OBJECT OCCLUSION AND OBJECT REMOVAL DETECTION

By

CHAI YUNG JOON

A dissertation submitted to the Department of Internet Engineering and Computer Science,
Lee Kong Chian Faculty of Engineering and Science,
Universiti Tunku Abdul Rahman,
in partial fulfillment of the requirements for the degree of
Master of Computer Science
March 2015

ABSTRACT

OBJECT OCCLUSION AND OBJECT REMOVAL DETECTION

Chai Yung Joon

In this day and age, with more and more burglary cases happening, the society pays more attention to security issues. Thus there are many researches that are being carried out to improve the security system. There are many methods to detect burglary. One of the surveillance systems that are commonly used to detect burglary is the closed circuit television (CCTV). However, there are some drawbacks in this system. In some situations, when a person is blocking an object in a particular scene in CCTV, it is possible that the system assumes that as burglary due to the disappearance of the object. Thus, if we can make a system that can differentiate between removal and occlusion when the object disappears in a scene, this can increase the intelligence of the system as well as reduce false alarms.

The main objective of this project is to detect object removal and object occlusion events that happen in a scene. In the project, all moving objects appeared in the video scene are detected and kept track to monitor their location. When the object is removed or occluded, the system needs to detect the event correctly. Besides that, the system can identify a suspected person. Colour histogram and Canny edge detector are used to detect moving object in the scene, while Kalman tracker is opted to track the location of moving

objects. The event is being detected using Canny edge, object region and outer region.

ACKNOWLEDGEMENT

Firstly, I would like to express my deepest gratitude to my supervisor, Dr. Khor Siak Wang and my co-supervisor, Dr. Tay Yong Haur. They share a lot of knowledge and guidance to me in my study. Without their guidance, I would not have been able to have this precious experience to study in image processing field.

Secondly, I would like to thank my lovely family and dearest friends who gave me support during my study. When I felt upset with the progress of the project, they comforted me. When I was lacking of ideas on the project, they gave me plenty of ideas.

Besides that, I would also like to thank UTAR and MIMOS Bhd for providing place, tools, and equipment for me to do my research.

APPROVAL SHEET

This dissertation entitled "**OBJECT OCCLUSION AND OBJECT REMOVAL DETECTION**" was prepared by CHAI YUNG JOON and submitted as partial fulfilment of the requirements for the degree of Master of Computer Science at Universiti Tunku Abdul Rahman.

Approved by:

_____

(Dr. Khor Siak Wang) Date:……..

Assistant Professor/Supervisor

Department of Information System

Faculty of Information Communication and Technology

Universiti Tunku Abdul Rahman

_____

(Dr. Tay Yong Haur) Date:……..

Associate Professor/Co-supervisor

Department of Internet Engineering and Computer Science

Lee Kong Chian Faculty of Engineering and Science

Universiti Tunku Abdul Rahman

FACULTY OF ENGINEERING AND SCIENCE

UNIVERSITI TUNKU ABDUL RAHMAN

Date: _____

SUBMISSION OF DISSERTATION

It is hereby certified that **CHAI YUNG JOON** (ID No: **11UEM01637**) has completed this dissertation entitled "OBJECT OCCLUSSION AND OBJECT REMOVAL DETECTION" under the supervision of Dr. Khor Siak Wang (Supervisor) from the Department of Information System, Faculty of Information Communication and Technology, and Dr. Tay Yong Haur (Co-Supervisor) from the Department of Internet Engineering and Computer Science, Lee Kong Chian Faculty of Engineering and Science.

I understand that the University will upload softcopy of my dissertation in PDF format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

_____

Chai Yung Joon

DECLARATION

I Chai Yung Joon hereby declare that the dissertation is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTAR or other institutions.

_____

CHAI YUNG JOON

Date _____

TABLE OF CONTENTS

		Page
ABSTRACT		**ii**
ACKNOWLEDGEMENT		**iii**
APPROVAL SHEET		**iv**
SUBMISSION SHEET		**v**
DECLARATION		**vi**
TABLE OF CONTENTS		**vii**
LIST OF TABLES		**ix**
LIST OF FIGURES		**xi**
LIST OF ABBREVIATIONS		**xiii**
CHAPTERS		
1.0	**INTRODUCTION**	**1**
1.1	Introduction	1
1.2	Problem Statements	2
1.3	Project Objectives	2
1.3.1	Automate the detection of moving objects	2
1.3.2	Automate the tracking of moving objects	3
1.3.3	Object removal detection	3
1.4	Project Scope	4
1.5	Dissertation Organization	4
1.6	Contribution	5
1.7	Potential Application	6
1.8	Project Schedule	8
1.9	Chapter Summary	8
2.0	**LITERATURE REVIEW**	**9**
2.1	Introduction	9
2.2	Moving Object Detection	10
2.3	Object Tracking	13
2.4	Object Removal Detection	16
2.5	Chapter Summary	20
3.0	**SYSTEM DESCRIPTION**	**22**
3.1	Introduction	22
3.2	System Overview	22
3.2.1	Pre-process stage	23
3.2.2	Moving object extraction	25
3.2.3	Moving object tracking	26
3.2.4	Object removal detection	26
3.3	Hardware and Software Requirement	27
3.4	Chapter Summary	28

4.0 SYSTEM IMPLEMENTATION **29**
4.1 Introduction 29
4.2 Project Setting and Pre-Process 30
4.3 Moving Object Detection 33
4.3.1 Foreground Extraction 34
4.3.2 Shadow Removal 40
4.4 Moving Object Tracking 44
4.4.1 Kalman Filter 44
4.4.2 Motion Estimation 46
4.4.3 Object Matching 47
4.4.4 Tracker Update 49
4.5 Object Removal Detection 50
4.5.1 Abnormal Event Detection 52
4.5.2 Event Classification 54
4.5.3 Identify Suspected Person 57
4.6 Chapter Summary 59

5.0 SYSTEM TESTING **60**
5.1 Introduction 60
5.2 Testing Data Collection 60
5.3 Testing Result and Analysis 65
5.4 Chapter Summary 72

6.0 CONCLUSION **73**
6.1 Conclusion 73
6.2 Future Enhancement 74

REFERENCES **76**
APPENDIES **79**
A Detection Result for Experiment 1 79
B Processing Time of Experiment 82
C Sample Detection 85
D Detection Result for Experiment 2 88
E Detection Result for Experiment 3 91
F Publication – Object Occlusion and Object Removal Detection 94

LIST OF TABLES

Table		Page
3.1	Functionality of process	25
3.2	Minimum hardware requirements	27
3.3	Minimum software requirements	27
4.1	Pseudocode of the system	30
4.2	Pseudocode for pre-process stage	32
4.3	Pseudocode for moving object detection	34
4.4	Pseudocode of foreground extraction	34
4.5	Pseudocode of shadow removal	41
4.6	Pseudocode of moving object tracking	44
4.7	Pseudocode of motion estimation	47
4.8	Pseudocode of object matching	49
4.9	Pseudocode of tracker update	50
4.10	Pseudocode for object removal detection	51
4.11	Pseudocode for event detection	53
4.12	Pseudocode for event classification	56
4.13	Pseudocode of identify of suspected person	59
5.1	Event in simple scenarios	62
5.2	Event in medium scenarios	63
5.3	Event in complex scenarios	64
5.4	Summary of fine tuning	65
5.5	Summary of TPR, FPR and accuracy rate for test 1	68
5.6	Summary of TPR, FPR and accuracy rate for test 2	69
5.7	Summary of TPR, FPR and accuracy rate for test 3	69

LIST OF FIGURES

Figures		Page
1.1	Modules of the research project	4
1.2	Gantt chart of the project	7
3.1	Overview diagram	24
4.1	Example of running average with selectivity	31
4.2	Sample result of drawing OI process	33
4.3	Overview of foreground extraction	35
4.4	Example of R, G, B channel images	37
4.5	Example of summation result	38
4.6	Sample of thresholded image	39
4.7	Sample of combined image after opening operation	39
4.8	Final result of foreground extraction	40
4.9	Example of edge subtraction	42
4.10	Example of AND operation	42
4.11	Example of shadow removal	43
4.12	Example of motion estimation	47
4.13	Example of objects merge and split	48
4.14	Overview of object removal detection	51
4.15	Example of object region	52
4.16	Example of event detection	54
4.17	Example of outer region	55
4.18	Example of event classification	56
4.19	Process of identifying suspected person	58
4.20	Sample result of object removal module	58
5.1	Testing scene	61
5.2	Example of Threshold T_M Fine Tuning	66

5.3 Example of Threshold T_E Fine Tuning 66

5.4 Example of Threshold T_C Fine Tuning 67

5.5 Example of wrong detection 70

5.6 Example of multiple regions detection 71

5.7 Example of miss tracking 72

LIST OF ABBREVIATIONS

e.g.	For example
CCTV	Closed-circuit television
et al.	and others
RGB	Red, green, blue
OI	Object of interest
TP	True positive
FP	False positive
TPR	True positive rate
FPR	False negative rate
FPS	Frame per second

CHAPTER 1

INTRODUCTION

1.1 Introduction

Thievery has become a common issue in our society and it increases from time to time, regardless of any circumstances. Thus, the society becomes concerned on the thievery issue. To overcome this alarming issue, many researches are being carried out to improve the security.

The common way to address thievery is the use of the surveillance system in strategic places. Most of the surveillance system is monitored by security guards. Once thievery occurs, either the security guard or owner will get an alert. However, it may not be the best practice due to some human limitations. For instance, security guards may not be able to monitor the scene continuously for a long time.

In order to increase the security level, we can have some methods or algorithms that can automate thievery detection. In this research work, we propose a method to detect thievery. This method will be able to detect object removal and object occlusion event that happen in the video scene.

1.2 Problem Statements

The use of surveillance system to detect any moving object in the scene is common. However, surveillance system lacks the functionalities to keep track of these objects.

Besides that, surveillance system also lack of the ability to detect stealing/ burglary. There are some existing research methods to detect stealing/ burglary, but in some situations, when a person is blocking an object in a particular scene, it is possible that the system assumes the blocking as a burglary due to the disappearance of the object.

1.3 Project Objectives

This research project is to detect object removal and occlusion event. In addition, it focuses on moving object detection and object tracking as well. Hence, the project objectives are:

1.3.1 Automate the detection of moving objects

Prior to the tracking of any objects of interests, any moving objects need to be detected. The objective here is to perform automatic detection on these objects.

1.3.2 Automate the tracking of moving objects

Once the moving objects have been detected, their moving directions and locations have to be determined. Hence, tracking of the moving path needs to be performed.

1.3.3 Object removal detection

Once all the moving objects have been detected and tracked for their moving directions, it is crucial to identify and determine if the objects of interests are removed from some locations. Nevertheless, disappearance of objects from some locations could happen in two scenarios:

- The objects are indeed removed
- The objects could have been blocked by other objects

Thus, the objective here is to detect occlusion and removal instead of just detecting a particular object that is only removed.

1.4 Project Scope

The research project can be divided into three modules as shown in Figure 1.1. The first module is to detect the presence of any moving object in the video scene. For second module, tracking of these objects are done here to determine their respective moving directions. The last module detects any objects that are being removed or occluded.

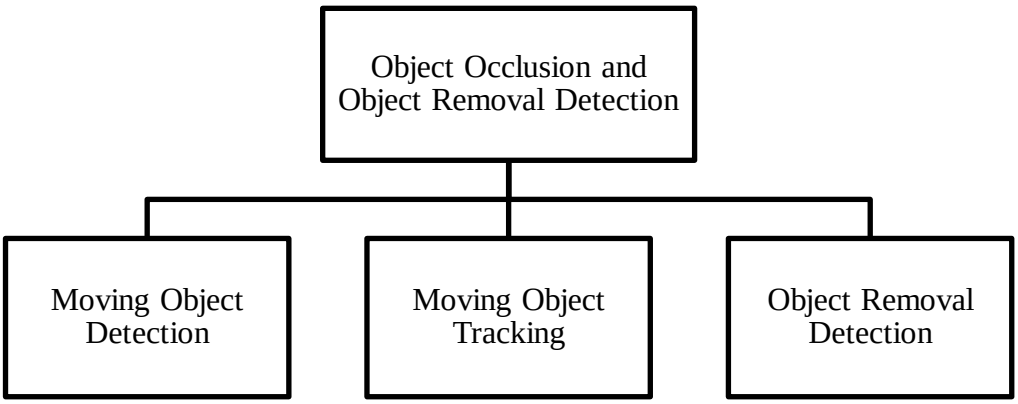


Figure 1.1: Modules of the Research Project

1.5 Dissertation Organization

This dissertation consists of six chapters. Chapter 1 presents the introduction, project background, problem statements, and objectives of the research work. Besides, it includes project scope, dissertation organization, contribution, potential application and schedule of the project.

Chapter 2 consists of reviews on some research papers that are related to the research work. All the reviewed papers can be divided into three parts, which are moving object detection, object tracking, and object removal.

Chapter 3 presents the system design of the project. This chapter describes the methodology. A general information, flow charts and system diagram are shown to provide an overview of the system. Besides that, hardware and software requirements are also furnished in this chapter.

Chapter 4 describes the system implementation. It explains how all modules in the system are implemented. Pseudocode is included to describe the algorithm that has been applied in the program. Some sample images are included to illustrate how each module work.

Chapter 5 focuses on the testing stage. It presents the explanation of the testing data collection, parameter tuning, testing results and analysis. Lastly, Chapter 6 consists of the conclusion of the entire project. Besides, future enhancements are briefly explained in this chapter.

1.6 Contribution

In the past researches, the general way to detect object removal event is by finding out static region and determining whether the object is removed or not. In some situations, occlusion event can also give some interesting

information to analyse events in video. In this project, the detection of object removal or occlusion event that happen on static object in the scene is expected. The event detection can allow the system to have a better abnormal event detection.

When the event detection on static object is mature enough, we can modify the algorithm slightly to apply on moving object. We can use the idea of object removal detection to do moving object tracking. For the occlusion event detection, it can identify occlusion between moving objects and it can tell the relationship on the moving objects, for example, A occluding B. This can be helpful for other analysis on computer vision.

Furthermore, the algorithm is also able to identify the person who removes or occludes an object. It can be further used on other research, such as for identifying the owner of the object or the thief.

1.7 Potential Application

There are some applications that are suitable to be applied on this project, such as snatch thief system. The developed system can be used in several areas such as offices, residential areas, museums and art galleries. The developed system can help security guard to minimize the thefts of valuable belongings or items.

Task	2011			2012		2013	2014			2015
	Mar – May	Jun – Aug	Sep – Dec	Jan – Apr	May–Dec	Jan – Dec	Jan – Mar	Mar – Aug	Sep – Dec	Jan – Feb
Research Introduction										
Literature Review										
Data Collection										
System Design										
System Implementation										
System Testing										
Finalize Dissertation										

Figure 1.2: **Gantt Chart of the Project**

1.8 Project Schedule

The project is divided into few milestones. The milestones of this project are shown in Figure 1.2 above.

1.9 Chapter Summary

In this research work, it further addresses the issues of detecting of any objects being removed from a particular scene. It is very useful to help detect thievery. The system consists of three modules including moving object detection, moving object tracking and object removal detection. The detailed system implementation is given in the next few chapters.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, literatures written by other researchers will be reviewed. From the literature reviews, we can know about current existing methods to solve problems and a better solution may be found to solve problems which cannot be solved by existing methods. This project is divided into three areas which are moving object detection, moving object tracking, and object removal detection. Therefore, the literature review is focused on these areas. The following sections in this chapter are constructed as below:

- Literature reviews on moving object detection
- Literature reviews on moving object tracking
- Literature reviews on object removal detection
- Conclusion on literature reviews

2.2 Moving Object Detection

Wang et al. (2008) presented three methods to model background from compressed video. These methods are running average, median, and mixture of Gaussians. Based on these background models in discrete cosine transform domain, the segmentation of moving objects can be done effectively. The running average algorithm gives a lower computational complexity, but same accuracy than median and mixture of Gaussians methods by using mathematically proven in the experiment. The discrete cosine transform domain is used as coefficients in the background model to segment moving objects, and it can give a better result than spatial domain.

Guan (2009) proposed an algorithm to extract moving objects based on multi-scale wavelet transform. This algorithm can choose a suitable threshold value automatically. The shadow can be suppressed by using temporal motion analysis on saturation component of hue-saturation-value colour space. Background update has also been done by analysing and performing algorithm on the chromatic characteristic of the previous frame and the current frame. For evaluation, receiver operating characteristic is used to compare the performance of the proposed algorithm with graph cut method and improved Gaussian mixture model, and the result shows that the proposed algorithm has higher true positive rate.

Liu et al. (2010) introduced another method to detect the foreground object of video with shape and colour information. This method contains two

steps which are shape-based background subtraction and colour-based background subtraction. Shape-based background subtraction uses shape context descriptor, nearest neighbour algorithm, and polygonal approximation algorithm to determine foreground regions. All pixels outside the foreground regions are classified as background. Other pixels need to be judged by colour-based background subtraction. For colour-based background subtraction, modified Gaussian mixture model is used to perform background subtraction. This method was compared with Gaussian mixture model and local binary pattern algorithm on five video sequences. The proposed method had a lower false positive rate in average and the false negative result is comparable with other two algorithms.

The next technique was introduced by Wang and Yung (2010). This method extracts moving objects from background based on multiple threshold and boundary evaluation. This method requires a background image and an input image. The differences between images are determined based on texture, luminance and chrominance. Three thresholds are selected using triangle algorithm, the object's boundary and threshold masks can be calculated from the thresholds. Evaluation is done by using information from objects' boundary and threshold masks. In this stage, the shadow of objects is also removed. Boundaries of objects are refined to get a more accurate result. Some experiments are conducted to do a comparison with other existing methods: minimum description length, shading model, derivative model, Li’s texture-based approach and single-threshold. The experiment result shows that this method gave the lowest error rate under different illumination conditions.

Lee et al. (2010) proposed a real time moving object detection method under global various illumination conditions. This method does not require users to change parameters manually as it used level set based bimodal segmentation algorithm to decide the threshold automatically. Temporal and spatial filter is used to eliminate spurious regions due to noise. A comparison was done between the method and other existing methods which are histograms of oriented gradients, mixture of Gaussians, Argawal's method. The result shows that this method can eliminate more noises than other methods.

Bhaskar et al. (2010) used extended cluster background subtraction with symmetric alpha-stable mixture model to detect the moving object. Symmetric alpha-stable distribution can solve problem of slight movements of background, camera shakes, and clutter noise. The algorithm was compared with Gaussian mixture model based cluster background subtraction, and Li's algorithm. The result proved that the algorithm provided a more efficient performance by precision, recall and *S*-ratios.

In the same year, Chiu et al. (2010) introduced a technique that can segment objects by using probability-based background extraction algorithm. The method uses colour image instead of greyscale image to extract initial background to obtain better performance. The total frames that needed to construct the background completely were determined automatically by the proposed algorithm. Object segmentation and background update can be done through colour segmentation, connect-component labelling, shadow removal,

hollow filling, noise deletion and background updating. Moving objects can be segmented by using colour segmentation, and connect-component labelling. Shadow removal algorithm also applied to suppress shadow of moving objects. Hollow filling and noise deletion were used after the shadow had been removed. The background image is sequentially updated periodically by using a background update algorithm.

2.3 Object Tracking

For moving object tracking module, Leibe et al. (2008) presented a system that can complete multiple objects detection and trajectory estimation. This system is able to recover trajectories from temporary lost track and mismatch by using minimum description length hypothesis selection. It can do multiple category object recognition. Trajectory estimation was also applied in this method to track objects in the scene. The method can focus on cars and pedestrians on present experiment and the method can still be extended to other object categories.

Li and Bian (2008) introduced the object detection and tracking by combining Gaussian mixture model, principal component analysis, and expectation maximization Kalman filter. Principal component analysis was used to extract features from video, and then the tracked object feature is described by using Gaussian mixture model. Objects were detected by posterior estimation. Moving objects can be detected by checking intensive

different value of consecutive frames. Expectation maximization Kalman filter algorithm was used to predict the position of the targeted object. From the experiment, it showed that the method gave a better result than classic mean shift algorithm under conditions where objects were crossover, partial occlusion and change of moving speed where targeted object and background has high contrast.

Hong et al. (2009) proposed real time moving object tracking using low computational complexity level-set-based contour tracking and foreground extraction method. The shadows and noises were eliminated in foreground extraction by analysing RGB colour space information. The foreground extraction can get the coarse contour of the object. The moving objects were tracked and inserted into a list and using level set function to track it. From the result of experiment, it showed that the method gave a faster processing time and a more accurate result than traditional level-set method.

Zeng et al. (2009) also introduced a tracking method by using point matching estimation on Kalman filter. Moving object extraction was done by using adaptive Gaussian mixture models. Location and speed of moving object were selected as features for object tracking. Kalman filter uses the features above to predict the location of moving objects. The corner of each moving object was obtained using Harris corner detector. Corner point matched between the current image and previous image was done to track moving objects.

Pathan et al. (2009) proposed Kalman filter tracking on traffic. The method used centre point of a moving object as the feature of Kalman filter. Correlation-Weighted Histogram Intersection method was introduced to assist Kalman tracker to keep track moving object under occlusion and separation. The hue and saturation of image were extracted in the method in order to keep track moving object. This method was tested in several environments and it showed that the method gave a satisfy result.

Zhang et al. (2010) proposed multiple objects tracking using species-based annealed Gaussion particle swarm optimization algorithm to deal with multiple objects occlusion in scene. In this algorithm, it divided the global swarm into many species based on number of objects in the scene. Each swarm species search for its target object and keep track of the targeted object. The occlusion between objects was modelled as species competition and repulsion which implicitly reduce the power of each swarm species. From the experimental result, it showed the algorithm was more efficient and more effective than the previous works which had been done by other researchers.

Khan and Gu (2010) also provided a novel object tracking scheme by using joint feature correspondences and object appearance similarity. For joint feature correspondences part, the feature points were estimated by using scale-invariant feature transform descriptor, and then random sample consensus was used to estimate consensus correspondences points for foreground and background. For object appearance similarity part, enhanced anisotropic mean shift was applied and it was guided by the area centre generated by a joint

feature correspondences part. An optimal selection criterion was applied to final tracker based on the results from the previous parts.

Li et al. (2010) also presented a multiple object tracking method using Kalman filter. Centroid, size of tracking window, and speed of centroid and tracking window were chosen as feature of moving objects. Some algorithms were deployed to match the related features between previous image and the current image. Eigenvalue was used to solve occlusion problems in the scene.

2.4 Object Removal Detection

Spagnolo et al. (2006) proposed an algorithm that is able to detect abandoned or removed objects by using edge information. The background image was built and updated automatically by temporal statistical analysis. Moving objects without shadows and noise can be extracted by background subtraction, shadow removing algorithm. The shape of moving objects is extracted to get segmented image. Abandoned or removed objects can be detected by matching the shape of the detected static foreground objects in the current image and segmented image. The algorithm can decide a moving object as abandoned, removed or ambiguous. The algorithm tested on some sequences in PETS 2006 public datasets, and the result shown this algorithm able to work well in the experiments.

Ferrando et al. (2006) presented a system that can classify unattended and stolen objects by using three level modules. The first level, mainly focuses on background update, identify pixel regions, blobs through bounding box's coordinates. Second level extracts colour feature from image and performs object tracking. Third level aims to detect suspicious events using colour information. At this level, a split between moving objects can be classified into one “human” class and one “non-human” class. For “non-human” class, it can be either static or dynamic. If the moving object was a static “non-human” class, an algorithm was applied to change detection image, background image and the current image to determine the event was either abandoned or removed.

Tian et al. (2010) introduced a framework to detect abandoned and removed objects in complex surveillance videos. Three Gaussian mixtures were used to model background and detect static region. The static region was moving blob that static in the scene for a relatively long time. The use of three Gaussian mixtures can help the system to push the static region into the background, to reduce the static region fragment and to adjust the background model updating rate. After the static region was defined, region growing method was used in the background image and the current image on the static region part. From region growing method, a removed object can be found if the segmented region in background image was larger than the segmented region in the current image. When the segmented regions had a similar size, the algorithm is not be able to decide the static region was abandoned or removed, and it concluded the static region in an unclear condition. A time range was set at the detection to avoid occlusion event mistakenly treated as abandoned or

removed event. Conclusion of event only is given when the total time of static region is greater than the threshold set by the user.

San Miguel and Martínez (2008) presented an approach to detect unattended object and stolen object based on the result from fusing three simple detectors. Moving objects were extracted after noise filtering by using mathematical morphology and background update based on a mixture between average and running average. The moving object is kept track based on different information such as colour, distance and object size. When moving objects remain static at the scene for a long time, the objects are classified as human or non-human. If a moving object is classified as static and non-human object, it is either an abandoned object or a removed object. Three detectors are applied in this phase, namely low-gradient detector, high-gradient detector and colour histogram detector. Active contour function is used to adjust the object shape before gradient detectors check gradient similarity. High-gradient detector finds high-gradient value point along the object in the current image, while low-gradient detector finds low-gradient value point. Colour histogram detector gets a colour similarity measurement for static moving object. The calculation is applied on the detectors to obtain two values for each detector. The last step is fusing these six values into the formula, and the detection of abandoned or removed object is obtained.

Li et al. (2009) presented a method that can detect objects from surveillance system based on colour richness. The method constructs two background models. The first background is used to generate a foreground

mask, while second background generates stationary mask. A static region confidence map can be created based on these two masks. Thresholding technique is used on the static region confidence map to identify static regions in the scene. Once static region is obtained, it is classified to either abandoned object or removed object using algorithm based on the colour richness between the current image and background image.

Wang and Liu (2010) presented a method for abandoned and stolen object detection in real time. This method involves three steps. Two background models are constructed and updated using different update rates. The moving objects are extracted from background by using these background models. The extracted moving objects are classified as either moving objects or static objects. Abandoned or stolen objects can be detected from background images and static image by using a decision making model based on colour information. This method is tested on several sequences from AVSS2007 public data sets and various environments. This method successfully detects most of the abandoned or removed objects in the simple video scene.

Caro Campos et al. (2011) used active contour to decide whether a static object is an abandoned object or a removed object. When a moving object is static in the scene, the boundary of the static object is extracted. Contour adjustment using active contour is applied to the current image and background image to targeted moving object. For removed object after contour adjustment, the contour in background image is expected to similar to initial contour, while the contour in current image may shrink or disappear. They also

proposed using different active contour algorithms to solve the problem. The selected active contour algorithms are parametric, geometric region-based and geometric edge-based. The proposed active contour algorithms are compared with some existing methods and the result shows that the geometric edge-based algorithm gives the best performance.

SanMiguel et al (2012) provided a pixel based colour contrast approach to identify abandoned and removed objects. In this method, the colour contrast of pixel inside and outside the boundary of a moving object is compared in the current image and background image. To identify removed objects, the colour contrast in background image need to be high, while the colour contrast in current image is low enough.

2.5 Chapter Summary

The literature review above shows existing methodologies applied for moving object detection, moving object tracking and object removal detection. For moving object detection, there are few ways to detect moving object which are colour, shape, texture, luminance, and etc. Besides that, shadow removal and background update are also recommended to be used, as they can improve accuracy and reduce noises. For object tracking methods, the Kalman filter is one of the popular methods. It can give a good prediction to track object's location. Other methods also include Gaussian mixture model, principle component analysis, level-set-based contour tracking, and swarm optimization

algorithm. For object removal detection, it can be done by using contour based, region growing, colour, and gradient. Although there are few methods to do the above mentioned detection, the most important thing is to review past researches and to come with a method that can deal with as many environments as possible and enhance performance in term of accuracy, consumed time, and other factors. After consideration of pros and cons of all existing methods, we have decided a suitable method for us to build the system. It is discussed in the following chapters.

CHAPTER 3

SYSTEM DESCRIPTION

3.1 Introduction

In this chapter, we will describe the system overview and methodology of this project. From literature reviews in the previous chapter, we constructed the design of the system. The system overview section below highlights the system overview and general description of each module of the project. Software and hardware requirements are also described in this chapter.

3.2 System Overview

Based on literature reviews and researches in the previous chapter, we decided to utilize the system into three modules, namely moving object extraction, moving object tracking and object removal detection. Besides that, the system also contains pre-process stage. The overview diagram of the project is shown in Figure 3.1.

The input of the system is image sequences or video. When the system receives the input, it identifies all objects of interest in the scene. It also constructs a background model and the background model is updated continuously. The background model and current image are used to extract

moving object. The system keeps track of the moving object until it left the video scene. Objects of interest which are defined at the beginning of the video is keep tracked to determine whether it is taken or occluded.

In pre-process, there is one process, named P1. In P1, it constructs a background model and gets information about the object of interest (OI). After pre-process, it is the moving object extraction module. In this module, there are two processes, namely P2 and P3. P4, P5 and P6 are processes in moving object tracking module. For object removal detection module, it consists of process P7, P8, and P9. The functionality of all processes is shown in Table 3.1. The description of all modules is explained in following subsections.

3.2.1 Pre-process stage

In this stage, the background model is created and the information about OI is collected. The information collected about OI includes the location and size of the OI.

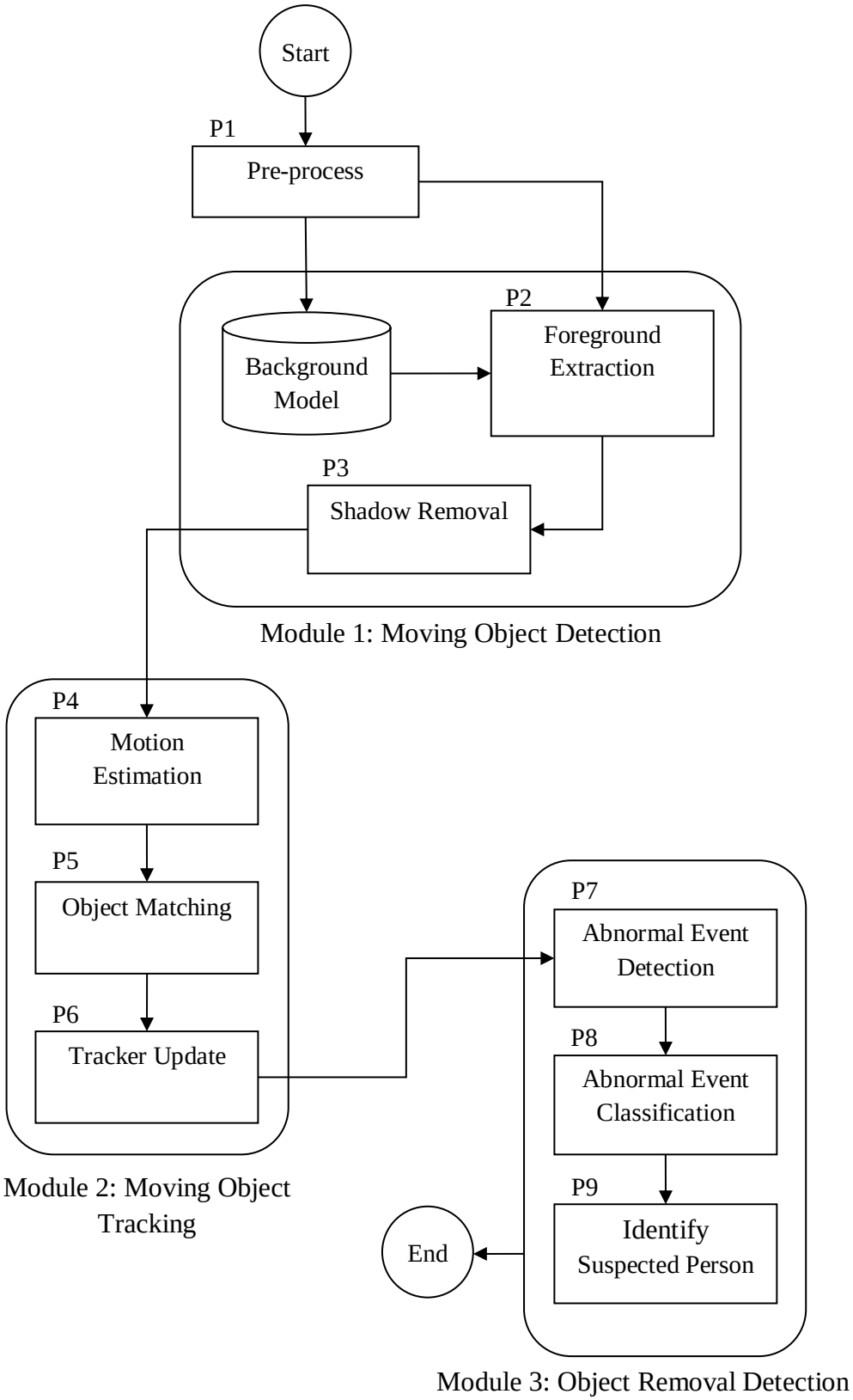


Figure 3.1: Overview Diagram

Table 3.1: Functionality of Processes

Process Symbol	Process Name	Description
P1	Pre-process	Background model is initialized. The information of all OI is set.
P2	Foreground extraction	Moving object extraction is done by using colour information.
P3	Shadow removal	Shadow and noise are removed.
P4	Motion estimation	Location of moving objects are predicted by trackers.
P5	Object matching	The moving object is matched with predicted location.
P6	Tracker update	Using actual location to update tracker.
P7	Abnormal event detection	Determine any abnormal event happen.
P8	Abnormal event classification	Identify the abnormal event that is happening.
P9	Identify suspected person	Identify person who triggers an abnormal event.

3.2.2 Moving object extraction

This module aims to detect all moving objects that appear in the video scene. The first process of this module, P2, uses the colour information to detect moving object in the scene. This can be achieved by determining the colour value different of every pixel for current image and background model; all moving objects in the scene are drawn by using rectangular box. The pixel with big different colour value is considered as moving object pixels. For P3, this process first extracts the shape of moving objects. It takes edges detected

in current image to subtract the edges detected in background model. After that, it integrates the information from P2 to remove noise in the scene.

3.2.3 Moving object tracking

This module uses trackers to keep track and locate all moving objects inside the scene. Each tracker only keeps track of single moving object. In P4, tracker predicts the location of moving objects. For P5, the tracker uses predicted information to match its belonging moving object. For P6, the tracker uses actual information of its belonging moving object to update its parameter for next prediction. The tracker keep track and updates the location of moving object until the moving object has left the scene.

3.2.4 Object removal detection

This module detects event that happens on OI. The possible events that can happen are no event, object occlusion and object removal. The module first runs process P7, which determines whether an abnormal event is happening on the OI. When an abnormal event is detected in P7, P8 classifies the event as either occlusion or removal. When the event is determined, the next process, P9 finds out which moving object removes or occludes the OI.

3.3 Hardware and Software Requirement

Table 3.2 shows the minimum hardware requirements for project implementation while Table 3.3 shows minimum software requirements.

Table 3.2: Minimum Hardware Requirements

Hardware	Description
Intel Core 2 Duo or higher	This is the minimum requirement to run all software needed.
2GB RAM or higher	Many images are needed to run the process, thus there is a need for more memory in the computer.
Keyboard and Mouse	These are the basic input devices that are required for user to send signal to the computer.
Monitor	It is the primary output device to display the result of the project.
Camera	It captures the videos for training and testing of the project.

Table 3.3: Minimum Software Requirements

Software	Description
Windows XP	A recommended operating system that is more suitable for Microsoft Visual Studio to implement the project.
Microsoft Visual Studio 2008	Software that is useful to build and compile the system.
OpenCV 2.0 or above	A library that provides lots of functions for image processing.
WinX HD Video Converter	Software that can convert video from camera to a format that is supported by

	OpenCV library.

3.4 Chapter Summary

In this chapter, general overview of the project is discussed. Moving object extraction is the first module of the project. It uses two processes to detect moving object in the scene. These processes are foreground extraction and shadow removal. For second module, moving object tracking module, it keeps track moving object that detected in the first module. The processes in tracking module are motion estimation, object matching and tracker update. The last module of the project is object removal detection. It uses three processes to detect happening event, namely abnormal event detection, abnormal event classification and suspected person identification. Besides overview, this chapter also describes about software requirements and hardware requirements for the implementation. The detailed implementation is described in the next chapter.

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 Introduction

In this chapter, the system implementation of this project will be discussed. This chapter also describes the implementation of the modules in this project. Besides that, it discusses about the project setting and pre-process in the system. Table 4.1 shows the pseudocode of the system. In the table, Moving_Object_Detection, Object_Tracking and Object_Removal_Detection are the functions for all modules. The details of each module are discussed in following section. Below shows the organization of this chapter:

- Project setting and pre-process
- Methodology of moving object detection
- Methodology of moving object tracking
- Methodology of object removal detection
- Chapter summary

Table 4.1: Pseudocode of the System

```
Object_removal_detection
    Load all setting value
    Load video
    Load first_frame
    OI_list = Pre-process(first_frame, background)
    WHILE video is running
        Load current_image
        Update background
        moving_box_list = Moving_Object_Detection
                                (current_image, background)
        tracking_list  = Object_Tracking(current_image, moving_box_list)
        event_list = Object_Removal_Detection(current_image, background,
                                                    OI_list, tracking_list)
    ENDWHILE
STOP
```

4.2 Project Setting and Pre-Process

In this project, the system can load video or image sequences. There are some processes that need to be done when the video first runs. First, the first frame of the video can be used as an initial background model. This background model is updated continuously by using running average with selectivity. The system updates all pixels in current image which is not marked as foreground pixels. Equation for using running average with selectivity is shown in the following equation:

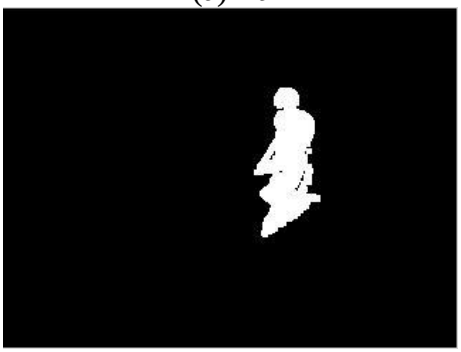
$$B_{t+1}(x,y) = \begin{cases} \propto C_t(x,y) + (1-\propto)B_t(x,y) if\ C_t(x,y) is\ background \\ B_t(x,y), otherwise \end{cases} \quad (4.1)$$

where B_t is the background model at time *t*, C_t is the entire current image at time *t*, and $\propto$ is learning rate which set by the user.

Figure 4.1 shows a sample of running average with selectivity. (a) and (b) are background and current image respectively. (c) is the foreground mask detected by the system. In (d), all white pixels in foreground mask do not update into background model.

(a) Bt

(b) Ct



(c) Foreground mask

(d) Part that is updated into B_{t+1}

Figure 4.1: Example of Running Average with Selectivity

In the video scene, there are some targeted objects that are the focus for the system to do event detection, named as object of interest (OI). OI is the static object in the scene and our goal is to detect object removal on this object. All OI are placed inside the scene when the video is started. Information of OI are extracted including the rectangular box that bound exactly on the OI, the location of the box, and its size. The information of the box of OI can be obtained by either reading from file or by letting the user to draw the boxes in video scene. In real situation, OI box's information can be obtained from object abandon detection.

After information of OI is known, the system assigns a name to each OI by using capital letter (e.g.: A, B, AA, AB and etc.). Table 4.2 shows the pseudocode for pre-process stage.

Table 4.2: Pseudocode for Pre-Process Stage

```
Pre-process(current_image, background)
    Declare OI_list
    background = current_image
    IF OI coordinates file exists THEN
        OI_list = information from the file
    ELSE
        User draw a rectangular box for OI
        OI_list = rectangular box
    ENDIF
RETURN with OI_list
```

Figure 4.2: Sample Result of Drawing OI Process

4.3 Moving Object Detection

When the video is running, there are always some moving objects the scene. The task of this module is to detect all moving objects and draw rectangle box to mark them out. This module consists of two processes: Foreground extraction and shadow removal. Foreground extraction module extracts moving objects in the scene. For shadow removal, shape of the moving object is extracted to remove noise in the image. Table 4.3 shows the pseudocode of this module. All processes of this module are discussed in following subsections.

Table 4.3: Pseudocode for Moving Object Detection

```
Moving_Object_Detection(current_image, background)
    Declare moving_object_list
    Declare foreground_result, moving_object_list
    foreground_result = Foreground_extraction(current_image, background)
    moving_object_list = Shadow_removal(current_image,
                                        background, foreground_result)
RETURN with moving_object_list
```

4.3.1 Foreground Extraction

This method uses RGB colour space image instead of greyscale image. This is because it can obtain better performance but computational complexity is not substantially greater (Chiu et al., 2010). Figure 4.3 shows the overview of this method, while Table 4.4 is the pseudocode of foreground extraction.

Table 4.4: Pseudocode of Foreground Extraction

```
Foreground_extraction(current_image, background)
    Split current_image into red, green, blue channels
    Split background into red, green, blue channels
    Background subtraction on red channel
    Background subtraction on green channel
    Background subtraction on blue channel
    result = combine red, green and blue channels
    Thresholding technique on result
    Morphology opening operation on the result
RETURN with result
```

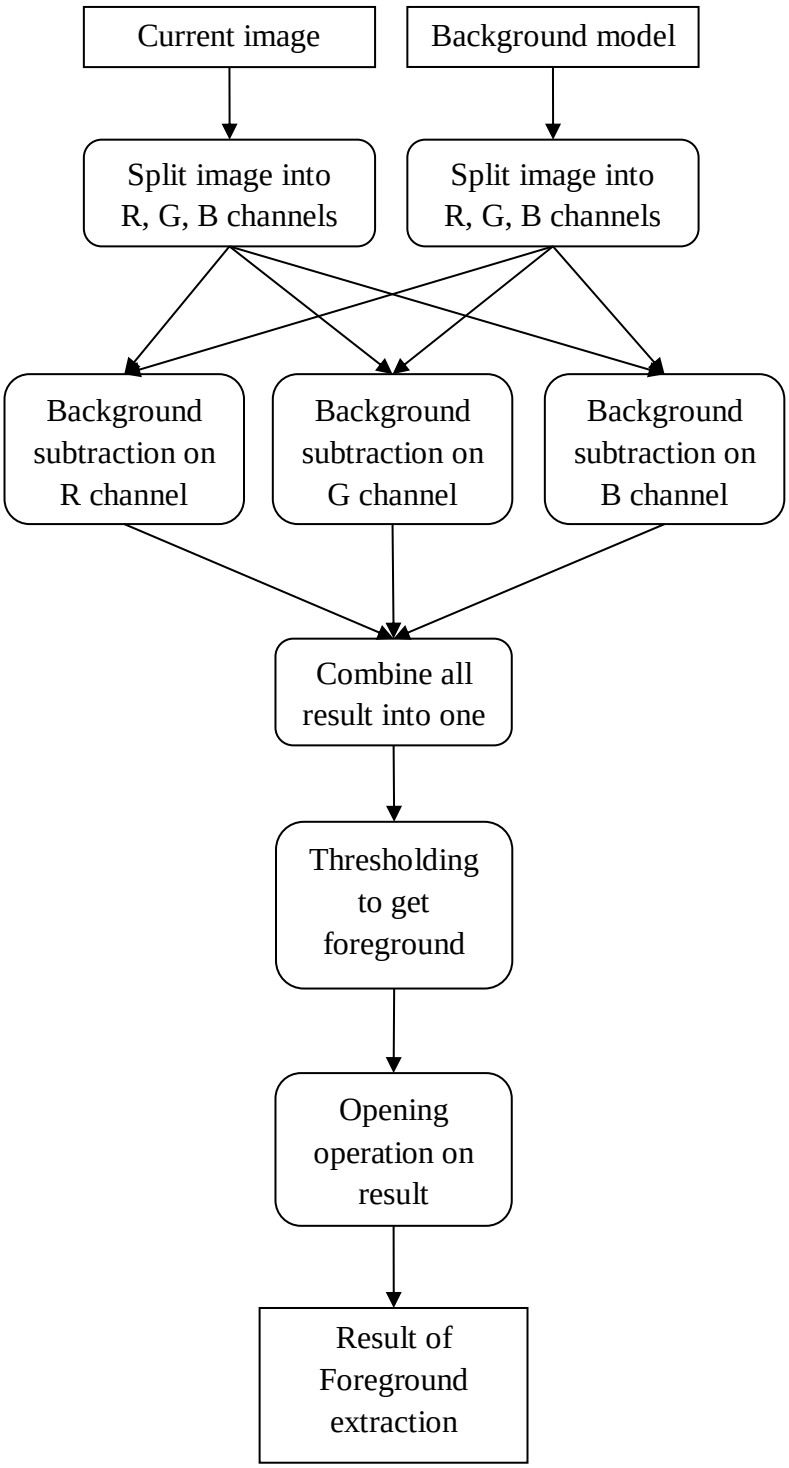


Figure 4.3: **Overview of Foreground Extraction**

Firstly, the current image is split into three channels which are red, green and blue. The same method is also applied to the background model. Next, background subtraction technique is applied to each channel of the image and three result images come out in greyscale form. Figure 4.4 displays an example of RGB channels for current image and background. Equations 4.2 to 4.4 are the background subtraction equation for R, G, and B channels.

$$R(x,y) = |C_R(x,y) - B_R(x,y)| \tag{4.2}$$

$$G(x,y) = |C_G(x,y) - B_G(x,y)| \tag{4.3}$$

$$B(x,y) = |C_B(x,y) - B_B(x,y)| \tag{4.4}$$

The next step, all result images are combined into one image. The system sums up all result images into one. The maximum value of each pixel in the sum result image is 255, any summation result that's bigger than the maximum value is set to 255. Equation 4.5 is equation of summation. Figure 4.5 shows an example of results summation.

$$Sum(x,y) = \begin{cases} 255 \; if \; R(x,y) + G(x,y) + B(x,y) > 255 \\ R(x,y) + G(x,y) + B(x,y), otherwise \end{cases} \tag{4.5}$$

(a) Current image

(b) Background

(c) R channel of current image

(d) R channel of background

(e) G channel of current image

(f) G channel of background

(g) B channel of current image

(h) B channel of background

Figure 4.4: Example of R, G, B Channel Images

(a) R result

(b) G result

(c) B result

(d) sum result

Figure 4.5: Example of Summation Result

Then, the sum result image is applied with thresholding by using equation 4.6. The result from equation 4.6 is an image that only contains black and white colour, white part in the image refers to moving object while black part is the background. The value range for T_M is from 0 to 255. Figure 4.6 displays a sample of thresholded image.

$$\mathrm{dst}(x,y) = \begin{cases} 255 & \text{if } \mathrm{sum}(x,y) > T_M \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

Figure 4.6: Sample of Thresholded Image

The final step is using opening operation (Erode before dilate) in mathematical morphology on the combined image to obtain the final result. The dilation operation is slightly greater than erosion, so that the system can combine some of the separated part together. The opening operation can make some small noises eliminated from the image. Figure 4.7 displays the sample after opening operation image.

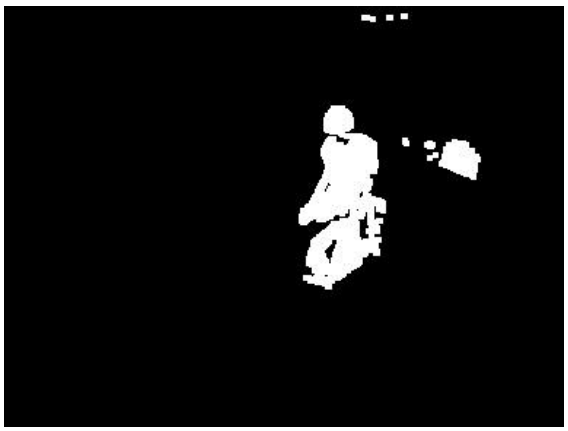


Figure 4.7: Sample of Combined Image after Opening Operation

After this step, all the white parts that are smaller than the minimum size are treated as noises and are ignored. The minimum size is set by the user. Then, all the white parts are considered as a moving object, the system draws a blue rectangular box on the moving object on the current image to show the moving object is detected. Figure 4.8 shows a sample result of foreground extraction. This result may not be that precise as colour is easy to be affected by noises which cannot be solved by using colour information such as lighting inconsistence, shadows and etc.

(a) Result in colour image (b) Result in binary image

Figure 4.8: Final Result of Foreground Extraction

4.3.2 Shadow Removal

Since the natural characteristic of the colour information is sensitive to illumination changes, thus some other information is needed to help increase the performance of moving object detection. Liu et al. (2010) shows that shape information has good capability of tolerance against illumination changes. Thus, we use shape information to help eliminate shadows and noises in the scene after foreground extraction process. Canny edge detector is chosen as shape descriptor. The reason to use Canny algorithm is this algorithm had

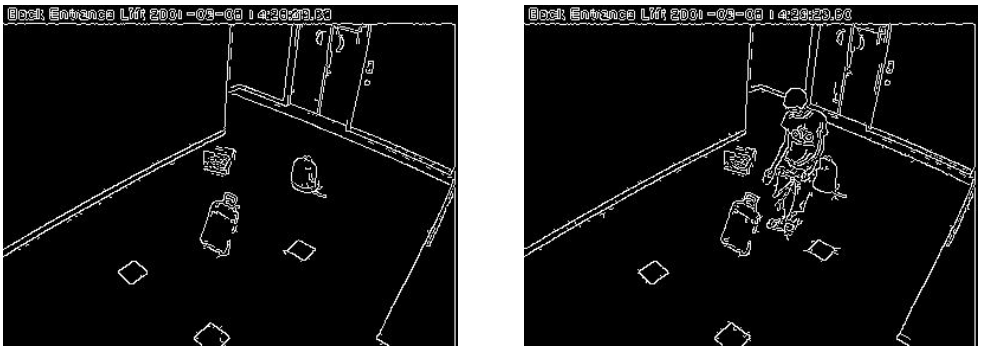
better performance and better robustness in convergence and it is one of the best faster-executing algorithm (Shin et al., 2001). Maini and Addarwal (2009) also mentioned that Canny algorithm perform better than Sobel, Prewitt and Robert's operator under almost all scenarios. Besides that, Canny algorithm is preferred compared to other edge detector since it produces single pixel thick, continuous edges (Nadernejad et al., 2008). Table 4.5 shows the pseudocode of shadow removal.

Table 4.5: Pseudocode of Shadow Removal

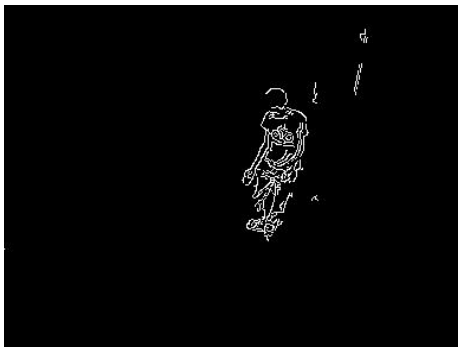
```
Shadow_removal(current_image, background, foreground_result)
    Get edge of current_image
    Get edge of background
    edge_result = current_edge - background_edge
    object_shape = edge_result AND foreground_result
    result = Adjust foreground_result base on object_shape
RETURN with result
```

The first step is using Canny shape descriptor to extract all edges in current image and background model. The second step uses extracted edge images of current image subtract with background edge image. Then, the result only contains the shape of moving objects. **Error! Reference source not found.** shows example of the subtraction. Equation 4.7 is the formula for edge subtraction.

$$\text{result}(x,y) = \begin{cases} 255 & \text{if } C(x,y) - B(x,y) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (4.7)$$

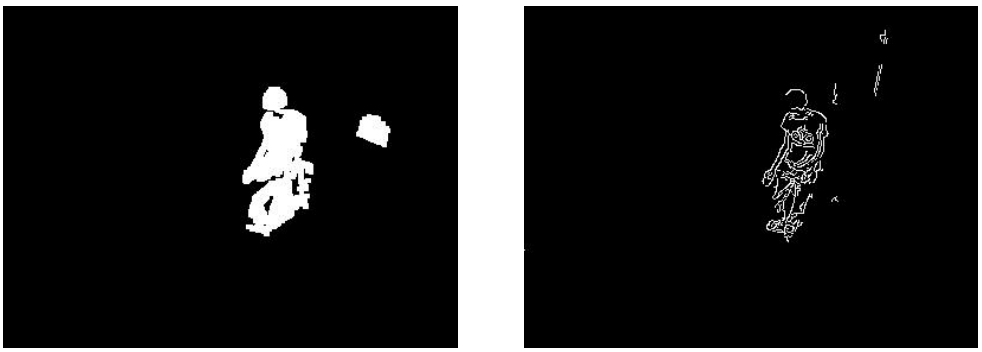


(a) Background (b) Current image

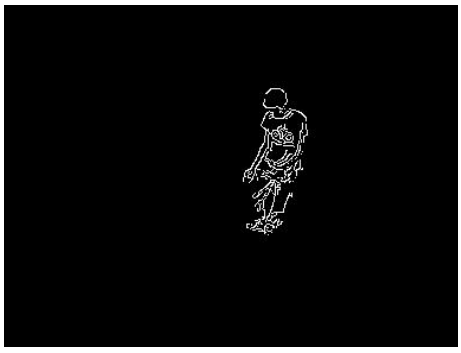


(c) Result

Figure 4.9: Example of Edge Subtraction



(a) Foreground extraction result (b) Edge subtraction result



(a) Final result

Figure 4.10: Example of AND Operation

The third step is to apply AND operation on edge subtraction and foreground result to get the shape of foreground objects. **Error! Reference source not found.** shows sample of AND operation.

The final step is to adjust the moving object boundary by using the AND result image. The system finds the most top and the most bottom y-axis values that consist of white pixel for each moving object box. Besides that, the most right and the most left x-axis values also are determined for each moving object box. After that, the system adjusts the moving object box boundary depending on top, bottom left and right value. If white pixel does not exist in the moving object box or the adjusted moving object box is too small, it is considered as a false detection and the moving object box is deleted from moving objects list. Figure 4.10 shows an example of before and after shadow removal process.

(a) Before shadow removal (b) After shadow removal

Figure 4.11: Example of Shadow Removal

4.4 Moving Object Tracking

After the previous module detected all moving objects in the scene, the information is passed to this module to keep track of moving object. The centre and the speed of the moving object box is chosen as tracking feature, and the feature is keep tracked by using Kalman filter. Basically, this module can be divided into three processes which are: motion estimation, object matching and tracker update. Table 4.6 is the pseudocode of this module. The details of Kalman tracker and sub-modules are explained in the following subsection.

Table 4.6: Pseudocode of Moving Object Tracking

```
Object_Tracking( current_image, moving_box_list)
        Declare tracking_list
        Estimate_location(moving_box_list, tracking_list)
        Macth_object(moving_box_list, tracking_list)
        Update_tracker(tracking_list)
RETURN with tracking_list
```

4.4.1 Kalman Filter

Kalman filter is an estimator that predicts targeted state. Kalman filter is widely used because of its simple, optimal and robust method (Zeng, et al., 2009). It is also an estimator that is efficient in practical and attractive. The optimal state can be found with the smallest possible variance error recursively (Li, et al., 2010). Kalman filter has two phases which are prediction and measurement update. These two phases are kept repeating after another phase.

The prediction phase has two equations. The first equation is used to predict value of the state estimate at a particular time, while the second one finds prior estimate error covariance. The equations are shown in Equation 4.8 and 4.9.

$$x_k^- = Ax_{k-1} + Bu_k \tag{4.8}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{4.9}$$

where x_k^- is prior state estimate at time *k*, A and B is transition matrix, x_{k-1} is posterior state estimate at time *k*-1, u_k is external control. P_k^- is priori estimate error covariance at time *k*, while P_{k-1} is posterior error covariance estimate at time *k*-1. *Q* is process noise covariance.

For measurement update phase, there are three equations. The objective of this phase is to give feedback on the accuracy of prediction phase and update predicted state to have a more accurate prediction for the next state. Equation 4.10 is to determine Kalman gain value at time *k*. Equation 4.11 is to obtain a posterior state estimate at time *k*, while Equation 4.12 finds out the posterior error covariance estimate at time *k*.

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{4.10}$$

$$x_k = x_k^- + K_k(Z_k - Hx_k^-) \tag{4.11}$$

$$P_k = (I - K_k H)P_k^- \tag{4.12}$$

where K_k is Kalman gain at time *k*, *H* is measurement matrix, *R* is measurement error covariance, Z_k is actual measurement at time *k*, and *I* is the identity matrix.

4.4.2 Motion Estimation

In this sub-module, the prediction phase of Kalman filter which are equation 4.8 and 4.9 is applied. The system state is a four dimensional matrix which contains x, y, dx, and dy. x and y are the location of the centroid of moving object in x-axis and y-axis, while dx and dy are the speed of the moving object. The moving object's feature is structured as:

$$x_k = \begin{bmatrix} x \\ y \\ dx \\ dy \end{bmatrix}$$

The transition matrix *A* is defined as:

$$A = \begin{bmatrix} 1 & 0 & t & 0 \\ 0 & 1 & 0 & t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where *t* is sample time.

Here, the system assumes there is no any external control that can affect the state estimation, so Bu_k is ignored in the equation. Process noise covariance, Q is set as a 4x4 identity matrix.

When a new moving object appears in the scene, a Kalman tracker is assigned to it and centroid of the moving object is initialized as the posterior state and posterior error covariance is set as a 4x4 identity matrix. After that, equation 4.8 and 4.9 are applied in order to predict the location of the moving object in the next frame. Table 4.7 shows the pseudocode of this process. Figure 4.12 shows an example of motion estimation. In the figure, white dot

inside moving object box is the actual location of centroid while black dot is centroid location estimated by tracker.

Table 4.7: Pseudocode of Motion Estimation

```
Estimate_location(moving_box_list, tracking_list)
    FOR moving_box  in moving_box_list DO
        centre_point = centroid of moving_box
        predicted_point = apply Equation 4.8 & 4.9 using centre_point
        Update tracker_list
    ENDFOR
RETURN
```

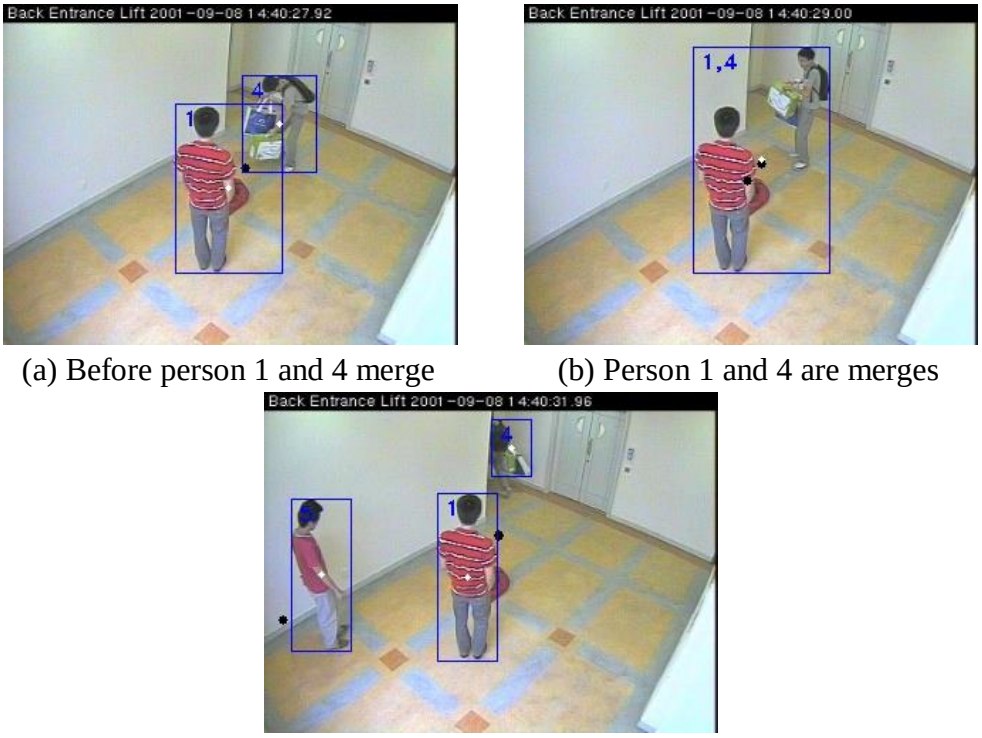
Figure 4.12: Example of Motion Estimation

4.4.3 Object Matching

After the Kalman tracker predicts the location of the moving object, the tracker finds relevant moving box in the scene. As the moving object in the scene is marked using rectangular box, the first step is to determine if the coordinates of the predicted centroid falls inside or nearby any moving boxes.

If the centroid is inside a specific moving object box, it means the tracker successfully tracks the moving object and the tracker saves the current moving object box information such as coordination, height, width, and colour histogram.

In a situation where two moving objects intersect or they are close enough, moving object detection detects them as one object. The related trackers find out their own target by using the predicted centroid and colour histogram. Predicted centroid continues to keep track merged object. Colour histogram is to help the trackers to determine the correct target when multiple moving objects are merged and split. Figure 4.13 displays a sample of multiple object merge and split.



(a) Before person 1 and 4 merge (b) Person 1 and 4 are merges

(c) After person 1 and 4 split

Figure 4.13: Example of Objects Merge and Split

If the predicted centroid cannot match any moving object box for a few frames, it means the moving object leaves the scene. When the moving object has left the scene, the tracker is removed from tracker list. Table 4.8 is the pseudocode of object matching.

Table 4.8: Pseudocode of Object Matching

```
Macth_object(moving_box_list, tracking_list)
    FOR tracker in tracking_list DO
        IF tracker prediction match moving_box  in moving_box_list THEN
            Update tracker
        ELSE
            Delete tracker from tracking_list
        ENDIF
    ENDFOR
RETURN
```

4.4.4 Tracker Update

After the tracker successfully track its moving object box, Equation 4.10, 4.11 and 4.12 are applied to update the parameter of Kalman tracker. In Equation 4.10, measurement matrix H and measurement error covariance R are set as below:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

After the parameter of the Kalman tracker is updated, the parameter is used as input for Kalman tracker prediction in the next frame. These processes are repeated until the moving object leaves the scene. Table 4.9 is the pseudocode of this process.

Table 4.9: Pseudocode of Tracker Update

```
Update_tracker(tracking_list)
    FOR tracker in tracking_list DO
        Apply Equation 4.5, 4.6 & 4.7 using actual centroid
        Update tracker
    ENDFOR
RETURN
```

4.5 Object Removal Detection

In this module, the system detects any abnormal event happening on OI. The event that can happen is either normal or abnormal. Normal event means nothing happens, while abnormal event can be divided into two: object removal and object occlusion. Object removal is defined as the OI is taken by someone from the original place and left the scene. Object occlusion is defined as the OI is occluded by moving objects.

This module contains three processes: Abnormal event detection, abnormal event classification and suspected person identification. Figure 4.14 illustrates the overview for this module.

Table 4.10: Pseudocode for Object Removal Detection

```
Object_Removal_Detection(current_image, background, OI_list,
                                        tracking_list )
    event_list  = event_detection(current_image, background, OI_list)
    IF abnormal event in event_list THEN
        classify_event(current_image, background,   OI_list, event_list)
        event_list_with_people =
            detect_person_trigger_event(OI_list, event_list, tracking_list)
    ENDIF
RETURN with event_list_with_people
```

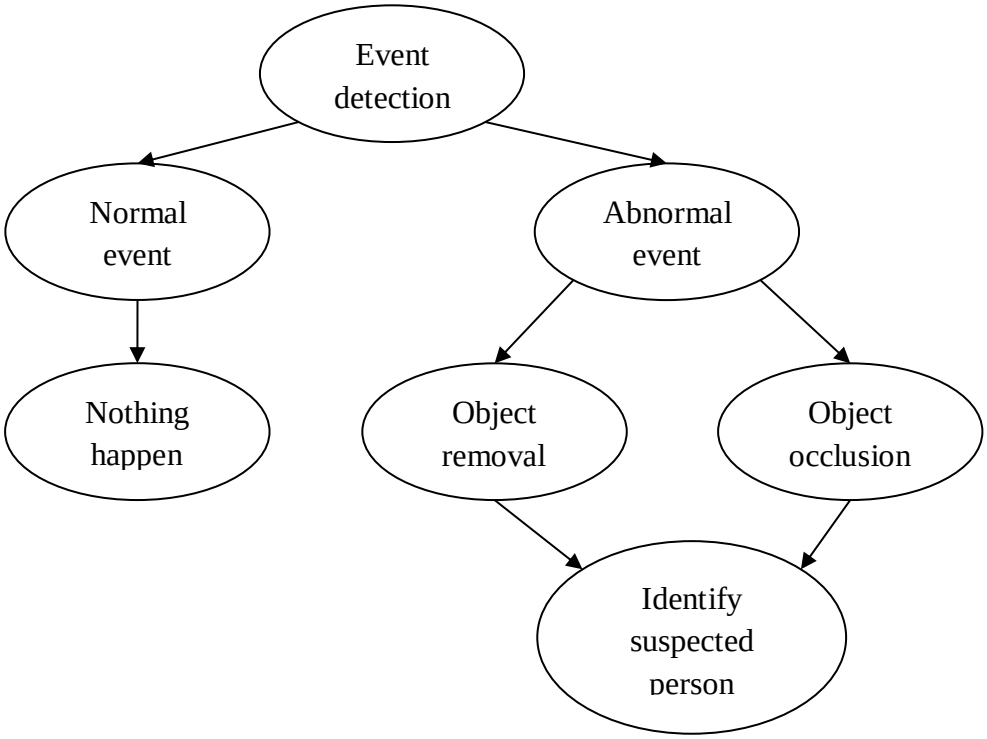


Figure 4.14: Overview of Object Removal Detection

4.5.1 Abnormal Event Detection

This process detects any event that happens on the OI in the scene. The event that can happen on the OI are normal or abnormal event. The edge information on OI is used to detect the happening event. First, we define region around OI as object region and this region is output from the pre-process module. Figure 4.15 shows an example of object region. The region inside red rectangular box is object region. The OI is from the output in pre-process stage.

(a) Original image (b) Object region

Figure 4.15: Example of Object Region

The abnormal event can be detected by comparing the edge similarity between background model and current image inside object region. When there is a normal event, the edge similarity for object region is always high, but when there is an abnormal event (object removal or object occlusion) happening on OI, the edge similarity for object region is low. The following paragraph explains the details of abnormal event detection.

First, the edge is extracted for each object region in the background. If a pixel is detected as edge, it is drawn as white pixel. For non-edge pixel, it is

drawn as black pixel. Next, the edge information is also generated for the object region in the current image.

After that, current edge mask and background edge mask are compared to check the edge similarity to detect happening event. In normal event, edge of the current image and the background is similar, so the difference between background edge and current image edge is smaller. For abnormal event, it is vice versa. The event detection can be done using the equation below.

$$R_E = \begin{cases} \text{Abnormal event, if } \dfrac{\sum |\mathrm{B}(x,y) - \mathrm{C}(x,y)|}{\sum \mathrm{B}(x,y)} > \mathrm{T}_E \\ \text{Normal event , otherwise} \end{cases} \quad (4.13)$$

where B is background edge mask, C is current edge mask, and T_E is threshold value set by the user.

Figure 4.16 shows an example of event detection. In figure(a), nothing happens and figure (b) shows OI is removed. Table 4.11 is the pseudocode for event detection.

Table 4.11: Pseudocode for Event Detection

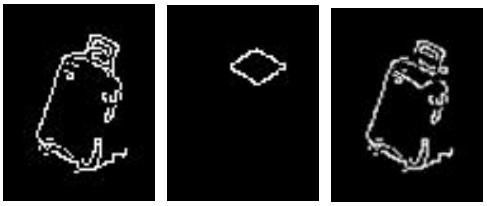
```
event_detection(current_image, background, OI_list)
    FOR object_region in OI_list DO
        B = Edge extracted from background
        C = Edge extracted from current_image
        event = Equation 4.13
        Update event into event_list
    ENDFOR
RETURN with event_list
```

(a) Normal event (b) Object removal event

Background current image

Background current image

B C R_E



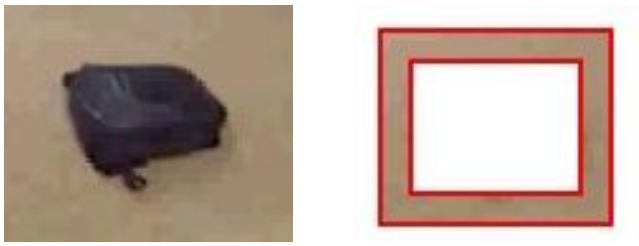
B C R_E

(c) Graphical result of (a) (d) Graphical result of (b)

Figure 4.16: Example of Event Detection

4.5.2 Event Classification

When the previous process detects abnormal event, the system needs to classify if the abnormal event happening on OI is either occlusion or removal. The classification can be done by comparing the outer region. Outer region is a rectangle region that has a bigger size than object region and this region is excluding from the region part that belongs to object region. The width and height of outer region are 20px bigger than object region. Figure 4.17 shows an example of outer region.



(a) Original image (b) Outer region

Figure 4.17: Example of Outer Region

First, the edge similarity between background model and the current image is determined for outer region. For occlusion event, the edge similarity of outer region is low. When comparing the difference between current edge and background edge, the result is smaller. For removal event, it is vice versa. The following equation shows the formula to classify object occlusion and object removal.

$$R_C = \begin{cases} \text{Object occlusion, if } \dfrac{\sum |\text{C`}(x,y) - \text{B`}(x,y)|}{\sum \text{C}(x,y)} > \text{T}_C \\ \text{Object removal, otherwise} \end{cases} \quad (4.14)$$

where B` is background edge on outer region, C` is current edge on outer region, and T_C, is the threshold set by the user.

Table 4.12 shows pseudocode for event classification module and Figure 4.18 illustrates an example for object removal and object occlusion.

Table 4.12: Pseudocode for Event Classification

```
classify_event(current_image, background, OI_list, event_list)
    FOR event in event_list is abnormal DO
        C`= edge information for outer region in current_image
        B`= edge information for outer region in background
        result = (C`-B`)/C`
        IF result > threshold THEN
            event = object occluded
        ELSE
            event = object removed
        ENDIF
        Update event into event_list
    ENDDO
RETURN
```

Current image Background

C` B` R_C

(a) Graphical result of object removal event

Current image Background

C` B` R_C

(b) Graphical result of object occlusion event

Figure 4.18: Example of Event Classification

4.5.3 Identify Suspected Person

After the system classifies the type of abnormal event happen on OI, the next step is determining the person who triggered the abnormal event. From moving object tracking module, the system is able to detect people who are moving around in the scene. When an abnormal event happens, the system checks any moving object that is intersected with the object.

If there is only a single moving object that intersects with OI, the system concludes that the moving object triggers the abnormal event. If there is multiple moving objects that intersect with the OI, the system first calculates the centre points of moving objects and OI. Then, the system finds out which moving object is the nearest to the OI and concludes it is that moving object that triggers the abnormal event. The distance between the centre points can be calculated by using equation below.

$$Distance = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} \tag{4.15}$$

If there is no any moving object that intersects with the OI when the abnormal event is detected, the system traces back the person who had intersected with the OI. The moving object that is the nearest to the object region at that moment is identified as suspected person. Figure 4.19 shows the process of identifying suspected person. Figure 4.20 is a sample results of object removal detection.

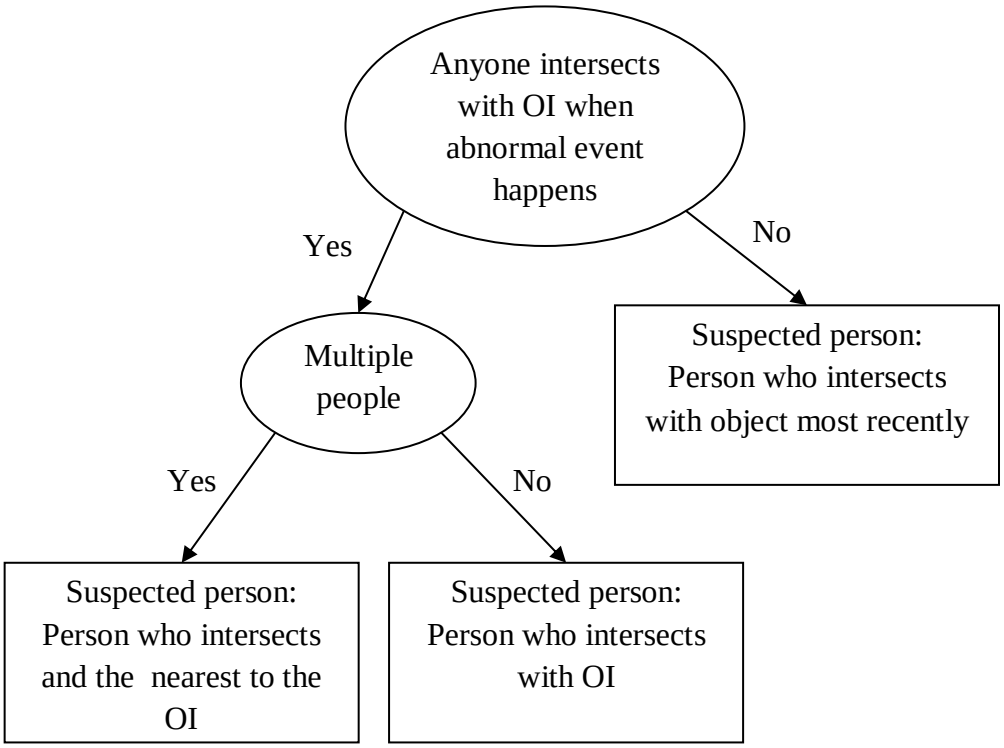
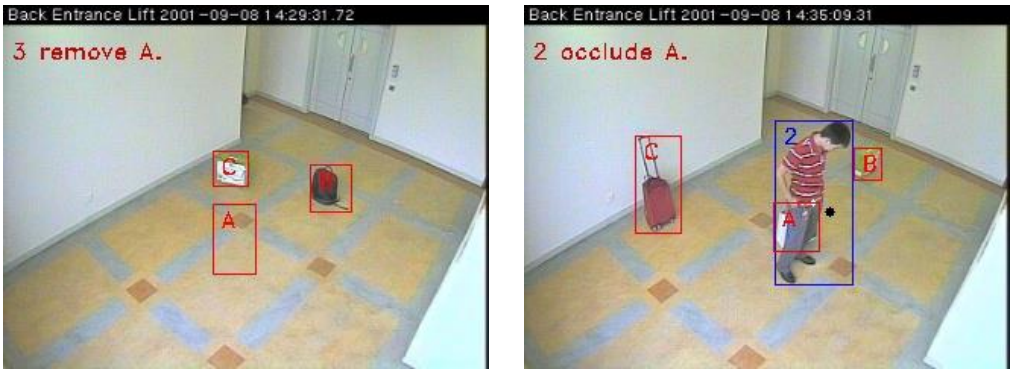


Figure 4.19: **Process of Identify Suspected Person**



(a) Object removal event (b) Object occlusion event

Figure 4.20: Sample Result of Object Removal Module

Table 4.13: Pseudocode of Identify of Suspected Person

```
detect_person_trigger_event(OI_list, event_list, tracking_list )
    Declare event_list_with_people
    FOR event in event_list is abnormal DO
        Determine any moving object intersect with specified OI
        IF moving object intersect with OI THEN
            IF total moving object intersect with OI is 1 THEN
                person = person who intersects with OI
            ELSE
                person = person who intersects with OI and nearest to OI
            ENDIF
        ELSE
            person = person who previously intersects with OI
        ENDIF
        Update person and event into event_list_with_people
    ENDFOR
RETURN with event_list_with_people
```

4.6 Chapter Summary

All implementations of modules in this project are explained in detail in this chapter. Pseudocode and equations are included to explain the system internal works. Besides this, figures are used to illustrate how the system runs. In conclusion, a system that is able to detect object occlusion and object removal event is implemented. The system can also determine the person who triggers the event. The performance of the system is discussed in the next chapter.

CHAPTER 5

SYSTEM TESTING

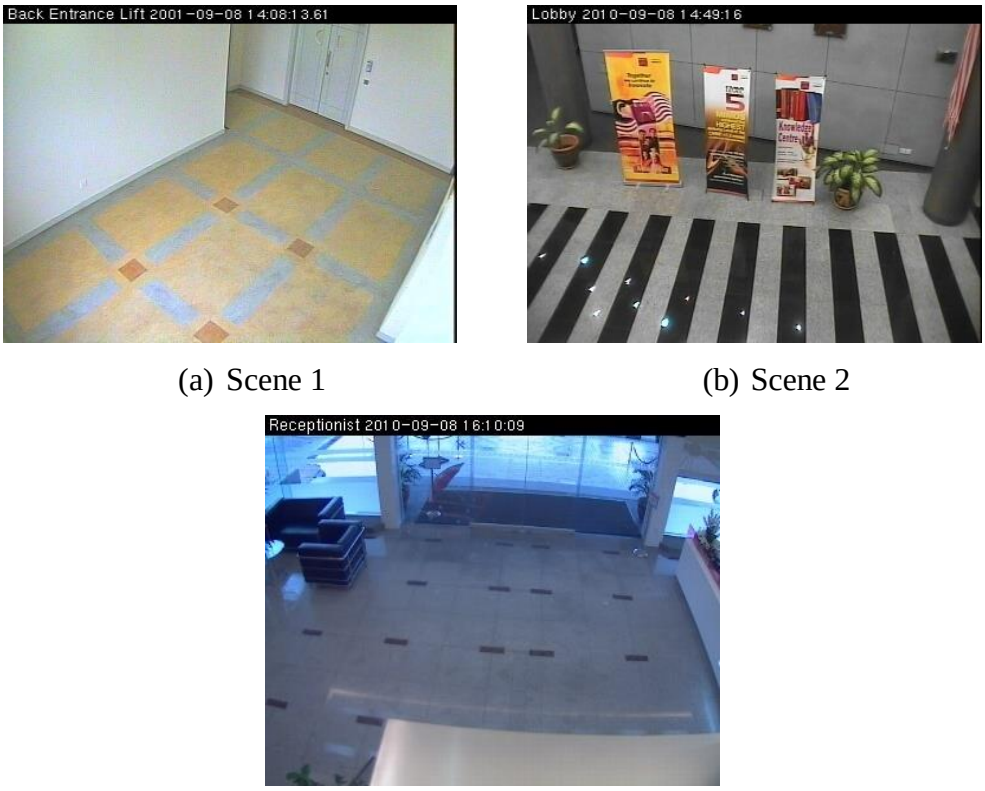
5.1 Introduction

This chapter focuses on testing of all modules in this project to test their performance. The sections of this chapter are organized as below:

- Testing data collection
- Testing result and Analysis
- Chapter summary

5.2 Testing Data Collection

For testing the data in this project, due to the available public dataset are more suitable for unattended object detection, so we use dataset from a research and development centre. This dataset contains 81 scenarios in total and there are one to three events in each scenario. These data are captured in three different places in a building. Each place has 27 videos in total. Figure 5.1 shows the environment of testing scene.



(a) Scene 1 (b) Scene 2

(c) scene 3

Figure 5.1: Testing Scene

These scenarios can be divided into three categories, which are simple (one to three persons), medium (three to five persons) and complex (maximum 10 persons). In each scenario, there is a minimum of one OI and maximum 3 OIs. The tasks of people in these videos include those who walk around in the scene, block OI, remove OI, remain static and pass by the scene. Table 5.1, Table 5.2 and Table 5.3 display events in scenarios in each testing place. The testing result is discussed in following section.

Table 5.1: Event in Simple Scenarios

Video Name	Total abnormal events	Description
Video 1	1	Person 1 occludes OI A
Video 2	1	Person 1 removes OI A
Video 3	2	Person 1 occludes OI A Person 2 removes OI B Person 3 walks around
Video 4	2	Person 1 and 2 remove OI A and B Person 3 walks around
Video 5	2	Person 1 removes OI A Person 2 occludes and removes OI B
Video 6	3	Person 1, 2 and 3 remove OI A, B and C
Video 7	3	Person 1 and 2 remove OI A and B Person 3 occludes OI C
Video 8	3	Person 1 removes OI A and B Person 2 occludes OI C and Person 3 removes OI C
Video 9	3	Person 1 removes OI A Person 2 removes OI B and C, then Person 3 remains static in front of OI C

Table 5.2: Event in Medium Scenarios

Video Name	Total abnormal events	Description
Video 1	1	Person 1 occludes OI A Person 2 and 3 remain static in the scene Person 4 and 5 walk around
Video 2	1	Person 1 removes OI A Person 2 walks around Person 3 remains static on the scene Person 4 and 5 pass by the scene
Video 3	2	Person 1 and 2 remove OI A and B Person 3, 4 and 5 pass by the scene
Video 4	2	Person 1 and 2 remove OI A and B Person 3 remains static on the scene Person 4 passes by the scene
Video 5	2	Person 1 and 2 remove OI A and B Person 3 remains static on the scene Person 4 walks around Person 5 passes by the scene
Video 6	3	Person 1 removes OI A Person 2 and 3 occlude OI B and C Person 4 and 5 walk around
Video 7	3	Person 1 and 2 remove OI A and B Person 3 occludes OI C Person 4 walks around Person 5 passes by the scene
Video 8	3	Person 1, 2 and 3 remove OI A, B and C Person 4 remains static on the scene
Video 9	3	Person 1, 2 and 3 remove OI A, B and C Person 4 remains static on the scene Person 5 passes by the scene

Table 5.3: Event in Complex Scenarios

Video Name	Total abnormal events	Description
Video 1	1	Person 1 occludes OI A Person 2 and 3 remain static on the scene Person 4, 5, 6, 7, and 8 walk around Person 9 and 10 pass by the scene
Video 2	1	Person 1 removes OI A Person 2 and 3 walk around Person 4 and 5 remain static on the scene Person 6 and 7 pass by the scene
Video 3	2	Person 1 and 2 remove OI 1 and B Person 3, 4, 5, 6, 7, 8, 9 and 10 pass by scene
Video 4	2	Person 1 removes OI A Person 2 occludes OI B Person 3 and 4 walk around Person 5, 6, 7, 8 and 9 pass by the scene Person 10 remains static on the scene
Video 5	3	Person 1 and 2 remove OI A and B Person 3 occludes OI C Person 4 and 5 walk around Person 6, 7, 8 and 9 pass by the scene Person 10 remains static on the scene
Video 6	3	Person 1, 2 and 3 remove OI A, B and C Person 4 and 5 walk around Person 6 and 7 remain static on the scene Person 8 and 9 pass by the scene
Video 7	3	Person 1, 2 and 3 remove OI A, B and C Person 4, 5, 6, 7, 8, 9 and 10 pass by the scene
Video 8	3	Person 1, 2 and 3 remove OI A, B and C Person 4 remains static on the scene Person 5 walks around
Video 9	3	Person 1, 2 and 3 remove OI A, B and C Person 4, 5, 6, 7, 8, 9 and 10 pass by the scene

5.3 Testing Result and Analysis

In this section, the focus is to determine the performance of the algorithm using collected dataset. The testing is done using computer with Intel core 2 Quad 2.83GHz processor and 4GB RAM. Before the testing process is started, suitable parameter needs to be found out of the 15 videos that are chosen from the dataset (roughly 20% of the dataset) to perform fine tuning process. In each scene, there are five randomly selected videos for fine tuning, where two videos are from simple scenario and medium scenario respectively, and one video from a complex scenario. Table 5.4 gives a summary for the fine tuning process. The table explains the effect if the threshold value is too small or too large. Figure 5.2 to Figure 5.4 show some example of fine tuning.

Table 5.4: Summary of Fine Tuning

Threshold	Threshold value too small	Threshold value too large
T_E	Tends to give false positive results for event detection	Tends to give false negative event for event detection
T_C	Tends to conclude an abnormal event as occlusion event	Tends to conclude abnormal event as removal event
T_M	Tends to merge multiple moving object into one and give false positive results	Tends to split a single object into multiple and give false negative results
Edge detection threshold	Tends to get false edges from image, give false positive results for moving object detection	Tends to get less edges from image, give false negative results for moving object detection and event detection

(a) 10

(b) 30

(c) 50

(d) 70

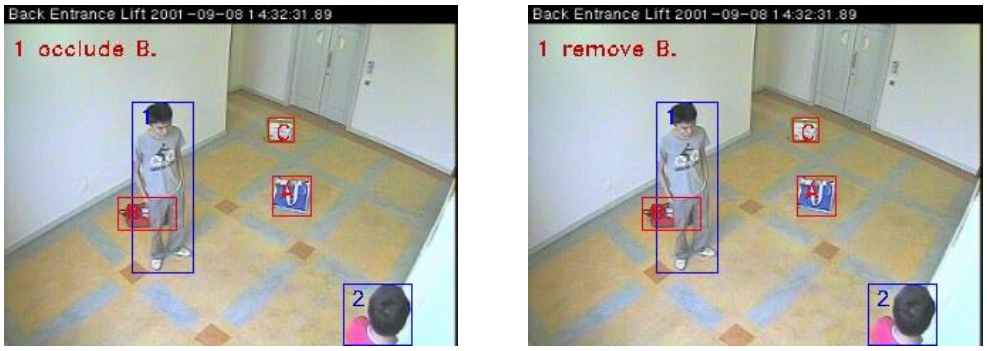
(e) 90

Figure 5.2: Example of Threshold T_M Fine Tuning

(a) 0.1

(b) 0.9

Figure 5.3: Example of Threshold T_E Fine Tuning



(a) 0.1 (b) 0.9

Figure 5.4: Example of Threshold T_C Fine Tuning

The next step is using the remaining videos to determine the performance of the algorithm. In the testing process, the parameter settings used are 40 for T_M, 50 for low edge detection threshold, 150 for high edge detection threshold, 0.5 for T_E and 0.6 for T_C as it is the best parameter taken in fine tuning. True positive (TP) and false positive (FP) are used to evaluate the algorithm. In this, TP means the algorithm detects an abnormal event successfully, while FP means the system detects a wrong abnormal event. The detailed detection result for all test scenarios is appended in appendix A. In summary, scene 1 has 51 abnormal events, 46 TPs and 5 FPs. For scene 2, there are 51 events, 42 TPs and 5 FPs. For scene 3, there are 51 events, 45 TPs and 3 FPs. In total, there are 153 abnormal events, 133 TPs and 13 FPs. True positive rate (TPR), false positive rate (FPR) and accuracy rate are applied to check the accuracy of the algorithm. The formulas are shown below:

$$TPR = \frac{total\ true\ positive}{total\ actual\ positive} \times 100\% \tag{5.1}$$

$$FPR = \frac{total\ false\ positive}{total\ actual\ positive} \times 100\% \tag{5.2}$$

$$Accuracy\ Rate = TPR - FPR \tag{5.3}$$

Table 5.5: Summary of TPR, FPR and Accuracy Rate for Test 1

Scene	Total events	Total TP	TPR (%)	Total FP	FPR (%)	Accuracy Rate
Scene 1	51	46	90.20	5	9.80	80.40
Scene 2	51	42	82.35	5	9.80	72.55
Scene 3	51	45	88.24	3	5.88	82.36
TOTAL	153	133	86.93	13	8.50	78.43

The speed of the system is also determined in the testing process. First, the time (in millisecond) to run each video image is figured out. Next, frame rate, frame per second (fps), is applied to find out total frame a system can process in one second. Furthermore, the average fps of the whole video is calculated by sum up the fps of all images in the video and then divided by the total image in the video. The following equations are the formula for fps and average fps.

$$\text{fps} = \frac{1000}{\text{time to run an image}} \tag{5.4}$$

$$\text{average fps} = \frac{\text{sum of fps from first image until last image}}{\text{total image in video}} \tag{5.5}$$

Overall, the maximum processing time delay is 78ms or frame rate is 7.15fps, while the minimum processing time is 15ms or frame rate is 66.67fps. The average time delay is 36.96ms or frame rate is 27.06fps. The minimum frame rate that is suitable for the human eye is 25fps, so the algorithm is suitable to use in real time environment. Appendix B illustrates average fps for all scenarios. Besides that, there are some examples of system detection in Appendix C.

We also used different threshold value for T_E and T_C in the testing process to test accuracy of object removal and object occlusion. We set 0.4 for T_E and 0.5 for T_C. Other threshold values are remained as we focus on object removal detection parameter to test accuracy of different threshold values. Table 5.6 shows result of using these threshold values and Appendix D displays detailed result.

Table 5.6: Summary of TPR, FPR and Accuracy Rate for Test 2

Scene	Total events	Total TP	TPR (%)	Total FP	FPR (%)	Accuracy Rate
Scene 1	51	40	78.43	8	15.69	62.75
Scene 2	51	39	76.47	7	13.73	62.75
Scene 3	51	43	84.31	5	9.80	74.51
TOTAL	153	122	79.74	20	13.07	66.67

Next, we set 0.6 for T_E and 0.7 for T_C. Table 5.7 shows result of test 3 and appendix E displays detailed result.

Table 5.7: Summary of TPR, FPR and Accuracy Rate for Test 3

Scene	Total events	Total TP	TPR (%)	Total FP	FPR (%)	Accuracy Rate
Scene 1	51	42	82.35	11	21.57	60.78
Scene 2	51	40	78.43	9	17.65	60.78
Scene 3	51	43	84.31	8	15.69	68.63
TOTAL	153	125	81.70	28	18.30	63.40

There are some factors that affect performance of the system. The first one is a removal event may be detected as occlusion event when there are people occluding the object while another person removes it.

Figure 5.5: Example of Wrong Detection

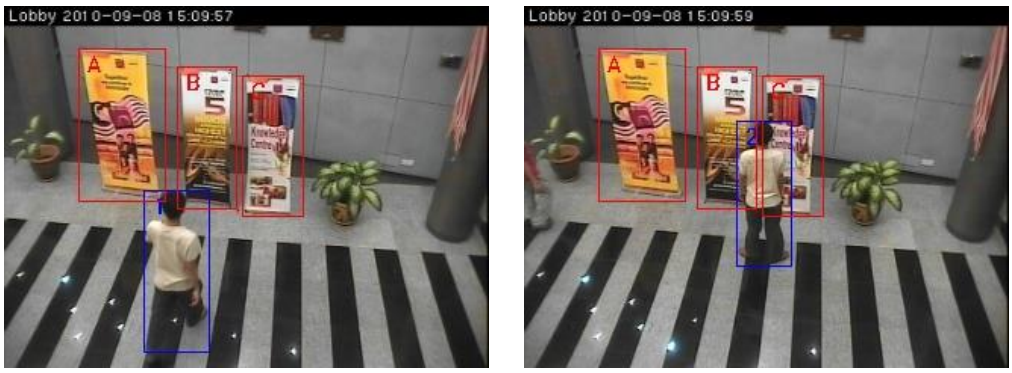
Figure 5.5 shows one example of wrong detection. In the example, person 4 is blocking the object A and person 6 passes by and takes away object A. But the system concludes person 4 occluding object A. This has happened because we only use one region to detect events. To solve this problem, we can divide the region into a few parts and do detection separately. When a removal event happens, if all regions detected removal event, then it is considered as removal event. But if some regions detected removal event while other regions detected occlusion event, this is considered as removal event and occlusion event happen at the same time. Figure 5.6 shows an example of detection using multiple regions. In the figure, the two regions at the right should detect a removal event, while another two are occlusion events. In this case, it is considered as the object is occluded by one person while another person

removes it. With multiple region detection, the system can detect full and partial occlusion event.

Figure 5.6: Example of Multiple Regions Detection

Another problem is when the video is having problems and skip some frames, the tracker may not able to find out the location of moving objects and the system assigns a new tracker to existing moving object. Figure 5.7 shows an example of unable to keep track moving object. In the figure (a), the system assigned a tracker to the person with white shirt namely 1. But after that, the video is a bit delayed and there are some missing frames, so tracker 1 unable to find the location of the white shirt person in figure (b). The system assumes the white shirt person leaves the scene and deletes tracker 1 from its tracker list. Besides that, the system also assigns another tracker, tracker 2 for the white shirt person. This kind of situation degrades the performance of the tracking process. To solve the problem, we can add in some feature and tracking method into the tracker. For example, we can add in colour, gradient, supervised learning method, tracking learning detection, and so on to allow the

tracker to be able to track the moving object when it encounters missing frames.



(a) (b)

Figure 5.7: Example of Miss Tracking

Another common problem is the object colour that is similar to the background, and this causes the system unable to detect objects. For this problem, we can add other features in detection to improve the performance.

5.4 Chapter Summary

In this chapter, it determines the performance of the algorithm in this project. 81 videos are selected to evaluate the algorithm. 15 videos are used to fine tune parameter, while other videos are used to test the performance. There are three testing environments with three different complexities. From the testing result, the algorithm gives 86.93% TPR, 8.50% FPR, and 78.43% accuracy rate. Besides, the algorithm can give a real time processing speed.

CHAPTER 6

CONCLUSION

6.1 Conclusion

In this project, a system that is able to detect object removal and occlusion event is created. This system can increase the performance of detection of the object removal event by differentiating between object occlusion and object removal event. First, the system detects all moving objects in the scene. The system uses background subtraction on colour channels and edge subtraction to remove noises around the moving object. Next, the system keeps track all moving objects in the scene using Kalman filter trackers. The tracker uses Kalman filter method to predict the location of moving objects. The system can track the moving object by using a prediction point to match location of the moving object. After that, the system detects abnormal event that is happening on a specific object in the scene. The abnormal events that can happen in the scene are occlusion and removal event. Edge information is used to identify event happening on the specific object. After the system detected abnormal event, the system finds out the person who triggers the abnormal event. From the testing result, it shows that the performance of the system is at an acceptable rate, where the TPR is 86.93%, FPR is 8.5% and the accuracy rate is 78.43%. For the processing speed, the average frame rate is 27.06fps.

6.2 Future Enhancement

In this project, we focus on detection of static objects. In future, we can use the concept of event detection algorithm on moving object with some modification. We can modify event detection to keep track of moving object. When the moving object is in the scene, the modified detection can know the location of the moving object in the scene. This can give some information to other research, for example, we can use it to further analyse why people will always move or stop at some point. When the moving object is moving, we can update the position of the modified detection to keep track of the object. When two moving objects crossover, the modified object occlusion detection can tell which object occludes another.

Besides that, we can add on module in the project to identify the owner of the object in the scene. When someone takes the specific object, the system needs to analyse whether that person is the owner of the object. The alarm will only ring when the person is not the owner. Parameter in this project can also be adjusted automatically by adding parameter tuning adjustment algorithm.

There are some enhancements that can be done to further improve the performance. First, in some situations, the system may detect a removal event as occlusion event when a person occluded an object while another person removes it. We can solve this problem by dividing the object region and outer region into few parts. If one of the regions detects object removal event, then the system can conclude this as object removal event. If there is no removal

event detected but occlusion event is detected by some or all parts, then the system can conclude it as partial or full occlusion event.

The tracker may not be able to track the object correctly when the image sequence is not continuous. However, we can improve it by adding other features or methods such as colour, gradient, Tracking Learning Detection, and any matching algorithm to improve tracking. We can enhance moving object detection when the moving object has similar colour with background by adding other features, such as gradient, active contour, and so on.

REFERENCES

Bhaskar, H., Mihaylova, L. and Achim, A., 2010. Video foreground detection based on symmetric alpha-stable mixture models. *IEEE Transaction on Circuits and Systems for Video Technology,* 20 (8), pp. 1133-1138.

Caro Campos, L., SanMiguel, J.C. and Martínez, J.M., 2011. Discrimination of abandoned and stolen object based on active contours. *Proceedings of the Eighth IEEE International Conference on Advanced Video and Signal-Based Surveillance,* 30 August -2 September 2011 Klagenfurt, Austria. Klagenfurt: IEEE, pp. 101-106.

Chiu, C., Ku, M. and Liang, L., 2010. A robust object segmentation system using a probability-based background extraction algorithm. *IEEE Transaction on Circuits and Systems for Video Technology,* 20 (24), pp. 518-528.

Ferrando, S., Gera, G. and Regazzoni, C., 2006. Classification of unattended and stolen objects in video-surveillance system. *Proceedings of the International Conference on Video and Signal Based Surveillance*, 22 -24 November 2006 Sydney, Australia. Syndey: IEEE, pp. 21-26.

Guan, Y., 2009. Spatio-temporal motion-based foreground segmentation and shadow suppression. *IET Computer Vision,* 4 (1), pp. 50-60.

Hong, C., Yang, Z., Bu, J., Chen, C. and Deng, X., 2009. A real-time approach of level-set-based contour tracking for accurate foreground extraction. *Proceedings of the 2009 WRI World Congress on Computer Science and Information Engineering*, 31 March -2 April 2009 Los Angeles, California. Los Angeles: IEEE, pp. 596-600.

Khan, Z. and Gu, I., 2010. Joint feature correspondences and appearance similarity for robust visual object tracking. *IEEE Transaction on Information Forensics and Security,* 5 (3), pp. 591-606.

Lee, S., Woo, H. and Kang, M., 2010. Global illumination invariant object detection with level set based bimodal segmentation. *IEEE Transaction on Circuits and Systems for Video Technology,* 20 (4), pp. 616-620.

Leibe, B., Schindler, K., Cornelis, N. and Gool, L., 2008. Coupled object detection and tracking from static cameras and moving vehicles. *IEEE Transaction on Pattern Analysis and Machine Intelligence,* 30(10), pp. 1683-1698.

Li, Q., Mao, Y., Wang, Z. and Xiang, W., 2009. Robust real-time detection of abandoned and removed objects. *Proceedings of the Fifth International Conference on Image and Graphics,* 20 -23 September 2009 Shanxi, China. Shanxi: IEEE, pp. 156-161.

Liu, L.Y., Sang, N. and Huang, R., 2010. Background subtraction using shape and colour information. *Electronics Letters,* 46 (1), pp. 41-43.

Li, X. and Bian, S., 2008. Robust object detection and tracking using a space-temporal mutual feedback scheme. *Proceddings of the IEEE International Symposium on Knowledge Acquisition and Modeling Workshop*, 21-22 December 2008 Wuhan, China. Wuhan: IEEE, pp. 212-216.

Li, X., Wang, K., Wang, W. and Li, Y., 2010. A multiple object tracking method using Kalman filter. *Proceddings of the IEEE International Conference on Information and Automation*, 20-23 June 2010 Harbin, China. Harbin: IEEE, pp. 1862-1866.

Maini, R. and Addarwal, H., 2009. Study and comparison of various image edge detection techniques. *International Journal of Image Processing*, 3 (1), pp. 1-11.

Nadernejad, E., Sharifzadeh, S. and Hassanpour, H., 2008. Edge detection techniques: evaluations and comparisons. *Applied Mathematical Sciences*, 2 (31), pp. 1507-1520.

Pathan, S., Al-Hamadi, A. and Michaelis, B., 2009. Intelligent feature-guided multi-object tracking using Kalman filter. *Proceedings of the second International Conference on Computer, Control & Communication*, 17-18 February 2009 Karachi, Pakistan. Karachi: IEEE, pp. 1-6.

San Miguel, J.C. and Martínez, J.M., 2008. Robust unattended and stolen object detection by fusing simple algorithms. *Proceedings of the Fifth International Conference on Advanced Video and Signal Based Surveillance*, 1-3 September 2008 Santa Fe, New Mexico. Santa Fe: IEEE, pp. 18-25.

SanMiguel, J., Caro, L. and Martinez, J., 2012. Pixel-based colour contrast for abandoned and stolen object discrimination in video surveillance. *Electronics Letters*, 48 (2), pp. 86-87.

Shin, M.C., Goldgof, D.B., Bowyer, K.W. and Nikiforou, S., 2001. Comparison of edge detection algorithms using a structure from motion task. *IEEE Transaction on Systems, Man and Cybernetics, Part B: Cybernetics*, 31 (4), pp. 589-601.

Spagnolo, P., Caroppo, A., Leo, M., Martiriggiano, T. and D'Orazio, T., 2006. An abandoned removed objects detection algorithm and its evaluation on PETS datasets. *Proceedings of the International Conference on Video and Signal Based Surveillance*, 22-24 November 2006 Sydney, Australia. Sydney: IEEE, pp. 17.

Tian, Y., Feris, R.S., Liu, H., Hampapur, A. and Sun, M., 2010. Robust detection of abandoned and removed objects incomplex surveillance videos. *IEEE Transaction on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 41 (5), pp. 565-576.

Wang, L. and Yung, N., 2010. Extraction of moving objects from their background based on multiple adaptive thresholds and boundary evaluation. *IEEE Transaction on Intelligent Transportation Systems,* 11 (1), pp. 40-51.

Wang, W. and Liu, Z., 2010. A new approach for real-time detection of abandoned and stolen objects. *Proceedings of the 2010 International Conference on Electrical and Control Engineering*, 25-27 June 2010 Wuhan, China. Wuhan:IEEE, pp. 128-131.

Wang, W., Yang, J. and Gao, W., 2008. Modeling background and segmenting moving objects from compressed video. *IEEE Transaction on Circuits and Systems for Video Technology,* 18 (5), pp. 670-681.

Zeng, W., Zhu, G. and Li, Y., 2009. Point matching estimation for moving object tracking based on Kalman filter. *Proceedings of the eighth IEEE/ACIS International Conference on Computer and Information Science*, 1-3 June 2009 Shanghai, China. Shanghai:IEEE Computer Society Press, pp. 1115-1119.

Zhang, X., Hu, W. and Qu, W., 2010. Multiple object tracking via species-based particle swarm optimization. *IEEE Transaction on Circuits and Systems for Video Technology*, 20 (11), pp. 1590-1602.

Appendix A

Result for Simple Scenarios in Scene 1

Video name	Total events	Total TP	total FP
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 6	3	2	1
Video 7	3	3	0
Video 8	3	2	1
Video 9	3	3	0

Result for Medium Scenarios in Scene 1

Video name	Total events	Total TP	total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 6	3	3	0
Video 7	3	2	1
Video 9	3	2	1

Result for Complex Scenarios in Scene 1

Video name	Total events	Total TP	total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	2	0
Video 5	3	3	0
Video 6	3	3	0
Video 7	3	3	0
Video 8	3	2	1
Video 9	3	3	0

Result for Simple Scenarios in Scene 2

Video name	Total events	Total TP	total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	1	0
Video 4	2	2	0
Video 6	3	3	0
Video 7	3	2	0
Video 9	3	2	1

Result for Medium Scenarios in Scene 2

Video name	Total events	Total TP	total FP
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	1	1
Video 6	3	2	0
Video 7	3	2	1
Video 8	3	3	0
Video 9	3	2	1

Result for Complex Scenarios in Scene 2

Video name	Total events	Total TP	total FP
Video 1	1	0	0
Video 2	1	1	0
Video 3	2	2	0
Video 5	3	3	0
Video 6	3	2	0
Video 7	3	3	1
Video 8	3	3	0
Video 9	3	3	0

Result for Simple Scenarios in Scene 3

Video name	Total events	Total TP	total FP
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 5	2	2	0
Video 6	3	3	0
Video 7	3	2	0
Video 9	3	3	0

Result for Medium Scenarios in Scene 3

Video name	Total events	Total TP	total FP
Video 2	1	1	0
Video 3	2	1	0
Video 4	2	2	0
Video 6	3	3	0
Video 7	3	3	0
Video 8	3	2	0
Video 9	3	2	1

Result for Complex Scenarios in Scene 3

Video name	Total events	Total TP	total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 6	3	1	1
Video 7	3	3	0
Video 8	3	3	1
Video 9	3	3	0

Appendix B

Processing Time for Simple Scenarios in Scene 1

Video name	Average fps
Video 2	31.83
Video 3	28.56
Video 4	30.47
Video 6	39.67
Video 7	28.04
Video 8	30.76
Video 9	30.18

Processing Time for Medium Scenarios in Scene 1

Video name	Average fps
Video 1	29.01
Video 2	25.60
Video 3	25.09
Video 4	25.24
Video 6	25.54
Video 7	28.05
Video 9	30.83

Processing Time for Complex Scenarios in Scene 1

Video name	Average fps
Video 1	23.13
Video 2	25.60
Video 3	26.86
Video 5	28.01
Video 6	29.28
Video 7	27.95
Video 8	29.20
Video 9	29.18

Processing Time for Simple Scenarios in Scene 2

Video name	Average fps
Video 1	30.49
Video 2	22.83
Video 3	26.65
Video 4	22.45
Video 6	30.25
Video 7	26.77
Video 9	31.43

Processing Time for Medium Scenarios in Scene 2

Video name	Average fps
Video 2	25.66
Video 3	27.09
Video 4	22.24
Video 6	30.22
Video 7	23.68
Video 8	27.89
Video 9	28.60

Processing Time for Complex Scenarios in Scene 2

Video name	Average fps
Video 1	31.15
Video 2	30.81
Video 3	27.39
Video 5	32.28
Video 6	29.07
Video 7	27.97
Video 8	25.48
Video 9	29.62

Processing Time for Simple Scenarios in Scene 3

Video name	Average fps
Video 2	55.39
Video 3	32.67
Video 4	38.64
Video 5	33.75
Video 6	38.84
Video 7	40.36
Video 9	27.06

Processing Time for Medium Scenarios in Scene 3

Video name	Average fps
Video 2	24.69
Video 3	27.87
Video 4	24.73
Video 6	23.47
Video 7	27.48
Video 8	26.43
Video 9	21.36

Processing Time for Complex Scenarios in Scene 3

Video name	Average fps
Video 1	26.68
Video 2	25.36
Video 3	34.93
Video 4	23.44
Video 6	34.39
Video 7	36.86
Video 8	23.62
Video 9	24.21

Appendix C

(a)

(b)

(c)

(d)

(e)

(f)

(g)

(h)

Example from First Scene

(a)

(b)

(c)

(d)

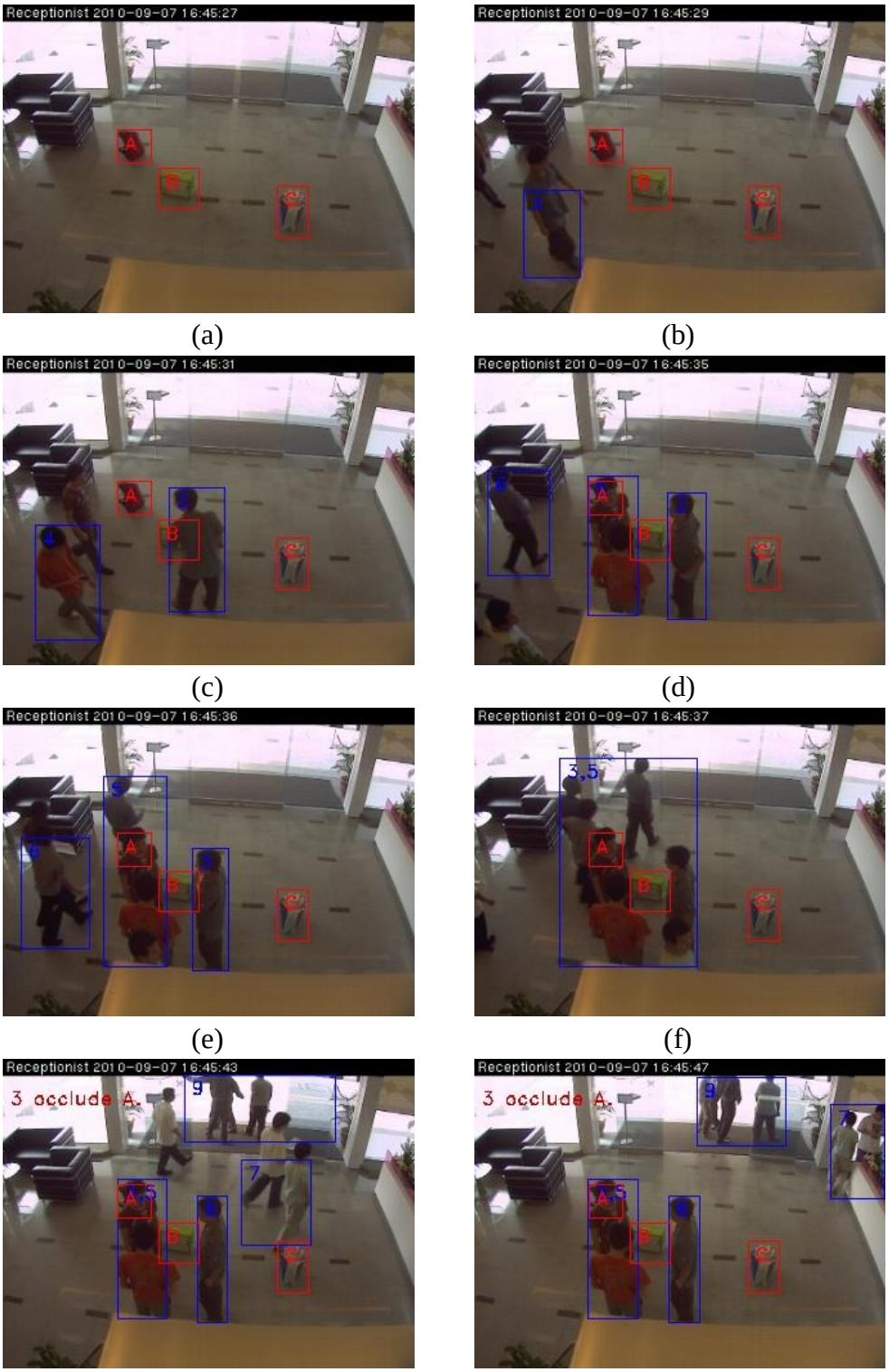
(e)

(f)

(g)

(h)

Example from Second Scene



(a) (b)

(c) (d)

(e) (f)

(g) (h)

Example from Third Scene

Appendix D

Result for Simple Scenarios in Scene 1

Video name	Total events	Total TP	Total FP
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 6	3	2	1
Video 7	3	2	0
Video 8	3	2	1
Video 9	3	3	0

Result for Medium Scenarios in Scene 1

Video name	Total events	Total TP	Total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	1	1
Video 4	2	1	0
Video 6	3	3	0
Video 7	3	2	1
Video 9	3	2	1

Result for Complex Scenarios in Scene 1

Video name	Total events	Total TP	Total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	1	1
Video 5	3	3	0
Video 6	3	2	1
Video 7	3	3	0
Video 8	3	2	1
Video 9	3	2	0

Result for Simple Scenarios in Scene 2

Video name	Total events	Total TP	Total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	1	0
Video 4	2	2	0
Video 6	3	2	1
Video 7	3	2	0
Video 9	3	2	1

Result for Medium Scenarios in Scene 2

Video name	Total events	Total TP	Total FP
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	1	1
Video 6	3	2	0
Video 7	3	2	1
Video 8	3	3	0
Video 9	3	2	1

Result for Complex Scenarios in Scene 2

Video name	Total events	Total TP	Total FP
Video 1	1	0	0
Video 2	1	1	0
Video 3	2	2	0
Video 5	3	3	0
Video 6	3	1	0
Video 7	3	3	1
Video 8	3	2	1
Video 9	3	3	0

Result for Simple Scenarios in Scene 3

Video name	Total events	Total TP	Total FP
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 5	2	2	0
Video 6	3	3	0
Video 7	3	2	0
Video 9	3	2	1

Result for Medium Scenarios in Scene 3

Video name	Total events	Total TP	Total FP
Video 2	1	1	0
Video 3	2	1	0
Video 4	2	2	0
Video 6	3	3	0
Video 7	3	2	1
Video 8	3	2	0
Video 9	3	2	1

Result for Complex Scenarios in Scene 3

Video name	Total events	Total TP	Total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 6	3	1	1
Video 7	3	3	0
Video 8	3	3	1
Video 9	3	3	0

Appendix E

Result for Simple Scenarios in Scene 1

Video name	Total events	Total TP	Total FP
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 6	3	3	1
Video 7	3	2	0
Video 8	3	2	1
Video 9	3	3	1

Result for Medium Scenarios in Scene 1

Video name	Total events	Total TP	Total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	1	1
Video 4	2	2	0
Video 6	3	3	0
Video 7	3	3	1
Video 9	3	2	1

Result for Complex Scenarios in Scene 1

Video name	Total events	Total TP	Total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	1	1
Video 5	3	3	0
Video 6	3	2	1
Video 7	3	2	1
Video 8	3	2	1
Video 9	3	2	1

Result for Simple Scenarios in Scene 2

Video name	Total events	Total TP	Total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	1	0
Video 4	2	2	0
Video 6	3	2	1
Video 7	3	3	0
Video 9	3	2	1

Result for Medium Scenarios in Scene 2

Video name	Total events	Total TP	Total FP
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	1	1
Video 6	3	2	0
Video 7	3	2	1
Video 8	3	2	1
Video 9	3	2	1

Result for Complex Scenarios in Scene 2

Video name	Total events	Total TP	Total FP
Video 1	1	0	0
Video 2	1	1	0
Video 3	2	2	0
Video 5	3	3	0
Video 6	3	2	1
Video 7	3	3	1
Video 8	3	2	1
Video 9	3	3	0

Result for Simple Scenarios in Scene 3

Video name	Total events	Total TP	Total FP
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 5	2	2	0
Video 6	3	3	0
Video 7	3	2	1
Video 9	3	2	1

Result for Medium Scenarios in Scene 3

Video name	Total events	Total TP	Total FP
Video 2	1	1	0
Video 3	2	1	0
Video 4	2	2	1
Video 6	3	3	0
Video 7	3	2	1
Video 8	3	3	0
Video 9	3	2	1

Result for Complex Scenarios in Scene 3

Video name	Total events	Total TP	Total FP
Video 1	1	1	0
Video 2	1	1	0
Video 3	2	2	0
Video 4	2	2	0
Video 6	3	2	1
Video 7	3	2	1
Video 8	3	3	1
Video 9	3	2	0

Appendix F

PUBLICATION- OBJECT OCCLUSION AND OBJECT REMOVAL DETECTION

Object Occlusion and Object Removal Detection

Yung Joon Chai*[a], Siak Wang Khor*[b], Yong Haur Tay*[a]
[a]Faculty of Engineering and Science, Universiti Tunku Abdul Rahman, Malaysia; [b]Faculty of Information and Communication Technology, Universiti Tunku Abdul Rahman, Malaysia

ABSTRACT

In general, there are methods to detect object removal event, but when a person is blocking the object, it is possible to classify the event as removal. Thus, if we can make a system that can differentiate between occlusion and removal event, this can increase the accuracy and performance of the system. In this paper, we present a method that can detect and classify object occlusion and object removal event. The detection and classification uses Canny edge detector, and the classification can be done by determining the edge similarity of the object between background and current image. The system is tested in different places and it gives an acceptable and satisfying result.

Keywords: removed object; object occlusion; object removal; abnormal event detection

1. INTRODUCTION

Theft of valuable belongings is common, and this occurs in many places such as office, art galleries, museums and even private residential areas, so the use of surveillance system to track any missing valuable items is common. Normally the system is monitored by human guard. When the system monitoring process is too long, the tiredness may make the guard miss some incidents, so some methods are also proposed to help detect object removal event.

[1] proposed an algorithm that is able to detect abandoned or removed objects from static moving object by matching the contour of detected static foreground objects in current image and segmented image. [2] presented a system that can classify unattended and stolen objects by detecting non-human static moving object and apply algorithm to determine if the object is either abandoned or removed.

[3] proposed a framework to detect abandoned and removed objects by using region growing method. Object removal event can be detected when the region growing in background image is larger than in current image. Few years later,

the algorithm is further improved by adding trajectories tracking algorithm to reduce false alarm rate [4].

[5] provided a pixel based colour contrast approach to identify abandoned and removed objects. The colour contrast of pixel inside and outside the boundary of object is compared in current image and background. For removed object, the colour contrast in background should be low, while the colour contrast in current image is high.

[6] presented a method to detect abandoned and removed objects based on colour richness. The known region is classified into few colour bins. Event can be detected by checking total colour bin with value higher than threshold in current image and background.

[7] used active contour to find out abandoned object or removed object. First, the contour of object is adjusted on current image and background. For removed event, the adjusted contour in background is similar with initial contour, while the adjusted contour in current image may shrink or disappear.

Although the researches are useful to detect object removal event, but when an object is occluded/ blocked by other object but is not removed, the detection may face failure. Thus, this paper proposes a method that can classify occlusion event and removal event by using edge approach.

The remaining sections of the paper are organized as below. Section 2 explains the methodology of the system. Section 3 is experimental result and discussion. The last section is conclusion of the whole paper.

2. METHODOLOGY

In this section a method to detect object removal and object occlusion events is introduced. First, coordinate of targeted object is set and it can be come from abandoned object, user predefines, and etc. The event that can happen is either normal or abnormal. Normal event means nothing happens, while abnormal event can be divided into two: object removal and object occlusion. Figure 1 illustrates the overview of the event detection.

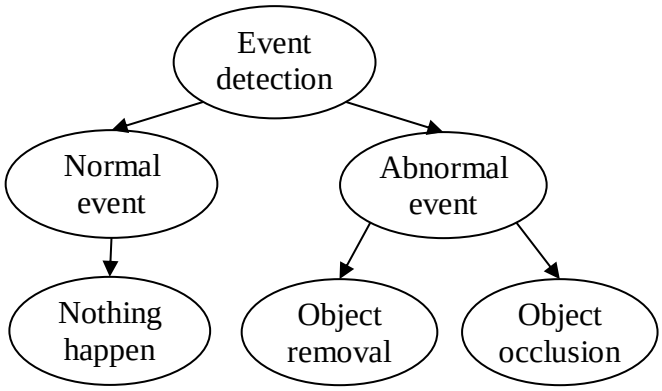


Figure 1. Overview for detect object removal and occlusion event

This method contains two phases: abnormal event detection and abnormal event classification.

Abnormal event detection phase detects event happen on a targeted object. Object region is introduced to complete the detection. Object region is a region part on the targeted object. Figure 2 shows an example for object region.

Figure 2(a) is plain image for an object. For Figure 2(b), the region inside the red rectangular box is object region of the object.

(a) Original image (b) Object region

Figure 2. Example of object region

When abnormal event happened, the texture in the object region will be different for background and current image. But for normal event, the texture will remain the same. First, edge in the images will be extracted by using Canny detector. The edge will be used to determine the texture similarity on the targeted object. Next, subtraction between background and current image on targeted object is done, and the total white pixel is determined to classify either abnormal event or normal event happened. The formula to detect abnormal event is shown in Equation 1.

$$R_E = \begin{cases} Abnormal\ event, if \dfrac{\sum (B(x, y) - C(x, y)}{\sum B(x, y)} > T_E \\ Normal\ event, otherwise \end{cases} \quad (1)$$

where B is background edge mask inside object region, C is current edge mask inside object region, and T_E is threshold.

Figure 3 and Figure 4 show image result example for abnormal event and normal event. (a) and (b) are original image for background and current image respectively. (c) and (d) are edge images for (a) and (b). For (e), it is result for (c) – (d).

(a) Background (b) Current image (c) B (d) C (e) Result image

Figure 3. Example of abnormal event

(a) Background (b) Current image (c) B (d) C (e) Result image

Figure 4. Example of normal event

After abnormal event is detected, abnormal event classification phase will classify the abnormal event is either occlusion or removal. The classification can be done by comparing the outer region. Outer region is a region which has a bigger size than object and this region will exclude the region part that

belongs to object. Figure 5 shows an example of outer region. Figure 5(a) is original image for a targeted object. For Figure 5(b), the region inside the two red rectangular boxes is outer region of the object.

(a) Original image (b) Outer region

Figure 5. Example of outer region

For occlusion event, the texture in outer region for current image and background is different. For removal event, the texture in outer region for current image and background is same. First, the edge on outer region for background model and current image is extracted. Edge similarity on outer region can be calculated by using Equation 2.

$$R_C = \begin{cases} Object\ occlusion, if\ \dfrac{\sum (C'(x, y) - B'(x, y)}{\sum C'(x, y)} > T_C \\ Object\ removal, otherwise \end{cases} \quad (2)$$

where B` is background edge mask on outer region, C` is current edge mask on outer region, and T_C, is threshold.

Figure 6 and Figure 7 show image result examples for removal event and occlusion event. (a) and (b) are original image for current image and background respectively. For (c) and (d), these are edge images for (a) and (b). There are two red rectangular boxes drawn to display the outer region part of the object. For (e), it is result for (c) – (d).

(a) Current image (b) Background (c) C` (d) B` (e) Result image

Figure 6. Example of removal event

(a) Current image (b) Background (c) C` (d) B` (e) Result image

Figure 7. Example of normal event

Figure 8 shows an example of the detection by using proposed method. In this example, the targeted object A is occluded by a red shirt guy, while object B and C are taken by other people.

(a) Background (b) Result

Figure 8. Example of detection of proposed method

3. Experimental Result and Discussion

In the experiment, due to the lack of database video for object occlusion and object removal, we used dataset from a research and development centre. The method is tested in three places. Figure 9 shows the environment of the places. For each place, there are three different complexities which are simple (one to three persons), medium (three to five persons) and complex (maximum 10 persons). Each place has 22 scenarios in total. The tasks of people in these videos include block targeted object, remove the object, walk around, remain static and pass by the scene.

Place 1 Place 2 Place 3

Figure 9. Testing scenes

In here, true positive (TP) means the algorithm detects and classifies abnormal event successfully, while false positive (FP) means the system detect event wrongly. True positive rate (TPR) and false positive rate (FPR) is applied to check the accuracy of the algorithm. The formulas are shown below and table 1 shows the TPR, FPR and accuracy rate.

$$TPR = \frac{Total\ TP}{Total\ event} \times 100\% \quad (3)$$

$$FPR = \frac{Total\ FP}{Total\ event} \times 100\% \quad (4)$$

$$Accuracy\ rate = \frac{Total\ TP - total\ FP}{Total\ event} \times 100\% \quad (5)$$

Table 1. Result of the proposed method

Place	TPR	FPR	Accuracy rate
Place 1	92,10	7,90	84,20
Place 2	81,98	7,90	74,08
Place 3	90,29	5,66	84,63

Place	TPR	FPR	Accuracy rate
TOTAL	88.24	7.19	81.05

In the experiment, we successfully detected most of the abnormal events. There are situation where the proposed method may face failure. For instance, when targeted object is occluded by a person while another person removes the object. The system may conclude the event as occlusion event. Figure 8 shows an example of the failed case. In Figure 8(a), there are two people, the first person occluding object A. At the same time, the second person walks toward object A. In Figure 8(b), the second person takes the object A and is going to leave the scene while first person is still standing at the same place.

There are some solutions that can solve the problem above. We can first divide the object region and outer region in few parts and combine the detection results. Besides that, we also can combine our idea with idea from other researchers, such as we can combine the region growing method from [3] and [4], active contour from [7] and etc.

(a) Before object removed

(b) After object removed

Figure 8. Example of failed case

4. Conclusion

We have proposed a method that can detect object removal and object occlusion event. This abnormal event can be detected using edge information. By checking the similarity of the edge of the object, it can detect event when someone occludes or removes the object. From the experiment, the system achieves an acceptable performance. Although the system may fail when a person occludes an object, while the other removes the object, but this problem can be solved by adding other methods into the detection. This system can help to detect removal event more accurately by differentiating between occlusion and removal.

5. ACKNOWLEDGEMENT

This work has been supported by Fundamental Research Grant Scheme (FRGS).

References

[1] Paolo Spagnolo, Andrea Caroppo, Marco Leo, Tommaso Martiriggiano, and Tiziana D'Orazio, "An Abandoned Removed Objects Detection Algorithm and Its Evaluation on PETS Datasets," in IEEE International Conference on Video and Signal Based Surveillance, 2006 (AVSS '06), 2006, pp. 17.

[2] Silvia Ferrando, Gianluca Gera, and Carlo Regazzoni, "Classification of Unattended and Stolen Objects in Video-Surveillance System," in IEEE

International Conference on Video and Signal Based Surveillance, 2006 (AVSS '06), 2006, pp. 21.

[3] Ying-li Tian, Rogerio Feris, and Arun Hampapur, "Real-Time Detection of Abandoned and Removed Objects in Complex Environment," in IEEE International Workshop on Visual Surveillance (in conjunction with ECCV'08), Marseille, France, 2008.

[4] YingLi Tian, Rogerio Schmidt Feris, Haowei Liu, Arun Hampapur, and Ming-Ting Sun, "Robust Detection of Abandoned and Removed Objects in Complex Surveillance Videos," in IEEE Transactions on Systems, Man, and Cybernetics, 2010, pp. 565-576.

[5] J.C. SanMiguel, L. Caro, and J.M. Martinez, "Pixel-based colour contrast for abandoned and stolen object discrimination in video surveillance," Electronics Letters, pp. 86-87, 2012.

[6] Qiujie Li, Yaobin Mao, Zhiquan Wang, and Wenbo Xiang, "Robust Real-time Detection of Abandoned and Removed Objects," in Fifth International Conference on Image and Graphics, 2009 (ICIG '09), 2009, pp. 156-161.

[7] Luis Caro Campos, Juan Carlos SanMiguel, and José M. Martínez, "Discrimination of abandoned and stolen object based on active contours," in 8th IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS), 2011, 2011, pp. 101-106.