposed SIFT codebook background model against the Gaussian Mixture Model (GMM) background subtraction algorithm (Stauffer and Grimson, 2000). For both methods, the background model construction and the detection are performed for each seat region instead of the whole image. A seat is considered occupied when 75% of its area is detected as foreground. To evaluate the performance of each algorithm, we compute the percentage of errors (false positives) metric, which can be expressed as:  $ErrRates = \frac{\#false\ positives}{\#test\_frames \times \#seats}$ . Since the bus is empty, the ideal performance is to detect no seat occupancy (0% false positives).

**Results:** The performance of both methods is provided in Table 3.4, while the snapshot results are shown in Figure 3.20. It shows the superiority of our proposed method against the GMM algorithm. The small number of errors in our detection method is mainly caused by the “unseen” or “rare” seat appearances (due to different lighting conditions) resulting in most of the codewords having low likelihood values below the detection threshold. On the other hand, the GMM method is not able to model the lighting changes correctly due to random intensities produced by severe lighting conditions. This causes the GMM background model to be noisy resulting in high false positives as shown in Figure 3.20 (bottom-row). In Experiment 3.4.5.4, we will use the constructed background model in this experiment to detect the passengers.

Methods	Error rates (% false positives)
SIFT Codebook Background Modelling	9.26%
GMM (3 number of Gaussians) Stauffer and Grimson 2000	87.67%

Table 3.4: Comparison results for video 1.

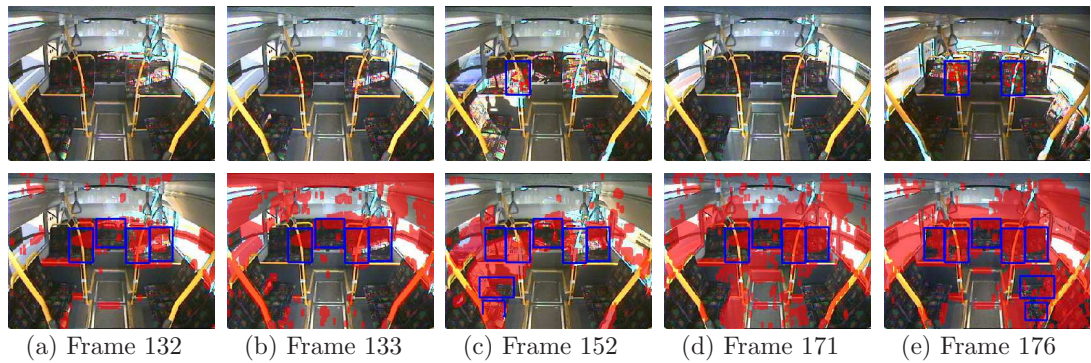


Figure 3.20: Snapshot comparisons of the SIFT codebook background model (top row) and the GMM background subtractions (bottom row). Foreground pixels are marked with red, while a seat that is considered occupied is drawn with a blue bounding box. The performance is measured using the bounding boxes or seats.

### 3.4.5.2 Experiment II: Single Passenger with Lighting Changes (Video 2)

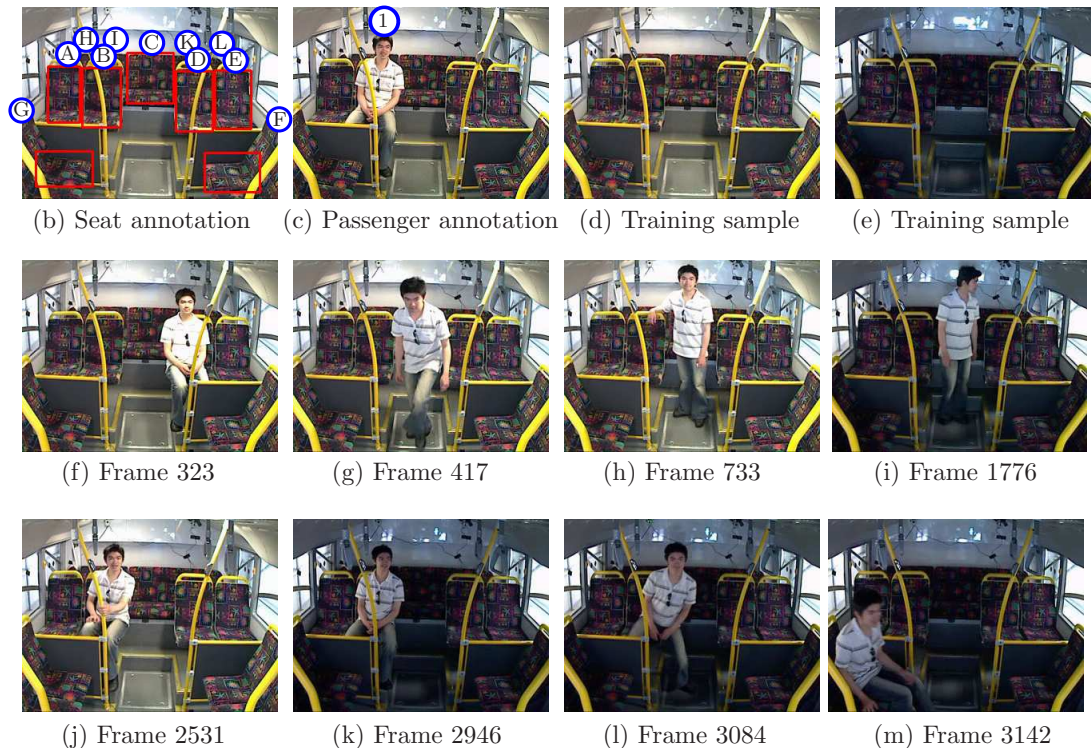


Figure 3.21: Snapshots of video 2.



This experiment demonstrates the performances of the proposed detection method under the following two scenarios: (1) different lighting conditions and (2) a simple case of a passenger moving and switching seats inside the bus. Since all the CAT 1’s trial data are collected when the bus is stationary, the different lighting conditions are simulated by switching the light on and off inside the bus.

**Description:** In this experiment we consider a simple scenario in which a single passenger moves while the lighting inside the bus is adjusted. Figure 3.21 shows the annotated snapshot images of video 2. Similar to the first experiment, we build a SIFT codebook background model using the empty bus data with different lighting conditions. Examples of the training images are shown in Figures 3.21 (d) and (e) respectively. In this video, initially a passenger, wearing white striped shirt, enters the bus and occupies seat D (from Frame 250–400), as shown in Figures 3.21 (f) and (g). He then leaves the bus and re-enters the scene at frame 653 to 987, mostly standing in the aisle (blocking seat C). Different lighting changes occurred in a number of instances, between Frames 1774 and 1775; Frames 2945–2946; and Frames 3085 and 3086. In summary, the passenger switched seats twice as he moved from F to B; and from B to G.

**Results:** Figures 3.22, 3.23, and 3.24 show some of the detection snapshots, performance graph, and summarised detection result of video 2 respectively. In this experiment, we achieve 89.2% true positives detection with 10.3% false positives. As can be seen from Figure 3.22, in most frames, for example between frames 733 and 1776; frames 2945 and 2946; and frames 3084 to 3142, our detection method is able to detect the passenger correctly despite the sudden changes in illumination. It is also able to detect passenger movements around the environment, this can be noted from frames 323–417; and 2946–3084 (when the passenger moves from the seating position to the aisle). On the other hand, there were a number of false positive detections when the passenger occludes multiple seats, as in frames 2498, 3084, and 3142. In addition, the algorithm does not detect the head or body location correctly in some frames, due to the weak edges around the human boundary in frames 2960 and 3142.

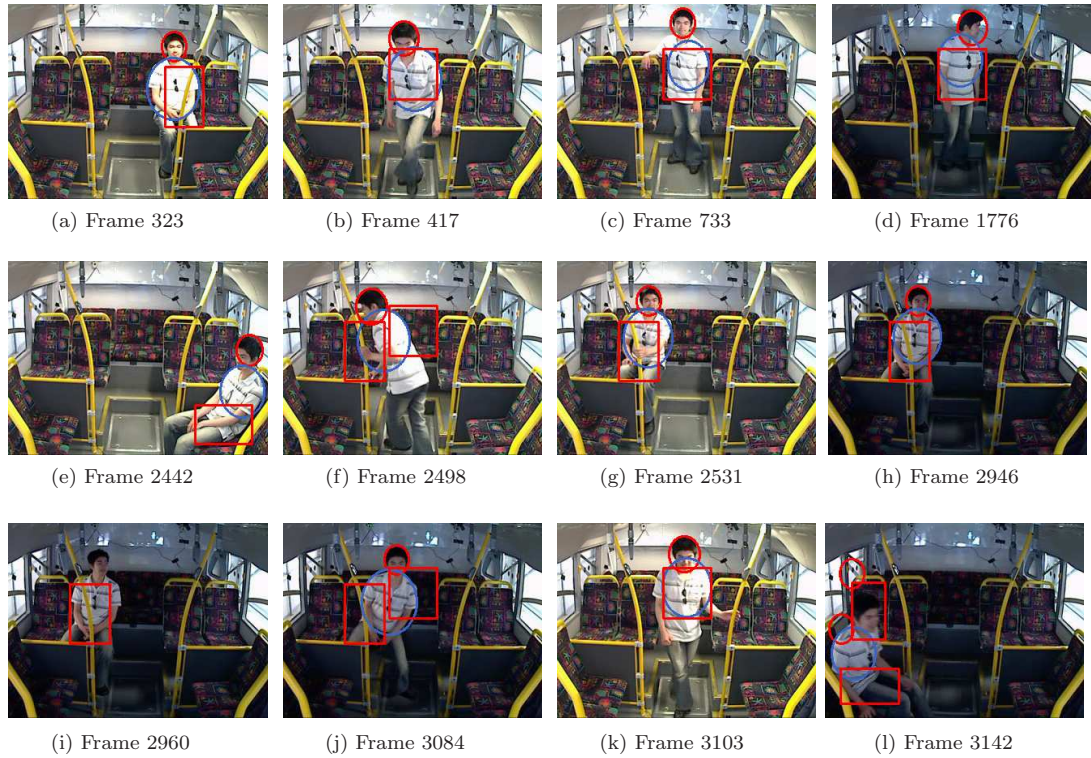


Figure 3.22: Snapshot images of the passenger detection in video 2 (CAT 1).

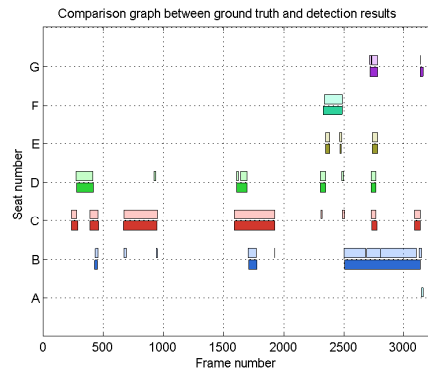


Figure 3.23: Comparison of the detection results (light colour bar) and the ground truth (dark colour bar).

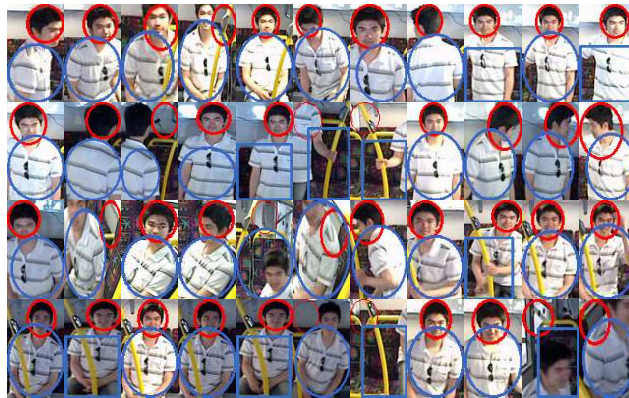


Figure 3.24: Detection results of video 2.

### 3.4.5.3 Experiment III: Crowded Passengers at Night Time (Video 3)



Figure 3.25: Snapshots for video 3 (CAT 3).

This experiment demonstrates the performance of the proposed detection algorithm under various crowd levels. The video was taken at night time where the lighting condition is

fairly stable. The challenges in this video include background clutter and object occlusions. Figure 3.25 shows annotated snapshots of video 3, where the passengers are annotated in the order they appear in the video.

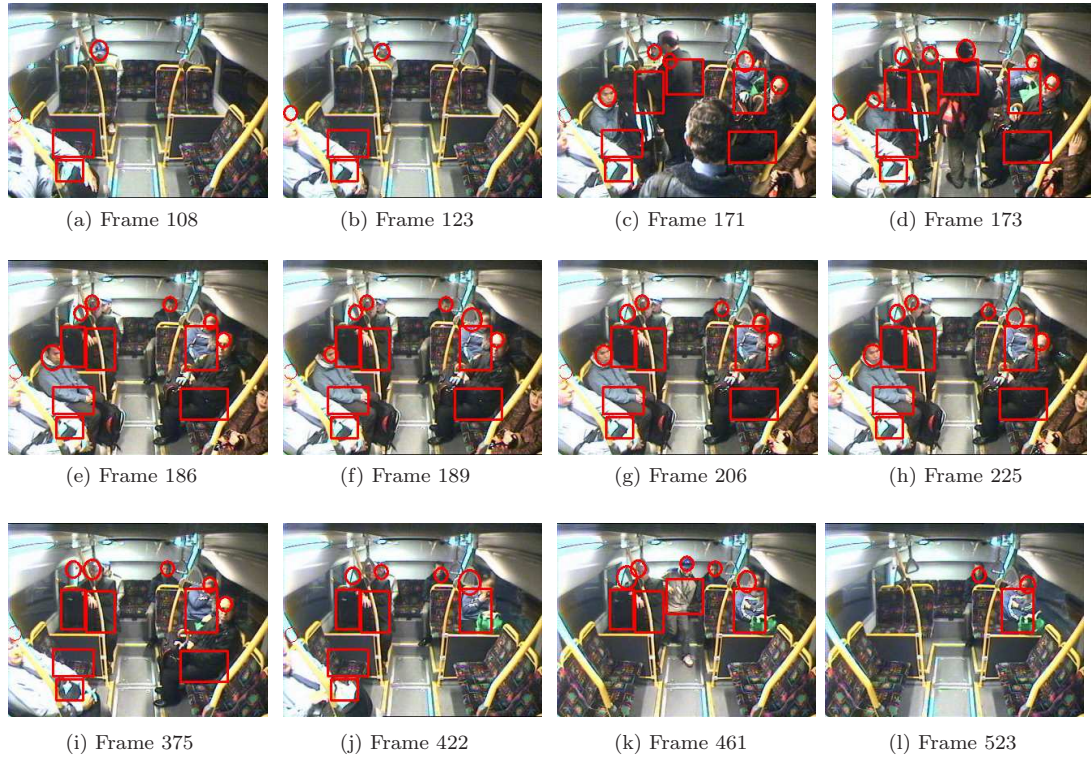


Figure 3.26: Snapshot images of the passenger detection in video 3 (only the head and the seat occupancy are shown for simplicity purposes).

**Results:** Figures 3.26, 3.27, and 3.28 show selected snapshot results, performance graph, and summarised detection results of video 3 respectively. In this experiment, we achieve 79.4% true positives detection with 22.8% false positives. Despite the crowded scenes in Figure 3.26 (d), our proposed method is able to pick up 7 correct foregrounds with 1 mis-detection. From frame 171–502, we show that our method can also detect the black suitcase object as the foreground. The reason we have a lower overall true detection is mainly caused by occlusion. During frames 171–460, passenger 3 sitting at the rear side of the bus was heavily occluded by passenger 7, making it difficult to be detected. Similar situation occurs for passenger 4 who sat close to the camera and was occluded by the bus handle bar (only half of the passenger’s body parts were visible in the camera’s field of view). The false positives are caused by passengers occupying multiple seats, for example, part of the body of passenger 1 simultaneously occludes seat M and G in frames 108 and 123. This experiment demonstrates that the proposed method is able to detect multiple targets under occlusion and background clutter situations with reasonable

detection accuracy.

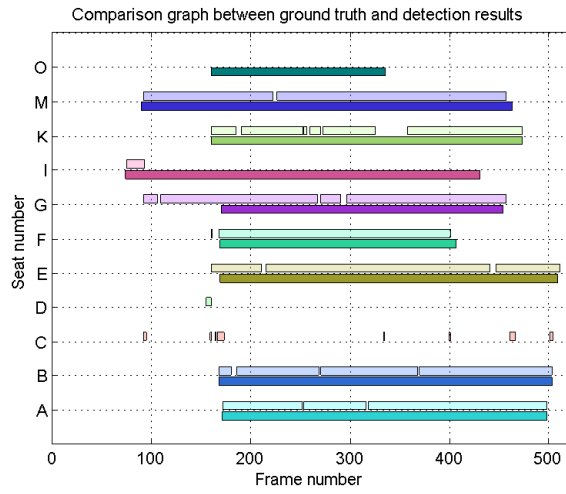


Figure 3.27: Comparison of the detection results (light colour bar) and the ground truth (dark colour bar).

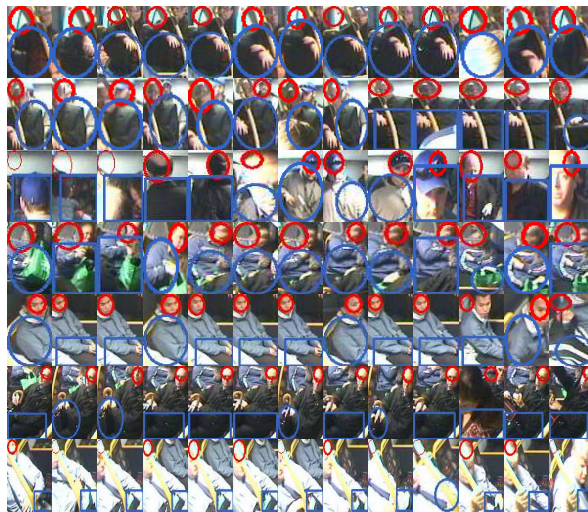


Figure 3.28: Detection results of video 3, arranged according to the seat number (from top row to bottom, we have seat A–G)

3.4.5.4 Experiment IV: Crowded Passengers with Lighting Changes (Video 4)

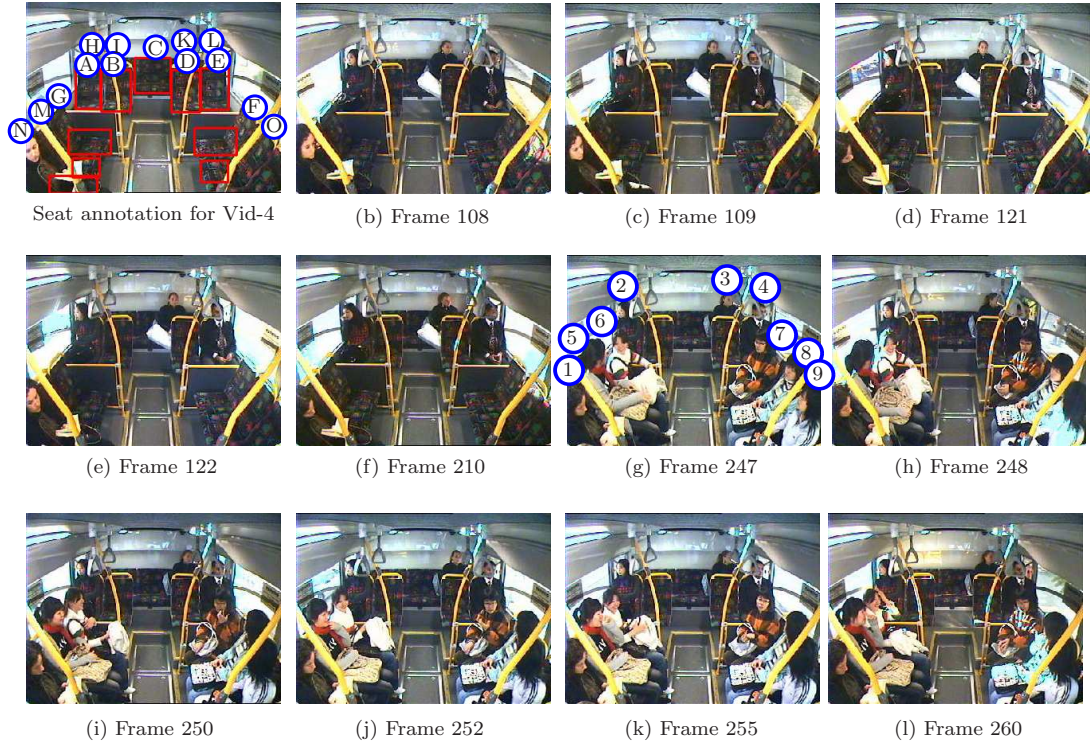


Figure 3.29: Snapshots for video 4 (CAT 2).

This experiment demonstrates the performance of the detection method in a challenging real bus scenario with crowds and severe lighting change situations.

**Description:** Figure 3.29 shows snapshots of video 4. This video comprises a series of passengers entering and leaving the field-of-view of the camera at random intervals. The scene through the window keeps changing while the bus is moving as in frames 108 and 109; 121 and 122. Between frames 247 and 248, the lighting conditions change rather dramatically due to external lighting; between frames 255 and 260 illumination changes exist on the rear and right seats of the bus.

**Results:** Figures 3.30, 3.31, and 3.32 show selected snapshot results, performance graph, and summarised detection results of video 4 respectively. In this experiment we obtain 70.2% true positives, with 28.3% false positives. Despite failing to detect the human shape due to weak edges at the back of the bus (for example in frames 88, 108, 109), we are able

to detect the passengers correctly since the probability of seat occupancy is high. Similarly, between frames 247-248 and 255-260, even when the lighting in the side and rear of the bus changes drastically, the system is able to detect passengers correctly without introducing false positives. This demonstrates the strength of the invariant lighting features used to model the background comprising highly structured texture such as the seats. Mostly, the “clothing” of passengers has very different textures from the seats, making them distinguishable from the background.

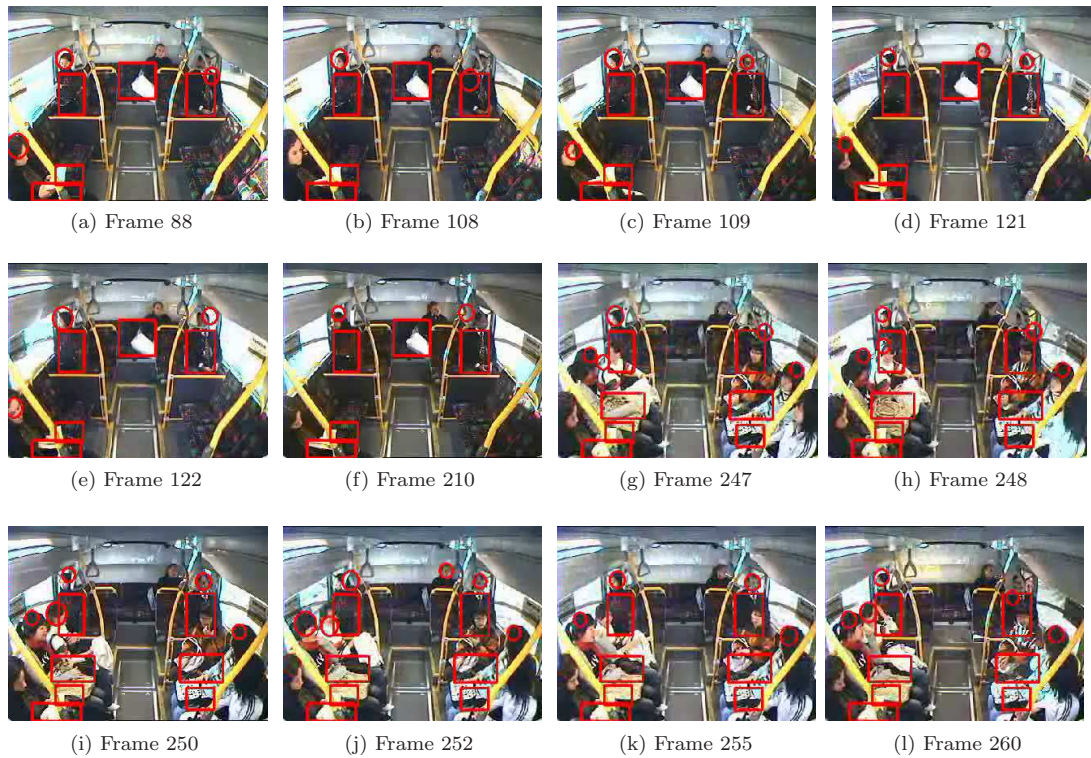


Figure 3.30: Snapshot images of the passenger detection in video 4.

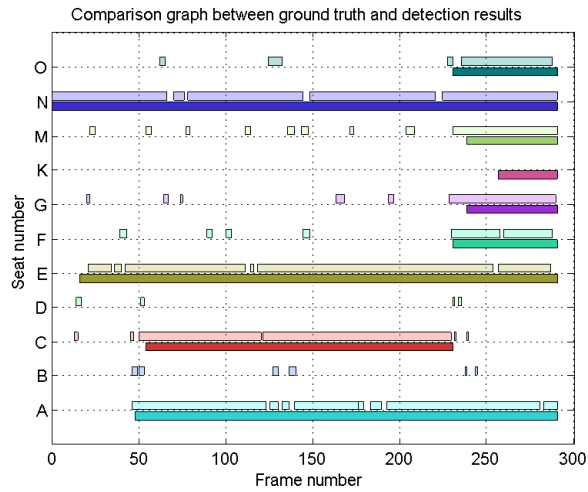


Figure 3.31: Comparison of the detection results (light colour bar) and the ground truth (dark colour bar).

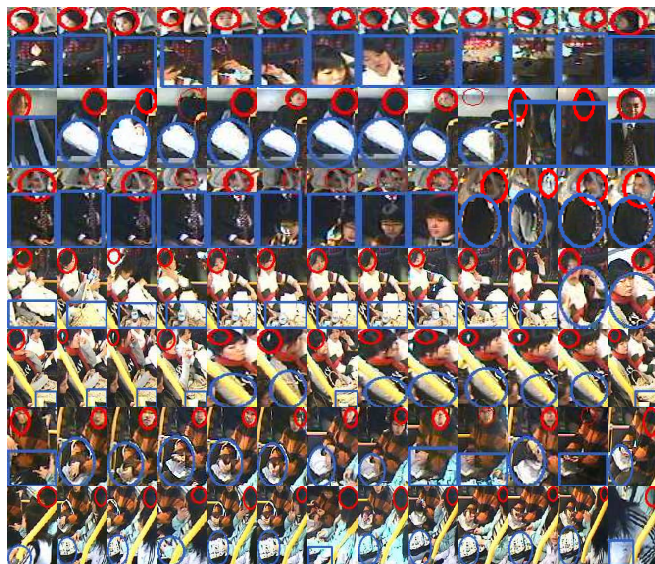


Figure 3.32: Detection results of video 4, arranged according to the seat number (Top-Bottom: Seat A, C, E, G, F, M, O)



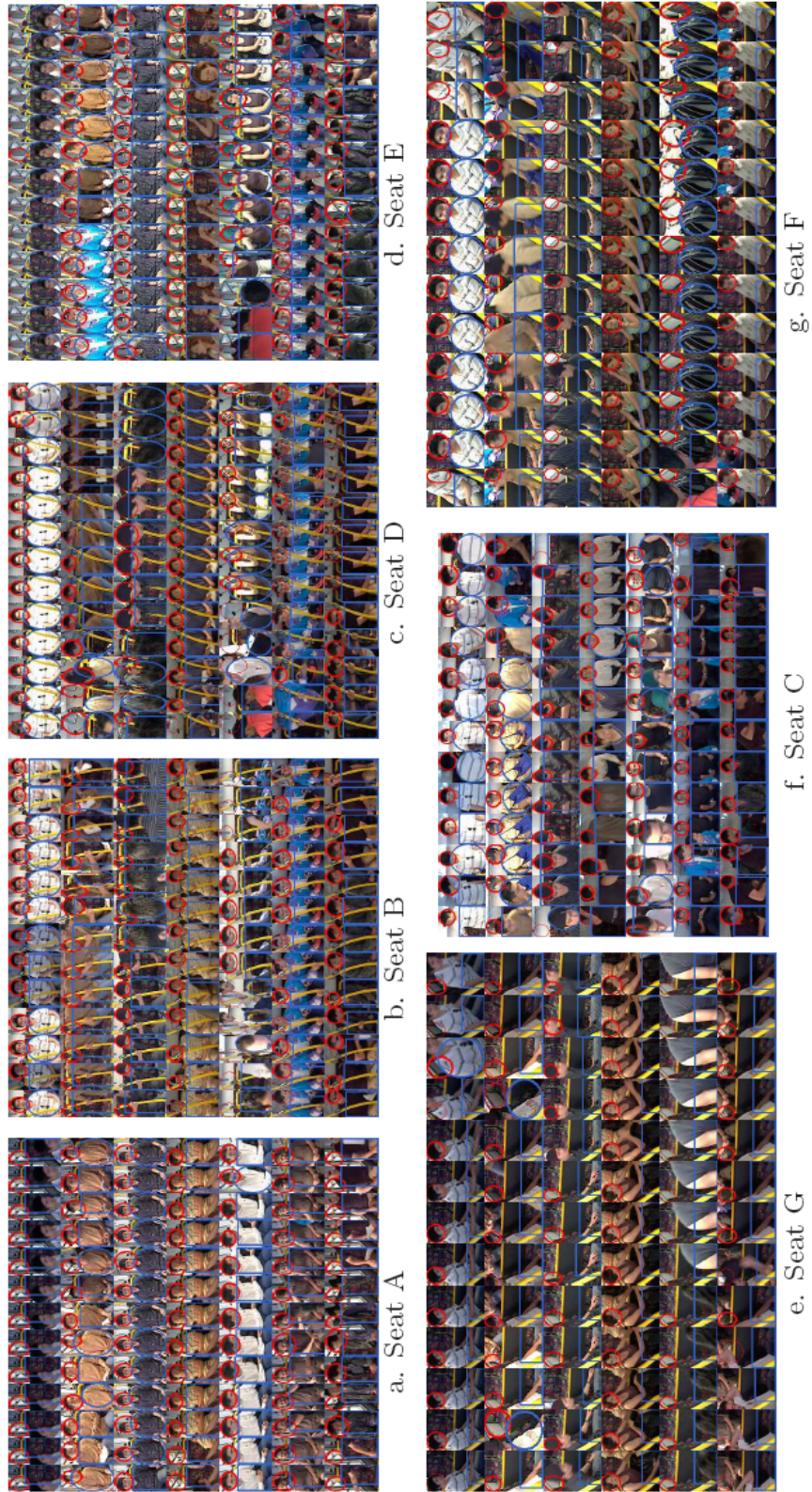


Figure 3.33: Detection results for video 2, 5, 6, 7, 8, 10, and 11 (from top row to bottom). The results are shown according to the seat number.

## 3.4.5.5 Experiment V: Comparison With Other Detection Approaches

Results	(a) Our method		(b) GMM		(c) Head detector	
	TP	FP	TP	FP	TP	FP
Video 2	89.2	10.3	57.3	80.3	92.3	89.5
Video 3	79.4	22.8	42.2	78.4	83.5	84.7
Video 4	70.2	28.3	40.7	72.6	78.6	76.5
Video 5	84.1	27.2	53.8	76.6	86.9	83.2
Video 6	78.8	23.4	56.2	80.2	79.1	78.4
Video 7	82.9	24.8	54.7	79.4	85.8	84.3
Video 8	83.7	25.1	46.2	72.6	87.7	93.6
Video 9	78.2	22.9	42.2	71.6	80.2	96.1
Video 10	76.2	18.7	32.6	62.1	78.4	77.2
Video 11	90.6	31.3	52.2	76.7	90.4	88.9
Average performance	81.3%	23.5%	47.8%	75.1%	84.3%	85.2%

TP = True positive rates, FP = False positive rates

Table 3.5: Summary of different method performances on the CAT video.

The performance and detection outputs of six other video are summarised in Table 3.5 (a) and Figure 3.33. We further evaluate the performance of our proposed method against two standard foreground detection approaches: (1) Stauffer-Grimson Gaussian Mixture Model (background subtraction) (Stauffer and Grimson, 2000) and (2) Birchfield’s head detector (Birchfield, 1998).

**Description:** For the Stauffer-Grimson background subtraction, we use 3 Gaussian distributions to create a background model from the empty bus video and use it to detect foreground pixels during run-time. Once the foreground pixels are detected, we further perform operations to attain the connected components on the foreground pixels and then use morphological operations to arrive at the final results. The head detection module is implemented to locate the best fitting ellipse to the human head on the edge image. We first convert each image observation into an edge image, and then compute the ellipse curvature difference by scanning through the entire image at different scales, setting a threshold to eliminate false positives. Since the ground truth is labelled with the head and body bounding boxes information, the evaluation of each method is performed according to its output, i.e. since the head detection only produces the head ellipses output, it is tested against the head bounding box. Similarly, the GMM is evaluated for both the head and body bounding boxes.

**Results:** Figure 3.34 and Table 3.5 show snapshots of the detection results and the quantitative comparison between the different approaches respectively. Based on the results in Table 3.5, we see that the Stauffer-Grimson background subtraction and the Birchfield’s head detection approaches both have relatively high false positive rates. On the other hand, our proposed method is able to achieve balance between the true and false positive detections. We obtain an average performance of 81.3% true positive and 23.5% false positive detections.

From Figure 3.34 and Table 3.5, we can see that the Stauffer-Grimson background subtraction approach fails badly when tested on the CAT datasets. There are three main reasons why it fails: (1) lighting changes; (2) moving scenery seen through the window; and (3) crowded situations. From Figures 3.34 (a) and (c), when lighting changes occur, the GMM method produces a high number of false detections and, at the same time, have low true detection because most of the detected candidates do not fit the matching criteria which are counted as mis-detection. Even by adjusting the number of the Gaussians ( $n$ ), it is still not able to adequately capture the variability of lighting changes. Setting  $n$  to be a large number results in under segmentation/detection, and when  $n$  is small, the false positive rates become high. Because passengers are generally sitting close to each other, the GMM method generally detects multiple passengers as a single object, which results in a lower overall true detection.

The head detection method used by itself also faces a number of problems, such as high false detection rates and speed. Figure 3.34 shows that head detection produces a lot of false positives. Although the head detection is done on an edge image, which is partially invariant to lighting, this method used by itself is prone to error, as there are many artefacts that resemble ellipses inside the bus (e.g. bus handle, corner of the window, top part of the seat for example). Detecting a cascaded human configuration model like the pedestrian detection approaches (Dalal and Triggs, 2005) is hard to achieve due to heavy occlusion and irregular poses of passengers. Finally, scanning the entire image at different scales for head detection is computationally demanding.

In contrast, our proposed method greatly reduces the false detection and at the same time delivers a reasonable detection rate, even when severe lighting changes occur. The advantage of our detection algorithm lies in the use of a strong background model to detect the seat occupancy, by means of estimation we compute the statistics of the SIFT features which are partially invariant to lighting. The resultant seat occupancy detection then helps the human detection module to significantly reduce the amount of false positives given the evidence of seat occupancy status. The availability of the bus internal structure provides an effective way to disregard areas where it is not possible to detect passengers. However,

in some instances, mis-detections occur which is the result of the lack of foreground edges and objects that are very close to the camera. False positives occur when capturing an insufficient variety of lighting changes during the creation of the background model or when a passenger occupies multiple seats at the same time.

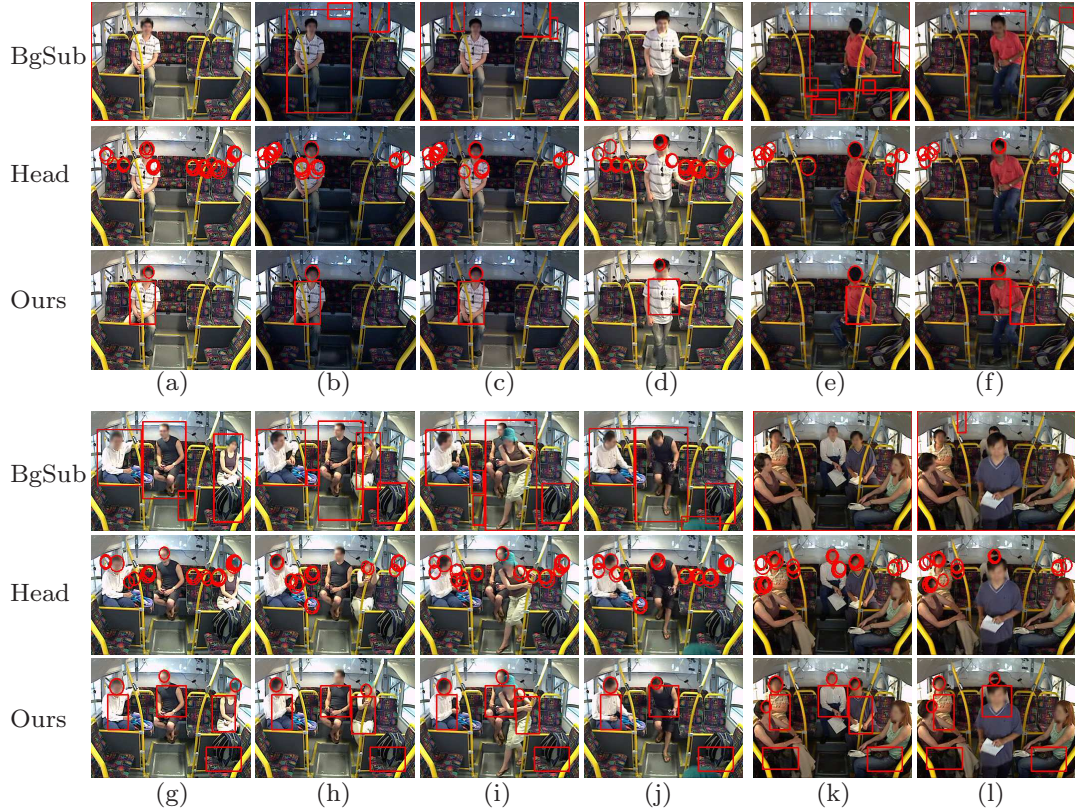


Figure 3.34: Comparison of the passenger detection results on the selected CAT datasets. (First and fourth rows) Background subtraction results; (Second and fifth rows) Head detection results; (Third and sixth rows) Passenger detection results using our proposed. Even when the light changes, our method is still able to detect the passengers correctly, while the background subtraction result fails badly (this can be seen from the red bounding boxes indicating the foreground objects being detected). Similarly, the head detection alone produces a lot of false positives (it detects the bus handles and corners of the window as human heads).

### 3.4.6 Experiments (Tracking)

To evaluate the proposed tracking method, we conducted two experiments. The first experiment was performed in an indoor setting to test the proposed appearance transformation technique on a conventional tracking problem involving non overlapping cameras.

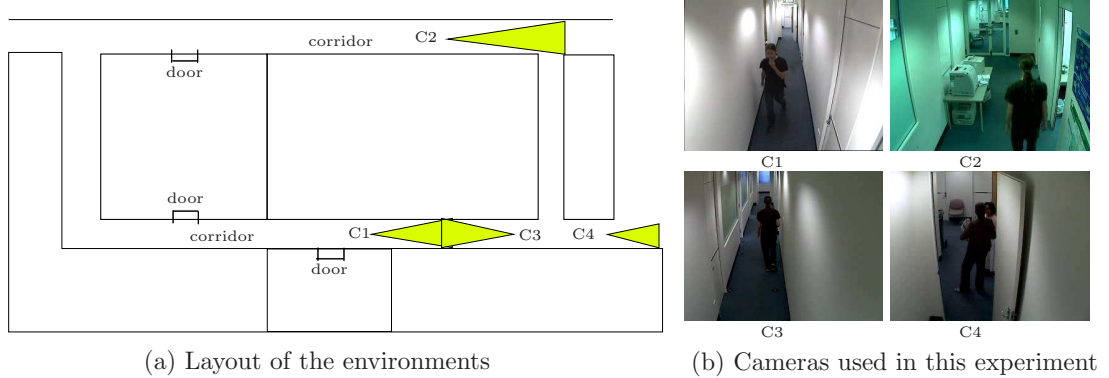


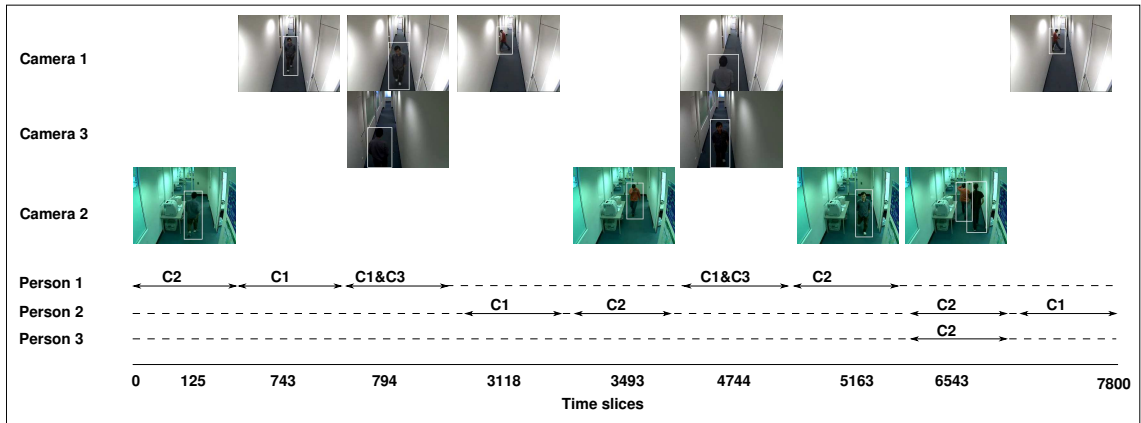
Figure 3.35: Experimental environment.

The second experiment presents results and quantitative comparison between our proposed method and the state-of-art tracking algorithms on the bus dataset.

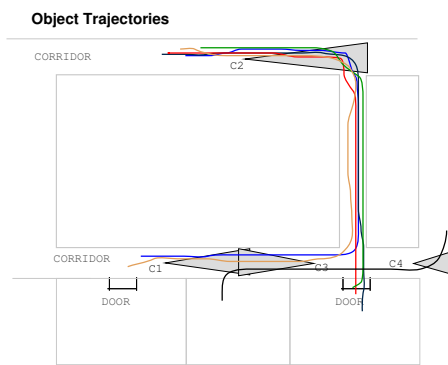
#### 3.4.6.1 Experiment VI: Non-Overlapping People Tracking (Indoor Environment)

The task of the first experiment is to track moving targets in an indoor surveillance environment installed with multiple non-overlapping cameras. In this situation, the appearance of the target may appear differently from one camera to another due to varying target poses, illumination, and camera parameters. Hence, the problem is similar to tracking passengers under low frame rate and different lighting conditions inside the moving bus.

The environment under surveillance consists of rooms and corridors typifying a conventional hallway surveillance installation. Figure 3.35 (a) shows the layout of this environment, while Figure 3.35 (b) shows images captured by the cameras C1, C2, C3, and C4. In this experiment, a week’s worth of video footage was collected using these cameras. We first implemented the Gaussian Mixture Model (GMM) segmentation by Stauffer and Grimson (2000) and extracted a total of 16,048 people images (without the background). From the segmented data (see Figure 3.36), we proceed to learn the appearance transformation models for all combination pairs of cameras using the proposed method stated in Section 3.3.7. We divided the datasets into positive and negative training samples. The positive training samples contain pairs of the same person captured by two cameras at different time instances as shown in Figure 3.36 (c); while the negative training samples contain two different people captured from the camera pair. Note that unlike Javed *et al.* (2005) who builds the appearance model for a pair of cameras, in our case, we only build



(a)



(b)



(c)

Figure 3.36: Example of the annotated ground truth: (a) the extracted foregrounds and (b) the object trajectories. (c) Examples of the positive training samples used in this experiment, where same foregrounds are viewed differently in two cameras. Between camera 1 and 3, same objects are viewed brighter in one camera than the other. Between camera 1 and 2, same objects have different appearance colours.

a single SVM classifier for all combination cameras. Hence our appearance transformation model takes into account a larger appearance variation.

In order to evaluate the tracking performances of our system, we define a quantitative measure (success and failure rates) as the performance metric, i.e. by calculating the rate of true and false positive rates for the non-overlapped region tracking. Success and failure rates refer to the success of the system in matching targets “leaving” and “entering” any pair of the non-overlapped surveillance cameras. During testing we perform cross validation with 40 iterations, with random division of the data into training and testing samples, to validate the accuracy of the proposed method. The results of this experiment are shown in Table 3.6.

C1_C2		C1_C3		C1_C4	
TP	FP	TP	FP	TP	FP
66.31%	25.28%	86.91%	23.42%	87.53%	16.16%
C2_C3		C2_C4		C3_C4	
TP	FP	TP	FP	TP	FP
71.02%	23.03%	69.21%	18.92%	85.73%	22.76%

Table 3.6: System performance of the proposed appearance transformation model.

The experimental results in Table 3.6 show that our approach is able to operate on an average of 77.8% tracking accuracy and 21.6% false positives. The reason C1\_C2, C2\_C3, and C2\_C4 have lower true positives compared to the other camera pairs is because camera C2 has large colour variations compared to the other three cameras, and thus contributes less training samples for the SVM during training. Hence, during testing, it has smaller weight as compared to the other transformation features. On the other hand, between cameras C1\_C3, despite the fact that objects in camera 1 generally have a brighter appearance model than objects on camera 3 we achieve 86.91% true positive.

The results and the success rate of the system justify our proposed appearance transformation technique as being suitable in low frame rate tracking situations with reasonable accuracy. Based on this result, we use the appearance transformation model method on real bus footage. The following subsection demonstrates the performance of the RJMCMC-ATMT tracking on real bus footage

### 3.4.6.2 Experiment: Seat-switch Detection

We will now evaluate the ability of the RJMCMC-ATMT technique to track passengers for detecting seat-switch events. The tracking performance is evaluated based on the number of correctly detected seat switching instances against the number of ground truth seat switching instances

#### Implementation details

*RJMCMC parameters.* We set the parameters for RJMCMC in a similar way as Smith *et al.* (2005). In the bus scenario, we define the exit region as the seats that are close to the end of the camera’s FOV. The move types are chosen according to the prior distribution from the previous state. The priors for birth ( $p(v = birth) = [.07, 0.4]$ ) and death ( $p(v = death) = [.007, .0004]$ ) increase when the passenger is occupying seats that are close to the

exit region or when two passengers are close together, i.e. occlusion. Similarly, the prior of the swapping is increased when the two objects are within a certain distance threshold  $d_s$ .

*Appearance Transformation Model:* Similar to the previous tracking experiment, we learn the appearance transformation model using outputs from the passenger detection results. We extracted a total of 1728 passenger images (without the background). Figure 3.33 shows some of the extracted passenger images. We divide this data into positive and negative training samples. The positive training samples contain pairs of the same passengers captured under low frame rates with various motion patterns including seating-to-standing, standing-to-seating, moving, switching seats, etc. The negative training samples contain two different passengers. We randomly select 2542 pairs as positives samples and 3179 pairs as negative samples. Figure 3.37 shows examples of the positive training samples used in the bus experiment.



Figure 3.37: Examples of the positive training samples used in the bus experiment.

*Seat switching detection:* At the end of the tracking program execution we obtain a set of passenger trajectories (tracklets) along with their appearances during tracking, which is stored in the “main database”. The seat-switch events can be detected by analysing the trajectory’s results; but this information is not reliable due to noise, discontinuity of tracklets, etc. For example, a passenger who moves from one seat to another may have discontinuity of tracklets in the middle of the video. Therefore, we utilise the appearance records of the passenger to detect the seat-switch activities even when the trajectories are not smooth. When a seat is occupied by the same passenger for at least three consecutive frames, we refer to this as the stable appearance profile of the passenger. A “main database” is used to store all target appearance profiles. Every time a new appearance profile is introduced, we compare it against all other appearance profiles in the main database. The



similarity measure between two appearance profiles is performed by cross validating all appearances inside the sets. If the average similarity is above 70%, we consider the two appearances coming from the same target and if the seats they occupied are different, this is identified as a “seat switch” activity.

### Experimental Objectives

The following tracking experiments are used to illustrate the performance of RJMCMC-ATMT under a series of passenger movement patterns, including normal motions, object occlusions, merging and splitting along with various lighting conditions. The main objectives of the experiments are to test the:

1. Performance of RJMCMC-ATMT on low frame rate video.
2. Performance of RJMCMC-ATMT under different lighting conditions (Experiment 3.4.6.3).
3. Performance of RJMCMC-ATMT in crowded scenes. (Experiment 3.4.6.4).
4. Performance of RJMCMC-ATMT against similar foreground appearance scenarios (Experiment 3.4.6.5).
5. Quantitative comparison between RJMCMC-ATMT against other tracking methods (Experiment 3.4.6.6).

#### 3.4.6.3 Experiment VII: Different Lighting Scenario (video 2)

This experiment examines the performance of RJMCMC-ATMT to correctly track a passenger under low frame rates and different lighting situations.

The key challenges in this video involve tracking a target that has different colour appearances due to lighting changes and different pose due to low frame rates. Figure 3.38 shows the snapshots of the passenger tracking while Figure 3.39 shows the comparison between the RJMCMC-ATMT tracking results and the ground truth (targets movement). In this experiment, we are able to detect 2 seat-switch activities correctly. RJMCMC-ATMT successfully tracks passenger 1 despite changes in illumination between frames 84 and 85. Under the same situations, Particle Filter with HSV appearance based association technique (Okuma *et al.*, 2004) fails to track the passenger as shown in Figure 3.40. In frame 124, RJMCMC-ATMT fails to associate the detected blob to passenger 1 due to a

large change in object appearances (sitting vs standing poses). Despite failing to track the passengers correctly in the middle of the video, we are able to detect the seat-switch activities correctly.

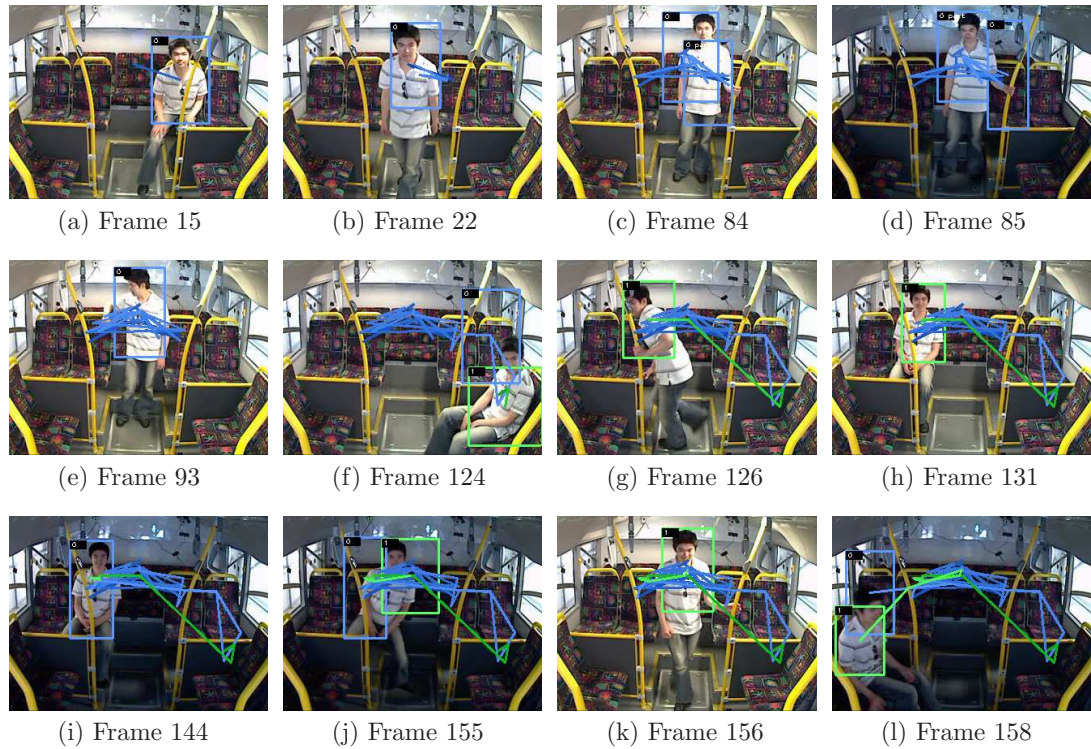


Figure 3.38: Snapshots of the passenger tracking in video 2 (CAT 1).

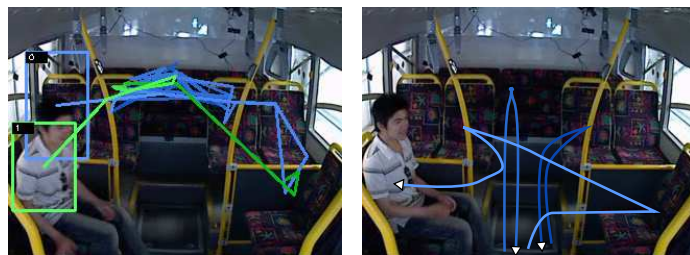


Figure 3.39: Comparison between the tracking results and the ground truth.

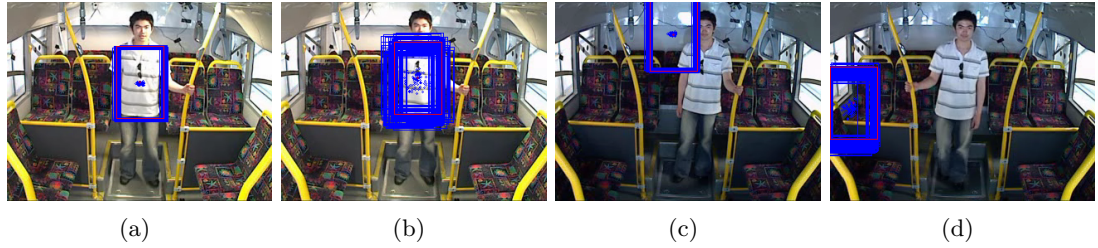


Figure 3.40: Examples of the Particle Filter with HSV appearance model fails to track the passengers when the lighting changes.

#### 3.4.6.4 Experiment VIII: Crowded Scenario

The second set of experiments shows two types of walking patterns under crowded scenarios. The first type involves object occlusion, while the second type involves walking patterns of merging and splitting. Both sets of experiments are conducted to show the performance of RJMCMC-ATMT in handling noisy environments.

This experiment determines the performance of the RJMCMC-ATMT technique on complex crowded scenes where movements of passengers cause occlusion as well as merging and splitting scenarios.

##### Object occlusion’s scenario (video 5)

The sequence of movements (ground truth) for an object occlusion scenario is shown in Figure 3.42 (b), while Figure 3.41 shows some snapshots of the tracking results. There is a total of 5 passengers detected in this video. Throughout the video, there are no significant movements from passengers 1, 2, 4, and 5. On the other hand, passenger 3 moves from seat D to B during frames 16–25 and from seat B to E during frames 32–38. During these instances, passenger 2 is occluded by passenger 3 as he moves. In this experiment, we correctly identify 2 seat-switch activities without false detection. Figures 3.41 (f) and (j) show that RJMCMC-ATMT is able to track passenger 3 when he switched seats, and correctly returns the tracker to passenger 2. Note that in this experiment, the *birth* and *death* moves in RJMCMC play an important role to the success of the tracking under occlusions. We also show in Figure 3.43, that using the MCMC tracking results in tracking failures as it does not have the dynamic model to handle a varying number of targets. In addition, the standard Particle Filter with second-order regressive model is not able to follow the target (see Figure 3.44). Note that passengers sitting at the rear of the

bus are not detected due to heavy occlusion (it is hard to detect them even by a human operator), so they are not counted in the ground truth. Finally, Figure 3.42 shows the comparison between the final tracking results and the ground truth. Note that the false association in frame 30 is not counted as seat-switch activity because the tracker returns back to passenger 2 in the next frame.

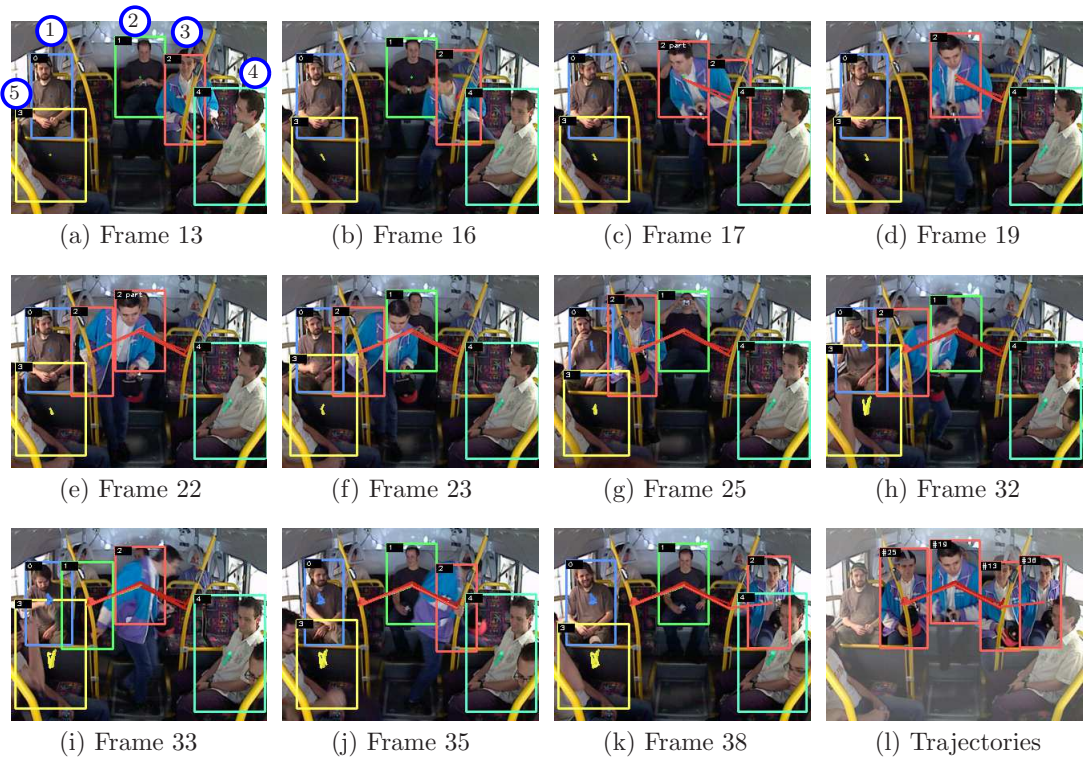


Figure 3.41: Snapshots of passenger tracking in video 5 (CAT 1).

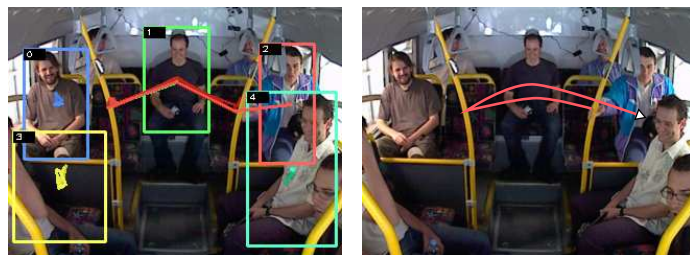


Figure 3.42: Comparison between the tracking results and the ground truth.



Figure 3.43: Examples of frames when the MCMC tracking technique fails to track passenger 2 under occlusion situations.

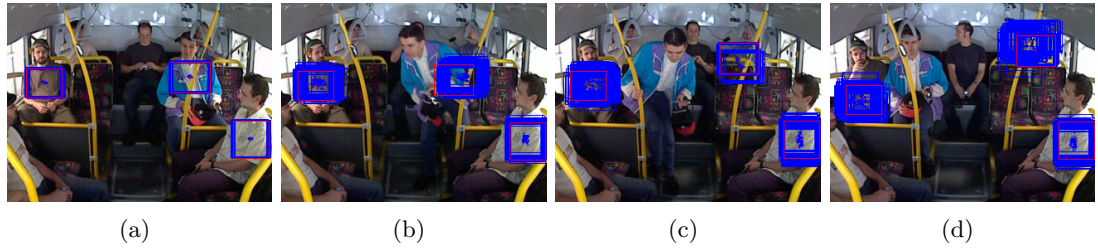


Figure 3.44: Examples of frames when Particle Filter method fails to track passenger 3 under the LFR situations.

### Merge and split scenarios (video 6)

This experiment was conducted to deal with the moving pattern of merging and splitting involving two passengers. The challenge of this experiment is to show a successful scenario when tracking multiple targets that occlude one another for a short period of time. In this experiment, we correctly identify 4 out of 6 seat-switch activities. Snapshots of the tracking results are shown in Figure 3.45 while the comparison of the trajectory results between the RJMCMC-ATMT tracking and the ground truth is shown in Figure 3.46. The first object occlusion occurs during frames 27–29, when passenger 1 moved from seat B to seat D and RJMCMC-ATMT is able to track the passengers correctly. The two passengers then decided to switch their seats; merging occurs at frames 54 and 55 while splitting occurs at frame 56. During merging, *RJMCMC death configuration* holds the highest probability because one target is occluded by the other. When the targets split, the *RJMCMC birth configuration* and appearance transformation model are then used to identify each individual target, as shown in frames 56 (see figure 3.45 (1)).

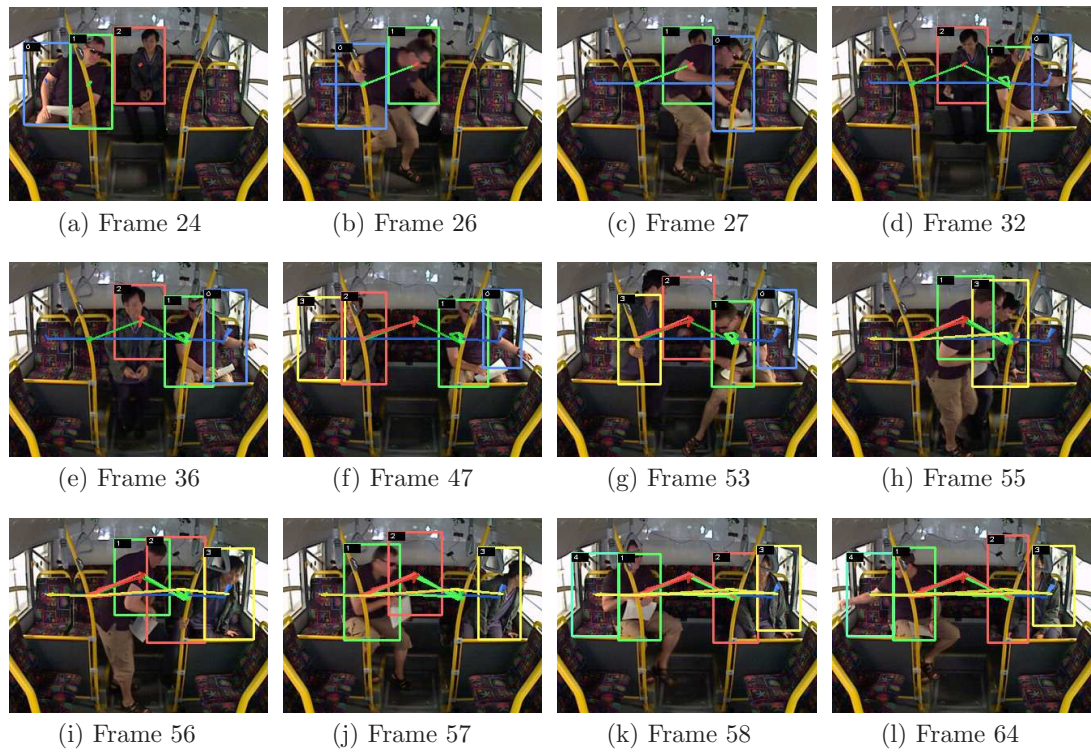


Figure 3.45: Some of the tracking results of video 6 (CAT 1).



Figure 3.46: Comparison between the tracking results and the ground truth.

### 3.4.6.5 Experiment IX: Similar Appearance Scenario (video 7)

This experiment investigates the tracking of four passengers undergoing seat-switch actions under different lighting conditions. Two of them have similar appearances (brown and blue appearances). The main objective of this experiment is to show the need of integrating RJMCMC PF and the appearance transformation method, in order to successfully track

two targets with similar colour profiles. Figure 3.47 shows snapshots of the tracking results while Figure 3.48 shows a comparison of the trajectory results between the RJMCMC-ATMT tracking and the ground truth (targets movements). In this sequence, we obtain 2 correct seat-switch activities out of 4. We can see that RJMCMC-ATMT is able to correctly track and identify the first passenger correctly even though there is another passenger with a similar appearance in the bus (frames 38–42 in Figures 3.47 (l–o)). This is due to the trajectory model that provides spatial constraint. Using appearance based association techniques alone in this scenario would fail because both trackers compete to hijack the one with higher likelihood, which will result in tracking failure.

The overall performance of the seat-switch detection results are summarized in Table 3.7.

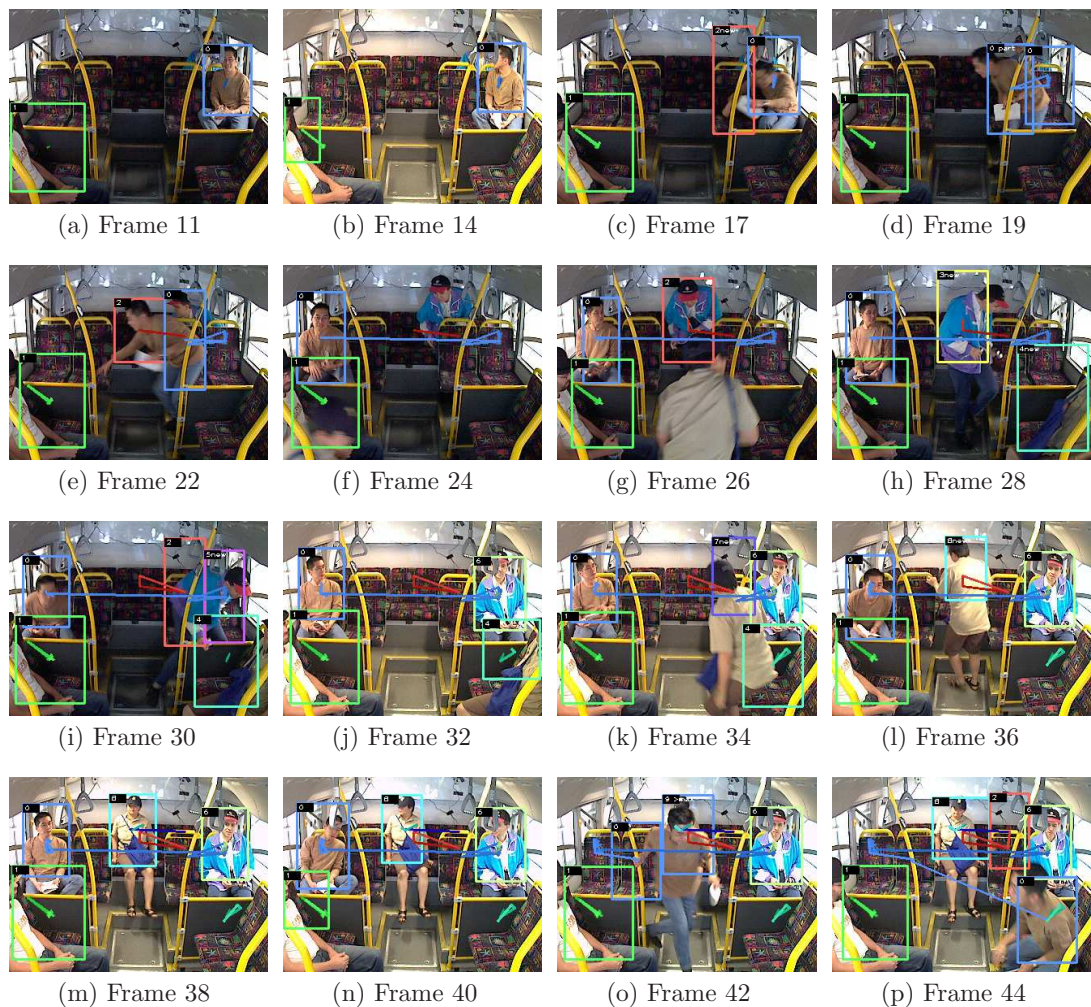


Figure 3.47: Some of the tracking results of video 7 (CAT 1).



Figure 3.48: Comparison between the tracking results and the ground truth.

Test video	Number of detected seat switching instances	Number of ground truth seat switching instances
Video 2	2	2
Video 5	2	2
Video 6	4	6
Video 7	2	4
Video 8	1	0
Video 10	1	2
Video 11	2	2

Table 3.7: Summarised seat switching detection results for all test video in CAT 1.

#### 3.4.6.6 Experiment X: Quantitative Comparison For Different Tracking Methods Against Low Frame Rate

The last experiment was conducted using all CAT 1's video. We evaluate the tracking performance by computing the normalised errors in tracking against the different frame capture rates. We down sample the test video frame capture rates from 25 fps to 20, 15, 10, and 5 fps to simulate the low frame rate conditions. We compare our proposed method against mean shift (colour) (Comaniciu and Meer, 2002), Particle Filter with HSV (Okuma *et al.*, 2004), and SURF feature tracking (texture) (Bay *et al.*, 2006). The experimental results in Figure 3.49 show an increase in errors in tracking for all methods as the frame capture rate decreases. In addition, it shows the superiority of our approach against standard approaches such as mean shift (Comaniciu and Meer, 2002), Particle Filter with HSV (Okuma *et al.*, 2004), and SURF tracking (Bay *et al.*, 2006). The mean shift and Particle Filter tracking algorithms do not perform well in the bus environment because the trajectory of the passengers are not smooth, thus violating the basic principal of these



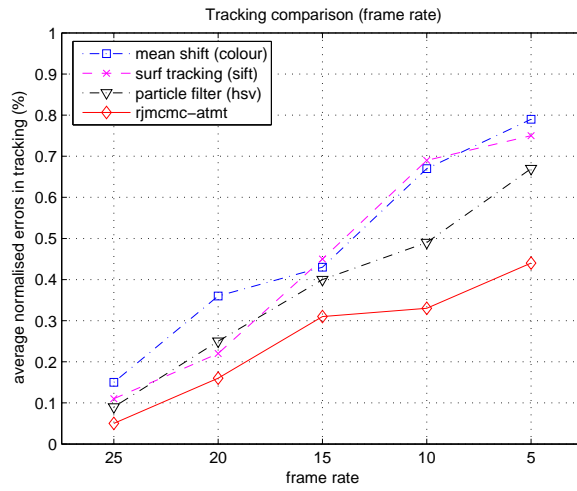


Figure 3.49: Comparison projection errors graphs between four different tracking algorithms under low frame rate situation

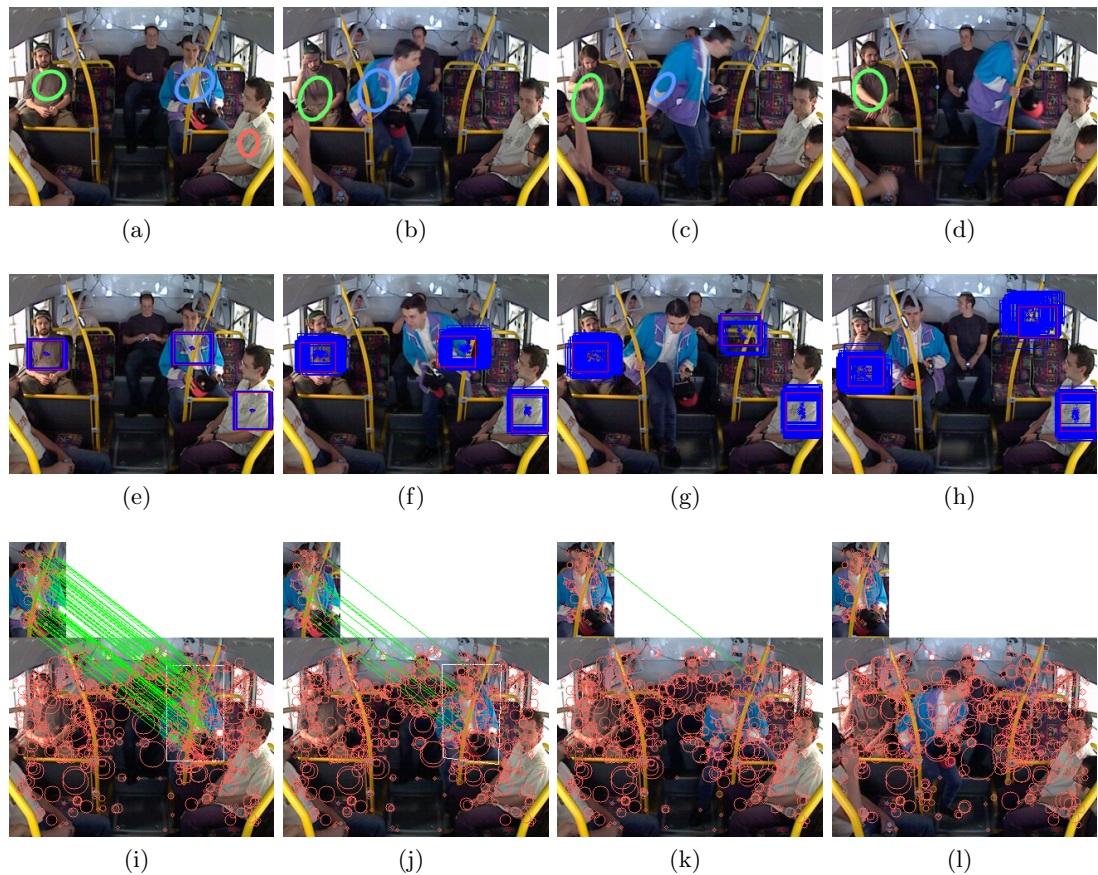


Figure 3.50: Examples of frames where other methods fail under the LFR situations. (Top row): Colour Mean Shift, (Middle row): Particle Filter with HSV, and (Bottom row): SURF tracking.

tracking algorithms, i.e. the distance of an object between two frames must be relatively small. The appearance model of using either colour or HSV histogram generally fails to associate (track) the foreground correctly when lighting changes occur because these features are not stable over different lighting conditions. Similarly, the use of SURF features to perform object correspondence between frames generally has low success rates given the features in the human body appear and disappear when they move (although it has the lighting invariant properties). Figure 3.50 shows examples of the tracking failures under low frame rates. On the other hand, our proposed tracking framework models the target movements according to the bus environment and incorporates spatiogram of oriented gradient (with rotational invariant property) and inverted cumulative colour histogram to represent the changes in spatial, textures, and colours under the LFR conditions. In general we could achieve a balance between the number of true positives and false positives, which is demonstrated from our experimental results.

### 3.5 Chapter Summary

In this chapter we have presented a system for passenger detection and tracking in moving bus environments, under low frame rates and varying lighting situations. The novelty of the detection lies on the use of prior knowledge about the environment, in this case the seat structures of the bus. This offers a number of advantages: (1) It allows the combination of foreground detection (best fitted human model) and background seat modelling (SIFT codebook background modelling) in a weighted Bayesian model using the assumption that a passenger can only occupy a single seat at one time instance. Since SIFT features and human models are both partially invariant to lighting, the algorithm can be used to effectively detect passengers while the bus is moving (i.e. even when different lighting situations occur). (2) It saves the overall computational time in building the background model and searching for head candidates. (3) It greatly reduces false detections and at the same time has a reasonable detection rate (small numbers of mis-detections). In addition, we propose a multi-target tracking framework that combines the prediction of the object movement using the trajectory model and an appearance transformation model that learns the foreground's colour (ICH), texture (DHOG), and spatiogram (SHOG) using SVM classifiers, which is suitable for tracking foregrounds in a low video frame rate.

We have demonstrated the results of our system performance tested on real bus footage. The experimental results demonstrate the robustness of the proposed detection and the tracking algorithms in dealing with low frame rates and lighting changes. It also shows the superiority of our proposed methods against other existing techniques in these fields.

Finally, the “switching seat” results on real world public transport systems demonstrate the effectiveness and usefulness of our approach.

In the next chapter, we look at the pedestrian detection problem for mobile bus surveillance.

---

## CHAPTER 4

# PEDESTRIAN DETECTION FOR MOBILE BUS SURVEILLANCE

---

In the previous chapter, we have discussed a Bayesian framework to detect and track passengers inside moving buses. In this chapter, we consider pedestrian detection scenarios involving scenes captured by mobile bus surveillance cameras in busy city streets. Unlike the passenger detection problem, we are not able to construct the background model in a traditional way, because the background scenes are constantly changing while the bus moves. A recent technology allows the use of front facing cameras mounted on bus fleets to collect large quantities of similar scene video over time. The *VirtualObserver* (Greenhill and Venkatesh, 2006) technology allows the footage retrieval for a given GPS location and time of interest. It returns a “time-ordered” sequence of footage acquired by a number of sensors, in a pseudo time-line, as shown in Figure 4.1. The footage at a given GPS location, returned by a number of buses can be different because: (a) GPS data may be noisy, (b) The sensor in one bus may have a different view point to that in another bus, (c) Even if the footage for a sensor is amalgamated, it may view the environment from slightly different positions in the road (e.g. different lane). Further for each sensor, the data is taken under differing lighting conditions, contains moving scenes, and is of low frame rate low resolution, imposing great challenges for pedestrian detection from these video images.

In this chapter we present a new foreground detection system that tackles these challenges. Our approach integrates scene localization, foreground-background separation, and pedestrian detection modules into a unified detection framework. The scene localization module transforms sequences of video data into arrays of images grouped as “view sites”. A view site is a “unique region” storing images captured from different times, but having *highly similar background structure and lighting* (i.e. they can be aligned in static-camera-manner), which can then be used for foreground-background separation. To do this, we first use the *VirtualObserver* to retrieve all video images captured at a particular GPS location. Then, SIFT-Homography is applied on these images to effectively estimate the scene-to-view-site mapping, measured by the scene structural similarities. For each

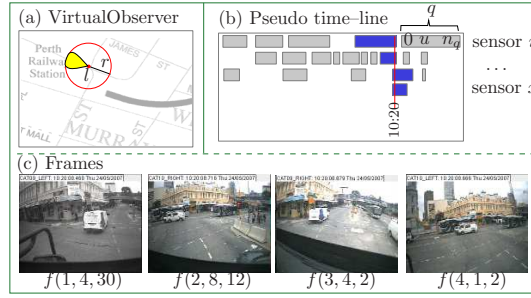


Figure 4.1: A VirtualObserver. Given a location  $l$  with radius  $r$  (a) and the pseudo time-line (b), these can be used to navigate the interested frames at the particular region and time. (c) The snapshot images from different sensors  $\{i, \dots, x\}$  taken at similar GPS location.

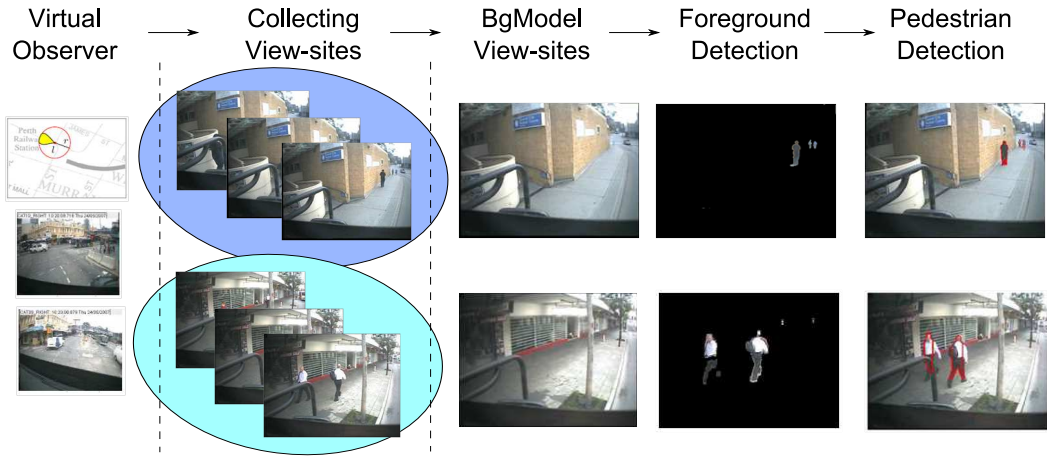


Figure 4.2: Overview of the proposed pedestrian detection algorithm.

view site, the images are further clustered according to their background intensities to produce clusters of images having similar background lightings. With the large amount of data, we are able to generate a rich profile of spatially aligned images for any view site along the bus route. This way, the kernel density estimation (KDE) methods for colour and gradient foreground-background separation can be used to construct the background model for each intensity cluster and used to detect the foreground objects. Finally, using a hierarchical template matching approach on the foregrounds from KDE, pedestrians can be identified. The overview of our proposed framework is shown in Figure 4.2.

The rest of the chapter is organised as follows. Section 4.1 and 4.2 discuss each of the algorithms in detail. Section 4.3 presents the experiment results. Finally, Section 4.4 concludes the chapter.

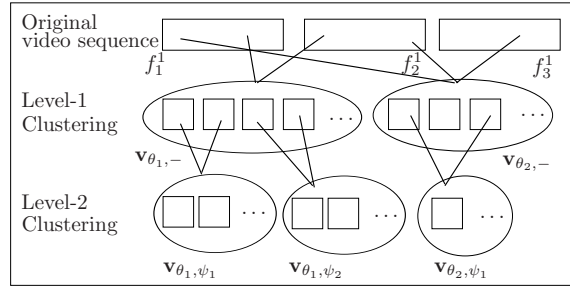


Figure 4.3: Scene localization.

## 4.1 Scene Localization

The video is captured using a *VirtualObserver* and a two stage clustering is performed: Level-1 extracts frames that are aligned in terms of structure for each sensor, and Level-2 further clusters these aligned frames in terms of lighting. This two stage approach caters for differential viewpoint (Level-1 clustering) and differential lighting (Level-2 clustering). Figure 4.3 shows the clustering stages. The clustering results at Level-2 are then used to learn the background model for each sensor.

### 4.1.1 Location Based Clustering (L1)

The *VirtualObserver* performs geometric queries, given a GPS location  $l$ , radius  $r$ , and time period of interest. It returns all geo-referenced tracks and associated video footage that satisfies the query. For example, given a query location shown in Figure 4.1, the *VirtualObserver* returns a “time-ordered” sequence of footage acquired by a number of sensors in a pseudo time-line. The pseudo time-line shows that although the footage is discontinuous, it can be navigated in an orderly fashion.

The footage at a given GPS location, returned by a bus can be different because (a) GPS data may be noisy (b) The sensor in one bus may have a different view point to that in another bus (c) Even if the footage for a sensor is amalgamated, it may view the environment from slightly different positions in the road (e.g. different lane). Thus, for a given GPS location, we define a view site as  $(\mathbf{v}_\theta = \{f_1, \dots, f_{m_\theta}\})$ , which corresponds to a region in which sensors return visually similar footage (where  $\theta$  is the view-site index,  $\{f_j : j = 1, \dots, m\}$  are the images in the cluster view-site  $\mathbf{v}_\theta$  and any frame in a view site can be indexed as  $\mathbf{v}(\theta, e)$ , where  $e$  is the index for the frame).

We introduce the following terminology. Let the footage from sensor  $i$  be amalgamated. Let the pseudo time-line block index, as shown in Figure 4.1 be  $q$ , with  $n_q$  number of frames in each block. Then, any frame can be indexed as  $f(i, q, u)$  or  $f_{q_u}^i$ , where  $1 \leq u \leq n_q$ . The task is to find the mapping from the set of frames  $f_{q_u}^i$  to the “most appropriate” view site  $\mathbf{v}$  from  $\mathbf{V} = \{\mathbf{v}_\theta\}_{\theta \in \Theta}$ , where  $\Theta$  defines the entire view site space. This can be done by a process of incremental clustering. The measure of similarity between the cluster center in a view site ( $c_{\mathbf{v}_\theta}$ ) and an incoming frame  $f_{q_u}^i$  is calculated as

$$s(c_{\mathbf{v}_\theta}, f(i, q, u)) = \text{homog}(\mathbf{v}_\theta, f_{q_u}^i) \quad (4.1)$$

where  $s(\cdot)$  is the similarity function,  $\text{homog}(\cdot)$  is a function to compute a Homography matrix,  $\mathbf{H}$ , between a view-site  $\mathbf{v}_\theta$  and an image  $f_{q_u}^i$ . This function is defined as follows: given two points  $b$  and  $b'$ , which correspond to the same point in  $\mathbf{v}_\theta$  and  $f_{q_u}^i$  respectively, then there exists a 3x3 Homography matrix  $\mathbf{H}$  such that  $b = \mathbf{H}b'$  (Hartley and Zisserman, 2004). The Homography matrix can be estimated using the least square algorithm, given at least 4 or more corresponding points. Since one would expect the illumination conditions between the matching image pair to vary considerably, features that are resilient to illumination variations are used. In the implementation, the point matching is defined as corresponding between the stable SIFT features from the two scenes (Lowe, 2004; Hartley and Zisserman, 2004). In order to achieve a high alignment accuracy, we impose a strict constraint on  $\mathbf{H}^*$  matrix to only accept a small degree of changes between the two scenes. In this implementation, we restrict the deviation of the scene transformation between the scene and the view sites to just 2% before accepting them.

Given baseline view sites, this alignment process can be adopted to register further images collected from different sensors (i.e. buses) taken at different times, therefore generating a rich profile of spatially aligned images for any view site along the bus route.

### 4.1.2 Intensity Based Clustering (L2)

In a view site, since images are grouped based on the locations at different times, one has to deal with large variations in illumination and dynamic backgrounds (e.g. a store which is closed during early mornings and opened during afternoons). Hence, the aim is to group images in a view site into sub-clusters of images having similar background intensity properties.

A typical approach for scene grouping is to compute the closeness or the differences of the global intensities between two images, like the spatial envelope representations proposed by Oliva and Torralba (2001) for scene recognition. If the global intensity differences are small, then the two images are considered similar and assigned into one group. However, this approach is limited to images that contain predominantly background information only. But in our case, we want to avoid computing these differences when the foregrounds are present. We overcome this problem by applying SIFT matching across scenes at different times in a view-site set. Using SIFT matching, we can control the fact that the corresponding pixels between two images always belong to the background, and only the differences between these determined background pixels are computed.

Formally, the task is to break down the frames in a view site  $\mathbf{v}_\theta$  into lower level sub-clusters  $\mathbf{v}_{\theta,\psi}$  containing images with similar global intensity values, where  $\psi$  denotes the intensity cluster index (note that the index  $\theta$  represents the first level clustering to group images by location, whereas  $\psi$  is the image clustering based on their intensities). This again can be achieved by incremental clustering. Let  $S$  be the set of all SIFT keypoints in the intensity cluster center ( $c_{\mathbf{v}_{\theta,\psi}}$ ), that correspond to keypoints in the image  $\mathbf{v}(\theta, e)$  inside a view site. Thus, we define the measure of similarity between ( $c_{\mathbf{v}_{\theta,\psi}}$ ) and  $\mathbf{v}(\theta, e)$  as the accumulated differences of the region surrounding SIFT::

$$s(c_{\mathbf{v}_{\theta,\psi}}, \mathbf{v}(\theta, e), S) = \sum_{\{b,b'\} \in Sift} |\text{hoi}(b) - \text{hoi}(b')| \quad (4.2)$$

where  $b$  and  $b'$  denote the same SIFT keypoints (background pixels) that appear in  $c_{\mathbf{v}_{\theta,\psi}}$  and  $\mathbf{v}(\theta, e)$  respectively,  $\text{hoi}(\cdot)$  is a function that computes the Gaussian histogram-of-intensities (G-HOI) patches around each background pixel. In SIFT, each local keypoint descriptor is represented as a Gaussian weighted and interpolated histogram-of-gradient orientations (G-HOG) (Lowe, 2004; Dalal and Triggs, 2005). We adopt the same idea, but apply it on the intensity instead (G-HOI). We divide the region into 8 patches organized by  $2 \times 2$  grid, each with 16 bins. The histogram is then constructed by accumulating the weighted Gaussian kernel around the center of each keypoint. The resulting intensity patch descriptor is a 84 dimensional vector ( $2 \times 2 \times 16$ ). Intuitively, comparing two G-HOG determines the similarity of the gradient-patch structure, which is partially invariant to illumination (Lowe, 2004), while comparing two G-HOI allows us to further compute their intensity-patch closeness.

Since our objective is to detect the foreground regions of a given scene, we will use the



images taken from the intensity cluster from each view site,  $\mathbf{v}_{\theta,\psi}$ , to assist in segmentation. This foreground segmentation technique is proposed in the next Section.

## 4.2 Combined Foreground Background Separation and Pedestrian Detection

We formulate the problem of foreground-background separation using a non-parametric approach similar to (Elgammal *et al.*, 2000).

### 4.2.1 KDE on Colour

Given a set of images  $\mathbf{v}_{\theta,\psi} = \{f_1, \dots, f_n\}$  in a specific view-site index by  $\theta$  and  $\psi$ , then the background model can be estimated using the Parzen window technique (Scott, 1992) (because these images exhibit both similar scene structure and lightings). We assume that the background can be observed as the highest pixel occurrences and KDE method with Gaussian function is used to construct the background model. Let  $\mathbf{x}_1, \dots, \mathbf{x}_n$  be a sample of intensity values for a particular pixel on the entire image set  $\mathbf{v}_{\theta,\psi}$ , in  $d$  dimensional space from a multi-variate distribution  $P(\mathbf{x})$ , then an estimate of  $\widehat{P}(\mathbf{x})$  of the density at  $\mathbf{x}$  can be calculated using

$$\widehat{P}_{\mathbf{H}}(\mathbf{x}) = \frac{1}{n} |\mathbf{H}|^{-\frac{1}{2}} \sum_{i=1}^n K(\mathbf{H}^{-\frac{1}{2}}(\mathbf{x} - \mathbf{x}_i))$$

where  $K$  is a kernel estimator function and  $\mathbf{H}$  is a symmetric  $d \times d$  kernel bandwidth matrix. In the case where the kernel estimator is a Gaussian with  $(0, \Sigma)$ , i.e.  $\mathbf{H} = \Sigma$ . When dealing with colour (RGB) distribution, we can assume that each colour is independent of each other, thus  $\Sigma$  is a diagonal matrix, i.e.  $\Sigma = \text{diag}[\sigma_1^2, \sigma_2^2, \dots, \sigma_d^2]$ . The final density estimation is written as

$$\widehat{P}_{\sigma}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n \prod_{j=1}^d \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{1}{2} \frac{(x^j - x_i^j)^2}{\sigma_j^2}} \quad (4.3)$$

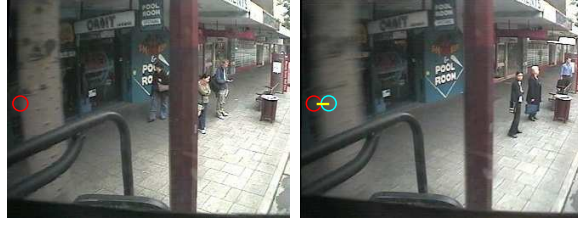


Figure 4.4: Example of misalignment. The two images are taken from a bus at different times. The red circle on the left image indicates a “tree” pixel that has moved to the cyan circle in the right image.

Intuitively, Equation 4.3 can be interpreted as computing the pixel occurrence across  $n$  samples, weighted using a Gaussian kernel around the central pixel with a fixed standard deviation. We choose three  $\mathbf{x}^*$  that have highest occurrences as the possible background pixels and the rest are set to foreground pixels. We further check if the probability score,  $\hat{P}(\mathbf{x}^*)$  is greater than a certain threshold, then they are assigned as background pixels, else they are considered as foreground pixels.

The result of this process is a separation of foreground and background pixels. However, false positives may be introduced as a result of image misalignment (from the scene localization process). Since these misalignments are due to camera motion, we can solve this problem by analyzing the neighbourhood properties, as motivated by the work of Elgammal *et al.* (2000). If the correlation of a pixel to its neighbourhood is high, then it is most likely to be a background pixel that has “moved” to a new location, otherwise, it is a foreground pixel that occludes the background. We define this probability as the neighbourhood probability. Let  $x_f$  be the observation of pixel  $x$ , that is detected as a foreground pixel. The neighbourhood probability is defined as:

$$P_{\mathcal{N}}(x_f) = \max_{y \in \mathcal{N}(x)} P(x_f | B_y)$$

where  $B_y$  is the background model at pixel location  $y$  and  $P(x_t | B_y)$  is computed using Equation 4.3. Although this approach addresses the misalignment problem, it introduces the problem of losing true detections, as mentioned by Elgammal *et al.* (2000). Hence we incorporate a further constraint on the neighbourhood conditions, i.e. if a pixel is truly a background pixel that happens to move to a new location, then some of its neighbourhood pixels should also move since we assume that the misalignment happens not only for one single pixel, but for a group of pixels as well, which can be seen in Figure 3. Therefore, we define the probability of displacement as:

$$P_C(x_f) = \prod_{y \in \{x_f \cup \mathcal{N}(x_f)\}} P_N(y) \quad (4.4)$$

where  $P_C(x_f)$  is computed as the product over the connected components of the neighbourhood pixels. Therefore, a pixel  $x$  is considered as background only if both  $P_N(x) > th1$  and  $P_C(x) > th2$ .

### 4.2.2 KDE on Gradient

We build a gradient background model in a similar way to the colour based approach described in Section 4.2. First, we transform the colour image into an intensity image and then compute the edge (gradient) image using  $\Delta(I) = \sqrt{d_x^2 + d_y^2}$ , where  $\Delta$  is the magnitude gradient image,  $d_x$  and  $d_y$  are the horizontal and vertical rates of changes of  $I$  respectively. Once we obtain a set of gradient images, we employ the same kernel density estimation method to search for the maximal occurrence of the edge pixels and mark them as background edges.

### 4.2.3 Combined KDE Colour and Gradient

The results of the colour-based and gradient-based approaches are finally combined to determine the foreground regions. This approach can be outlined as follows: given a set of foreground regions in colour and gradient images respectively, we assume that a region is a foreground if and only if they are both strong in the edge and gradient foregrounds. A similar method has been used in (Javed *et al.*, 2002). Therefore, we compute the combined score of a given foreground region as:

$$score = \frac{1}{|G|} \sum_{i \in \delta R_a} \Delta I(i).G(i)$$

where  $I$  is the original gradient image,  $G$  is the foreground gradient image,  $R_a$  is the foreground region. If the *score* is less than a threshold, then the region is considered as a background.

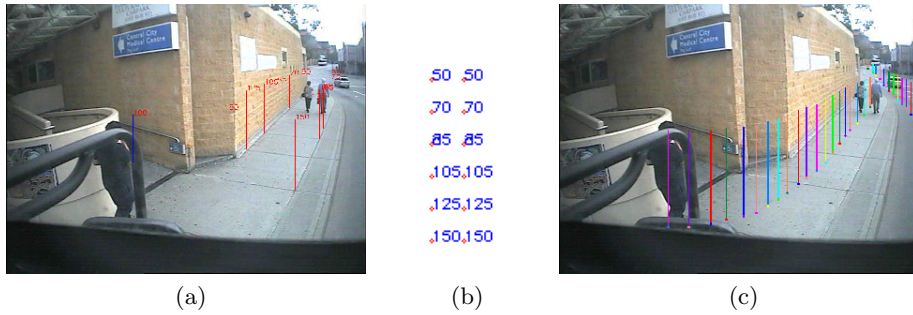


Figure 4.5: Foot-to-head calibration using Homography method. (a) An input image to be calibrated. A set of bars are manually labelled specifying approximated human height. (b) The corresponding 2D point mapping. (c) Given a set of foot positions, the predicted heights are estimated automatically.

#### 4.2.4 Pedestrian Detection

After the KDE subtraction process, areas of strong background gradient and colour have been removed. This way, the segmentation depicts the possible area of search for the pedestrians. The advantages are it can be efficiently computed and a lot of possible false positive candidates have been removed.

We assume that a pedestrian walks vertically on the ground plane. Under this assumption, we employ a similar approach for detecting pedestrians as described in (Zhe *et al.*, 2007), and apply it on the KDE foreground-background separation results. First, we construct a hierarchical set of templates for head, torso, and legs. During the detection, an optimized Bayesian formulation is used to find the best template configuration that matches the pedestrian. This hierarchical template matching can be performed efficiently for three reasons: First, using the hierarchical approach, we truncate unnecessary matches. Secondly, the result of the foreground-background separation allows us to narrow down the search for pedestrian locations in the frame of interest. And finally, the pedestrian’s scale (height) can be estimated using a priori information like offline foot-to-head Homography calibration. An example of the calibration process is shown in Figure 4.5.

### 4.3 Experimental Results

In order to verify the performance of our proposed system for foreground detection, we conduct a series of experiments using real world bus data. The experiments are comprised

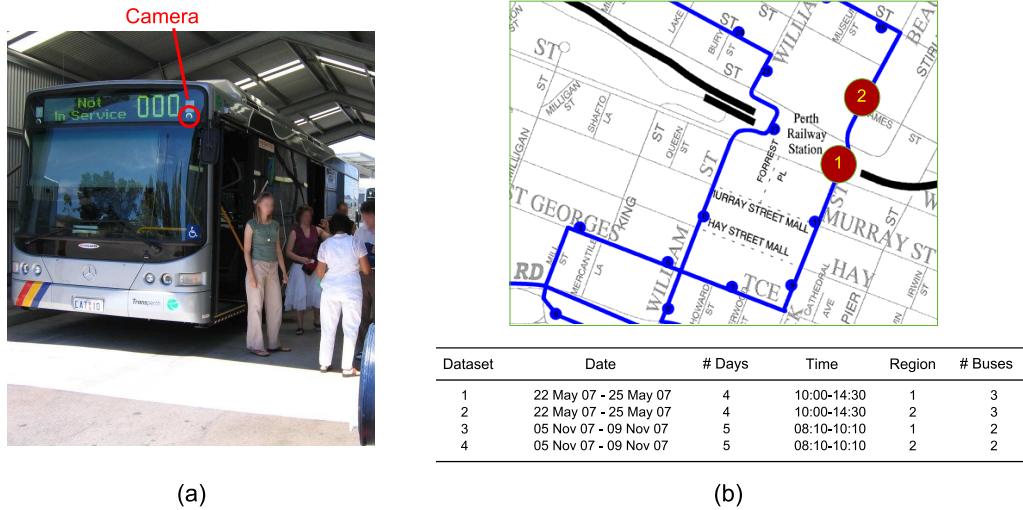


Figure 4.6: Perth dataset: (a) camera setup and (b) description of the dataset.

of two parts: (a) the comparison the proposed KDE approach against with Stauffer and Grimson (2000)’s GMM approach, in terms of foreground detection (Section 4.3.3) and (b) the performance evaluation of our system on different bus datasets (Section 4.3.4).

### 4.3.1 Perth Dataset

We tested our system on 4 sets of video streams taken by the outward facing cameras mounted on buses. The selected buses were BlueCat buses plying the major routes within the City of Perth, Western Australia, which we call the Perth dataset. The onboard GPS records the location of the bus synchronized with video frames. Figures 4.6 (a) and (b) show the physical camera location and the detailed information about the dataset.

Each dataset consists of video images recorded at approximately 10 fps with a resolution of  $768 \times 576$  pixels. Regions 1 and 2 have a coverage radius of 250 metres and each region dataset is collected at different dates and times (one is on May 2007 and the other is taken at November 2007) as shown in Figure 4.6 (b). We then evaluate the system performance separately based on the region. The first region dataset captures images on busy streets, including cluttered complex backgrounds (for example shops) and crowded pedestrians, while less frequent pedestrian traffic occurs in the second region dataset. For each region, we first perform the scene localization, construct the background model on 80% of the datasets (for both May and Nov), extract the foregrounds on the remaining 20%, and finally detect the pedestrians. Also, we process the video images specific to each sensor, avoiding the variability of the different intrinsic camera parameters. For testing

purposes, we extract 1281 frames from region 1 that contains 947 annotated ground truth pedestrians, and 1174 frames from region 2 that contains 722 pedestrians.

**Evaluation Criteria:** To evaluate the detection performance we apply two criteria: *relative distance* and *ratio of the cover* (Leibe *et al.*, 2005), as shown in Figure 4.7.

$$cover = \frac{area(detected \cap groundTruth)}{area(groundTruth)} \quad (4.5)$$

A detected candidate is considered to be true positive when its *relative distance* from the object is less than 0.5 times the actual size of the ground truth’s bounding box and the *cover* is above 50%. Anything else is considered as a false positive.

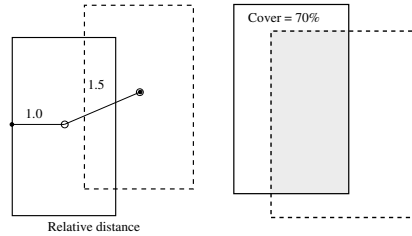


Figure 4.7: Evaluation criteria for comparing the ground truth (solid) bounding box with the detected candidate bounding box (dash).

### 4.3.2 Scene Localization Results

From the Perth datasets, we successfully registered 189 view-sites at Region 1 and 212 view-sites at Region 2 using the Homography image registration method. At each view-site, the data is further registered based on intensity values with the number of frames varying from 64 to 233. Figure 4.8 shows an example of the registered results from location to view-sites and then to intensity clusters. As can be seen, Figures 4.8 (f) and (g) show significant image differences due to time of the day differences. One group has the store door closed in the mornings and the other group has the door opened in the afternoons. Similarly, Figures 4.8 (h) and (i) also show the different lighting levels in the background. The first group has lights switched on, while the second group has the lights off. Figure 4.8 (e) shows images taken at different times can be grouped inside a similar view-site. This can be seen from the scenes containing large variability of people and cars.

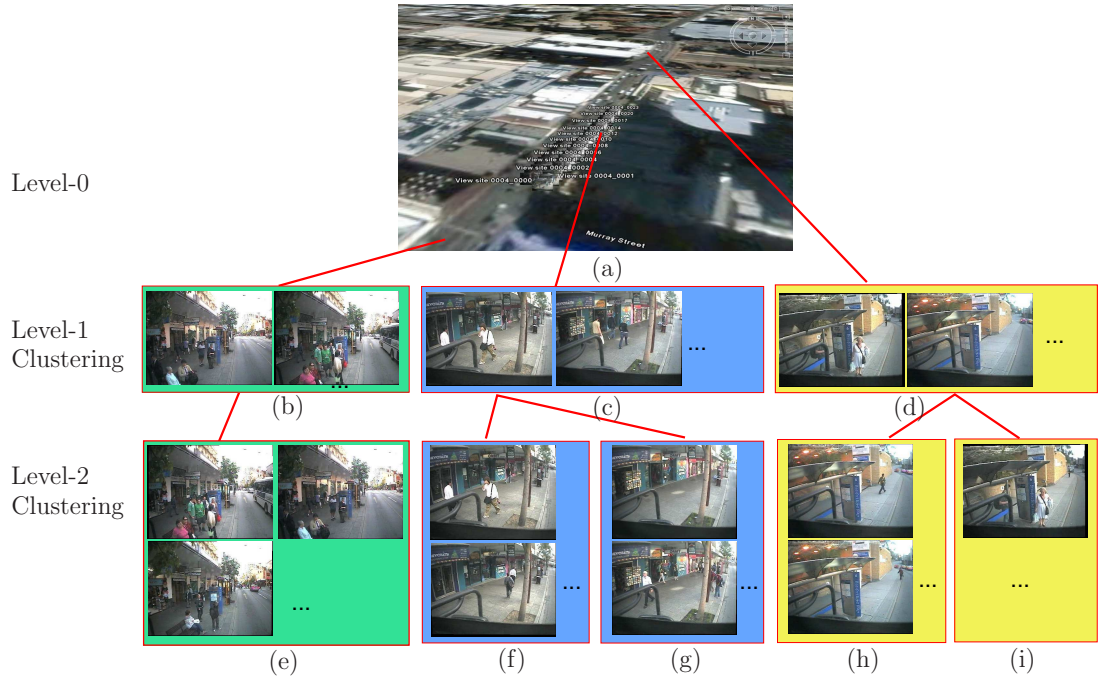


Figure 4.8: Scene localization results (hierarchical tree representation of scene localization process). First, images at Level-0 represent original video sequences captured from moving bus cameras. These images are then clustered into the “appropriate” view-sites as shown in Level-1 Clustering ( $L_1$ ) based on their *scene-transformation similarities*. The bottom layer Level-2 Clustering ( $L_2$ ) is obtained by sub-clustering  $L_1$  based on their *global background intensity values*. Images at  $L_2$  exhibit properties of both aligned in a static manner and having similar background intensities, which can be used to construct background models to detect foregrounds-of-interest.

We measure the performance of the scene localization by using known ground truth. The accuracy of the proposed scene localization method is shown in Table 4.1.

Region	True positives	False positives
1.	92%	10%
2.	89%	7%

Table 4.1: Scene localization accuracy tested on two different datasets.

From Table 4.1, we see that there exists a small percentage of false positives, which are mainly caused by light reflecting off one of the side mirrors causing the SIFT-Homography alignment process to fail.

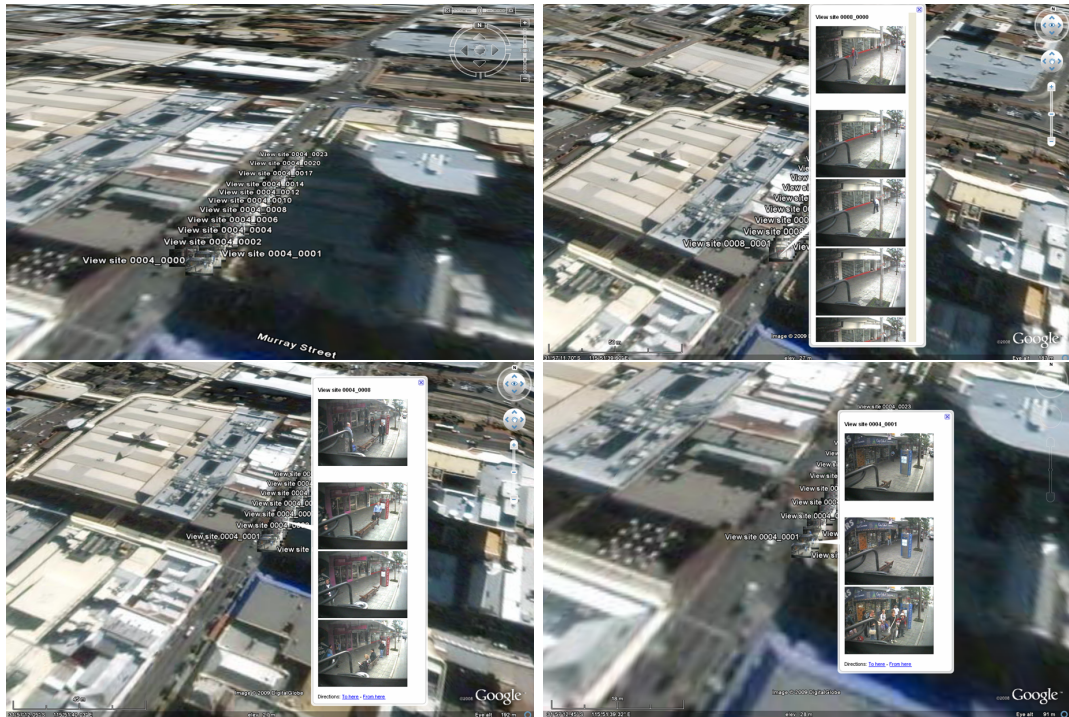


Figure 4.9: Visualising the view-sites on GoogleEarth (2009).

### 4.3.3 Comparison Between KDE and GMM Methods

We compare the performance of Stauffer and Grimson (2000) background subtraction and the proposed KDE-based method by computing the true and false segmentations against the manual ground truth. We test the segmentation results on the Perth dataset. Table 4.2 shows the comparison results of the two approaches, while the snapshots of the segmentation results are shown in Figure 4.10. In this figure, the background reconstruction models are shown on the first column. Next, we see that Stauffer’s approach (third column) produces more false positives as compared to our KDE background subtraction approach (fourth column). The advantage of our method is evident as it reduces false positives by incorporating the neighbourhood probability. On the other hand, the Stauffer-Grimson approach which deals only with pixel wise background models is unable to handle the pixel misalignment issues and results in higher false positive detections.

	Stauffer-Grimson	Our proposed method
True positives	86%	82%
False positives	28%	11%

Table 4.2: Comparison performance results between Stauffer-Grimson and the proposed KDE background subtraction.



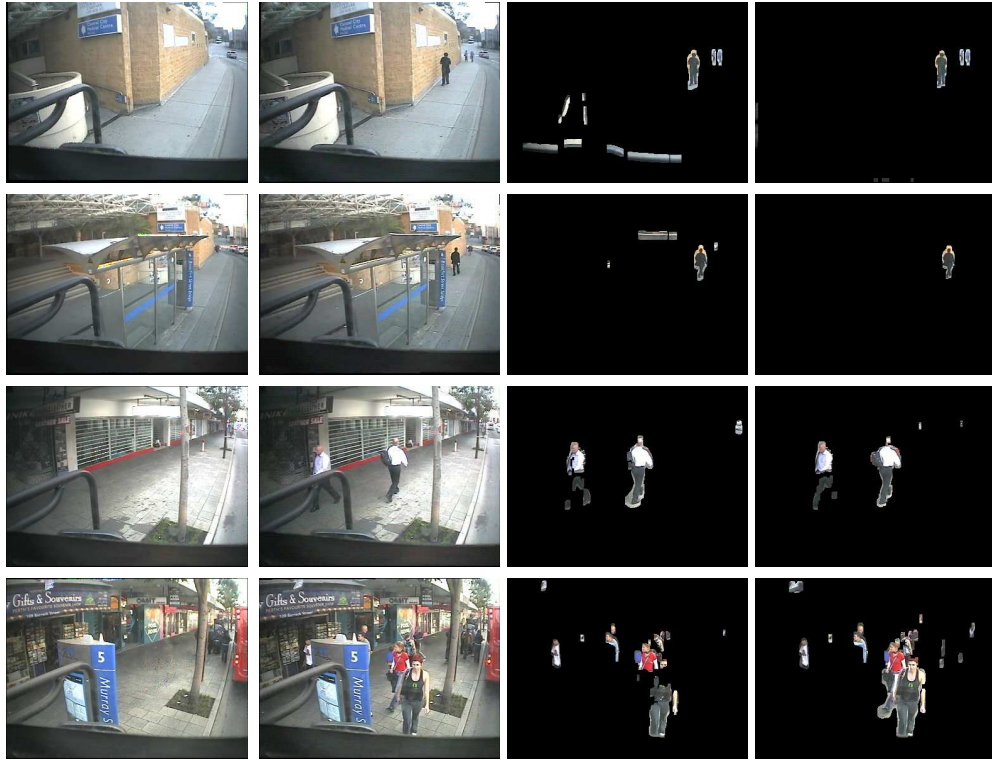


Figure 4.10: Comparison results of the foreground detections between Stauffer-and-Grimson’s based (third column), and our proposed KDE-based method (last-column). The first and second column images are the reconstructed background model and the original images to be compared.

#### 4.3.4 System Performance

We further evaluate the performance of our proposed algorithm against the hierarchical pedestrian detection (Zhe *et al.*, 2007). We measure the performance of each method using ROC curves, showing the trade-off between detection rate and false positives per frame as shown in Figure 4.11. The blue and red solid lines represent the performance of our proposed algorithm for regions 1 and 2 respectively, whilst the green and purple dotted lines represent the performance of the hierarchical pedestrian detection (Zhe *et al.*, 2007). From the figure, we can see that our proposed algorithm outperforms the pedestrian detection alone. This is mainly due to the high number of false positive candidates being discarded using the KDE background subtraction approach. It can also be seen that the performance plots for datasets in region 2 are slightly better compared to those in region 1. This is due to the complexity of the background and the degree of occlusion. Figure 4.12 shows the snapshots of the pedestrian detection results. The first three rows show the detection results in region 1, where cluttered complex backgrounds (for example shops) and crowded pedestrians are present. The last row of this image displays results of the

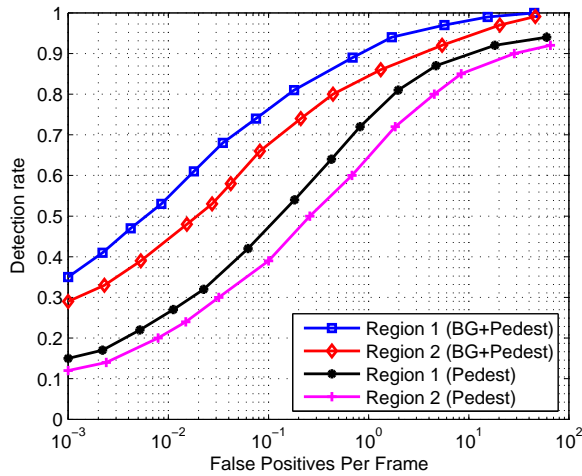


Figure 4.11: System performance.

detected pedestrians in region 2. Instances where our system returns false foreground detection are generally due to cluttered backgrounds or the misclassification of images into view-sites during the scene localization, resulting in comparison with the incorrect background model. The mis-detections are normally caused by the lack of foreground edges, under-segmentation and relatively small foreground regions.

## 4.4 Chapter Summary

In this chapter we have presented an integrated system for pedestrian detection involving mobile surveillance cameras. We have proposed a new approach to localize moving scenes captured from the bus camera into the appropriate view-sites (location) and further cluster them based on their global-background-intensity similarities. Over a period of time, the resulting view-sites construct a stable background model. A KDE background model is then employed for building the background model and is used to detect the objects of interest. Finally, hierarchical templates for a pedestrian model are generated to detect people. We have evaluated the performance of the system using the real world datasets, on city streets, demonstrating the feasibility of our method. By combining the results of the background model and the pedestrian detection we are able to achieve lower false positives whilst maintaining a reasonable detection rate. The only drawback of the proposed method is the computational cost of building the view sites. In the next chapter, we will evaluate different direct pedestrian detection methods for fast detection.

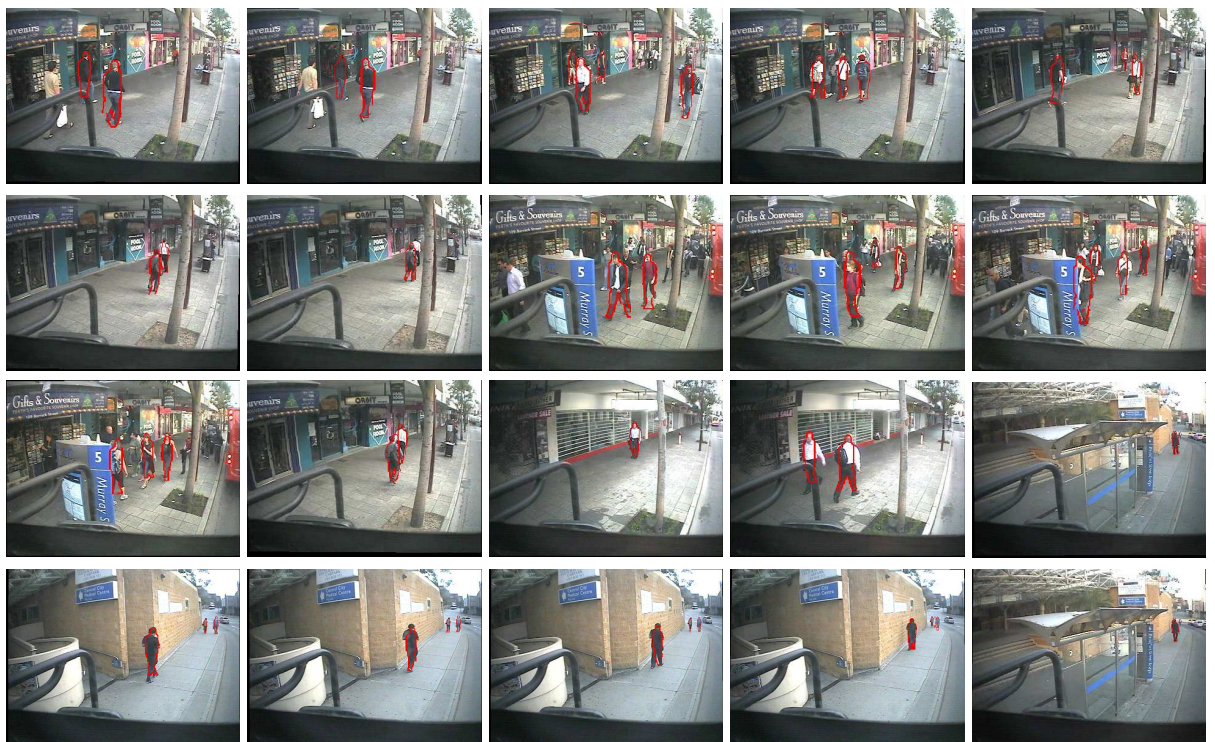


Figure 4.12: Pedestrian detection results on Perth datasets.

---

## CHAPTER 5

# COMPARATIVE EVALUATION OF PEDESTRIAN DETECTION METHODS

---

The previous chapter discussed a method to detect pedestrians by aligning images and building background model for the scene. Another way to detect pedestrian is to learn their shape through training samples. In this chapter we present three direct pedestrian detection methods that extend the HOG approach (Dalal and Triggs, 2005). In addition, we compare and evaluate their performance on the Perth dataset. The three approaches are: **a)** a new histogram feature that is formed by the weighted sum of both the gradient magnitude and the filter responses from a set of elongated Gaussian filters (Leung and Malik, 2001) corresponding to the quantised orientation, called Histogram of Oriented Gradient Banks (HOGB) approach; **b)** the codebook based HOG feature with branch-and-bound (efficient subwindow search) algorithm (Lampert *et al.*, 2008) and; **c)** the codebook based HOGB approach.

The layout of this chapter is organised as follows. Section 5.1 discusses each algorithm in detail. Section 5.2 presents the Perth pedestrian dataset. Section 5.3 provides the comparative evaluation performance of all approaches. Finally Section 5.4 concludes the chapter.

### 5.1 Different Pedestrian Detection Methods

#### 5.1.1 Method 1: Histogram of Oriented Gradient (HOG)

The HOG algorithm Dalal and Triggs (2005) operates in two steps: 1) building of pedestrian model; 2) pedestrian detection.

**Building of pedestrian model:** Each detection window is divided into cells of  $8 \times 8$

pixels. For each cell, HOG computes the accumulated gradient magnitude corresponds to a quantised orientation to form a 9-bin histogram:

$$\{h(\theta_i) = \sum_{p \in 8 \times 8} M_i^p : i = 1, \dots, 9\}$$

where  $h$  denotes the histogram,  $M_i^p$  refers to the gradient magnitude corresponding to a quantized orientation  $\theta_i$  at pixel  $p$ , which is computed using  $3 \times 3$  Sobel kernel. Each patch consists of  $2 \times 2$  cells. The four histograms in a patch are concatenated to produce a normalised 36-D feature vector. For a detection window of size  $96 \times 160$  pixels with  $7 \times 15$  patches, we obtain a total of 3780 features. These features are then used to train a linear Support Vector Machine (SVM) classifier.

**Pedestrian detection:** The detection window is scanned across the entire image to identify the pedestrians. In general, the scanning process has to take into consideration varying scales of the pedestrian, which thus suffers from high computational cost. To relax this problem, we assume that a pedestrian walks vertically on the ground plane and we employ a foot-to-head calibration strategy to estimate the scale of the pedestrian for a given point in the image. This is done using Homography similar to (Zhe *et al.*, 2007). Figure 5.1 shows an example of the estimation process. In our application, since the buses travel consistently on the same bus lane on a fixed route, we only need to perform the offline-calibration once for each bus.

### 5.1.2 Method 2: Histogram of Oriented Gradient Banks (HOGB)

We introduce a new feature for pedestrian detection namely Histogram of Oriented Gradient Banks (HOGB). We further process the gradient image using 9 elongated oriented Gaussian filters (Leung and Malik, 2001), and let  $\{R_i^p : i = 1, \dots, 9\}$  be the outputs recorded for each pixel. Figure 5.2 shows an example of the elongated Gaussian filters ( $11 \times 11$ ). Then for each cell, the histogram is computed as the weighted sum of the original gradient magnitude and the filter bank response corresponding to the orientation:

$$\{h(\theta_i) = \sum_{p \in 8 \times 8} \beta \times M_i^p + (1 - \beta) \times R_i^p\} \quad (5.1)$$

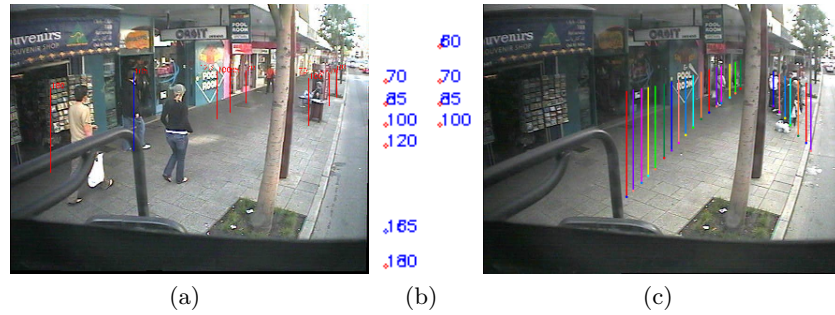


Figure 5.1: Calibration process. (a) An input image to be calibrated where a set of bars are manually labeled specifying the approximated human height at certain  $x$  and  $y$  location. (b) The corresponding 2D point mapping specifying the  $x$  and  $y$  location. (c) Once the calibration model is obtained, given a set of feet positions, the predicted heights are estimated automatically.

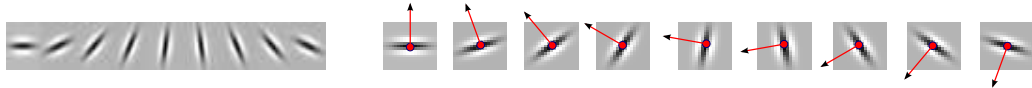


Figure 5.2: The nine banks of elongated Gaussian filters and their orientation.

where  $\beta$  defines the weight of choosing between the HOG or Leung Malik features. Intuitively, the first and second terms in Equation 5.1 model the shape of a pedestrian at a fine and coarse scales respectively. With the same detection window structure as HOG described in Section 5.1.1, our final feature vector also consists of 3780 entries per detection window, and we employ a similar training strategy to obtain the classifier for detecting the pedestrians.

### 5.1.3 Method 3: Codebook HOG with ESS (CHOG)

Recently, Lampert *et al.* (2008) propose a fast object detection method based on an efficient branch and bound algorithm. The technique is divided into two steps: 1) creation of codebooks; 2) feature-codebook quantization followed by an efficient search for peak responses. In this chapter we employ a similar idea but use HOG features to build the codebook as an alternative of the Speeded Up Robust Features (SURF) (Bay *et al.*, 2006).

**Building of codebook:** The codebook is created by first computing a set of 36-D HOG descriptors from patches of size  $20 \times 20$  pixels followed by the K-mean clustering. We build two codebooks for positive and negative samples respectively. These codebooks are concatenated and trained using linear SVM classifier to obtain a set of positive and



Figure 5.3: Codebook for pedestrians.

negative alpha (weight),  $\alpha_i$  for each codebook entry  $c_i$ . Figure 5.3 shows examples of the codebooks.

**Detection:** Given an image, we extract the HOG descriptors and quantize them using the codebook built during offline processing. Thus, each pixel can be represented as  $w_i = \sum_i \alpha_i o_i$ , where  $w_i$  is a score of confidence (or define the probability of object presence) and  $o_i$  is the count of the codebook occurrences during training. Let the quality function be  $f(I) = \sum w_{c_i}$ . Hence, we design a function  $\hat{f}$  that bounds the values of  $f$  over sets of rectangles as:

$$\hat{f}(R) = f^+(R_{max}) + f^-(R_{min}) \quad (5.2)$$

where  $R = [T, L, B, R]$  is a rectangle defining the [Top, Left, Bottom, and Right] interval coordinates, and each coordinate is defined as  $T = [t_{low}, t_{high}]$ ,  $f^+(R_{max})$  represents the positive weight responses for the largest rectangle and  $f^-(R_{min})$  is the negative weight responses under the smallest rectangle. Intuitively, Equation 5.2 always maintains the maximum response for region  $R$ , as it satisfies the bound conditions in (Lampert *et al.*, 2008). By combining function  $\hat{f}$  in Equation 5.2 with the branch and bound algorithm, we are able to detect pedestrians efficiently.

#### Method 4: Codebook HOGB with ESS (CHOGB)

The CHOGB approach uses the same implementation of CHOG but uses the HOGB feature instead.

## 5.2 Datasets and Evaluation Criteria

The two datasets used in our experiments are: the INRIA dataset (Dalal, 2005) and the dataset which we collected from 5 buses operated by the Public Transport Authority in Western Australia (Perth dataset). Figure 5.4 shows examples of the datasets used in our experiments.



Figure 5.4: (a) Examples of the INRIA dataset and (b–c) the Perth dataset that were used for training and testing in our experiments.

**Training dataset:** We divide the training dataset into positive and negative samples. Each training image is  $96 \times 160$  pixels in size. For positive training samples we use the INRIA pedestrian datasets, which consists of 2416 positives images. Each image contains a person standing against a wide variety of backgrounds including crowds. For negative training samples, we provide a total of 3145 manually cropped background images from the Perth dataset.

**Testing dataset and evaluation criteria:** We use the same testing data and evaluation criteria mentioned in Chapter 4 (Section 4.3.1).

## 5.3 Experiments

This section presents a comparative evaluation of four pedestrian detectors: HOG, HOGB, codebook HOG, and codebook HOGB. First, we present the implementation details for each detector, including the choice of selecting the parameters and the pre-processing. We then present discussion about the relative performances of these approaches based on detection accuracy and computational speed.



### 5.3.1 Implementation Details

**HOGB:** We assess the influence of different scales of the elongated Gaussian filter and the weight ( $\beta$  in Equation 5.1) on the performance.

◊ Scale and weight: Figure 5.5 shows the detection accuracy for varying  $\beta$  in Equation 5.1 and for different scales:  $(7 \times 7)$ ,  $(11 \times 11)$ ,  $(15 \times 15)$ ,  $(19 \times 19)$ . We observe that there is a reduction in both the false alarm and true detection rates with decreasing  $\beta$  (HOG), which implies that by introducing the filter banks, we obtain lower false positives, but sacrifice detection accuracy. In addition, we also observe that  $(11 \times 11)$  scale gives the best detection performance for the Perth dataset that consists of pedestrians of height ranging between 125-175 pixels. Based on the empirical results in Figure 5.5, we choose  $\beta = 0.9$  and scale of  $(11 \times 11)$  during evaluation.

◊ Similar to the HOG approach, we apply a Gaussian spatial mask and tri-linear interpolation in constructing the HOGB for each patch.

**CHOG:** The codebook is built as follows:

◊ Pre-processing: We perform pre-processing on the INRIA pedestrian datasets by selecting patches of size  $20 \times 20$  pixels around the shape of the pedestrians (shoulders, legs, and body, excluding the inner parts of the pedestrian) to obtain clean positive samples. We observe an increased detection performance of 5% as a result of this pre-processing step, as the pre-processing helps reduce false quantization during the detection process.

◊ Clustering: As mentioned in Dalal and Triggs (2005), the important cues for detecting pedestrian are head, shoulder, leg, and silhouettes. During the creation of codebook, we divide the patches based on its location into two groups: upper and lower parts of the pedestrian (ratio of 30:70); and perform the K-mean clustering independently for each group. This strategy helps improve the final quality of the codebook, since the codebook for the upper body part mainly consists of high curvature features (head and shoulder), whilst the lower part consists of more concentrated straight/vertical features (body and leg).

◊ Number of clusters: During the K-mean clustering process, we set  $K_u = 100$  and  $K_l = 400$  for creating the positive codebook, and  $K = 500$  for building the negative codebook, where  $K_u$  and  $K_l$  are the number of clusters/codebook entries for the upper and lower

parts of the pedestrian respectively. Thus, our final codebook consists of 1000 entries.

◊ CHOGB: The CHOGB approach uses the same implementation of CHOG, but using the HOGB feature instead.

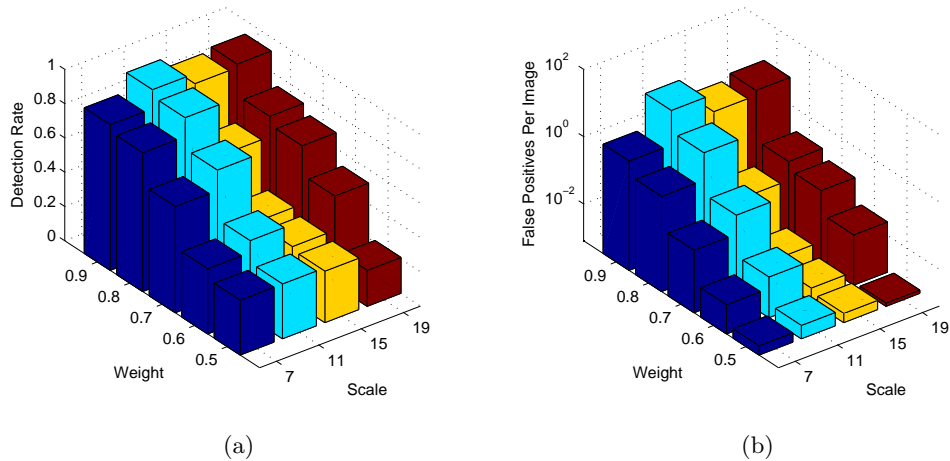


Figure 5.5: Performance of HOGB: (a) detection rate (b) false positives per image.

### 5.3.2 Detection Performance

Figure 5.6 (a) shows the comparative performance of the HOG, HOGB, CHOG, and CHOGB. The HOG based detector achieves the highest performance in terms of true positive detection. The HOGB approach has a comparable performance compared to the HOG. However, we notice a reduction in the performance for the codebook approaches, i.e. CHOG and CHOGB. This is mainly due to the low resolution images and highly cluttered background occurring in the video sequences, leading to false quantization when detecting parts of the pedestrian. Unlike the codebook approach, HOG and HOGB approaches are less sensitive to this problem, as they involve detecting a complete profile of the pedestrian.

We further present another comparative evaluation of the methods discussed in this chapter against the methods being proposed in the previous chapter (KDE + hierarchical template for pedestrian detection), since they use the same dataset and evaluation criteria. Figure 5.6 (b) shows the overall evaluation results. From the figure, we can see that the method that by combining the background subtraction method and direct foreground detection approach outperforms other pedestrian methods. Thus, it justifies the usefulness of combining both the background subtraction and direct foreground detection

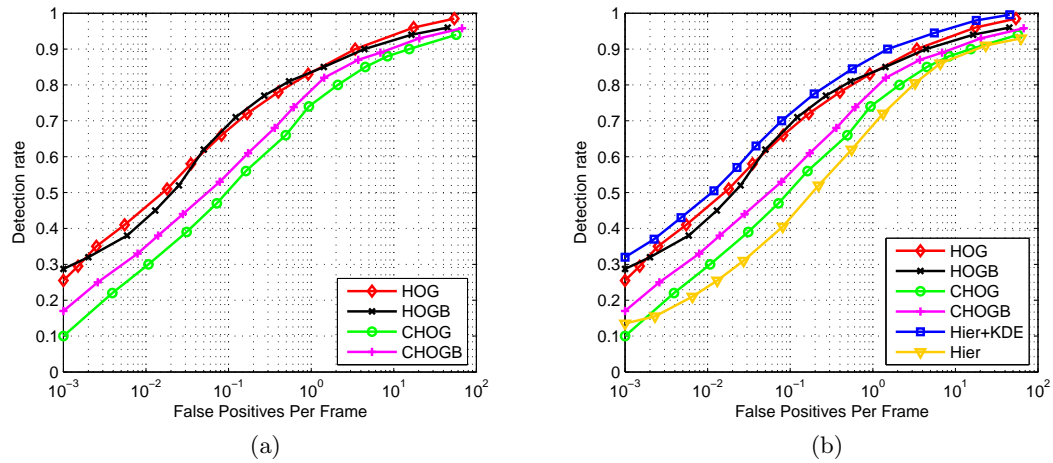


Figure 5.6: Evaluation of different pedestrian detection methods.

approaches.

### 5.3.3 Speed Performance

Table 5.1 shows the speed performance of the four approaches. We see that the CHOG approach is the most computationally efficient. It performs 3 times and 4 times faster compared to the original HOG and the HOGB approaches respectively. Central to the efficiency is that the codebook approach allows quick search of peak responses on the whole image rather than detection with scanning windows.

	HOG	HOGB	CHOG	CHOG
Overall speed ( <i>in secs</i> )	45	60	14	19

Table 5.1: Speed performance for the 4 approaches.

## 5.4 Chapter Summary

In this chapter we present a comparative evaluation of four algorithms for pedestrian detection and the implementation detail of each approach. We study the influence of various parameters and detail their performance on each application. Results show that the HOG based detector achieves the highest performance, the HOGB approach has a comparable performance to the HOG, and the codebook approaches allow computationally efficient

detection. Finally, the detection performance improves by incorporating the background subtraction approach, as it eliminates high number of false positive candidates.

---

## CHAPTER 6

# AUTOMATIC VIDEO SCENE CHANGE DETECTION

---

In the previous chapter we presented pedestrian detection algorithms for mobile bus surveillance. In this chapter we aim to generalise the detection method to automatically detect scene changes in video obtained from a mobile platform. Unlike the pedestrian detection method, here the *foreground* refers to any short term changes, such as objects that appear randomly in the scene (cars and pedestrians), or stable aspects of background being removed, for example a house being renovated or trees being removed. This task is challenging for the following reasons: (a) the footage is collected at different times (different lighting conditions), (b) the camera is mobile (different camera poses), and (c) the foreground covers various types of object classes (people, cars, buildings etc). In this chapter we present a novel scene change detection algorithm that integrates sparse 3D scene background modelling and dense 2D image background modelling into a unified framework. The process consists of two major steps: data alignment followed by a change detection process.

The first step involves transforming the input video that are time-ordered image sequences into: (a) time-ordered 3D scene data, (b) motion-field data, and (c) 2D aligned image data (view sites), as shown in Figures 6.1 (c), (d), and (e) respectively. In these representations, images and features from different video are geometrically aligned together and hence can be used for background modelling purposes.

The second step involves detecting changes on this data. Our scene change detection is motivated by the following observations:

1. Spatial consistency. Background features should be consistent over time. This is measured by the frequency of feature occurrence across all video. Intuitively, areas with random feature occurrence generally translate to foregrounds, like passengers and moving cars because they are “moveable” objects and they do not appear at

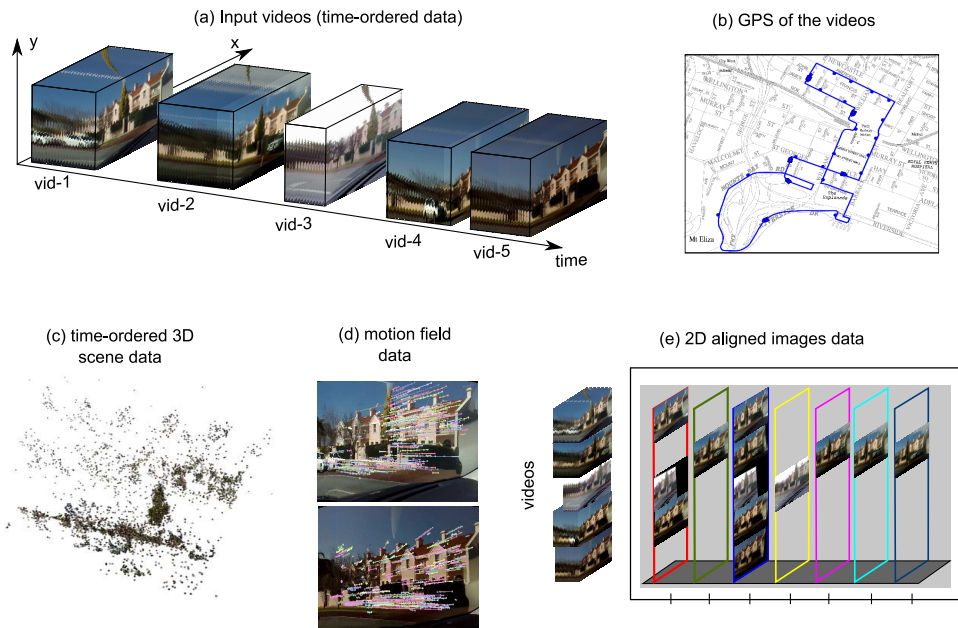


Figure 6.1: Illustration of the aligned data. (a) Input video that are time-ordered data; (b) GPS of the video; (c) time-ordered 3D reconstructed scene data; (d) motion field feature data; and (e) 2D aligned image data.

the same location over time. On the other hand, buildings, vegetation, and sky are usually background areas, with high feature spatial consistency.

2. Speed. Non-stationary objects, for example moving vehicles or pedestrians, have a higher probability of being foreground as compared to stationary objects, for example buildings, vegetation, and sky.
3. Appearance consistency. Background areas should have consistent structural (gradient) appearance over time. Areas with high appearance consistency normally belong to background objects (e.g. houses, sky, etc), while low appearance consistency areas imply changes in background.

## 6.1 Overview of The Algorithm

Our approach is illustrated in Figure 6.2. The 3D scene of each video is reconstructed using the Structure-From-Motion method (Snavely *et al.*, 2008). This provides sparse three dimensional points of the scene relative to the camera motion. The Iterative Closest Point method (Besl and Mckay, 1992) is then applied to align multiple 3D scenes, forming the time-ordered 3D scene data. We formulate 3D scene change detection in a clustering

framework (Ertöz *et al.*, 2003), finding neighbourhood point correspondences or spatial consistency in the time-ordered 3D scene data. Intuitively, points that have high neighbourhood correspondence belong to background objects because it implies that the objects do not move, whilst points that appear and disappear have a higher likelihood of being foreground objects.

Stable features are tracked over consecutively aligned images to form the motion-field data. Since moving objects have different velocities compared to static objects we are able to extract different class features according to their relative speed.

Using the recovered camera geometry, images from different video can be efficiently aligned to a fixed coordinate system to form a “sparse image matrix” (view-sites), where each row consists of image sequences from a single video and each column consists of images with similar scene views. Note that the column entries are generally sparse because not all 2D images from different video cycles can be aligned together. To model the background, we use the HOG codebook modelling approach (Dalal and Triggs, 2005) due to its robustness against illumination changes. In addition, we incorporate a Markov Random Field spatio-temporal framework (Xu *et al.*, 2005) to obtain consistent image-video segmentation.

Finally, we combine all outputs using morphological operations to obtain the final scene change detection results. This method is detailed in the following sections.

## 6.2 3D Scene Change Detection

The 3D scene change detection involves constructing the time-ordered 3D scene data, determining the feature occurrence consistency and finding the final foreground clusters from the time-ordered 3D scene data.

### 6.2.1 Time-ordered 3D Scene Construction

The time-ordered 3D scene data is constructed from a set of video. The process involves iteratively aligning the 3D Structure-From-Motion (SFM) of a video to the existing time-ordered 3D scene data using Iterative Closest Point (ICP).

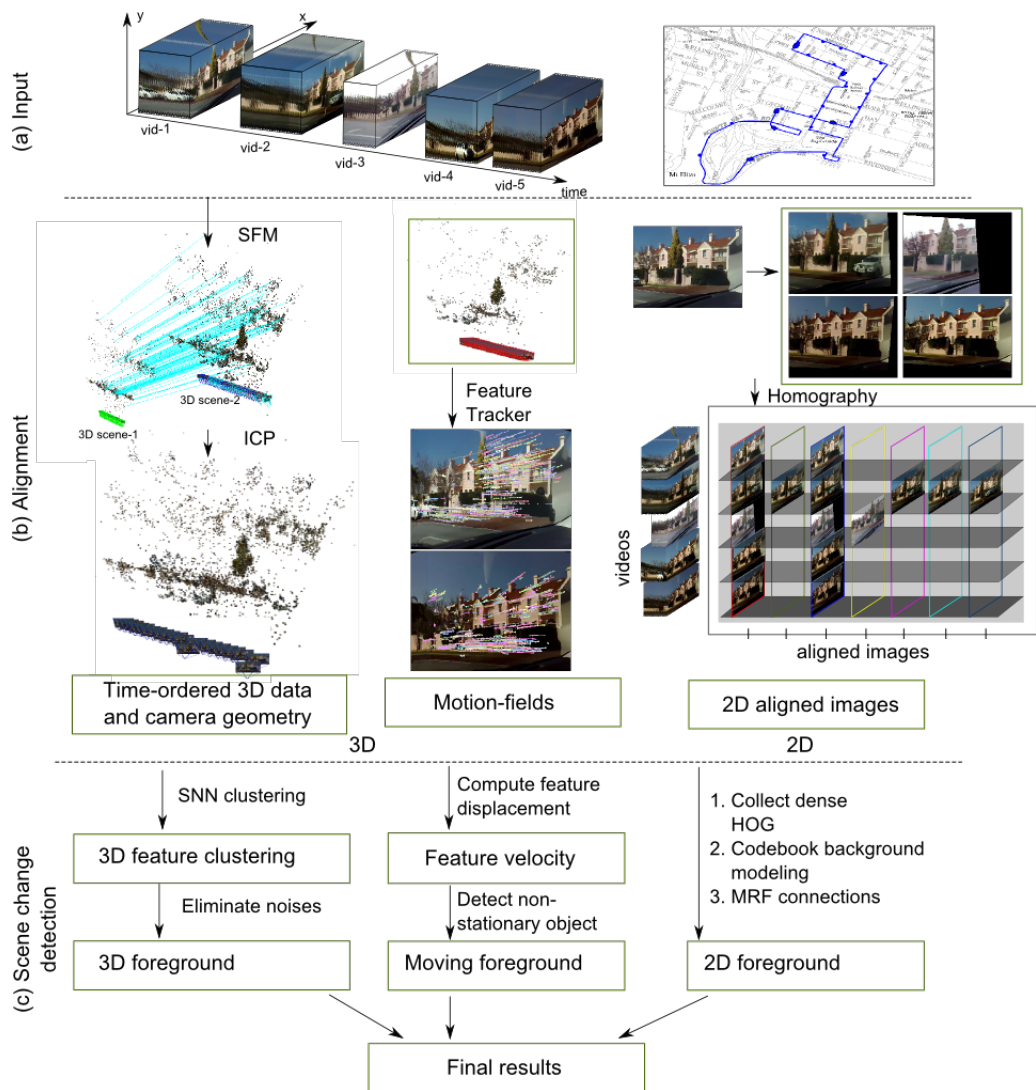


Figure 6.2: Overview of our proposed scene change detection framework.



### 6.2.1.1 Structure From Motion (SFM)

Structure From Motion (Snavely *et al.*, 2008) is a method to recover the camera geometry and three-dimensional structure of a scene by analysing motion of images over time. To do this, stable feature points, like SIFT (Lowe, 2004) or SURF (Bay *et al.*, 2006), are extracted and tracked over a number of images and the three-dimensional structure information is obtained by minimising the sum of the squared projection error using bundle adjustment as:

$$\min_{T_j, X_i} \sum_{i=1}^n \sum_{j=1}^m e_{ij}^2 \quad \text{with } e_{ij} = |\mathcal{P}(T_j, X_i) - x_{ij}| \quad (6.1)$$

where  $X_i$  and  $x_{ij}$  denote the 3D and 2D coordinates of feature  $i$  in image  $j$  respectively. Function  $\mathcal{P}$  is the projection of 3D object points to image coordinates and  $T_j$  denotes the transformation of the camera coordinate system of image  $j$  to an arbitrary world coordinate system. This error function can be minimised using non-linear least-square optimisation (Horn, 1986) or linear approaches based on projective geometry (Quan and Lan, 1999). According to Horn (1999), perspective methods should be favoured over methods based on projective geometry as they produce more accurate results.

The results of SFM are sparse 3D cloud points relative to the estimated camera geometry (position and orientation). Next, we use the ICP method to align multiple SFM data to form the time-ordered 3D scene data.

### 6.2.1.2 Iterative Closest Point (ICP)

Instead of recomputing the time-ordered 3D scene each time a new video is introduced, we employ the Iterative Closest Point approach (Besl and Mckay, 1992) to merge the new SFM scene to the existing time-ordered 3D scene data. Iterative Closest Point (Besl and Mckay, 1992) is a method to fit points in a target model to points in a source model. The final goal of the algorithm is to find the 3D translation  $t$  vector and rotation  $R$  matrix that minimizes the sum of square errors with respect to the source points and their corresponding target points:

$$E(R, t) = \frac{1}{N} \sum_{i=1}^N \|X_i - RY_i - t\|^2 \quad (6.2)$$

where  $X_i$  and  $Y_i$  are corresponding points. The algorithm follows 4 major steps:

1. Correlate points using feature correspondence (or associate points by the nearest neighbour criteria)
2. Estimate the parameters using a mean square cost function as in Equation 6.2.
3. Transform the points using the estimated parameters.
4. Iterate until the error is below a certain threshold.

Note that the algorithm above is similar to the Homography method discussed in Chapter 3 (Section 3.1) and the difference between the two is that  $t$  and  $R$  are in 3D for the ICP method, instead of 2D as in the Homography problem. Since each point in the SFM not only encodes the location, but also the SIFT descriptors, the point correspondence between SFM can be done simply by computing the matches between these descriptors. Table 6.1 shows an example of the 3D alignment process of multiple SFMs using the ICP algorithm.

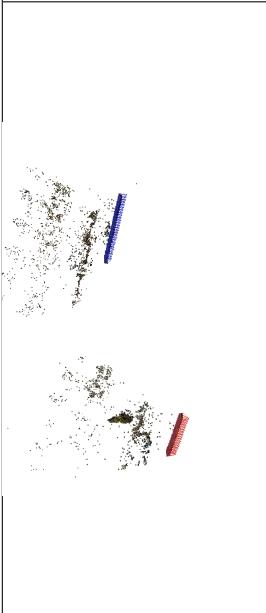
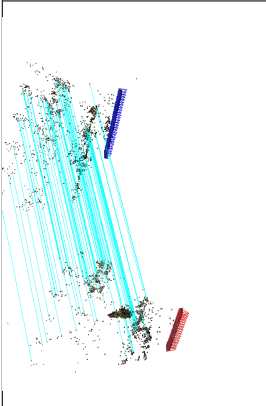
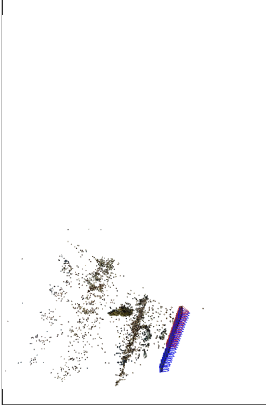
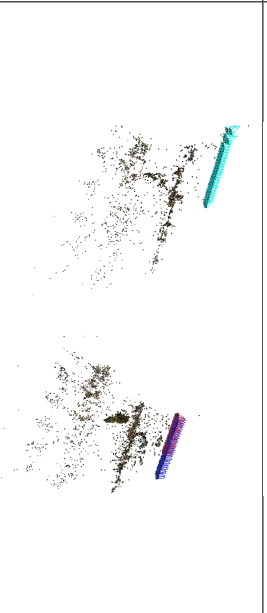
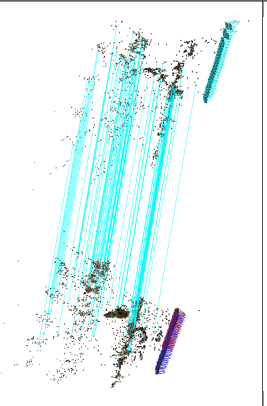

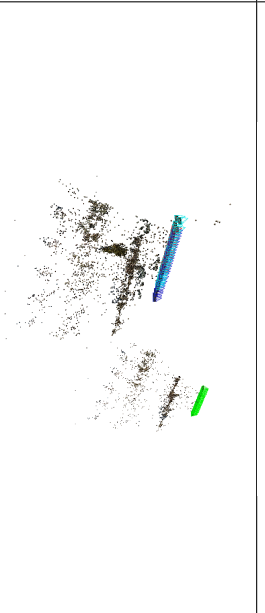
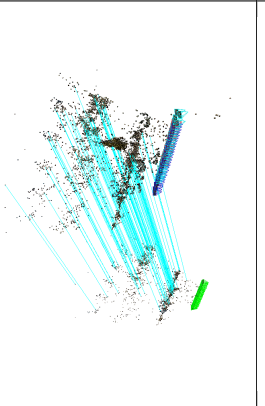
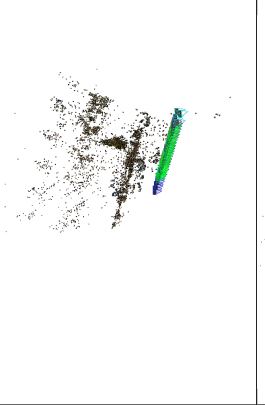
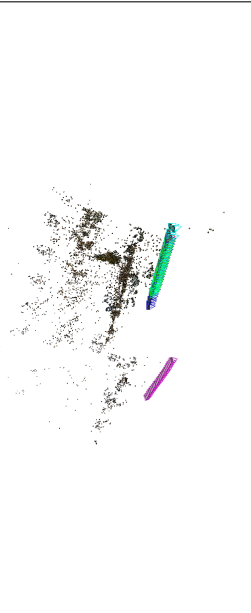
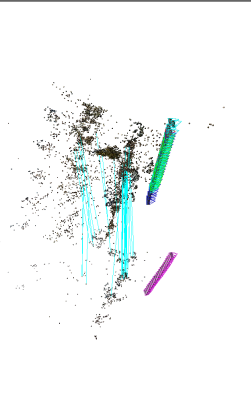
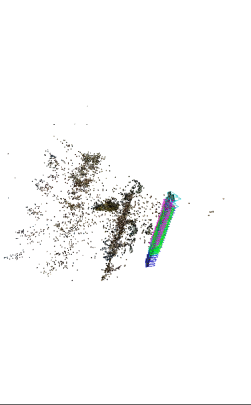
Existing time-ordered 3D scene & a new 3D scene	SIFT matching + ICP	Time-ordered 3D scene
		
		
		
		

Table 6.1: Alignment of multiple 3D scenes using the Iterative Closest Point method.

### 6.2.2 3D Cloud Point Subtraction

Once we obtain the time-ordered 3D scene data, the task is to identify different spatial patterns from a target scene to all other scenes. Intuitively, a feature is considered as “background” if it appears frequently throughout the entire video, otherwise it is considered as “foreground”. This problem can be viewed as a clustering problem of finding neighbourhood point correspondences. In this work, we employ a similar idea as the shared nearest neighbour (SNN) clustering algorithm (Ertöz *et al.*, 2003). Given two points (a source and a target point) from two different scenes, we consider them as the same point if they satisfy the following three conditions: first, the corresponding target point has to be one of  $K$ -Nearest Neighbours from the nearest neighbours of the source point and vice versa. Secondly, their distance must lie within a distance threshold  $Eps$ . Finally, both points have to share minimal of  $MinPts$  to the similar nearest neighbours. The last condition enforces a tighter connection between the two points. Following this process, each 3D point now has information about its neighbours, in other words, its occurrence frequency. Let  $L(X_i)$  be the frequency of a feature  $X_i$  over  $M$  number of video, where  $L(X_i) \leq M$ . By choosing a threshold  $th_L$  on  $L(X_i)$ , we are able to pick objects with different lifespan. If  $th_L$  is small, we are looking for short-duration foreground changes such as vehicles or pedestrians. As  $th_L$  increases, we obtain more background objects (high number of occurrences). Figure 6.3 shows an example of the 3D cloud point subtraction of two scenes based on their spatial point similarity. As the 3D features contain depth information, we obtain different foreground results depending on the choice of the target and source.

### 6.2.3 Spatial Clustering to Identify Foreground Objects and Eliminate Noise

Following the subtraction process, we obtain clusters that contain both noise and foreground points (see Figure 6.3). To eliminate the noise, we apply the SNN clustering algorithm (Ertöz *et al.*, 2003) on the remaining 3D points. Each point in a cluster is classified as one of the following classes: *core*, *border*, or *noise*. We define a point as

1. *Core* when its density is greater than  $Eps$ , where density of a point is computed by counting how many of its  $K$  nearest neighbours shares at least the same  $M$  number of nearest neighbours.
2. *Noise* for all non-core points which are outside the radius of  $Eps$  of a core point.
3. *Border* are the remaining points not classified as *core* or *noise*.

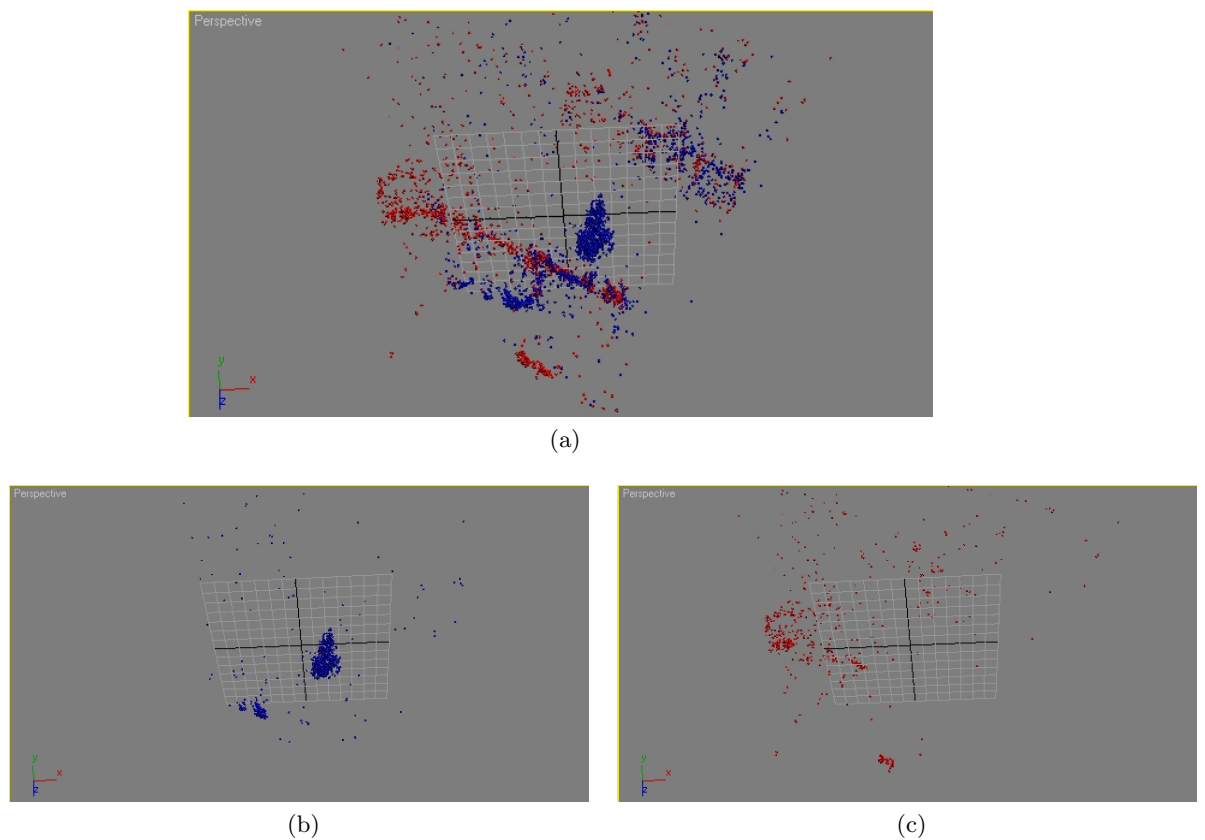


Figure 6.3: 3D cloud point subtraction based on the spatial point similarity. (a) The blue and red scenes are created from two video data taken at different times. (b) The result of the 3D cloud point subtraction for the blue scene. (c) The result of the 3D cloud point subtraction for the red scene.

In other words, a *core* point should satisfy two conditions: (1) it must have a high number of points that share the same nearest neighbours and (2) between the point and its neighbours, they must also share a high number of similar neighbours. *Border* points are points that share just enough nearest neighbours with *core* points. *Noise* points are all non *core* points where the number of shared nearest neighbours is less than the *Eps* (distance threshold).

The results of this process are clusters with different shapes and densities. In this work, we assume that the sparse clusters are noise. To eliminate noise clusters, we compute the mean and standard deviations of each cluster. We then eliminate clusters that are three standard deviations from the mean. Figures 6.4 (a), (b), and (c) show results of the SNN clustering algorithm, where the red points are noise and the rest are considered foreground clusters. In Figure 6.4 (b), we see that the SNN clustering algorithm is able to separate different density clusters based on their spatial characteristics. The final 3D scene change detection results are shown in Figure 6.4 (d).

The SNN clustering algorithm is suitable for our problem for two reasons. First, because our foreground objects include different types of objects such as pedestrians, people on bicycles, cars, trees, etc, all having different shapes and densities, and SNN being a density based algorithm can handle this problem. Secondly the SNN algorithm is more flexible since it only requires prior knowledge about the number of shared nearest neighbours, as compared to the *K*-means clustering algorithm (Hartigan and Wong, 1979) that requires the number of clusters to be known before hand or DBScan (Ester *et al.*, 1996) that requires explicit knowledge about the distance threshold.

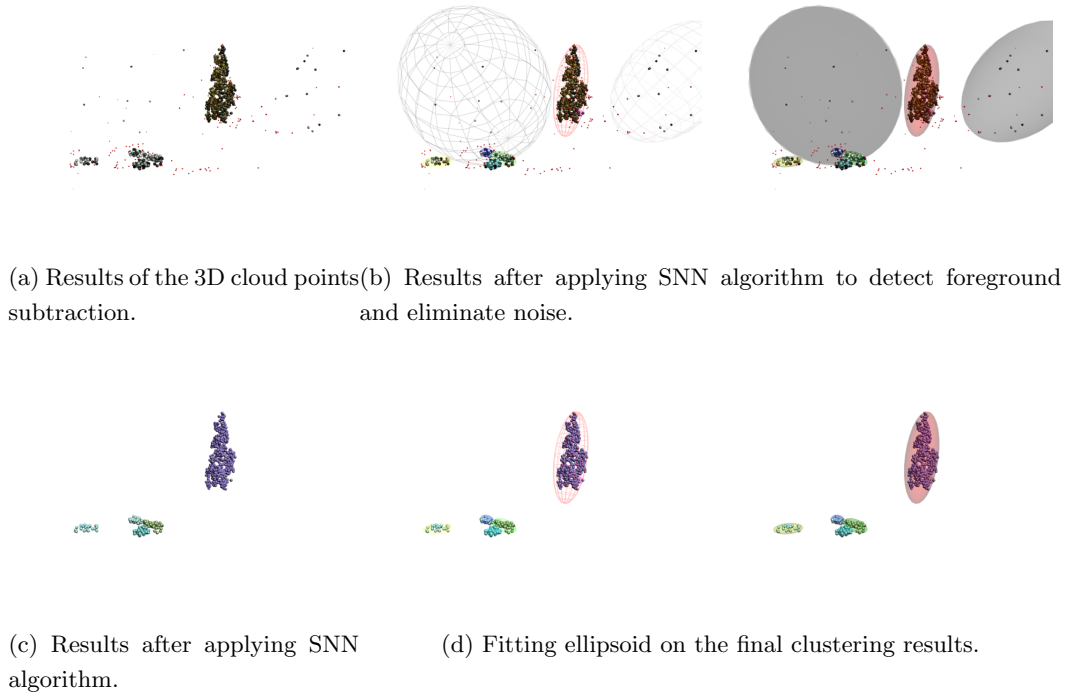


Figure 6.4: SNN clustering algorithm to detect foreground and eliminate noise.

### 6.3 Stationary/Non-stationary Object Probability

In consecutive images, an object can be categorised as either stationary or non-stationary based on velocity. While stationary objects can either be background or foreground, non-stationary objects generally belong to the foreground. Examples of stationary objects include buildings and vegetation, which are normally background, and cars parked on the side of the road, which are foreground. On the other hand, moving objects like walking pedestrians and moving vehicles always belong to foreground. Based on this, we assume that the non-stationary features belong to foreground objects.

To estimate the object velocity, we need to track and compute the feature displacement over the consecutive images. In a static camera, the feature displacement is simply the spatial pixel differences of the same feature over two consecutive frames. The problem becomes more complicated when the camera itself is moving, as in our case. We also need to estimate the camera displacement over consecutive sequences. Intuitively, since we have recovered the camera geometry from SFM, we can compute the relative distance from one camera to the other (see Figure 6.6), which is equivalent to stitching the consecutive image

sequences using the view frustum concept (Assarsson and Möller, 2000).

In computer graphics, a view frustum (Assarsson and Möller, 2000) is the field-of-view of a camera or a region of space that appears on the screen, as shown in Figure 6.5. Our proposed solution is to project all images to the *far plane* of the camera frustum. At the end of the projection process we flatten all the far planes into a single plane. In this plane all consecutive images are roughly stitched together (see Figure 6.6). The average velocity of a feature can then be computed as the relative distance between the feature displacement in the motion-field image. Note that although the images do not stitch perfectly this strategy allows us to estimate the relative feature velocity, which is enough for our problem. Figure 6.6 shows examples of the motion-field images. The larger the average velocity of a feature, the higher the probability that it is a foreground. Feature with average velocity that is greater than two standard deviations of the entire motion fields are considered as foreground points. Figure 6.7 shows motion-fields of stationary and moving objects.

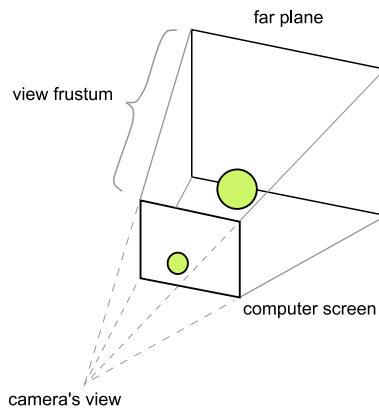


Figure 6.5: View frustum points.

We then combine the motion-field and the SNN clustering results to form the final 3D scene change detection results. The 3D result is generally too sparse to provide full segmentation of the objects. Next, we present a Markov Random Field (MRF) framework based on robust feature background modelling to enhance the video-image segmentation.



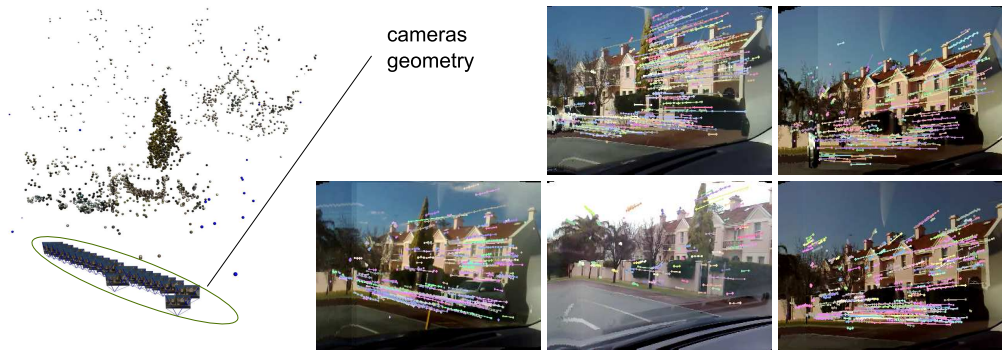


Figure 6.6: Motion-field data: consecutive images are stitched together using the camera geometry information. The feature velocity is then represented by the length of the lines in the motion-field images.



Figure 6.7: Example of two scenes with objects of different velocity. (a) A moving car has longer motion fields compared to other objects in the scene such as houses and trees. (b) Static objects normally have similar average velocity.

## 6.4 2D Change Detection

### 6.4.1 2D Data Alignment

Similar to traditional methods for foreground-background modelling, it is required for images to be aligned before the background model can be created. In Chapter 4 (Section 4.1) we scanned a large number of image sequences to find the best image alignment, which is computationally costly. In this work, since we have recovered the camera location and orientation from SFM, the best Homography matrix for each image can be searched efficiently in the order of its nearest neighbour cameras. This is motivated by the fact that the best alignment target image has no transformation changes (in terms of skew, rotation, scale, and translation), and thus cameras that are closest to the source camera will have a higher probability of being aligned compared to those that are far from the camera. Once the ordering is determined we employ the same Homography method as described in Chapter 4 (Section 4.1) to align images from different video.

### 6.4.2 Spatio-Temporal MRFs for Scene Change Detection

Since our problem of detecting scene changes is an image-labelling problem it can be viewed as a problem of inferring the Maximum A Posterior (MAP) solution of a MRF model. We employ a multi-resolution MRF background modelling approach similar to that in Xu *et al.* (2005) to model the spatio-temporal connections between images in the video. The multi-resolution model is achieved using the image pyramid. Let  $I_t^l$  denote the image at pyramid level  $l$  at time  $t$ ,  $\mathbf{X} = \{X_1, \dots, X_N\}$  be the hidden state, where each pixel state takes a class label of either foreground or background  $X_i \in \{F, B\}$ ,  $\mathbf{Z} = \{Z_1, \dots, Z_N\}$  be the pixel observation, and  $\Omega$  be the Markovian model.

According to this model, the optimal labelling can be written as:

$$\mathbf{X}_{\text{MAP}} = \arg \max_{\mathbf{X}} P(\mathbf{X}|\mathbf{Z}, \Omega) \quad (6.3)$$

which is equivalent to an optimization problem of minimizing the following energy function:

$$E(X) = \sum_{i \in \mathcal{V}} D(X_i) + \sum_k \sum_{(i,j) \in \mathcal{N}_k} C_k(i, j) \quad (6.4)$$

where  $\mathcal{V} = \{1, 2, \dots, N\}$  is the pixel index set,  $\mathcal{N}_k$  are the different edge sets connecting the pixels,  $D(\cdot)$  and  $C(\cdot)$  are the unary (association) and pair wise (interaction) potential functions respectively, and  $k \in \{1, 2, 3\}$  refers to three different spatio-temporal MRF interaction models. The three interaction models are defined as follows:

- $C_1$  = *Connection between two adjacent pixels at the same pyramid level.* This is designed to achieve coherence labelling among neighbourhood pixels, thus obtaining smooth foreground areas.
- $C_2$  = *Connection between corresponding pixels at two adjacent pyramid levels.* The motivation of using the pyramid image structure is that at the higher image resolution (the smaller the size of the image) we obtain less noise. By introducing the pixel relationship across pyramid levels, better background models can be achieved.
- $C_3$  = *Connection between two corresponding pixels at consecutive time instances.* The temporal connection captures the coherence labelling among pixels for different times. In this work, the temporal neighbours are obtained by feature correspondences between frames.

#### 6.4.2.1 MRF Model

The  $C_1$  connection defines the spatial relationship between a pixel and its neighbours at the same pyramid level. This is motivated by the assumption that foreground areas are continuous and smooth.  $C_1(i, j) = 0$  if  $i$  and  $j$  are not neighbouring pixels, otherwise it is defined as:

$$C_1(i, j) = \begin{cases} -\lambda_1 & \text{if } X_i = X_j \\ \lambda_1 & \text{if } X_i \neq X_j \end{cases} \quad (i \text{ and } j \text{ are adjacent pixels at the same image pyramid level})$$

where  $\lambda_1$  is the model parameter controlling the smoothness in the image region.  $C_1(i, j)$  has positive values if there is a discontinuity in the pixel label and 0 otherwise. Figure 6.8 shows examples of the  $C_1$  and  $C_2$  connections.

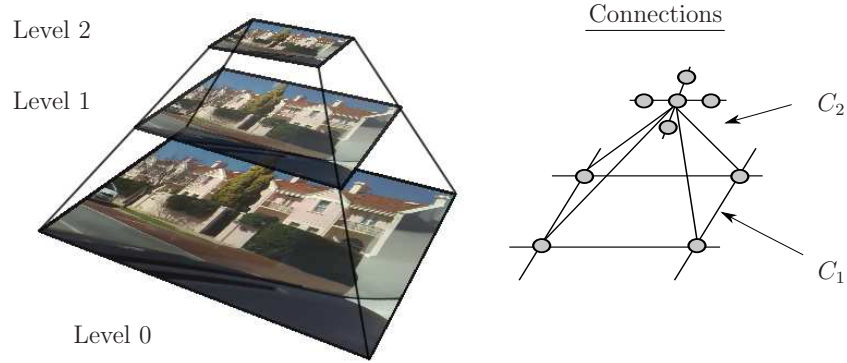


Figure 6.8: Spatial connections.

The  $C_2$  connection defines the relationship between corresponding pixels at two adjacent pyramid levels. The motivation is that the class labels should agree since they belong to the same pixel in the pyramid scheme.  $C_2(i, j) = 0$  if  $i$  and  $j$  are not corresponding pixels at two adjacent pyramid levels, otherwise it is defined as:

$$C_2(i, j) = \begin{cases} -\lambda_2 & \text{if } X_i = X_j \\ \lambda_2 & \text{if } X_i \neq X_j \end{cases} \quad (j \text{ is the corresponding pixel of } i \text{ at the adjacent pyramid level})$$

where  $\lambda_2$  is the model parameter controlling the label consistency between pyramid levels.  $C_2(i, j)$  has positive values if the pixel labels between two adjacent pyramid levels are inconsistent.

The  $C_3$  connection defines the temporal relationship between corresponding features (pixels).  $C_3(i, j) = 0$  if  $i$  and  $j$  are not corresponding SIFT pixels between two consecutive times, otherwise if the corresponding temporal features are considered as *stationary* features in the motion-field data, it is defined as:

$$C_3(i, j) = \begin{cases} -\lambda_3 & \text{if } X_i = X_j \\ \lambda_3 & \text{if } X_i \neq X_j \end{cases} \quad (j \text{ is the corresponding feature of } i \text{ in the previous frame})$$

and if the corresponding temporal features are considered as *moving* features in the motion-field data, it is defined as:

$$C_3(i, j) = \begin{cases} -\lambda_3 & \text{if } X_i = X_j = F \\ \lambda_3 & \text{if } X_i = X_j = B \text{ (} j \text{ is the corresponding feature of } i \text{ in the previous frame)} \\ \lambda_3 & \text{if } X_i \neq X_j \end{cases}$$

where  $\lambda_3$  is the model parameter controlling the label consistency between two corresponding features at two consecutive time instances.  $C_3(i, j)$  has positive values if the pixel labels between robust features do not agree and 0 otherwise. The second condition for the  $C_3$  connection is to ensure that we obtain foreground pixel labels since the motion label categorises it as moving features. Pixels with no temporal SIFT correspondence simply have  $C_3 = 0$ . Since  $C_3$  connection exists for the neighbourhood pixels between two consecutive frames, the MRF model is still preserved. Figure 6.9 shows the temporal connection.

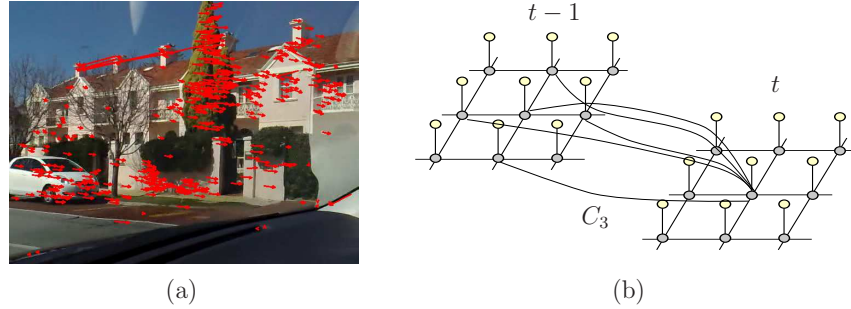


Figure 6.9: Temporal connection. (a) SIFT flow (b) Pixels that have SIFT correspondence between consecutive images are assigned MRF temporal connection.

#### 6.4.2.2 Likelihood

The observation model is expressed as:

$$p(Z_i|X_i, \theta) = p(Z_i|X_i, \theta_i) \quad (6.5)$$

where  $\theta_i$  is the background model for pixel  $i$ . Next we present the MRF algorithm to estimate the posterior probability and the details of the background model are discussed in Section 6.4.4.

### 6.4.3 Estimating the Posterior Probability

Once the prior and the likelihood models are defined, the posterior probability for the proposed MRF configuration in Equation 6.3 can be written as:

$$\begin{aligned} p(\mathbf{X}|\mathbf{Z}, \Omega) &\propto p(\mathbf{X}|\Omega) \times p(\mathbf{Z}|\mathbf{X}, \Omega) \\ &= \prod_{i,j} \exp(-C(i, j)) \times \prod_i P(Z_i|X_i, \theta_i) \end{aligned} \quad (6.6)$$

where  $\mathbf{X}$  is the hidden state,  $\mathbf{Z}$  is the observation,  $(i, j)$  is any pair of pixels in the model, the Markov connection  $C(i, j)$  is defined as:

$$C(i, j) = C_1(i, j) + C_2(i, j) + C_3(i, j) \quad (6.7)$$

and  $\Omega$  are the parameters of the Markov Network:

$$\Omega = \{R_1, R_2, R_3, \theta\}$$

where  $R_1$ ,  $R_2$ , and  $R_3$  are the set of parameters  $\lambda_1$ ,  $\lambda_2$ , and  $\lambda_3$  at different pyramid levels.  $\theta$  is the background model.

Similar to Xu *et al.* (2005), we employ Gibbs sampling with simulated annealing to estimate the posterior probability of the proposed Markov Random Field (MRF). Algorithm 3 shows a summary of the Gibbs sampling algorithm used in the scene change detection.

Sampling at high temperatures reduces the difference among peaks in the probability distribution and helps prevent the local maxima. At low temperatures, the mode of the posterior is amplified and this causes the algorithm to converge faster (Xu *et al.*, 2005).

---

**Algorithm 3:** Gibbs sampling with simulated annealing algorithm for image segmentation.

---

1. Initialise the start configuration  $X^{(0)} = (X_1^{(0)}, \dots, X_d^{(0)})$  using the MAP state estimation of previous frame, with iteration  $k = 0$  and temperature  $T = T(0)$  in simulated annealing, where  $d$  is the number of nodes in the MRF.
  2. At iteration  $k$ ,
    - (a) For each node  $X_i^k$  in the MRF
      - i. Select a new state  $X_i^+$  and let the proposed MRF configuration be  $\mathbf{X}^+ = (X_1^k, \dots, X_{i-1}^k, X_i^+, X_{i+1}^k, \dots, X_d^k)$
      - ii. Let the old state be denoted as  $X_i^-$  and its MRF configuration be  $\mathbf{X}^- = (X_1^k, \dots, X_{i-1}^k, X_i^-, X_{i+1}^k, \dots, X_d^k)$ .
      - iii. Compute the acceptance ratio as follows:
$$\alpha = \min \left\{ 1, \left( \frac{p(\mathbf{X}^+ | Z, \Omega)}{p(\mathbf{X}^- | Z, \Omega)} \right) \right\}$$
where  $\alpha$  is the acceptance ratio choosing the best between the new and old state's conditional posterior probability.
      - iv. A new label state  $X_i$  is accepted according to the following rule:
$$X_i^k = \begin{cases} X_i^+ & \text{if } \alpha \geq 1 \\ X_i^- & \text{otherwise} \end{cases}$$
  3. Decrease the temperature  $T = T(k + 1)$
  4. If  $T > 0$ , go to step 2
-

#### 6.4.4 Observation Model ( $\theta$ )

The observation model is one of the most important elements that significantly influences the segmentation results. Therefore, it should reflect to the characteristics of the dataset. Due to different lighting conditions in most of the dataset, we employ a HOG codebook based background modelling. We present each pixel of the background using the Histogram of Oriented Gradient, built from a window region centered around the pixel. The codebook for each pixel is constructed by grouping similar background HOG features to form a set of codewords. The weight of each codeword is defined by the number of the feature occurrences. The advantage of using the codebook based approach is because it preserves the original structure of the histograms (a distinctive property of the feature), creating more robust background models as compared to the KDE-HOG approach (Elgammal *et al.*, 2000) that treats each channel as independent attributes. Before constructing the codebook background model, we perform a pre-processing step to discard uniform regions.

##### 6.4.4.1 Discarding the Uniform Regions

The scene data consists of wide ranges of objects such as sky, houses, cars, road, vegetation etc. We choose to discard processing uniform areas (areas having weak gradients) for two reasons: (1) because the data is taken at different days and times, we generally encounter different sky conditions which are typically noise (see Figure 6.10); (2) areas with weak gradient structures do not give strong effects in the HOG formulation. To discard the uniform regions, we convert the normal edge image into the sum of HOG image followed by normalisation. Because we normalise the image based on its global gradient, we put higher weights on strong gradient areas. For areas with a gradient less than a threshold, the observation likelihood is equal to 0. Figure 6.10 shows an example of the results after eliminating the noise. Note that by doing this, we lose detailed information of the foreground objects with uniform colour regions. However, since our aim is to roughly locate the objects, this strategy would help eliminating false positives whilst preserving the gradient structure of the scene.

##### 6.4.4.2 HOG Codebook Observation Model

Given a target image, the observation likelihood of each pixel is computed using,



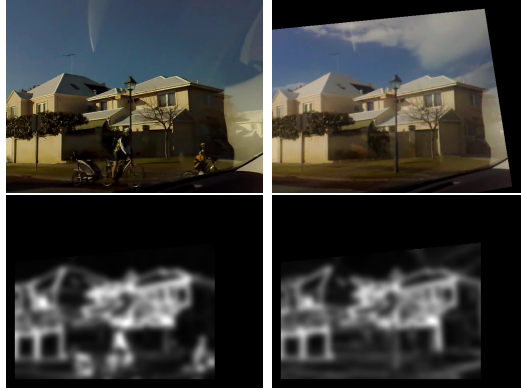


Figure 6.10: (Top) Two frames taken on different days, note that the sky has large differences. (Bottom) Results after noise elimination.

$$p(Z_i|\theta_i) = p(Z_i) = CB(Z_i, \theta_i) \quad (6.8)$$

where  $\theta_i$  be the background model for each pixel and  $CB(\cdot)$  is a function that returns the probability of the current feature matching the best background codeword. The comparison between HOG features and the codebook background features are performed using the Bhattacharyya coefficient. Since there is no guarantee that all the images from different times will align perfectly, we search for maximum responses from around the neighbourhood and redefine the background probability of a pixel as:

$$P_{\mathcal{N}}(Z_i|\theta_i) = \max_{j \in \mathcal{N}(i)} P(Z_i|\theta_j) \quad (6.9)$$

where  $\mathcal{N}$  is the neighbour pixel of  $i$ . Intuitively, this model searches for the closest candidate pixel around a set of neighbours, since we assume that even if misalignment exists, it lies around the neighbours. Although this approach addresses the misalignment problem, it introduces the problem of losing true detections, as mentioned by Elgammal *et al.* (2000). Hence we incorporate a further constraint on the neighbourhood conditions, i.e. if a pixel is truly a background pixel that happens to move to a new location, then some of its neighbourhood pixels should also move, since we assume that the misalignment happens not for one single pixel, but for a group of pixels. Therefore, we define the probability of displacement as:

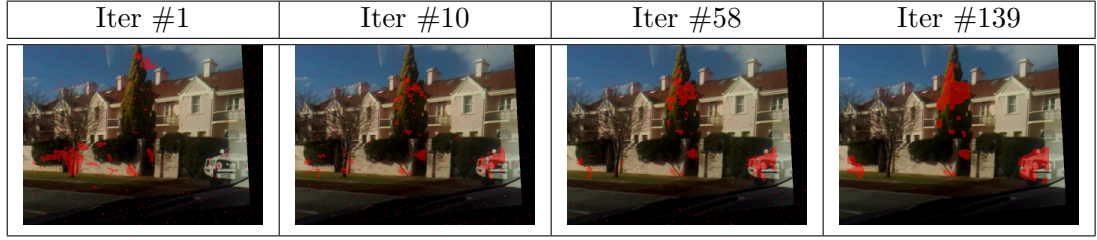


Figure 6.11: Spatio-temporal MRF segmentation results at different iterations. The segmentation results (red colour mask) are overlaid on top of the original images

$$P_C(Z_i) = \prod_{j \in \{i \cup \mathcal{N}(i)\}} P_N(Z_j) \quad (6.10)$$

where  $P_C(Z_i)$  is computed as the product over the connected components of the neighbourhood pixels.

Using Equations 6.7 and 6.10, the conditional posterior probability in Equation 6.6 can be computed. Figure 6.11 shows the MRF scene change results at different iterations before it converges.

## 6.5 Combining Results from 3D and 2D Scene Change Detection

The final scene change detection results are obtained by merging the 3D with 2D results based on their connected components. We name the combined approach as *SFM-HOG* scene detection. In general, although the 3D scene change detection is able to detect the scene changes effectively, the results are often under-segmented due to the sparsity of the features. On the other hand, the 2D scene change detection results include more details of the foreground with some noise. To merge the two results, we first project the 3D scene change detection results back to the 2D images using the view frustum method. Next, we compute the connected components for both 3D and 2D results to identify blob clusters. Finally, we find the overlap regions between the 3D and 2D results. If the number of overlap pixels between the two connected components is larger than a certain threshold, we assign it as a foreground object, otherwise it is assigned as background noise. In other words, this strategy chooses the final foreground results that agree with both the 3D and 2D results.

## 6.6 Experimental Results

### 6.6.1 Scene Changes Dataset

The experimental datasets were collected using a single front-facing camera (Nokia N95<sup>1</sup>) placed on the dash board of a moving car. Figure 6.12 shows the physical camera setup and the GPS locations of the scene where the video was taken. The camera has the following characteristics: it is a camera without pan/tilt/zoom functionalities, it was set up to record a video stream of size 640x480 at 30 frames per second (fps) and was located on the left side dash board looking to the right. During data collection, the vehicle travelled at 40–50 km/hour.

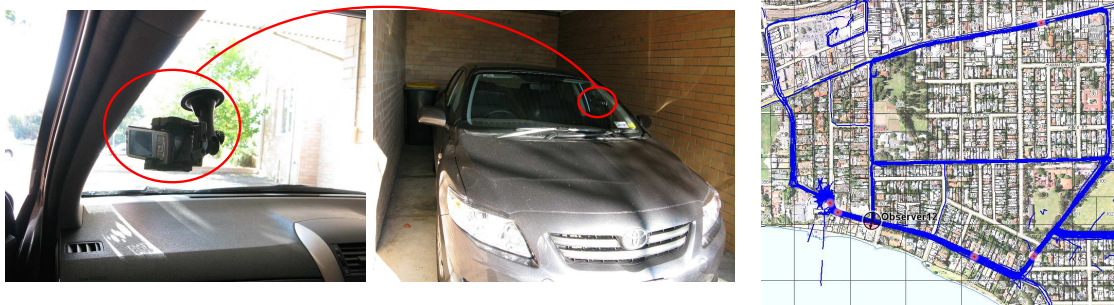


Figure 6.12: Camera setup and map where the video is recorded. The blue lines indicate the path of the vehicle.

The experimental data consists of three different scene datasets. Figures 6.13, 6.14, and 6.15 show the three datasets where all video are taken on different days. Table 6.2 summarises the descriptions of the video.

<sup>1</sup><http://www.nokia.co.uk/find-products/all-phones/nokia-n95/specifications>



Figure 6.13: Dataset 1 consists of two video. (Top) video 1.1. (Bottom) video 1.2.

Dataset	Number of video	Number of frames	Video ID	Frame index
1	2	39	1.1	1–20
			1.2	21–39
2	5	176	2.1	1–36
			2.2	37–54
			2.3	55–85
			2.4	86–112
			2.5	113–176
3	5	136	3.1	1–28
			3.2	29–51
			3.3	52–73
			3.4	74–100
			3.5	101–136

Table 6.2: Descriptions of the dataset acquired from vehicle.

### 6.6.2 Pre-processing

We perform some pre-processing to ignore parts of the video before performing 2D scene change detection due to the following two reasons:

- Physical placement of the camera. Because the camera is situated inside the car some parts of the video contain partial views of the dashboard or reflections of the windscreen (see Figure 6.16 (a)). We manually define an image mask to ignore computing scene change detection on those parts.
- Homography alignment results. Two images from different times generally have regions that do not overlap. However during the detection process we only perform the



Figure 6.14: Dataset 2 consists of five video. From top to bottom: video 2.1 to 2.5.



Figure 6.15: Dataset 3 consists of five video. From top to bottom: video 3.1 to 3.5.

foreground detection on the overlap image regions as shown in Figure 6.16 (b). This information is obtained automatically during the Homography alignment results.

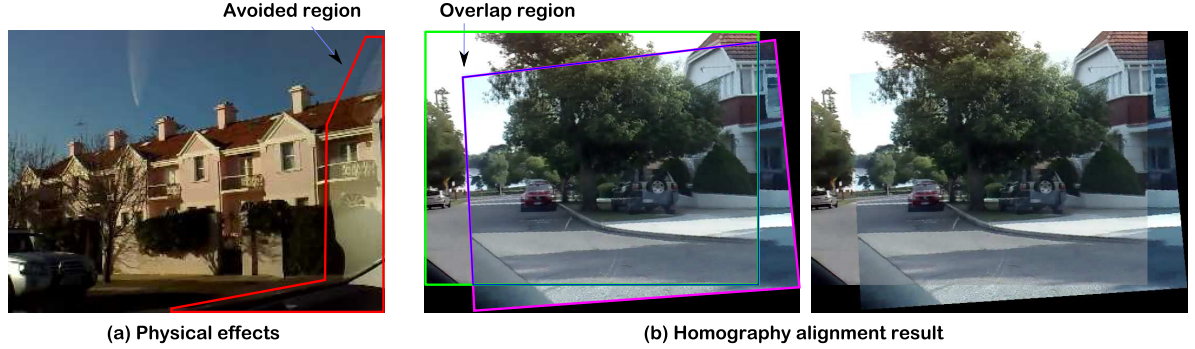


Figure 6.16: Pre-processing examples. (a) The red polygon defines areas to be ignored during 2D scene change detection process due to the effects related to the car; (b) Given a Homography alignment result of two images (the green and the pink colour frames), we detect the scene changes only on the overlap region.

### 6.6.3 Parameter Settings

#### Choice of $K$ , $Eps$ , and $MinPts$

As outlined in the Section 6.2.2 on 3D cloud point subtraction, the size of the neighbourhood,  $K$ , is an important parameter that controls the resolution of the clusters (Ertöz *et al.*, 2003), i.e. the accuracy of the detected objects.  $Eps$  and  $MinPts$  are the threshold density and the parameter that controls the core points respectively, which can be defined as a fraction of  $K$  (Ertöz *et al.*, 2003). Intuitively,  $K$  should reflect the density/size of the object. In our experiment we set  $K = 25$ ,  $Eps = K * 0.4$  and  $MinPts = K * 0.75$  as stated in (Ertöz *et al.*, 2003).

#### Choice of local feature

In this experiment, we study the performance when using different local features for background modelling purposes (this is related to the observation model in Section 6.4.4). The performance is quantified using the true positive and false positive rates. We manually label the image into two different classes: the positive and the negative samples. The positive samples contain patches of same scene under different lighting conditions, while the negative samples are any patches that are from different objects. Figure 6.17 shows examples of the scene patches. Seven different local features were evaluated, including:

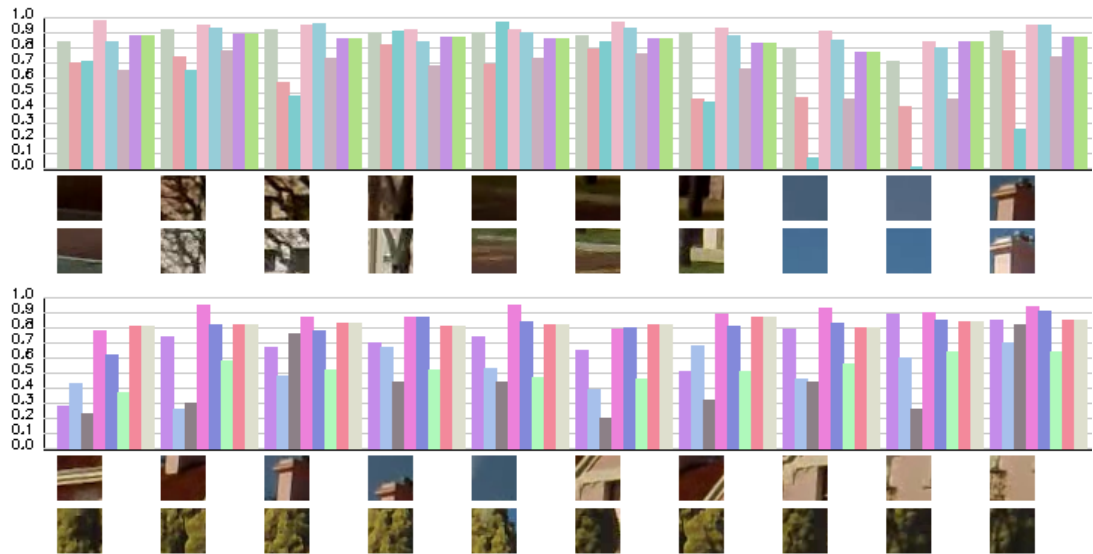


Figure 6.17: Evaluation using different features. Each pair of patches has eight bars representing the different features; from left to right: HOG, HOG Gauss, SIFT, Histogram of Leung Malik filters, Histogram of Gabor filters, PHOG, PHOG Gauss. Bhattacharyya is used as the comparison method: 1 is a perfect match while 0 is the poor match. (Top): comparison of the same objects. An ideal graph should have all bars as high as possible since they are the same object. (Bottom): comparison of different patches. An ideal graph should have all bars as low as possible.

1. Histogram of Oriented Gradient (HOG) (Dalal and Triggs, 2005),
2. HOG with interpolation using Gaussian kernel (Dalal and Triggs, 2005),
3. SIFT (HOG with central axis oriented to the dominant orientations and interpolated using Gaussian kernel) (Lowe, 2004),
4. Histogram of Leung Malik Texton response (Leung and Malik, 2001),
5. Histogram of Gabor Filter response (Jain and Farrokhnia, 1991),
6. Pyramid histogram of oriented gradient (PHOG) (Bosch *et al.*, 2007), and
7. Pyramid HOG with Gaussian interpolation (Dalal and Triggs, 2005)

Another parameter that determines the performance and the complexity is the size of the window. As the window size gets larger, the more neighbourhoods it covers and the computational cost increases.

Figure 6.18 shows the ROC curves for different local features by varying the thresholds and window sizes. From the figure, we can see that the HOG and PHOG features have the best performance compared to other features. The difference between HOG and PHOG

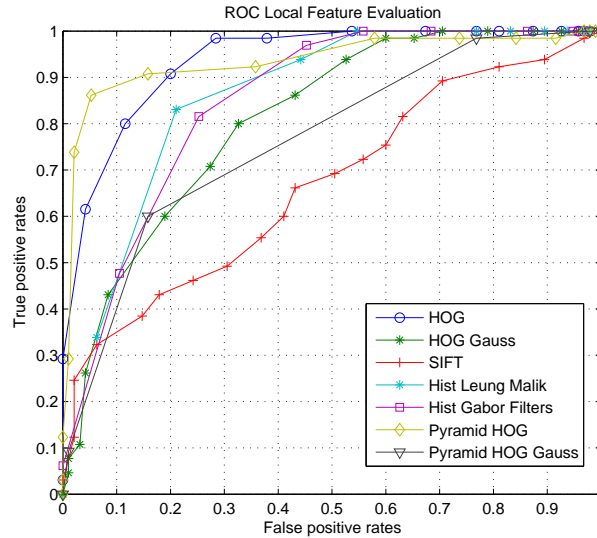


Figure 6.18: ROC curve for different local features.

is that PHOG captures more detailed descriptions of a particular patch or object. The only time PHOG has higher value than HOG is when it compares texture-less features like skies and walls. PHOG does not deal well with image misalignment as it captures more detailed information than HOG. Unlike the foreground detection (Dalal and Triggs, 2005), since we are modelling the background, it is preferable for the model to accept more variations as our data is not always well aligned. In the case of direct foreground modelling, like pedestrian detection, using PHOG should be better than HOG because it compares both the local and global features. The global features may be different but the local features may agree. In our case, if the global feature is different, the local feature would most probably be different thus shrinking the overall confidence score. Based on this evaluation we choose to use HOG as our pixel appearance feature and we set the window size to  $10 \times 10$  pixels wide.

### MRF parameters

Referring to Section 6.4.3 on MRF, we set the model parameters  $\lambda_1 = 0.5$ ,  $\lambda_2 \propto \log(s)$ , and  $\lambda_3 = 0.3$  similar to Xu *et al.* (2005), where  $s$  is the image size.



### 6.6.4 Evaluation Criteria

To evaluate the performance of our proposed framework, we present two quantitative measurements in terms of pixel and bounding box measurements. Both measurements are computed using the  $sensitivity(\%) = \frac{TP}{TP+FN}$  and  $recall = \frac{FP}{FP+TN}$  metrics, where  $TP$ ,  $TN$ ,  $FP$ ,  $FN$  are the true positives, true negatives, false positives, and false negatives respectively. The bounding box measurement provides the performance evaluation on the overall bounding box of the object, which is an adequate metric for our problem due to the different lighting conditions. In addition, we also provide the segmentation evaluation that measures how well the results are in terms of pixel segmentation. For the bounding box measurement we compute the *ratio of the overlap* as shown in Figure 6.19. The ratio of overlap is expressed as:

$$overlap = \frac{\text{area}(\text{detected} \cap \text{groundtruth})}{\text{area}(\text{detected})} \quad (6.11)$$

A detected candidate is considered to be a true positive when the *overlap ratio* between the detected and the ground truth bounding boxes is above 40%, otherwise it is considered as a false positive. To justify the performance of our framework we also compare the proposed SFM-HOG approach against the 2D and 3D scene change detection.

### 6.6.5 Experiments

The main objectives of the following three experiments are to test the:

1. Performance of SFM-HOG against lighting changes using only two video instances (Section 6.6.5.1).
2. Performance of SFM-HOG to detect sudden changes on multiple video instances (Section 6.6.5.2).
3. Performance of SFM-HOG to detect more complex changes on multiple video instances (Section 6.6.5.3).

For each dataset we show the results in the following order: first, we will show the time-ordered 3D scene reconstruction results along with the estimated camera motion to summarise the video sequences. Secondly, the motion field data results are used to summarise

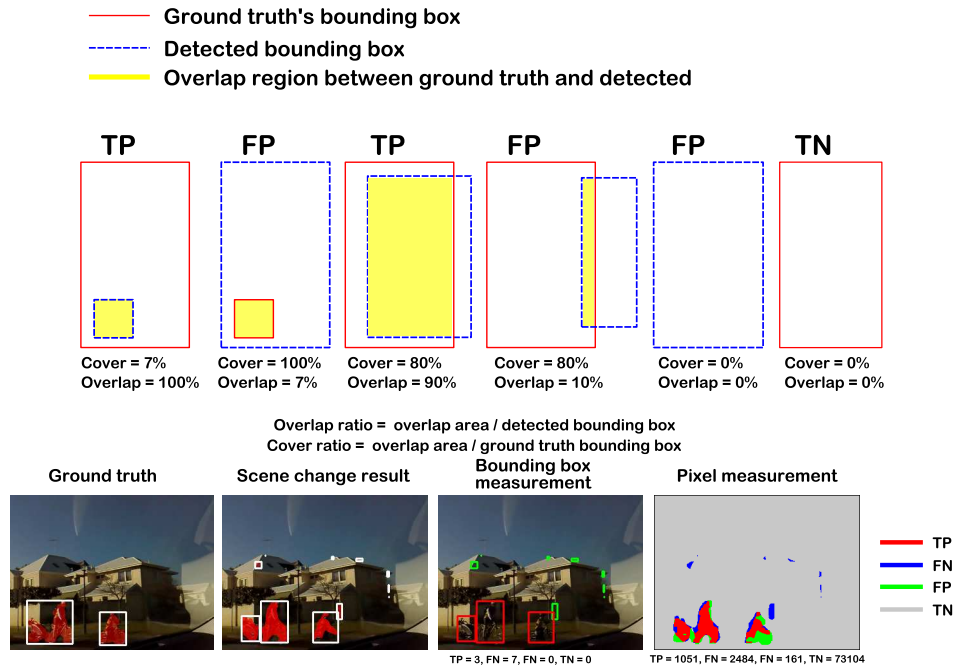


Figure 6.19: Evaluation criteria.

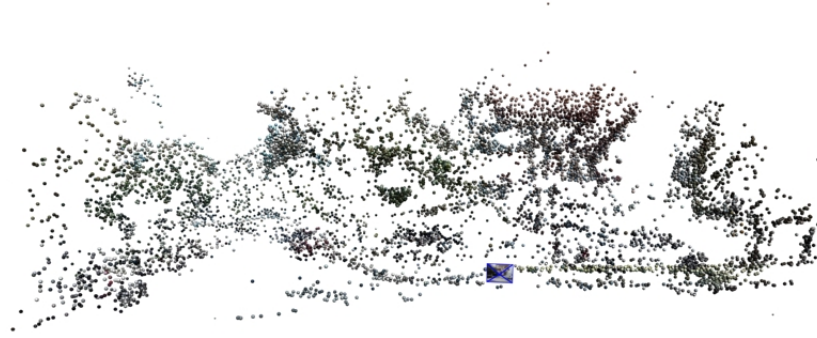
the moving and stationary objects. Thirdly, we will show the 3D scene change detection results for each video. Finally, we compare the 2D, 3D, and SFM-HOG scene change detection results.

### 6.6.5.1 Experiment I: Single Video Changes

This experiment demonstrates the performance of SFM-HOG to detect scene changes against lighting changes using two video instances.

**Description:** Figure 6.20 (a) shows the scene reconstruction of the two video overlaid on each other, whilst Figures 6.20 (b) and (c) show the motion-field data for video 1.1 and video 1.2 respectively. In this experiment, there are three *scene changes* between video 1.1 and video 1.2. First, there is a white pickup car in video 1.1 that does not appear in video 1.2. Similarly, there is a white jeep parked besides the tree in video 1.1 but not in video 1.2. Finally, the red sedan beside the tree in video 1.1 is replaced by another sedan in video 1.2. On the other hand, the *background objects* (*scenes that remain unchanged*) in this dataset include houses, vegetation, and shadows (appear in video 1.2). Since the scene contains stationary foreground and background objects, there are no significant differences in the motion fields data as shown in Figures 6.20 (d) and (c).

**Results:** Figure 6.21 shows some comparative results between 3D, 2D, and SFM-HOG approaches of dataset 1. Each column represents results from different approaches. From Figure 6.21, we can see that the high number of false detections for using either 3D or 2D result alone is greatly reduced when the two approaches are combined. The false detections in Figure 6.21 (a) and (c) are mainly caused by the limited number of video used in this experiment and there is large difference in lighting between these video. The shadow in video 1.2 does not present as part of the 3D foreground because the 3D feature density in that area is small and thus is discarded during the thresholding process. On the other hand, the 2D approach is not able to handle the shadow noise which results in false detections. In this experiment we achieve 66.62% true positives detection (TP) with 18.69% false positives (FP) for the SFM-HOG approach, 35.71/64.29% (TP/FP) for 3D, and 42.52/77.35% (TP/FP) for 2D approach. The summary of the SFM-HOG results are shown in Figure 6.22. This demonstrates the effectiveness of combining both the 2D and 3D scene detection results.



(a) Time-ordered 3D SFM data.



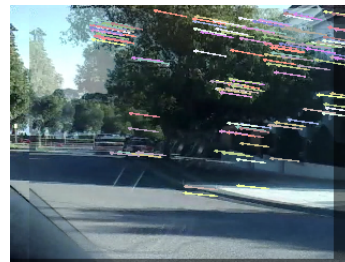
(b) Motion-fields for video 1.1.



(c) Motion-fields for video 1.2.



(d)



(e)

Figure 6.20: Dataset 1.

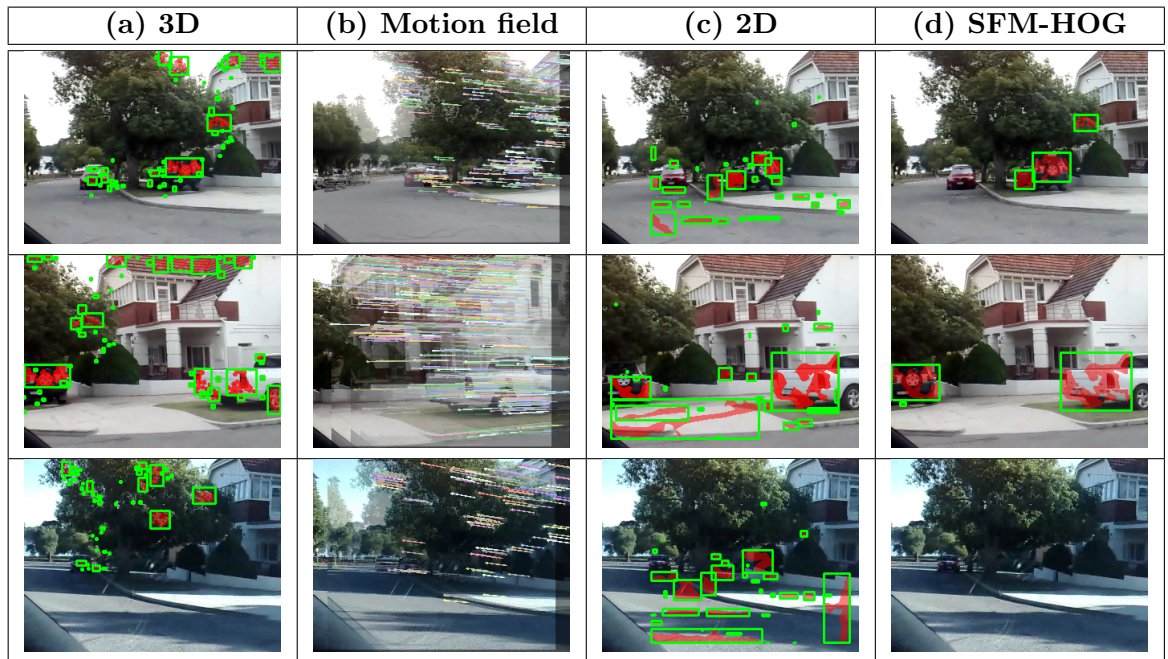


Figure 6.21: Scene change detection results.



Figure 6.22: SFM-HOG scene change detection results.

### 6.6.5.2 Experiment II: Single Video Changes

This experiment demonstrates the performance of SFM-HOG to detect scene changes in multiple video, where only one video contains moving foregrounds.

**Description:** Five video are used in the second experiment. Figure 6.23 shows the time-ordered 3D data and the estimated camera motion for dataset 2. In this experiment, video 2.1, 2.2, 2.3, and 2.5 do not contain any change, whilst there are *three scene changes* in video 2.4. The first two changes are a man and his child riding bicycles (between frames 80-89). Between frames 100 to 115 of video 2.4, there is a car driving on the opposite side of the road. In this experiment all foreground objects are moving, which can be seen from the motion field data shown in Figures 6.24 (b) and (d). On the other hand, objects in video 2.1 are stationary and have equal motion-field as shown in Figures 6.24 (a) and (c).

**Results:** Figure 6.23 (a) shows the 3D SNN subtraction results, while Figures 6.23 (b) and (c) show the noise eliminating process. Clusters that are not dense are considered as noise, which leads us to the final three clusters in Figure 6.23 (d). These clusters belong to the man, the child, and the cars respectively. This is confirmed by the results being projected back to the image as shown in Figure 6.26 (a). In addition, Figure 6.26 shows some comparative results between 2D, 3D, and SFM-HOG approaches. Although the lighting differs between all video, they still share similar gradient structures, hence we also obtain good image segmentation results as shown in Figure 6.26 (c). In this experiment, we achieve 79.29% true positive detection with 11.23% false positives for SFM-HOG approach, 57.72/10.15% (TP/FP) for 3D approach, and 67.11/82.75% (TP/FP) for 2D approach. In general, the SFM-HOG method is able to obtain the bounding boxes of the objects correctly, with a small number of false positives, as shown in Figure 6.27.

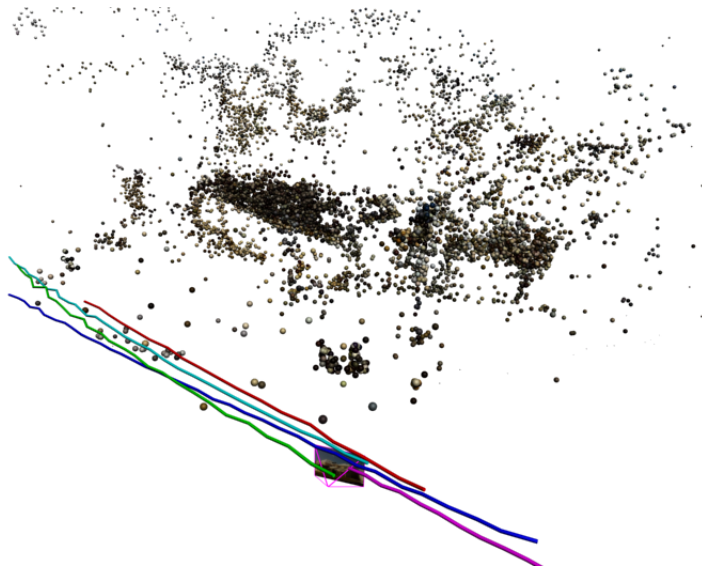


Figure 6.23: Time ordered 3D data and camera motion for experiment 2.

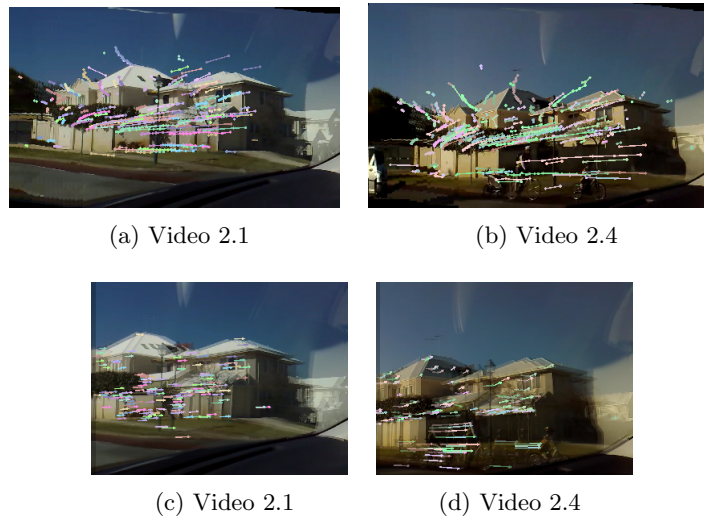


Figure 6.24: Comparison of different motion-field data for video set 2. (a) and (c) contain only stationary scene, whilst (b) and (d) contain moving foreground objects.

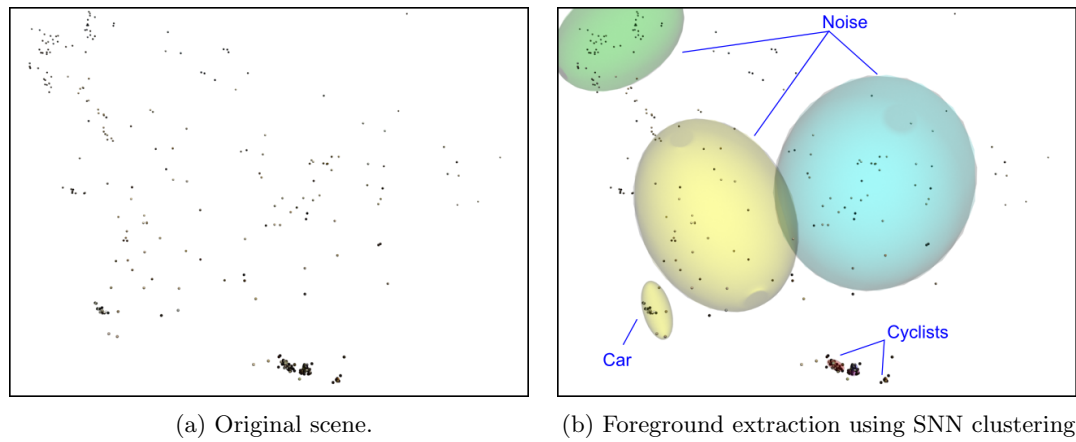


Figure 6.25: 3D SNN subtraction results. (Note that, for explanation purposes the SNN results in (b) are annotated. The actual results do not contains any annotations)

(a) 3D	(b) Motion-field	(c) 2D	(d) SFM-HOG

Figure 6.26: Scene change detection results.





Figure 6.27: SFM-HOG scene change detection results.

### 6.6.5.3 Experiment III: Multiple Video Changes

This experiment demonstrates the performance of SFM-HOG to detect scene changes in multiple video.

**Description:** Five video are used in this experiment. Figure 6.28 shows the constructed time-ordered 3D data and the estimated camera motion for dataset 3. Figure 6.29 shows the motion-field data for all video. In this experiment, the tree object in front of the house was cut off after video 3.3; therefore, it does not appear in video 3.4 and 3.5. Other changes include (1) a white jeep parked beside the house at video 3.1; (2) two white sedans appearing in video 3.3 but not in other video; and (3) a jeep travels in the opposite direction of the traffic in video 3.4.

**Results:** Figure 6.30 shows the results of the 3D scene change detections. From Figures 6.30 (a), (b), (c), we can see that the SFM-HOG is able to detect the foreground objects correctly, which consist of the tree and the car objects. From the projected 3D results (see Figure 6.31 (a)) we can see that it contains less foreground information as compared to the 2D results with a higher number of false positives, as shown in Figure 6.31 (c). By combining the results we show that the SFM-HOG is able to extract the changes correctly without large numbers of false positives, as shown in Figure 6.32. In this experiment, we achieve 77.06% true positive detection with 10.94% false positives for the SFM-HOG approach, 53.29/24.62% (TP/FP) for 3D approach, and 56.28/73.46% (TP/FP) for 2D approach.

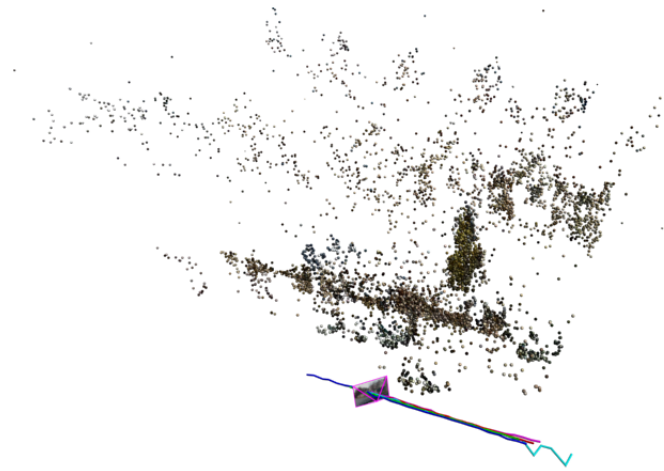
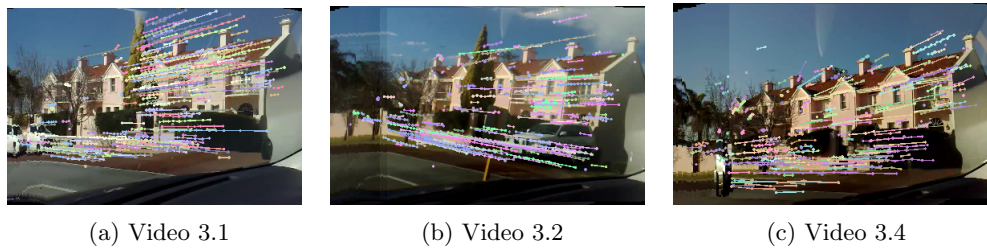


Figure 6.28: Time-ordered 3D scene reconstruction and its camera motion of dataset 3.

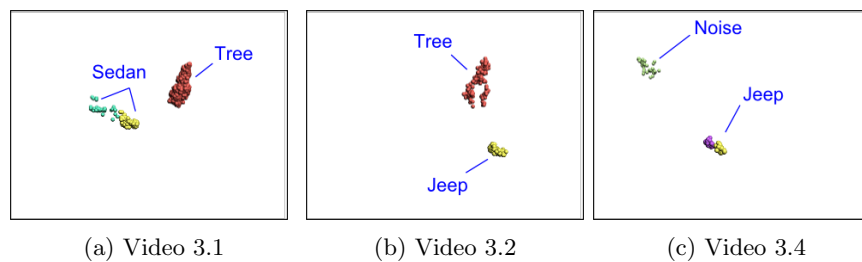


(a) Video 3.1

(b) Video 3.2

(c) Video 3.4

Figure 6.29: Motion-fields for video in dataset 3.



(a) Video 3.1

(b) Video 3.2

(c) Video 3.4

Figure 6.30: 3D scene change detection results. (Note that, for explanation purposes the SNN results in (a), (b), and (c) are annotated. The actual results do not contains any annotations)

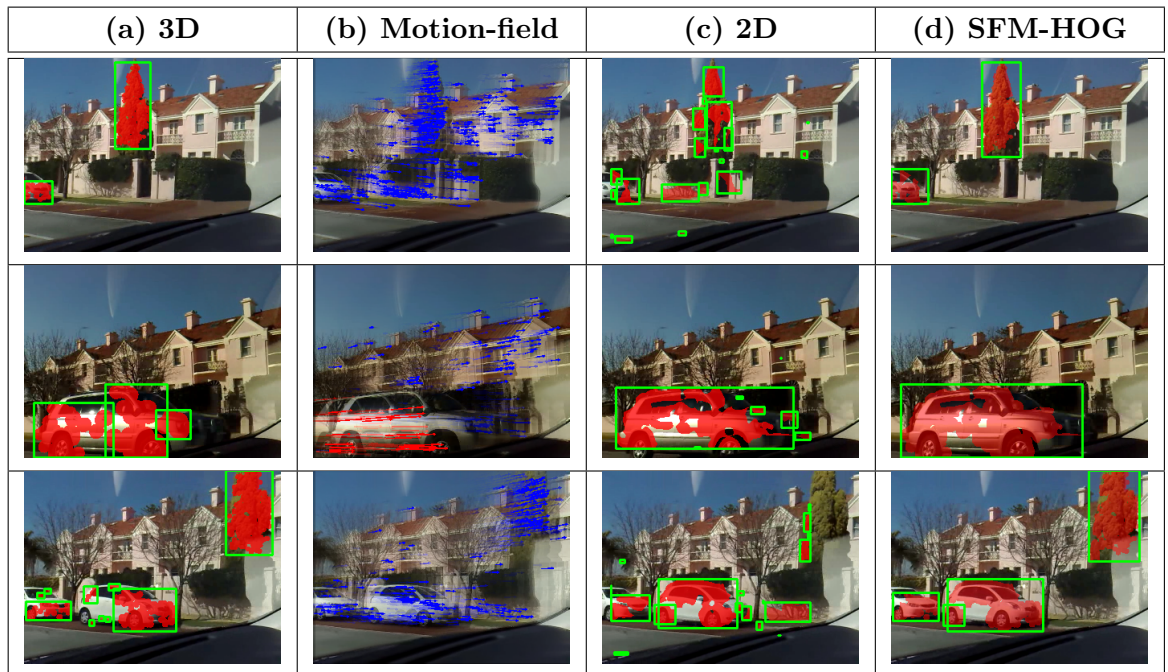


Figure 6.31: Scene change detection results.



Figure 6.32: SFM-HOG scene change detection results.

### 6.6.6 Discussion

Table 6.3 shows the overall performance of the different detection methods. The results show that by using the 2D scene change detection approach alone generally results in higher false positives as compared to SFM-HOG, while the 3D scene change detection approach alone gives lower true positives as compared to SFM-HOG. On the other hand, when they are combined together, the best performance is achieved. The intuitive reason

<i>Bounding box</i>	<i>True positive rate (%)</i>			<i>False positive rate (%)</i>		
<i>Sequence</i>	<i>3D</i>	<i>2D</i>	<i>SFM-HOG</i>	<i>3D</i>	<i>2D</i>	<i>SFM-HOG</i>
Dataset 1	35.71%	42.52%	66.62%	64.29%	77.35%	18.69%
Dataset 2	57.72%	67.11%	79.29%	10.15%	82.75%	11.23%
Dataset 3	53.29%	56.28%	77.06%	24.62%	73.46%	10.94%

<i>Pixel</i>	<i>True positive rate (%)</i>			<i>False positive rate (%)</i>		
<i>Sequence</i>	<i>3D</i>	<i>2D</i>	<i>SFM-HOG</i>	<i>3D</i>	<i>2D</i>	<i>SFM-HOG</i>
Dataset 1	28.53%	31.62%	41.36%	60.3%	70.97%	27.98%
Dataset 2	48.57%	54.21%	57.52%	12.28%	63.20%	11.23%
Dataset 3	55.07%	60.97%	65.88%	26.66%	56.11%	16.08%

Table 6.3: Evaluation results.

for this is because the noisy results of 3D and 2D do not overlap each other due to the different nature of the two techniques. The 2D results fill the sparse 3D results to give more information about the foreground objects. At the same time, since the 3D results do not contain noise, it reduces the false detections in the 2D results.

Based on the results at Table 6.3, we can also see that using less numbers of video such as experiment 1 results in low performance. As the dataset size increases, the noise of the dataset can be reduced by comparing multiple images and thus we obtain better overall performance.

## 6.7 Chapter Summary

In this chapter we have presented a system for scene changes detection. Our proposed framework is able to detect scene changes correctly despite the challenges in the datasets (variability in camera pose and lighting conditions). The main contribution lies in the use of 3D scene information to provide stable scene changes. The use of the 2D segmentation algorithm is to provide more detailed information about the object.

---

## CHAPTER 7

# CONCLUSION

---

This thesis has investigated the problems of foreground object detection for three challenging applications. In particular, we have aimed to improve the foreground object detection systems through the following tasks:

- Investigating state-of-art detection methods on our datasets.
- Proposing novel frameworks for foreground object detection that improve the accuracy rates of current approaches.
- Investigating the performance of the proposed detection frameworks in real-world datasets.

Chapter 3 investigated a surveillance system to detect and track passengers on moving buses. Specifically, we studied the lighting change issue and introduced a new approach that combines SIFT background modelling and elliptical human models to deal with passenger detection under different lighting conditions. In addition, we proposed the RJMCMC tracking algorithm with the trajectory and appearance transformation models to deal with low frame rate issues. Experiments were conducted on real bus footages. Results have shown that our proposed framework provides the best detection performance when compared to GMM and human detection approaches. Furthermore, the RJMCMC framework with appearance transformation model has outperformed MeanShift, Particle Filter, and appearance association method, demonstrating the usefulness of the proposed framework.

Chapter 4 discussed a pedestrian detection framework for front-facing bus cameras. Some important issues related to the moving cameras were investigated, in particular the problem of scene localization for background modelling. In addition, we employed the hierarchical template matching for pedestrian detection that is suitable for crowded scenes. Experimentation was conducted on the Perth dataset. Results have shown that our proposed framework significantly reduce the number of false positives as compared to the direct

pedestrian detection alone.

Chapter 5 investigated the performance of four pedestrian detection methods, namely HOG, HOGB, CHOG, and CHOGB. Experimentation was conducted using the Perth dataset. Results show that the HOG based detector achieves the highest performance, the HOGB approach has a comparable performance as compared to the HOG, and the codebook approaches allow computationally efficient detection. The detection performance is improved by incorporating the background subtraction approach as it eliminates high number of false positive candidates.

Chapter 6 discussed a scene change detection framework for a moving vehicle's camera. In particular, we studied the use of 3D scene information to improve the results of 2D segmentation approach. Experimentation was conducted on urban city street datasets. Results have shown that the combined 3D-and-2D approach outperforms the MRF background subtraction approach (Xu *et al.*, 2005).

## 7.1 Future Work

A number of possible improvements for future research can be identified through this research.

In Chapter 3, we have presented a surveillance system to detect and track passengers inside moving buses. In our approach, we use a single camera as the input to the system. This can be extended to using multiple distributed cameras, similar to (Mittal and Davis, 2003). The advantages of using multiple camera networks are the larger surveillance coverage and the ability to cross-check the detection results from multiple sources for reliable detection and tracking. In addition, we can learn the distribution of crowds based on the operating time and adjust the parameters of the algorithm accordingly, for example during peak hours, the system expects high number of occupancies and thus relaxing the detection rates in order to obtain high numbers of passengers, as compared to the normal hours where we expect small numbers of passengers.

In Chapters 4 and 5, we have investigated different pedestrian algorithms for mobile surveillance system. The output of this system is a set of people, which can be useful for a large scale tracking surveillance system. Methods such as (Wang *et al.*, 2003; Leibe *et al.*, 2008) can be used in conjunction to our system to localise pedestrians and track them in an urban city. In terms of the detection, our system does not incorporate any motion or

geometry cues such as SIFT flow (Liu *et al.*, 2008) or 3D scene information (Snavely *et al.*, 2008). When this information is available, it would improve the system performance.

In Chapter 6, we introduce a new framework for scene change detection using a monocular camera. We can improve the accuracy and speed of the 3D scene reconstruction method by using stereo vision similar to (Leibe *et al.*, 2008). Additional 3D cues such as the one proposed by Brostow *et al.* (2008) can be integrated with our proposed model to improve the 3D foreground detection method.

---

# BIBLIOGRAPHY

---

- Abdel-Hakim, A. E. and Farag, A. A. (2006). CSIFT: A SIFT Descriptor with Color Invariant Characteristics. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1978–1983, Washington, DC, USA. IEEE Computer Society.
- Ancuti, C. and Bekaert, P. (2007). SIFT-CCH: Increasing the SIFT distinctness by Color Co-occurrence Histograms. In *IEEE International Symposium on Image and Signal Processing and Analysis*, pages 27–29, Istambul.
- Assarsson, U. and Möller, T. (2000). Optimized View Frustum Culling Algorithms for Bounding Boxes. *Journal of Graphics Tools*, **5**, 9–22.
- Ballard, D. H. (1981). Generalizing the Hough Transform to Detect Arbitrary Shapes. *Pattern Recognition*, **13**(2), 111–122.
- Baumberg, A. (2000). Reliable Feature Matching Across Widely Separated Views. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 774–781.
- Bay, H., Tuytelaars, T., and Van Gool, L. (2006). SURF: Speeded Up Robust Features. In *IEEE European Conference on Computer Vision (ECCV)*, Graz Austria.
- Beaudet, P. R. (1978). Rotational Invariant Image Operators. In *IEEE International Conference on Pattern Recognition (ICPR)*, pages 579–583.
- Beis, J. S. and Lowe, D. G. (1997). Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 1000, Washington, DC, USA. IEEE Computer Society.
- Belongie, S., Malik, J., and Puzicha, J. (2000). Shape Context: A New Descriptor for Shape Matching and Object Recognition. In *Neural Information Processing Systems (NIPS)*, pages 831–837.
- Berg, A. C. and Malik, J. (2001). Geometric Blur for Template Matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 607–614.
- Besl, P. J. and McKay, N. D. (1992). A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **14**(2), 239–256.



- Beucher, S. and Meyer, F. (1993). The Morphological Approach to Segmentation: The Watershed Transformation. In *Mathematical Morphology in Image Processing*, pages 433–481.
- Bileschi, S. M. and Wolf, L. (2007). Image Representations Beyond Histograms of Gradients: The Role of Gestalt Descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Birchfield, S. (1998). Elliptical Head Tracking Using Intensity Gradients and Color Histograms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 232–237.
- Birchfield, S. T. and Rangarajan, S. (2005). Spatiograms Versus Histograms for Region-Based Tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Diego, California.
- Bookstein, F. L. (1989). Principal Warps: Thin-Plate Splines and The Decomposition of Deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **11**(6), 567–585.
- Bosch, A., Zisserman, A., and Munoz, X. (2007). Representing Shape with a Spatial Pyramid Kernel. In *IEEE International Conference on Image and Video Retrieval (CIVR)*, Amsterdam, The Netherlands.
- Bradshaw, K. J., Reid, I. D., and Murray, D. W. (1997). The Active Recovery of 3D Motion Trajectories and Their Use in Prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **19**(3), 219–234.
- Brostow, G. J., Shotton, J., Fauqueur, J., and Cipolla, R. (2008). Segmentation and Recognition Using Structure from Motion Point Clouds. In *IEEE European Conference on Computer Vision (ECCV)*, pages 44–57, Berlin, Heidelberg. Springer-Verlag.
- Brown, L. G. (1992). A Survey of Image Registration Techniques. *ACM Computing Surveys*, **24**(4), 325–376.
- Burghouts, G. J. and Geusebroek, J.-M. (2009). Performance Evaluation of Local Colour Invariants. *Computer Vision and Image Understanding*, **113**(1), 48–62.
- Chalechale, A., Mertins, A., and Naghdy, G. (2004). Edge Image Description using Angular Radial Partitioning. *International Conference on Computer Vision, Image and Signal Processing (ICCVISP)*, **151**(2), 93–101.
- Chang, P. and Krumm, J. (1999). Object Recognition with Color Cooccurrence Histogram. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

- Chen, C. H., Pau, L. F., and Wang, P. S. P., editors (2000). *Handbook of Pattern Recognition and Computer Vision*. World Scientific Publishing Co., Inc., River Edge, NJ, USA.
- Cheung, W. and Hamarneh, G. (2007). N-SIFT: N-Dimensional Scale Invariant Feature Transform For Matching Medical Images. In *IEEE International Symposium on Biomedical Imaging (ISBI)*.
- Comaniciu, D. and Meer, P. (2002). Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **24**(5), 603–619.
- Cortes, C. and Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, **20**(3), 273–297.
- Cramariuc, B., Shmulevich, I., Gabbouj, M., and Makela, A. (2000). A New Image Similarity Measure Based on Ordinal Correlation. In *International Conference on Image Processing (ICIP)*, volume 3, pages 718–721, Vancouver, BC, Canada.
- Cui, Y., Hasler, N., Thormahlen, T., and Seidel, H. (2009). Scale Invariant Feature Transform with Irregular Orientation Histogram Binning. In *International Conference on Image Analysis and Recognition (ICIAR)*, pages 258–267.
- Dalal, N. (2005). INRIA Person Dataset. <http://pascal.inrialpes.fr/data/human/>.
- Dalal, N. and Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 886–893, INRIA Rhône-Alpes, av. de l’Europe.
- Deselaers, T., Hegerath, A., Keysers, D., and Ney, H. (2006). Sparse Patch-Histograms for Object Classification in Cluttered Images. In *Proceedings of the German Symposium on Pattern Recognition (DAGM)*, volume 4174 of *LNCS*, pages 202–211, Berlin, Germany. Springer, Springer.
- Dollar, P., Wojek, C., Schiele, B., and Perona, P. (2009). Pedestrian Detection: A Benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami.
- Ekvall, S. and Kragic, D. (2005). Receptive Field Cooccurrence Histograms for Object Detection. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Edmonton, Canada.
- Elgammal, A. M., Harwood, D., and Davis, L. S. (2000). Non-parametric Model for Background Subtraction. In *IEEE European Conference on Computer Vision (ECCV)*, pages 751–767, London, UK. Springer-Verlag.

- Enzweiler, M. and Gavrilu, D. M. (2009). Monocular Pedestrian Detection: Survey and Experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **99**(1).
- Ertöz, L., Steinbach, M., and Kumar, V. (2003). Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data. In *Second SIAM International Conference on Data Mining*, San Francisco, CA, USA.
- Ester, M., Kriegel, H.-P., Jörg, S., and Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *The Second International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 226–231, Portland, Oregon, USA.
- Felzenszwalb, P., Mcallester, D., and Ramanan, D. (2008). A Discriminatively Trained, Multiscale, Deformable Part Model. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Anchorage, Alaska.
- Ferrari, V., Fevrier, L., Jurie, F., and Schmid, C. (2008). Groups of Adjacent Contour Segments for Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **30**(1), 36–51.
- Fischler, M. A. and Bolles, R. C. (1981). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM (CACM)*, **24**(6), 381–395.
- Forssén, P.-E. (2007). Maximally Stable Colour Regions for Recognition and Matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, USA. IEEE Computer Society, IEEE.
- Forstner, W. and Moonen, B. (1999). A Metric for Covariance Matrices. In *Technical report, Dept. of Geodesy and Geoinformatics, Stuttgart University*.
- Franz, A., Carlsen, I. C., and Renisch, S. (2006). An Adaptive Irregular Grid Approach Using SIFT Features for Elastic Medical Image Registration. In *Bildverarbeitung für die Medizin*.
- Freeman, W. T. and Adelson, E. H. (1991). The Design and Use of Steerable Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **13**(9), 891–906.
- Freund, Y. and Schapire, R. E. (1996). Experiments with a New Boosting Algorithm. In *International Conference on Machine Learning (ICML)*, pages 148–156.
- Gavrila, D. M. and Philomin, V. (1999). Real-Time Object Detection for Smart Vehicles. In *IEEE International Conference on Computer Vision (ICCV)*, pages 87–93.

- Geronimo, D., Sappa, A., Lopez, A., and Ponsa, D. (2007). Adaptive Image Sampling and Windows Classification for On-board Pedestrian Detection. In *IEEE International Conference on Computer Vision (ICCV)*, Bielefeld, Germany.
- GoogleEarth (2009). Version 4.2. <http://earth.google.com/> [Last accessed: 19-09-2009].
- Gordon, N., Salmond, D., and Smith, A. (1993). Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation. *IEEE Proceedings F on Radar and Signal Processing*, **140**(2), 107–113.
- Grabner, M., Grabner, H., and Bischof, H. (2006). Fast Approximated SIFT. In *Asian Conference on Computer Vision (ACCV)*, volume 1, pages 918–927.
- Grauman, K. and Darrell, T. (2005). The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features. In *IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1458–1465.
- Greenhill, S. and Venkatesh, S. (2006). Virtual Observers in a Mobile Surveillance System. In *ACM*, pages 579–588, New York, NY, USA.
- Harris, C. and Stephens, M. (1988). A Combined Corner and Edge Detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151.
- Hartigan, J. A. and Wong, M. A. (1979). Algorithm AS 136: A K-Means Clustering Algorithm. *Applied Statistics*, **28**(1), 100–108.
- Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition.
- Heikkilä, M., Pietikäinen, M., and Schmid, C. (2009). Description of Interest Regions with Local Binary Patterns. *Pattern Recognition*, **42**(3), 425–436.
- Horn, B. K. (1986). *Robot Vision*. McGraw-Hill Higher Education.
- Horn, B. K. P. (1999). Projective Geometry Considered Harmful.
- Huang, J., Kumar, S. R., Mitra, M., Zhu, W.-J., and Zabih, R. (1997). Image Indexing Using Color Correlograms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 762, Washington, DC, USA. IEEE Computer Society.
- Huang, R., Pavlovic, V., and Metaxas, D. (2008). A New Spatio-Temporal MRF Framework for Video-Based Object Segmentation. In *1st International Workshop on Machine Learning for Vision-based Motion Analysis (MLVMA'08), in conjunction with ECCV 2008*.

- İkizler, N. and Duygulu, P. (2007). Human Action Recognition Using Distribution of Oriented Rectangular Patches. *Human Motion - Understanding, Modeling, Capture and Animation*, pages 271–284.
- Jain, A. K. and Farrokhnia, F. (1991). Unsupervised Texture Segmentation using Gabor Filters. *Pattern Recognition*, **24**(12), 1167–1186.
- Javed, O., Shafique, K., and Shah, M. (2002). A Hierarchical Approach to Robust Background Subtraction using Color and Gradient Information. In *IEEE Workshop on Visual Motion Computing (MOTION)*, page 22, Washington, DC, USA.
- Javed, O., Shafique, K., and Shah, M. (2005). Appearance Modeling for Tracking in Multiple Non-Overlapping Cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26–33, Washington, DC, USA. IEEE Computer Society.
- Jeong, K. and Jaynes, C. (2006). Object Matching in Disjoint Cameras using a Color Transfer Approach. *Special Issue of Machine Vision and Applications Journal*.
- Jones, M. J., Viola, P., and Snow, D. (2003). Detecting Pedestrians Using Patterns of Motion and Appearance. In *IEEE International Conference on Computer Vision (ICCV)*, pages 734–741.
- Jurie, F. and Triggs, B. (2005). Creating Efficient Codebooks for Visual Recognition. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 604–610, Washington, DC, USA. IEEE Computer Society.
- Kadir, T., Zisserman, A., and Brady, M. (2004). An Affine Invariant Salient Region Detector. In *IEEE European Conference on Computer Vision (ECCV)*, volume 1, pages 228–241.
- Ke, Y. and Sukthankar, R. (2004). PCA-SIFT: A More Distinctive Representation for Local Image Descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 506–513.
- Kembhavi, A., Schwartz, W., and Davis, L. (2008). Resource Allocation for Tracking Multiple Targets using Particle Filters. *Workshop on Visual Surveillance (VS2008)*.
- Khan, Z., Balch, T. R., and Dellaert, F. (2004). An MCMC-Based Particle Filter for Tracking Multiple Interacting Targets. In *IEEE European Conference on Computer Vision (ECCV)*, volume 4, pages 279–290.
- Khan, Z., Balch, T., and Dellaert, F. (2005). MCMC-Based Particle Filtering for Tracking a Variable Number of Interacting Targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **27**(11), 1805–1918.

- Khan, Z., Balch, T., , and Dellaert., F. (2006). MCMC Data Association and Sparse Factorization Updating for Real Time Multitarget Tracking with Merged and Multiple Measurements. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*.
- Kim, K., Chalidabhongse, T. H., Harwood, D., and Davis, L. S. (2005). Real-time Foreground-Background Segmentation Using Codebook Model. *Real-Time Imaging*, **11**(3), 172–185.
- Kuhn, H. W. (1955). The Hungarian Method for The Assignment Problem. *Naval Research Logistic Quarterly*, **2**, 83–97.
- Lampert, C. H., Blaschko, M. B., and Hofmann, T. (2008). Beyond Sliding Windows: Object Localization by Efficient Subwindow Search. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.
- Laptev, I. (2009). Improving Object Detection with Boosted Histograms. *Image and Vision Computing*, **27**(5), 535–544.
- Lazebnik, S., Schmid, C., and Ponce, J. (2005). A Sparse Texture Representation Using Local Affine Regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **27**(8), 1265–1278.
- Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2169–2178.
- Ledwich, L. and Williams, S. (2005). Reduced SIFT Features For Image Retrieval and Indoor Localisation. In *Australian Conference on Robotics and Automation (ACRA)*.
- Lee, H.-Y., Kim, H., and Lee, H.-K. (2006). Robust Image Watermarking using Local Invariant Features. *Optical Engineering*, **3**, 45.
- Leibe, B., Seemann, E., and Schiele, B. (2005). Pedestrian Detection in Crowded Scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 878–885, San Diego, California, USA.
- Leibe, B., Schindler, K., Cornelis, N., and Van Gool, L. (2008). Coupled Object Detection and Tracking from Static Cameras and Moving Vehicles. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **30**(10), 1683–1698.
- Leoputra, W., Tan, T., and Lim, F. L. (2006). Non-overlapping Distributed Tracking using Particle Filter. In *IEEE International Conference on Pattern Recognition (ICPR)*, volume 3, pages 181–185, Los Alamitos, CA, USA. IEEE Computer Society.

- Leordeanu, M., Hebert, M., and Sukthankar, R. (2007). Beyond Local Appearance: Category Recognition from Pairwise Interactions of Simple Features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Leung, T. and Malik, J. (2001). Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons. *International Journal of Computer Vision (IJCV)*, **43**(1), 29–44.
- Levi, K. and Weiss, Y. (2004). Learning Object Detection from a Small Number of Examples: the Importance of Good Features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 53–60.
- Li, F.-F. and Perona, P. (2005). A Bayesian Hierarchical Model for Learning Natural Scene Categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 524–531, Washington, DC, USA. IEEE Computer Society.
- Li, Y., Ai, H., Yamashita, T., Lao, S., and Kawade, M. (2007). Tracking in Low Frame Rate Video: A Cascade Particle Filter with Discriminative Observers of Different Lifespans. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1728–1740.
- Lindeberg, T. (1998). Feature Detection with Automatic Scale Selection. *International Journal of Computer Vision (IJCV)*, **30**, 79–116.
- Ling, H. and Jacobs, D. W. (2005). Deformation Invariant Image Matching. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1466–1473, Washington, DC, USA. IEEE Computer Society.
- Lipton, A. J., Heartwell, C. H., Haering, N., and Madden, D. (2002). Critical Asset Protection, Perimeter Monitoring, and Threat Detection Using Automated Video Surveillance. In *Proceedings. 36th Annual 2002 International Carnahan Conference*, 11600 Sunrise Valley Dr, Suite 290 Reston, VA 20191.
- Liu, C., Yuen, J., Torralba, A., Sivic, J., and Freeman, W. T. (2008). SIFT Flow: Dense Correspondence across Different Scenes. In *IEEE European Conference on Computer Vision (ECCV)*, pages 28–42, Berlin, Heidelberg. Springer-Verlag.
- Lodha, S. K. and Xiao, Y. (2006). GSIFT: Geometric Scale Invariant Feature Transform for Terrain Data. In *Proceedings of SPIE*, volume 6066.
- Lowe, D. G. (2001). Local Feature View Clustering for 3D Object Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, Los Alamitos, CA, USA. IEEE Computer Society.

- Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision (IJCV)*, **60**(2), 91–110.
- Maji, S., Berg, A. C., and Malik, J. (2008). Classification Using Intersection Kernel Support Vector Machines is Efficient. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.
- Matas, J., Chum, O., Urban, M., and Pajdla, T. (2002). Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. In *British Machine Vision Conference (BMVC)*.
- Mikolajczyk, K. and Schmid, C. (2002). An Affine Invariant Interest Point Detector. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, Canada.
- Mikolajczyk, K. and Schmid, C. (2004). Scale and Affine Invariant Interest Point Detectors. *International Journal of Computer Vision (IJCV)*, **60**(1), 63–86.
- Mikolajczyk, K. and Schmid, C. (2005). A Performance Evaluation of Local Descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **27**(10), 1615–1630.
- Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and Gool, L. V. (2005). A Comparison of Affine Region Detectors. *International Journal of Computer Vision (IJCV)*, **65**(1/2), 43–72.
- Mittal, A. and Davis, L. S. (2003). M2Tracker: A Multi-View Approach to Segmenting and Tracking People in a Cluttered Scene. *International Journal of Computer Vision (IJCV)*, **51**(3), 189–203.
- Miyajima, K. and Ralescu, A. (1994). Spatial organization in 2D segmented images: representation and recognition of primitive spatial relations. *Fuzzy Sets Syst.*, **65**(2-3), 225–236.
- Mortensen, E. N., Deng, H., and Shapiro, L. G. (2005). A SIFT Descriptor with Global Context. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 184–190.
- Munder, S. and Gavrilu, D. (2006). An Experimental Study on Pedestrian Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **28**(11), 1863–1868.
- Nilsson, M., Bartunek, J., Nordberg, J., and Claesson, I. (2008). On histograms and spatiograms: Introduction of the mapogram. In *IEEE International Conference on Image Processing (ICIP)*, pages 973–976.



- Noriega, P. and Bernier, O. (2006). Real Time Illumination Invariant Background Subtraction Using Local Kernel Histograms. In *British Machine Vision Conference (BMVC)*, page III:979.
- Nowak, E., Jurie, F., and Triggs, B. (2006). Sampling Strategies for Bag-of-Features Image Classification. In *IEEE European Conference on Computer Vision (ECCV)*, pages 490–503.
- Ojala, T., Pietikainen, M., and Maenpaa, T. (2002). Multiresolution Gray-scale and Rotation Invariant Texture Classification with Local Binary Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **24**(7), 971–987.
- Okuma, K., Taleghani, A., and Freitas, N. D. (2004). A Boosted Particle Filter: Multitarget Detection and Tracking. In *IEEE European Conference on Computer Vision (ECCV)*, volume 1, pages 28–39, Prague, Czech Republic.
- Oliva, A. and Torralba, A. (2001). Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. *International Journal of Computer Vision (IJCV)*, **42**(3), 145–175.
- Oliver, N. M., Rosario, B., and Pentland, A. P. (2000). A Bayesian Computer Vision System for Modeling Human Interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **22**(8), 831–843.
- Opelt, A., Pinz, A., and Zisserman, A. (2006). A Boundary-Fragment-Model for Object Detection. In A. Leonardis, H. Bischof, and A. Pinz, editors, *IEEE European Conference on Computer Vision (ECCV)*, volume 2 of *Lecture Notes in Computer Science*, pages 575–588. Springer.
- Overett, G., Petersson, L., Brewer, N., Andersson, L., and Pettersson, N. (2008). A New Pedestrian Dataset for Supervised Learning. In *IEEE Intelligent Vehicles Symposium*.
- Papageorgiou, C. and Poggio, T. (1999). Trainable Pedestrian Detection. In *IEEE International Conference on Image Processing (ICIP)*, pages IV:35–39.
- Pass, G. and Zabih, R. (1996). Histogram Refinement for Content-Based Image Retrieval. In *IEEE Workshop on Applications of Computer Vision*, pages 96–102.
- Perlibakas, V. (2004). Distance measures for PCA-based face recognition. *Pattern Recognition Letters*, **25**(6), 711–724.
- Piccardi, M. (2004). Background Subtraction Techniques: A Review. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 4, pages 3099–3104 vol.4.

- Porikli, F. and Tuzel, O. (2005). Object tracking in low-frame-rate video. In A. Said and J. G. Apostolopoulos, editors, *Image and Video Communications and Processing 2005. Edited by Said, Amir; Apostolopoulos, John G. Proceedings of the SPIE, Volume 5685, pp. 72-79 (2005).*, volume 5685 of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, pages 72–79.
- Porikli, F., Tuzel, O., and Meer, P. (2005). Covariance Tracking using Model Update Based on Lie Algebra. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Qin, L. and Gao, W. (2005). Image matching based on a local invariant descriptor. In *IEEE International Conference on Image Processing (ICIP)*, volume 3, pages 377–380.
- Quan, L. and Lan, Z. (1999). Linear N-Point Camera Pose Determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **21**(8), 774–780.
- Reisert, M., Ronneberger, O., and Burkhardt, H. (2007). Holomorphic Filters for Object Detection. In *Proceedings of the German Symposium on Pattern Recognition (DAGM)*, pages 304–313.
- Rubner, Y., Tomasi, C., and Guibas, L. J. (1998). A Metric for Distributions with Applications to Image Databases. In *IEEE International Conference on Computer Vision (ICCV)*, page 59, Washington, DC, USA. IEEE Computer Society.
- Sabzmeydani, P. and Mori, G. (2007). Detecting Pedestrians by Learning Shapelet Features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Schaffalitzky, F. and Zisserman, A. (2001). Viewpoint invariant texture matching and wide baseline stereo. In *IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 636–643.
- Schiele, B. and Crowley, J. L. (1996). Object Recognition using Multidimensional Receptive Field Histograms. In *IEEE European Conference on Computer Vision (ECCV)*, pages 610–619.
- Schmid, C. (2001). Constructing models for content-based image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 39–45.
- Schneiderman, H. (2000). A Statistical Approach to 3D Object Detection Applied to Faces and Cars. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Scott, D. W. (1992). *Multivariate Density Estimation*. Wiley-Interscience.

- Serre, T., Wolf, L., and Poggio, T. (2005). Object Recognition with Features Inspired by Visual Cortex. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 994–1000, Washington, DC, USA. IEEE Computer Society.
- Shotton, J., Blake, A., and Cipolla, R. (2005). Contour-Based Learning for Object Detection. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 503–510, Washington, DC, USA. IEEE Computer Society.
- Simoncelli, E. and Farid, H. (1996). Steerable Wedge Filters for Local Orientation Analysis. *IEEE Transactions on Image Processing*, **5**(9), 1377–1382.
- Sinha, S. N., Frahm, J.-M., Pollefeys, M., and Gen, Y. (2006). GPU-Based Video Feature Tracking and Matching. In *techreport*.
- Smith, K., Gatica-Perez, D., and Odobez, J.-M. (2005). Using Particles to Track Varying Numbers of Interacting People. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 962–969, Washington, DC, USA. IEEE Computer Society.
- Snavely, N., Seitz, S. M., and Szeliski, R. (2008). Modeling the World from Internet Photo Collections. *International Journal of Computer Vision*, **80**(2), 189–210.
- Stanski, A. and Hellwich, O. (2005). Spiders as Robust Point Descriptors. In *Proceedings of the German Symposium on Pattern Recognition (DAGM)*, pages 262–268.
- Stauffer, C. and Grimson, W. E. L. (1999). Adaptive Background Mixture Models for Real-Time Tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2.
- Stauffer, C. and Grimson, W. E. L. (2000). Learning Patterns of Activity Using Real-Time Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **8**(22), 747–757.
- Stein, A. and Hebert, M. (2005). Incorporating Background Invariance into Feature-Based Object Recognition. In *IEEE Workshop on Applications of Computer Vision (WACV)*.
- Swain, M. J. and Ballard, D. H. (1991). Color Indexing. *International Journal of Computer Vision (IJCV)*, **7**, 11–32.
- Tamimi, H. and Zell, A. (2005). Global Robot Localization using Iterative Scale Invariant Feature Transform. In *International Symposium on Robotics (ISR)*, Tokyo, Japan.
- Tang, F., Brennan, S., Zhao, Q., and Tao, H. (2007). Co-Tracking Using Semi-Supervised Support Vector Machines. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1–8.

- Toews, M. and Wells, W. (2009). SIFT-Rank: Ordinal description for invariant feature correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 172–177.
- Tola, E., Lepetit, V., and Fua, P. (2009). DAISY: An Efficient Dense Descriptor Applied to Wide Baseline Stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*.
- Toyama, K., Krumm, J., Brumitt, B., and Meyers, B. (1999). Wallflower: Principles and Practice of Background Maintenance. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 255–261.
- Tuytelaars, T. and Gool, L. J. V. (2004). Matching Widely Separated Views Based on Affine Invariant Regions. *International Journal of Computer Vision (IJCV)*, **59**(1), 61–85.
- Tuytelaars, T. and Mikolajczyk, K. (2008). *Local Invariant Feature Detectors: A Survey*. 1601981384. Now Publishers Inc.
- Tuzel, O., Porikli, F., and Meer, P. (2007). Human Detection via Classification on Riemannian Manifolds. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.
- Viola, P. and Jones, M. (2002). Robust Real-time Object Detection. *International Journal of Computer Vision (IJCV)*.
- Wang, C.-C., Thorpe, C., and Thrun, S. (2003). Online Simultaneous Localization and Mapping with Detection and Tracking of Moving Objects: Theory and Results from a Ground Vehicle in Crowded Urban Areas. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 842–849.
- Wang, Y. and Makedon, F. (2003). R-Histogram: Quantitative Representation of Spatial-Relations for Similarity-Based Image Retrieval. In *ACM International Conference on Multimedia*, pages 323–326, Berkeley, California, USA.
- Winn, J., Criminisi, A., and Minka, T. (2005). Object Categorization by Learned Universal Visual Dictionary. In *IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1800–1807 Vol. 2.
- Wojek, C., Walk, S., and Schiele, B. (2009). Multi-Cue Onboard Pedestrian Detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.
- Wu, B. and Nevatia, R. (2005). Detection of Multiple, Partially Occluded Humans in a Single Image by Bayesian Combination of Edgelet Part Detectors. In *IEEE International Conference on Computer Vision (ICCV)*, pages I: 90–97.

- Wu, B. and Nevatia, R. (2007). Cluster Boosted Tree Classifier for Multi-View, Multi-Pose Object Detection. In *IEEE International Conference on Computer Vision (ICCV)*.
- Xiao, J. and Shah, M. (2005). Motion Layer Extraction in the Presence of Occlusion Using Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **27**(10), 1644–1659.
- Xu, W., Zhou, Y., Gong, Y., and Tao, H. (2005). Background Modeling Using Time Dependent Markov Random Field With Image Pyramid. In *IEEE Workshop on Visual Motion Computing (MOTION)*.
- Yu, G. and Morel, J.-M. (2009). A Fully Affine Invariant Image Comparison Method. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1597–1600, Washington, DC, USA. IEEE Computer Society.
- Yu, Q. and Medioni, G. (2007). Multiple-Target Tracking by Spatiotemporal Monte Carlo Markov Chain Data Association. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Yu, T. and Wu, Y. (2004). Collaborative Tracking of Multiple Targets. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 834–841, Los Alamitos, CA, USA. IEEE Computer Society.
- Yu, T. and Wu, Y. (2005). Decentralized Multiple Target Tracking Using Netted Collaborative Autonomous Trackers. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 939–946, Washington, DC, USA. IEEE Computer Society.
- Zhao, T. and Nevatia, R. (2004). Tracking multiple humans in crowded environment. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–406–II–413.
- Zhe, L., Davis, Larry, S., and Doermann, D. (2007). Hierarchical Part-Template Matching for Human Detection and Segmentation. In *IEEE International Conference on Computer Vision (ICCV)*, Rio de Janeiro, Brazil.
- Zhou, Y. and Huang, T. S. (2006). Weighted Bayesian Network for Visual Tracking. In *IEEE International Conference on Pattern Recognition (ICPR)*, pages 523–526.
- Zhu, Q., Yeh, M.-C., Cheng, K.-T., and Avidan, S. (2006). Fast Human Detection Using a Cascade of Histograms of Oriented Gradients. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1491–1498, Washington, USA.

*BIBLIOGRAPHY*

---

Zitova, B. and Flusser, J. (2003). Image Registration Methods: A Survey. *Image and Vision Computing*, **21**(11), 977–1000.

Every reasonable effort has been made to acknowledge the owners of copyright material. I would be pleased to hear from any copyright owner who has been omitted or incorrectly acknowledged.

---

## APPENDIX A

# COPYRIGHT CLEARANCE

---

From j.hansson@ieee.org  
Sent Monday, Nov 11, 2009 7:39 am  
To Wilson Suryajaya Leoputra <w.leoputra@postgrad.curtin.edu.au>  
Cc  
Bcc  
Subject Re: Using figures of IEEE papers for PhD Thesis

Comments/Response to Case ID: 002DDED3

ReplyTo: Copyrights@ieee.org

From: Jacqueline Hansson  
Send To: Wilson Suryajaya Leoputra  
<w.leoputra@postgrad.curtin.edu.au>  
Subject: Re: Using figures of  
IEEE papers for PhD  
Thesis  
Date: 11/11/2009

Dear Wilson:

This is in response to your letter below, in which you have requested permission to reprint, in your upcoming thesis/dissertation, the described IEEE copyrighted figures. We are happy to grant this permission.

Our only requirements are that you credit the original source (author, paper, and publication), and that the IEEE copyright line ( c [Year] IEEE) appears prominently with each reprinted figure.

Sincerely yours,

Jacqueline Hansson

-----  
IEEE Intellectual Property Rights Office  
445 Hoes Lane  
Piscataway, NJ 08855-1331 USA  
+1 732 562 3966 (phone)  
+1 732 562 1746 (fax)

IEEE-- Fostering technological innovation  
and excellence for the benefit of humanity.  
-----

\*\*\*\*\*

Dear Sir/Madam,

My name is Wilson Suryajaya Leoputra and I am a PhD student at Curtin University; Perth, Western Australia. I would like request the permission, if I can include any figure in the following IEEE papers for my PhD thesis:

- 1) Cramariuc, B., I. Shmulevich, M. Gabbouj, and A. Makela (2000, September). A new image similarity measure based on ordinal correlation. In IEEE International Conference on Image Processing, Volume 3, Vancouver, BC, Canada, pp. 718-721.
- 2) Mikolajczyk, K. and C. Schmid (2005). A Performance Evaluation of Local Descriptors. In IEEE Transactions on Pattern Analysis & Machine Intelligence (PAMI) 27 (10), 1615-1630.
- 3) Mikolajczyk, K., T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaalitzky, T. Kadir, and L. V. Gool (2005). A comparison of affine region detectors. In IEEE International Journal of Computer Vision 65 (1/2), 43-72.



- 4) Berg, A. C. and J. Malik (2001). Geometric blur for template matching. In IEEE Conference on Computer Vision and Pattern Recognition (1), pp. 607-614.
- 5) Lazebnik, S., C. Schmid, and J. Ponce (2005). A Sparse Texture Representation Using Local Affine Regions. In IEEE Transactions Pattern Analysis and Machine Intelligence, on 27 (8), 1265-1278.
- 6) Belongie, S., J. Malik, and J. Puzicha (2000). Shape Context: A New Descriptor for Shape Matching and Object Recognition. In IEEE NIPS, pp. 831-837.
- 7) Birchfield, S. T. and S. Rangarajan (2005, June). Spatiograms Versus Histograms for Region-Based Tracking. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, California
- 8) Enzweiler, M. and D. M. Gavrilu (2009). Monocular pedestrian detection: Survey and experiments. IEEE Transactions on Pattern Analysis and Machine Intelligence 99 (1).
- 9) Leoputra, W. S. and Venkatesh, S. and Tan, T. "Passenger Monitoring in Moving Bus Video", IEEE International Conference on Control, Automation, Robotics and Vision, December 17-20, Hanoi, Vietnam 2008.
- 10) Leoputra, W. S. and Venkatesh, S. and Tan, T. "Pedestrian Detection for Mobile Bus Surveillance", IEEE International Conference on Control, Automation, Robotics and Vision, December 17-20, Hanoi, Vietnam 2008.
- 11) Leoputra, W. S. and Venkatesh, S. and Tan, T. "Comparative Evaluation of Pedestrian Detection Methods for Mobile Bus Surveillance", IEEE International Conference on Acoustics, Speech, and Signal Processing, April 20-24, Taiwan 2009.

I am aware that if I include any figure that I will reference the paper as well as attaching the copyright symbol followed by the year and IEEE at the caption of the figure.

*APPENDIX A. COPYRIGHT CLEARANCE*

---

Thank you very much for your cooperation in this matter!

Kind Regards,

Wilson