Nonparametric pixel-wise background modelling and segmentation to detect moving object with RGB-D camera

By

Navid Dorudian

Department of Computer Science

College of Engineering, Design and Physical Sciences

Brunel University London

This thesis is submitted for the degree of

Doctor of Philosophy

July 2020

Declaration

I, Navid Dorudian, hereby declare that this thesis and the work presented is entirely my work. Some of the content has been previously published in journals or conference papers, and this has been mentioned in the thesis. Where I have consulted the work of others, this is always clearly stated and acknowledged.

Navid Dorudian
July 2020

Acknowledgements

I would like to express my gratitude to all my family and friends who consciously or unconsciously have inspired and helped me to undertake my research.

I would like to start with special thanks to my principal supervisor, Dr. Stasha Luria. Without his patience and guidance in supervising through the long years of research and revising this thesis, it would not have been possible to complete this journey. He has been exceptionally enthusiastic, continuous motivation, pleasant smile on his face and trusting my ideas, was the most crucial factors in the accomplishment of major challenges in my research. In many occasions, he has gone an extra mile to provide materials and resources required with my research.

I would like to take this opportunity to also express my gratitude to my second supervisor, Dr. Stephen Swift for his support and guidance in my research and writing my publications. I am also thankful to Dr. Armin Kashefi as my research development advisor for his support and encouragement.

My sincere thanks also go to Professor Xiaohui Liu, Dr Allan Tucker, Dr Arthur Money, Dr Yongmin Li, Dr Derek Groen, and Professor George Ghinea for their annual review on my research progress and advice given. I am also thankful to the academic and other staff of the department of Computer Science for their kind assistance. I want to thank the classmates and colleagues who helped and supported me in my study; Leila Yousefi, Mashael Al-Luhaybi, Roja Ahmadi Shirvani, Bashir Dodo and Siavash Esmaeili. I am also thankful to my wonderful friend Christopher Florides for his support in this journey and helping me to write my thesis more accurately. Thanks also goes to my other research colleagues and students with whom I have spent a great time at Brunel. Finally, I would like to thank my wife and my parents for their support and motivation.

Abstract

Moving object detection is one of the main fundamental parts in various computer vision applications, particularly for real-time object tracking and recognition in automated video surveillance. Although human eyes can simply recognise objects and changes in the scene, automated detection of moving objects in some scenarios are still a challenging task for existing systems. One of the main effective ways to improve the detection rate is to use colour and depth cameras together (RGB-D). Despite the effort of previous researchers with various sensors, moving object detection is still challenging in some scenarios such as dynamic background, sudden illumination-changes, colour and depth camouflage, intermittent motion, out of sensor range, bootstrapping, slow and stationary moving objects, etc. Thus, the aim of this thesis is to improve the detection accuracy and efficiency of moving object detection by achieving more precise and consistent detection in different challenging scenarios. To attain this, three new robust pixel-wise nonparametric methods for real-time automatic detection of moving objects in indoor environments using an external RGB-D sensor are presented. The methods introduced in chapter 4 and 5 (BSABU, NBM-GA, NBM-HC) are an improved version of the proposed method in chapter 3 called NBMS. These methods are able to deal with various complex scenarios and different type of moving objects such as high-speed drones or slow and stationary moving objects such as a human.

NBMS method first creates two background models by storing some observed colour and depth pixels. Then each pixel from the new frame will be compared with the stored models to mark the new pixel as foreground or background. These models require a continuous update to adapt to the changes in the environment. A novel regular update based on the distance of the pixels is proposed which only applies to the pixels marked as a background after pixels classification. Besides, the method also blindly updates the model to adapt to sudden changes in the background. This approach is compared with other methods in different collected datasets from the drone, a publicly available dataset and a live application. Results show improvements over current methods. In chapter four, an adaptive blind update policy has been added to the method to improve the detection accuracy of stationary moving objects. In particular, blind update frequency changes based on the speed of the moving object or any other changes in the background to prevent absorption of a stationary moving object in the background models. Besides, a new shadow detection method using CIEL*a*b* colour added to enhance the detection accuracy in the case of shadow and depth camouflage. Results show significant improvement compared to the original method. This method also evaluated in 32 datasets in the benchmark and the results have shown robust and consistent in different challenging scenarios. Due to a large number of samples in the model, optimisation algorithms such as Hill-Climbing, and Genetic Algorithm (GA) could help to improve the efficiency and accuracy even further. Instead of updating the models pixel by pixel, a fitness function calculates the fitness of each stored sample image and only one image will receive an update each time. GA selects this image by a Roulette Wheel. This updating mechanism allows all the pixels in the depth model to have a chance to get updated and therefore the system does not stop in the local optima, which is usually created by the noise of the sensors. Results indicate improvement in the depth-camouflage scenario and reduction of computational costs.

Nomenclature

Symbols

The following symbols are used in this thesis:

$\mathbb{R}^3$	The set of all three-dimensional space
X	The position of a point in three-dimensional space
X_t	The position of a point in three-dimensional space at time t
N	Number of samples
α	The frequently of blind update
$M(X)_{RGB}$	The Colour background Model
$M(X)_D$	The Depth Background Model
$M(X)_{Lab}$	The Colour Background Model based on LAB space
$v(x)$	The given colour value at pint X
$d(X)$	The given depth value at pint X
$\#_{Min}$	The minimum number of similar pixels
$\#_{Min-depth}$	The number of similar depth pixels
$\#_{Min-Colour}$	The number of similar colour pixels
Th_{RGB}	Acceptable colour threshold
Th_D	Acceptable depth threshold
$d_{\text{smallest}}(X)$	The smallest depth value in the model
µ	Population size
F	Fitness of solution
NG	Number of Generations
CP	Crossover Probability
MP	Mutation Probability
n	The number of integers (genes) making up each Chromosome

Acronyms / Abbreviations

The following abbreviations are used in this thesis:

ADO	Absent Depth Observation
Bg	Background
BSABU	Background subtraction using adaptive blind update
FP	False Positive
FSM	Finite State Machine
Fg	Foreground
FN	False negative
FPS	Frames Per Second
GPS	Global Positioning System
GMM	Gaussian Mixture Model
GSM	Generic Scene Modelling
IR	Infrared Radiation
KDE	Kernel Density Estimation
MAV	Micro Aerial Vehicle
MOG	Mixture of Gaussian
NBMS	Nonparametric background Modelling and Segmentation
PMD	Photonic Mixer Device
PDF	Probability Density Function
RGB-D	Red-Green-Blue-Depth
RM	Average Ranking of Method
RC	Overall Ranking Across Category
ROI	Region of Interest
SFO	Stationary Foreground Object
S	Similarity Measure
S_B	Similarity Measure Around Object Boundaries
TOF	Time of Flight

TE	Total Error
UAV	Unmanned Aerial Vehicle
ViBe	Visual Background Extractor
3D	Three-Dimension
HC	Hill Climbing
GA	Genetic Algorithm
EC	Evolutionary Computation
RMHC	Random Mutation Hill Climbing
NBM-HC	Nonparametric Background Modelling using RMHC
NBM-GA	Nonparametric Background Modelling using GA
St.Dev	Standard Deviation

Table of Content

DECLARATION .... **II**

ACKNOWLEDGEMENTS .... **III**

ABSTRACT .... **IV**

NOMENCLATURE .... **V**

SYMBOLS .... **V**

ACRONYMS / ABBREVIATIONS .... **VI**

TABLE OF CONTENT .... **VIII**

LIST OF FIGURES .... **XI**

LIST OF TABLES .... **XIII**

CHAPTER 1: INTRODUCTION **1**

1.1 MOTIVATION 1
1.2 CONTRIBUTIONS 3
1.3 PUBLICATIONS 5
1.4 THESIS STRUCTURE 5

CHAPTER 2: BACKGROUND **7**

2.1 INTRODUCTION 7
2.2 MOTION AND CHANGE DETECTION 7
2.3 BACKGROUND SUBTRACTION ALGORITHMS 8
2.4 RELATED WORK 10
2.5 NONPARAMETRIC METHODS 12
2.6 THE VIBE ALGORITHM 16
2.7 RGB-D SENSORS 18
2.8 RGB-D METHODS 20
2.9 CHALLENGING SCENARIOS IN RGB-D 24
2.10 CONCLUSION AND DISCUSSION 27

CHAPTER 3: NONPARAMETRIC BACKGROUND MODELLING AND SEGMENTATION TO DETECT MICRO AIR VEHICLES (MAV) USING RGB-D SENSOR 30

3.1 INTRODUCTION 30
3.2 PROPOSED BACKGROUND SUBTRACTION 33
3.2.1 SYSTEM INITIALISATION 34
3.2.2 ADO REMOVAL 35
3.2.3 BG/FG SEGMENTATION 36
3.2.4 BACKGROUND UPDATE 41
3.2.5 BLIND UPDATE 42
3.3 RESULTS 44
3.3.1 MAVS SEQUENCES 50
3.3.2 BENCHMARK DATASET 54
3.4 REAL-TIME EXPERIMENT 64
3.5 SUMMARY 67

CHAPTER 4: MOVING OBJECT DETECTION USING ADAPTIVE BLIND UPDATE POLICY AND RGB-D CAMERA 69

4.1 INTRODUCTION 69
4.2 BACKGROUND 70
4.3 PROPOSED METHOD 72
4.3.1 FG SEGMENTATION 74
4.3.2 BACKGROUND MODEL UPDATE 79
4.3.3 BLIND UPDATE POLICY 79
4.4 RESULTS AND EVALUATION 80
4.4.1 COMPARISON TO THE ORIGINAL METHOD 83
4.4.2 SBM-RGBD DATASET 88
4.4.3 REAL-TIME EXPERIMENT 93
4.5 SUMMARY 93

CHAPTER 5: BACKGROUND MODELLING AND SEGMENTATION USING A GENETIC ALGORITHM AND HILL CLIMBING 95

5.1 INTRODUCTION 95
5.2 RELATED WORK 96
5.3 PROPOSED METHOD 98
5.3.1 BACKGROUND MODELLING USING A GENETIC ALGORITHM (NBM-GA) 100
5.3.1.1 Fitness Function 101
5.3.1.2 Roulette Wheel Selection 102
5.3.1.3 Uniform Crossover 102
5.3.1.4 GA Mutation Update 103
5.3.2 BACKGROUND MODELLING USING A RANDOM MUTATION HILL CLIMBING (NBM-HC) 104
5.4 RESULTS 105
5.5 REAL-TIME EXPERIMENT 115
5.6 SUMMARY 116

CHAPTER 6: CONCLUSIONS AND FUTURE RESEARCH 118

6.1 COMPARISONS OF ALL PROPOSED METHODS IN THIS THESIS... 118
6.2 CONCLUDING REMARKS.. 120
6.3 RESEARCH ACHIEVEMENTS 123
6.4 DISCUSSION OF PROPOSED METHODS, LIMITATIONS AND PARAMETER'S OPTIMISATIONS............ 124
6.5 RECOMMENDATIONS FOR FUTURE RESEARCH.. 127

REFERENCES.. 129

APPENDIX A 140

List of figures

Figure 1.1. *Example of ROI in the green rectangle identified by the system.* *2*
Figure 2.1. *Background Subtraction Process described in [35].* *9*
Figure 2.2. *Foreground detection results in [58].* *15*
Figure 2.3. *The appearance of a ghost in ViBe background subtraction result in [63].* *16*
Figure 2.4. *Detail of Microsoft Kinect version one presented in [83].* *19*
Figure 2.5. *the amount of noise that could occur in a single depth frame.* *20*
Figure 2.6. *Related depth data problems (highlighted by red ellipses)identified in [38].* *27*
Figure 3.1*. Two types of different quadcopter.* *31*
Figure 3.2*. Proposed position tracking/marker recognition algorithm in [16].* *32*
Figure 3.3*. Flow chart of the proposed object detection method.* *34*
Figure 3.4*. An example of depth image.* *36*
Figure 3.5. *An example of shadow.* *37*
Figure 3.6. *Flow chart of the proposed classification method.* *40*
Figure 3.7*. an example of a good and bad sample pixel.* *42*
Figure 3.8*. Detection accuracy in a cluttered background.* *46*
Figure 3.9*. Detection accuracy in sharp illumination changes.* *46*
Figure 3.10*. Detection accuracy in the appearance of a shadow.* *47*
Figure 3.11*. Detection accuracy when the light is switched off.* *47*
Figure 3.12*. The result of micro UAV sequence.* *51*
Figure 3.13. *The result of AR.Drone sequence.* *53*
Figure 3.14*. Result of the proposed method in frame 1026 of GenSeq sequence.* *54*
Figure 3.15. *Result of the proposed method in frame 1073 of DCamSeq sequence.* *56*
Figure 3.16. *Result of the proposed method in frame 1078 of ColCamSeq sequence.* *57*
Figure 3.17. *An example of ShSeq sequences.* *60*
Figure 3.18*. RM chart.* *61*
Figure 3.19. *Example of qualitative comparison in a stationary moving object dataset.* *64*
Figure 3.20*. An AR.Drone was used to testing the proposed method in real-time.* *65*
Figure 3.21*. The computational time for the system to control the quadcopter.* *66*
Figure 4.1*. Flow chart of the proposed object detection method.* *73*
Figure 4.2*. Flowchart of the proposed segmentation method.* *78*
Figure 4.3. *Sequence ChairBox in [116].* *81*
Figure 4.4. *Example of qualitative comparison in a stationary moving object dataset.* *85*
Figure 4.5*. Total error (TE). The lower amount shows better performance.* *86*
Figure 4.6*. Similarity measure (S).* *87*
Figure 4.7. *Similarity measure in object boundaries (SB).* *87*
Figure 4.8*. Qualitative comparison in 7 challenging scenarios.* *91*
Figure 4.9. *Average ranking of all existing methods in each category of the dataset.* *92*
Figure 5.1. *Flow chart of the proposed classification method introduced in chapter 3 [3].* *98*
Figure 5.2. *The flow chart of proposed background modelling and object detection method based on Genetic Algorithm.* *101*
Figure 5.3. *An example of dataset introduced in [48].* *108*
Figure 5.4. *Frame 1061 of DCamSeq sequence introduced in [48].* *109*
Figure 5.5. *Frame 446 of ShSeq sequences.* *112*

Figure 5.6. *RM chart illustrates the overall performance.* *113*
Figure 5.7. *Convergence graph from NBM-HC and NBM-GA.* *113*
Figure 5.8. *Frame 534 of DCamSeq2 (Depth Camouflage) dataset.* *114*
Figure 5.9. *The computational time for NVM-GA to process each frame.* *115*
Figure 5.10. *The computational time for NVM-HC to process each frame.* *116*
Figure 6.1. *Total error (TE).* *118*
Figure 6.2. *Similarity measure (S).* *119*
Figure 6.3. *Similarity measure around object boundaries (SB).* *119*
Figure 6.4. *The efficiency (FPS) of the proposed methods in this thesis.* *119*
Figure 6.5. *Summary of this thesis.* *126*

List of tables

Table 2.1. *Classification of representative background modelling of Pixel-based methods.* *11*
Table 3.1. *Details of our dataset used for measuring the accuracy of the UAVs detection at 30fps.* .. *45*
Table 3.2. *Details of the dataset in [48] which used for evaluation in this section.* *48*
Table 3.3. *Craziflies sequence results.* *52*
Table 3.4. *AR.Drone sequence results*.. *52*
Table 3.5. *GenSeq sequence results*... *55*
Table 3.6. *DCamSeq sequence results.* *56*
Table 3.7. *ColCamSeq sequence results.*... *58*
Table 3.8. *ShSeq sequence results.* *59*
Table 4.1. *Full details of SBM-RGBD datasets introduced in [116].*... *82*
Table 4.2. *Craziflies sequence results.* *88*
Table 4.3. *AR.Drone sequence results*.. *88*
Table 4.4. *Average results of all available methods in SBM-RGBD datasets.*.................................... *89*
Table 5.1. *Craziflies sequence results.* *106*
Table 5.2. *AR.Drone sequence results*.. *107*
Table 5.3. *GenSeq sequence results*... *108*
Table 5.4. *DCamSeq sequence results.* *110*
Table 5.5. *ColCamSeq sequence results.*... *111*
Table 5.6. *ShSeq sequence results.* *111*

Chapter 1

Introduction

1.1 Motivation

Vision is considered as one of the most valuable senses human being possess. Just by simply glancing of the scene, a vast amount of valuable information can be easily obtained. This may not be possible to use all other senses combined as easily. Therefore, many researchers were inspired by this fact to emulate this natural ability into human-made systems by a combination of hardware and software. Thus, the visual sense is not only the primary source for humans but nowadays it is also crucial for many computer vision and robotics systems. One of the most essential steps in many computer vision systems is moving object detection, especially real-time object tracking and automated surveillance. Motion and change detection is typically the first step of computer vision applications and the outcomes are used for further processing activities [1]. Formally, change detection is defined as the process of automatically detecting and analysing substantial changes occurring in a scene by observing the frames in a video stream [2]. These changes frequently occur by appearing as a moving object in the scene such as human, animals or robots. Recognising detected moving object as the region of interest (ROI) allows the system to perform additional tasks such as robots localisation and tracking [3], people and object counting, detecting wild-life scenes and exploring [4], traffic monitoring [5] and etc. Figure 1.1 illustrates an example of localisation and navigation of a robot where the drone (moving object) in the green rectangle identified as an ROI and red rectangle is the predicted future position with the current direction.

Typically, the key step in the mentioned applications is to identify the foreground (moving object) from the scene known as background [6]. One of the most common ways is to compare the current image frame with former frames also known as "model" or "reference". These models could generate from a single image or more compound model called "scene model" [7]. In order for the scene model to adapt to changes in the background over the time, it requires a continuous update. Examples of these changes are illumination changes, appearing or removing an object in the scene or moving an object in the background. Also, to deal with these changes, how we collect these samples to create the model is important as it will be used for the segmentation of the frames. Bad samples in initialisation would cause misclassification of the pixels and delay in the update process and consequently create more false segmentation. This problem has been broadly discussed in the literature. Different techniques to update these samples have been suggested. For example, remove the old samples in the model and change them with the new samples which traditionally is known as first in first out. Naturally, these methods update all the examples in the model which some samples may still be valid and may not be necessary to change them. Similarly, using the background pixels only or also using foreground pixels as part of the samples are always argued. These approaches are called blind and conservative update [8]. Conservative update procedure only takes those pixels identified as part of the background and it never includes foreground pixels. This procedure can effectively produce sharp detection of moving objects. However, this update cannot cope with many real-world scenarios and therefore lead to the production of ghosts' phenomenon, permanent misclassification and deadlock situations. One of these scenarios is a change in the background of the scene which can cause permanent misclassification as some parts of the background will be classified as foreground and this area will never get updated.

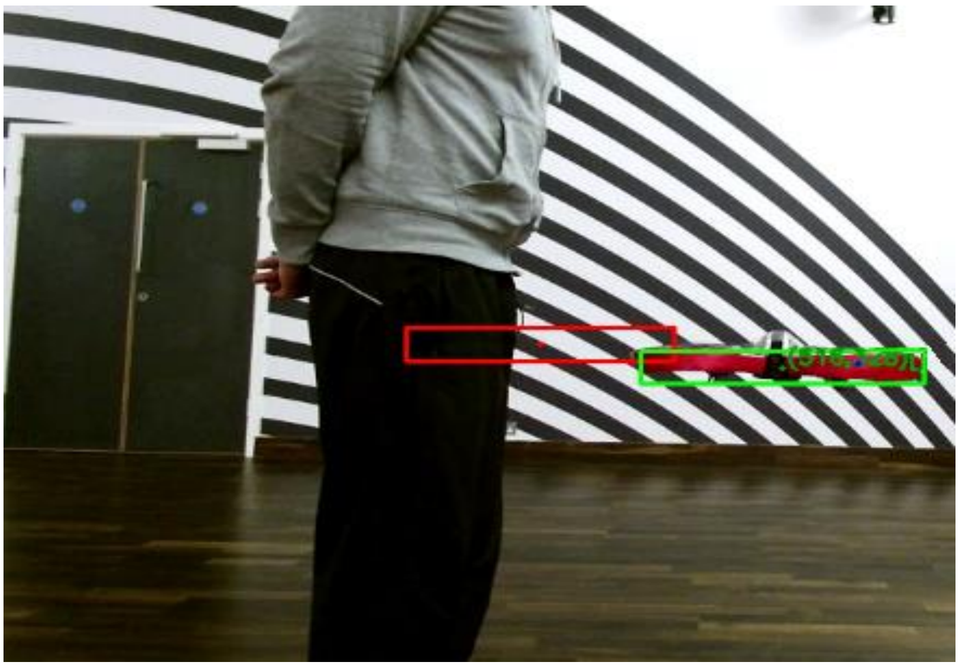


Figure 1.1. *Example of ROI in the green rectangle identified by the system.*

On the other hand, blind update policy which is used in ViBe [8] and MoG [9] algorithms includes all pixels into the background model update regardless of being part of the background or foreground. The drawback of this approach mainly is the production of ghost phenomenon, poor detection of a slow-moving object or stationary foreground objects (SFOs). Typically in the literature, these motionless objects are referred to a moving object which stops and remains static for some time. Their regions gradually absorb to the scene model and eventually, the scene model will adapt to the stationary object. Consequently, the object will incorrectly disappear from the foreground region. Segmentation of SFOs is still a challenging task for many background subtraction methods which attracted the attention of many researchers in the last few years [10]. Generally, we refer to "segmentation" as a classification of the pixels as foreground or background. By improving the mentioned weaknesses in segmentation, the accuracy and consistency of current methods can be enhanced. Therefore, the aim of this thesis is to improve the detection accuracy of moving object detection by achieving more precise and consistent detection in different challenging scenarios.

In the remainder of this thesis, first, we will overview existing methods and then we will investigate the detection of different type of moving objects such as fast, slow and stationary objects and propose new pixel-wise moving object detection solutions to improve the accuracy and efficiency in these problems by creating a model to store the samples and regular updates to keep the models adaptive to the changes in the scene.

1.2 Contributions

The main contributions of this thesis are below:

- ❖ Introducing a new pixel-wise nonparametric background modelling and segmentation to detect a moving object using colour frames and depth data. This method uses new segmentation criteria which are different from existing approaches. This method stores some colour and depth frames to create a model. Then, to find the moving object each pixel from the new frame will be compared to the models and a set of rules has been designed to segment the foreground and background. The segmentation method primarily relies on the depth sensor and compares each pixel with the model based on the distance the pixel is observed from the sensor. To the best of the author's knowledge, this is a new approach and it was never implemented in this way.
- ❖ This method is fully tested in three different structures. First, the method evaluated in two different fast-moving UAVs sequences (AR.Drone [11] and Crazyflies [12]). We have collected these data to measure the accuracy of the method to detect fast and small moving objects in a cluttered environment and other challenging conditions. This type of dataset

was not publicly available and we had to collect this dataset as part of our contribution to this research. Also, we have evaluated this method in other publicly available RGB-D datasets in four different challenging scenarios and compared them with the other state-of-the-art methods. In the end, we have tested the proposed method in a live application within an indoor environment to measure the accuracy of the proposed method in real-time.

- In chapter 4, a new adaptive blind update policy has been introduced and added to the proposed method. The blind update frequency in this method varies based on the speed of a moving object. This adaptive capability allows the system to adapt to the changes in the background, detect the fast and stationary moving object and reduce the ghost phenomenon. Additionally, a new shadow detection method based on CIE L*a*b* colour space and object boundary detection added to the algorithm to improve the accuracy in shadow and depth camouflage scenarios. Also, we have introduced a bootstrapping detection method, which is added to the system to increase the segmentation accuracy in the case of bootstrapping.
- A comprehensive evaluation of the proposed method in 33 different sequences in various challenging scenarios with a total of more than 15000 frames is added to this thesis. This is including the comparison to the original method and other state-of-the-art methods in benchmark and drone's datasets.
- Besides, another new moving object detection method using the optimisation algorithms (a Genetic Algorithm and Hill Climbing) is introduced. This method for the first time is using optimisation algorithms to update the background models in this way. In this method, instead of simple and classical approaches to update the background models, a fitness function has been implemented to rank each stored frame in the model and then a Roulette Wheel is used to choose which frame from the model needs to receive an updated after each new frame. This method allows all pixels to have a chance to get updated and prevents only the best pixels being selected. This capability will enable the system to not stop in local optima and prevent permanent misclassification. Also, it increases the efficiency of the proposed method compared to the original approach. This is also compared with the base and other existing methods in benchmark and drone's datasets.

1.3 Publications

1) N. Dorudian, S. Lauria, and S. Swift, “Nonparametric background modelling and segmentation to detect micro air vehicles using RGB-D sensor,” *International Journal fo Micro Aerial Vehicle*, vol. 11, p. 1756829318822327, 2019. https://journals.sagepub.com/doi/full/10.1177/1756829318822327

2) N. Dorudian, S. Lauria, and S. Swift, “Moving Object Detection using Adaptive Blind Update and RGB-D Camera,” *IEEE Sensor Journal*, vol. 19, no. 18, pp. 8191–8201, Sep, 2019. https://ieeexplore.ieee.org/document/8732428?source=authoralert

3) N. Dorudian, S. Lauria, and S. Swift, “Background modelling and segmentation using a Genetic Algorithm and Hill Climbing,” (to be submitted).

1.4 Thesis Structure

This thesis consists of 6 chapters in total (including this chapter). The contents of the remaining chapters are explained in the following section:

Chapter 2 is a review of various recent advanced methods on moving object detection and unsolved challenges which are still interesting for computer vision researchers and it is based on nonchronological order. In particular, we have discussed extensive types of background subtraction methods including parametric approaches, for example, Gaussian Mixture Model (GMM) [13], nonparametric methods such as ViBe [8] and Pbas [14], methods which have paid particular attention to detect stationary or slow-moving objects and methods which have also used depth sensors such as RGB-D camera, TOF (Time of flight) or stereo camera. Additionally, we have identified challenging scenarios which usually background subtraction algorithms have difficulties in detecting the whole silhouette of a moving object in some of these sequences.

In chapter 3, based on the motivation and the experience we have gained from previous works in chapter 2, we have proposed a new pixel-wise non-parametric moving object detection method. This method has improved the foreground detection accuracy in some challenging scenarios such as small and fast speed moving objects in an indoor environment using RGB-D camera. This method has different criteria compared to the current techniques. The evaluation results in this chapter are collected from a publicly available dataset and also from drone’s datasets which have been captured by us. Despite the great results this method has achieved in these initial evaluations, we have discovered the system still suffers in scenarios where slow and stationary moving objects exist, bootstrapping occurs, depth camouflage and sharp illumination affect the scene.

In chapter 4, we have investigated how to improve these weaknesses by adding an adapting blind update policy, bootstrapping and shadow detection to the method. The segmentation process and the set of rules that the system follows to segment the pixels as part of the background or foreground will be explained step by step in chapter four. We have investigated the effect of the proposed adaptive blind update method on detection accuracy of SFOs as well as fast-moving objects. A full comparison of the proposed method in a publicly available benchmark dataset and drone's datasets with other state-of-the-art methods and the original method is presented in the chapter.

In chapter 5, we have investigated the effects of artificial intelligence (AI) in nonparametric moving object detection to increase the efficiency of the proposed method. The goal is to use AI to develop an expert system which is able to extract the background models from the new frames more efficiently and accurately instead of traditional first-in-first-out, random sampling or the highest value replaces the lowest. To implement this system, Hill-Climbing and Genetic Algorithms has been used to update the models by the survival of the fittest theory. This has been applied by ranking the samples in the fitness function and choose the selected parents by a Roulette Wheel which allows all the pixels in the depth model to have a chance to get updated instead of only updating closest pixel to the camera. This allows the system to not stop in the local optima which is usually created by the sensor noise. The comparison of this method with the original and other methods will be discussed in the chapter.

In chapter 6, first, the results of all proposed methods in this thesis have been compared in terms of accuracy and efficiency. Then, all the contributions which were achieved during this research are discussed. Further, how to optimise the proposed methods and parameters, and how this work can be improved in further studies and future works is explained.

Chapter 2
Background

2.1 Introduction

In this chapter, the goal is to review the well-known and recent algorithms in moving object detection and change detection. The main focus of this literature review will be on moving object detection, especially those methods which used RGB-D camera. This will include different tasks such as segmentation, detection, and tracking.

This chapter is organized as follow: Section 2.1 will be a discussion about general motion and change detection as well as a review of different approaches to detect motion, the weaknesses and existing problems in this area. Section 2.2 will be an overview of related work of background subtraction techniques and different categories of background subtraction will be explained. Section 2.3 well-known nonparametric background subtraction methods will be reviewed and the advantages and weaknesses of these methods will be discussed in detail. In section 2.4, ViBe algorithm which is a well-known and efficient nonparametric method will be explained in detail. In section 2.5, recent RGB-D sensors along with their advantages and drawbacks will be discussed in detail. In section 2.6, recent RGB-D methods will be highlighted in detail. In section 2.7, scenarios which are typically challenging for background subtraction algorithms will be discussed and section 2.8 is a summary of this chapter.

2.2 Motion and Change Detection

There is a vast and growing number of research on motion and change detection in the field of computer vision. Observing the motion by a static camera and detecting it in the scene is an essential phase for many visual surveillance systems. This is mainly crucial for real-time automated surveillance and object tracking applications. Change detection is described as the process of detecting foreground regions which usually belong to moving objects such as people, animals or robots. We are also interested in detecting objects that become motionless for some time and then move as part of the foreground region which is known as stationary foreground objects (SFOs) [15]. For example, a person who stays stationary for a few seconds or a car stopped at a traffic light. However, we do not consider some small movements that continuously occur in the background as a change and therefore these objects will be classified as part of the background. For example, a wave on a water surface, tree branches or shakes of objects by wind (e.g. chandelier, curtains, light poles, flags, etc.).

The main aim of motion detection is to separate foreground (moving object) from the scene (background). By transforming the foreground area to the region of interest (ROI), many additional tasks will be possible to perform such as robots localisation and tracking [16], safe UAVs navigation [17][3], people counting [18], wild-life and traffic monitoring [19]. One of the most traditional methods is to compare the current frame with previously observed frames. These previous frames are called “reference”. This reference classically is created from a single frame or more compound model which is known as a background or scene model [7]. A scene model requires a constant update to adapt to the changes occurring in the background in real-world scenarios.

In the last few years, motion detection algorithms are divided into different categories such as optical flow [20][21], running average [22], cluster analysis [23], median filtering [24], frame differencing [25] and background subtraction [8]. Among these categories, the last two are mainly the most common methods [26]. Additionally, statistical background subtraction models are also broadly used. These models are divided into different types such as Midden Markov Models [27], Gaussian Mixture Models (GMM) [9], mean-shift clustering [28], nonparametric kernel density estimation [29], unimodal (single modelled) such as Gaussian [30] and Chi-Square distribution [31].

As background subtraction methods are a common way and the most effective to detect the moving object, these methods will be investigated in more detail in the next section.

2.3 Background Subtraction Algorithms

Detecting an object is usually an essential task in analysing the scene. If we have a scene model based on statistical data, an external object can be detected by recognising the parts of the frame that don’t exist in the model. This process in the literature is called “Background Subtraction” [32].

In general, a background subtraction procedure consists of at least four main steps. (1) background model initialisation, (2) background model representation, which defines what type of model represent the background, (3) background model maintenance, which is the update mechanism to allow the model to adapt to the changes in the scene, (4) foreground detection, which describes how to compare the current frame with the background model and classify the pixels as background or foreground [33][34]. Figure 2.1 illustrates general background subtraction algorithm process defined in [35], Where b_t is the background and i_t the current frame at time t. N is the total number of frames which is used for the background initialization.

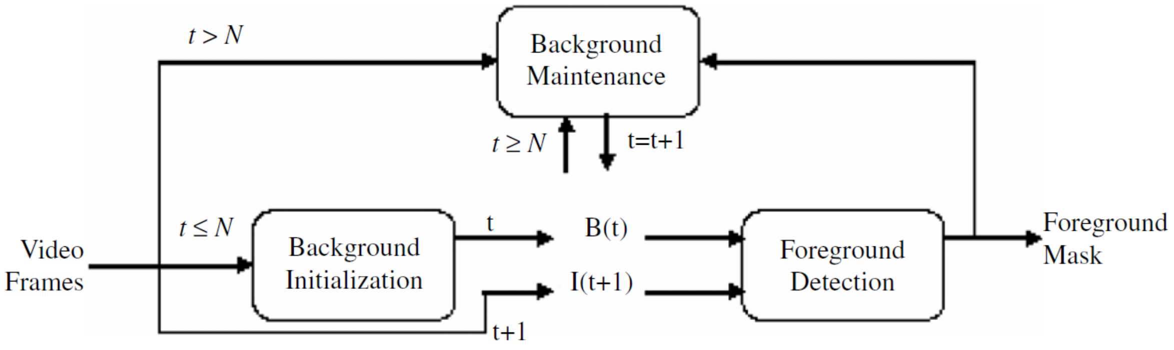


Figure 2.1. *Background Subtraction Process described in* [35].
b_t is the background and i_t the current frame at time t. N is the total number of frames which is used for the background initialization.

Background subtraction algorithms are mainly created based on a static background hypothesis [36]. Typically in this hypothesis, it has been assumed that there are no changes to the background in the scene in indoor environments. However, most practical scenarios lead to dynamic backgrounds such as animated images on screen, sunshine or reflections, moving chandelier or curtains by the wind. Therefore, most of the background subtraction algorithms have difficulties to deal with these conditions. To deal with these situations, many different kinds of research on foreground detection and background subtraction algorithms has been carried out. Although this fact can give the feeling that this problem is over-researched or it is solved, currently none of the algorithms seems to be able to simultaneously address all the key challenges that we can have in a scene [35]. Therefore, despite the effort of previous researchers, the problem of moving object detection remains challenging and there is no universal method to adequately detect the foreground in all practical scenarios. This was our main motivation to start the research introduced in this thesis to find a practical approach to combine the depth data and colour frames to obtain a more accurate and consistent outcome. As background subtraction methods could achieve the best results compared to the other approaches and are the most common way to detect the moving object, in the next section, we are going to review well-known and relevant background subtraction methods.

2.4 Related Work

In the last decade, background subtraction methods have been widely used to detect moving objects in the scene due to the robustness and effectiveness of these methods. During these years, many different background subtraction methods have been proposed to solve the moving object detection problem. Comprehensive reviews of this subject are available in [35] and [37], RGB-D based methods in [38] and stationary moving object detection is widely discussed in [6]. Therefore, in this section only recent and related works will be discussed.

Goyette *et al.* [15] categorised background subtraction (modelling) techniques into six families; basic, parametric, non-parametric and data-driven, matrix decomposition, motion segmentation, and machine learning.

One of the most common ways to detect the foreground of the scene is using parametric background modelling methods such as Mixtures of Gaussians (MoG or GMM) [9]. Normally, parametric methods are closely coupled with underlying assumptions and not always perfectly corresponding to the real data. Also, the choice of parameters can be bulky and consequently, it decreases automation [39]. GMM was later improved by Zivkovic to be more efficient in updating parameters known as adaptive GMM (AGMM) [13]. However, even this method fails in some sequences with complex dynamic backgrounds which parametric methods are not commonly able to properly model the dynamic backgrounds and pixel variations [40]. On the other hand, these methods are dependent on many parameters, which extremely reduces their usability [41]. For these reasons, some researchers expanded MoG and investigated a new way to detect static objects in the scene according to the foreground masks collected from two background models with different learning rates, a long-term foreground mask (F_L) and a short-term foreground mask (F_S) [42][36]. F_S model holds the moving objects and noises. On the other hand, $F_L$contains the pixel values which belong to temporary static objects or moving objects, illumination changes and shadows.

Despite the effort of these researchers, the MoG dual background model could not effectively detect temporarily static objects for a long period of time [36]. Instead of these complex methods, many researchers investigated other approaches, for example, the cluster-based models such as Codebooks [43] which background model is created from long observation of frames or other statistical methods such as consensus-based method (SACON) proposed by Wang et al. in [44] which estimates a statistical model of the scene by computing the consensus of the background samples. Instead of these compound approaches, nonparametric methods such as [29] are much easier to implement and these approaches are more effective models for complex dynamic scenes [32] which recently these methods become popular among researchers. Nonparametric models are referred to the methods which are tightly coupled with collecting and corresponding to the real data instead of assumption and estimation. This makes them flexible but severely data-dependent. The authors of [41] proposed a nonparametric background modelling to detect stationary moving objects in video sequences. This strategy is based on using a Probability Density Function (PDF). The background is modelled by two mixtures of Gaussians with the different learning rate. The state machine gives the meaning for the interpretation of the results taken from background subtraction. According to the authors, their proposed method classifies the pixels more accurately than MoG-based method introduced in [36]. Additionally, since they have used a nonparametric algorithm, their method could appropriately model complex background variations, whereas the detection proposed in [36] leads to many false detections. Their research demonstrates that nonparametric methods are able to deal with rapid background changes and complex dynamic scenes. Table 2.1 illustrates summaries of some state-of-the-art parametric and non-parametric methods based on colour only and RGBD data.

Table 2.1. *Classification of representative background modelling of Pixel-based methods.*

Category	RGB	RGB-D
Parametric Methods	MoG [9]	$\mathrm{MOG}_{\mathrm{RGB-D}}$[45], MOG4D [46], MOG_{bin}[47]
	AGMM [13]	CL_{W} [48]
Non-parametric Methods	PBAS [14]	
	ViBe [8]	ViBe $_{\mathrm{bin}}$[51], SCAD [52]
	SACON [44]	
	SOBS [49]	RGBD-SOBS [53]
	KDE [29]	GSM [7]
	DTNBM [50]	
Others	CodeBook [43]	DECB [54]
		MFCN [34]

In the above table, MOG [9] and AGMM [13] are popular parametric methods and $\mathrm{MOG}_{\mathrm{RGB-D}}$[45], MOG4D [46] are popular parametric methods based on MOG using RGB-D data. PBAS [14], ViBe [8], SACON [44], SOBS [49], KDE [29], DTNBM [50] are well-known nonparametric methods. All these methods will be discussed in more detail in this chapter.

According to previous studies such as [41], nonparametric methods have proven their accuracy and ability to accomplish high detection in some challenging scenarios such as complex dynamic background or in a rapidly changing environment. On the other hand, parametric methods such as MOG typically have some disadvantages including weak detection in the dynamic environments as the background with fast variations cannot be precisely modelled with only a few Gaussians. All these reasons inspired us to investigate nonparametric methods in greater depth. As a result, in the next section, state-of-the-art nonparametric algorithms will be discussed in further detail.

2.5 Nonparametric Methods

Typically, nonparametric methods statistically model the background by directly relying on the observed data [55]. While these methods can adapt to the fast changes in the background, it has some drawbacks such as high memory usage as well as time-consuming to process [2]. Many different nonparametric methods have been proposed to overcome these limitations. For instance, Elgammal et al. [29] developed a nonparametric method based on Kernel Density Estimation (KDE) from various recent samples. However, this method was still time-consuming. KDE is also improved in some other methods such as [56][57] to overcome these limitations.

Hofmann et al. proposed a nonparametric method called Pixel-Based Adaptive Segmented (PBAS) [14]. PBAS models the background based on the history of recently observed pixel values. The core part of PBAS is the decision block which is based on the per-pixel threshold to compare foreground with the background model and the existing frame. Additionally, the background model frequently receives an update to have the ability to deal with rapid background changes. This update mechanism is based on per-pixel learning parameter. The key innovation applied in PBAS approach is the estimate of the dynamics background which relies on per-pixel thresholds metric. Despite the outstanding performance of PBAS to gain detailed shapes of foreground objects, it can be affected by illumination changes, dynamic backgrounds and noise [58].

Maddalena and Petrosino [49] introduced a background modelling method known as and Self-Organizing Background Subtraction (SOBS) by using artificial neural networks. This method uses the neural background model by learning image sequence variations. It creates a neuronal map containing n × n weight vectors. Every incoming sample should be measured between weight vectors and also the minimum value must be determined, which makes this process time-consuming. This makes it computationally high cost and time consuming [58]. A comprehensive systematic review and comparative evaluation of SOBS method and its variants based on neural networks is available at [59].

Recently the authors of [50] introduced a dual-target nonparametric background modelling method called DTNBM which combines the grey value and gradient to represent each pixel and a classification rule that has been defined to classify a pixel as part of a dynamic background or motionless object. This method deals with low-speed and stationary moving objects by fast removing false detections which are produced by illumination changes or coarse background initialisation. Furthermore, DTNBM is computationally efficient and automatically adjusts the threshold level, which allows more applicability to various scenarios. However, the main drawback of DTNBM is that this method is not able to quickly identify shadows from illumination and therefore makes this system inefficient.

Barnich et al. introduced a new promising pixel-based nonparametric method called ViBe (Visual Background Extractor) to subtract the foreground from the background of the scene using an innovative random selection strategy [8]. This method has become popular among researchers due to its robustness and efficiency.

Example of an experimental qualitative comparison of state-of-the-art methods presented in [58] is illustrated in figure 2.2. This figure is the results of the evaluated methods in the CDnet2014 video dataset [49]. The scenarios shown in this evaluation are bad weather, baseline, camera jitter, dynamic background, and also intermittent object motion video, fighting and walking I [60]. Figure 2.2 (1) shows the original frame of the video and figure. 2.2 (2) is the ground truth data. Figure 2.2 (3-10) is the foreground detection results of background modelling methods. The methods in this evaluation are GMM (3), KDE (4), CodeBook (5), AGMM (6), SACON (7), SOBS (8), ViBe (9) and PBAS (10). In the first sequence (bad weather) KDE, CodeBook, SACON and SOBS failed to detect most pixels of the foreground. In baseline (b), all methods could detect the foreground area. However, the amount of noise in some of these methods such as KDE is extremely more than the others. In camera jitter (3) sequence, ViBe, SOBS, and KDE could achieve the best result and SACON totally failed in this sequence. In the sequence of dynamic background (4), SOBS could achieve the best result. Some methods such as SACON and KDE had difficulties in dealing with the water, which is a dynamic part of the background. In an intermittent object motion video, SACON totally failed to detect the foreground. In the last two sequences (fighting and walking I), ViBe, PBAS, and SACON could show robust accuracy compared to the other methods. Overall ViBe performed well and it proved that it is consistent in all scenarios.

ViBe algorithm is robust and can adapt to a dynamic background. Also, it can reveal exact background changes in current frames which it is facilitated to achieve one of the best results in comparative evaluation demonstrated in figure 2.2 [58] and outperform among many other nonparametric state-of-the-art methods such as KDE [29], CodeBook [43] and SACON [44]. According to the author of [35], ViBe could achieve the best results in the ChangeDetection.net dataset. This shows ViBe has a stable detection rate among different datasets. The ViBe algorithm was further investigated by Van Droogenbroeck and Paquot [61]. They have considered additional changes to progress the performance of ViBe. As ViBe has shown high accuracy and consistency in different scenarios, in the next section, we will review this algorithm in more detail and investigate the performance and weaknesses of ViBe in different situations.

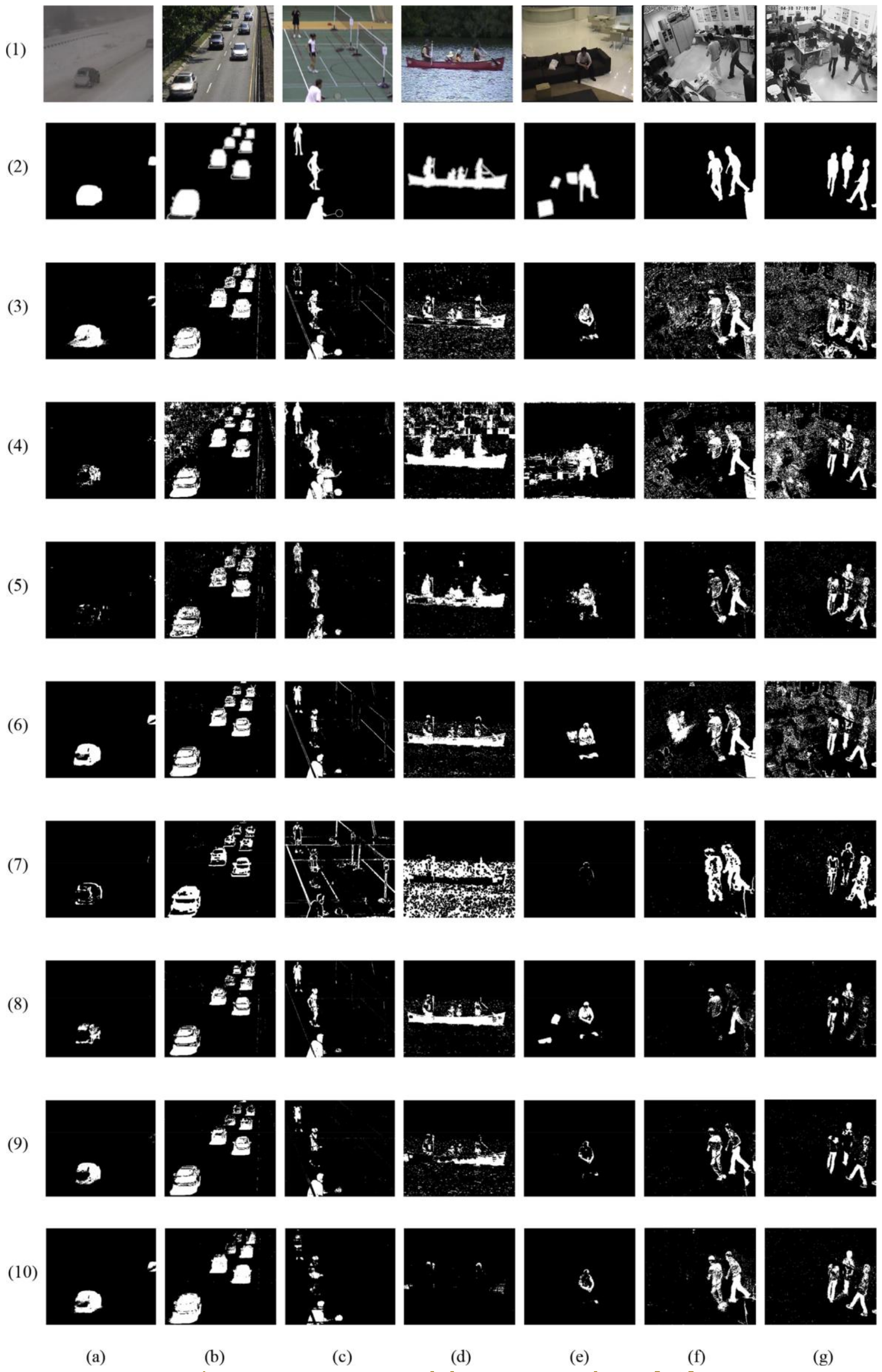


Figure 2.2. *Foreground detection results in* [58].

The CDnet2014 dataset [49],*(a) Bad weather,(b) Baseline, (c) Camera jitter,(d) Dynamic background,(e) Intermittent object motion, (f) Fighting, and (g) Walking I video from the dataset in* [60] *(1) Original frame. (2) Ground truth. (3) GMM. (4) KDE. (5) CodeBook. (6) AGMM. (7) SACON. (8) SOBS. (9) ViBe. (10) PBAS.*

2.6 The ViBe Algorithm

The ViBe algorithm initially produces a model by collecting previously observed pixel values at each location. Then it compares all pixels in the new frame with the created background model in order to identify each pixel as foreground or background. ViBe also benefits from taking two update methods in the processing stage by including recently observed pixels directly in the model to help the algorithm to adapt to the changes in the background instantly. The ViBe algorithm commonly exists in background subtraction applications for moving object detection as it proved that it is a fast and effective approach in many real-world challenging situations such as dynamic backgrounds. On the other hand, in benchmark evaluation ViBe proved that it is robust to artefacts stemming from irregular motion (Camera jitter) and background motion compared to the other state-of-the-art algorithms [62]. In addition, it is simple to implement. However, ViBe also suffers from some weaknesses in a few challenging scenarios which could result in incorrect classification of pixels as background or foreground. Consequently, this will lead to detection failure. An example of these situations is sudden illumination changes and shadow production in constant background changes [35]. Also, ViBe algorithm could easily generate ghost phenomenon during the completion of moving object detection [63], especially dealing with slow and stationary moving objects. In [64] Ghost is defined as "a set of connected points detected as in motion by means of background subtraction, but not corresponding to any real moving object". Figure 2.3 illustrates an example of a ghost created by ViBe in [63] to demonstrate the segmentation result of the original ViBe algorithm.

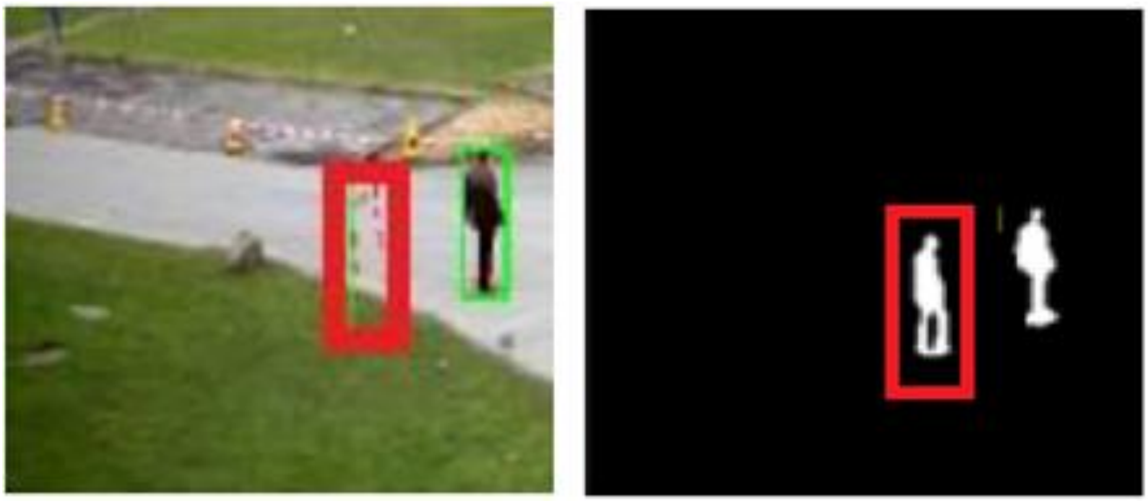


Figure 2.3. The appearance of a ghost in ViBe background subtraction result in [63]. only one person exists in the scene, but ViBe detected two moving objects, the ghost area manually marked with a red rectangle in both images.

The main reason ghost has appeared in the red rectangle is that the foreground pixels absorbed to the background model by the blind update. In this example, the man (foreground object) remains stationary for some time and therefore many foreground pixels absorbed to the background model over time. When he then moves to another location, ViBe incorrectly detects the background as a foreground because the background model does not match the real background. This detected object which doesn't exist in the red rectangle anymore is called ghost. Figure 2.3 (a) shows the colour frame that only one person exists in this frame and the red rectangle in this image indicates the moving object which ViBe has detected by error (ghost). Figure 2.3 (b) shows segmentation result achieved by ViBe which clearly contains ghost area which manually marked with the red rectangle.

To remove these limitations, many researchers have introduced an improved type of ViBe algorithm. For instance, some methods have been introduced to solve this problem by using a scene model of larger size. However, the drawback of such a system is that it requires high memory usage or time. In [63], the authors upgraded ViBe algorithm to eliminate the effect of ghost area based on the concept that the histogram of those fragments with existent of ghost has the same distribution characteristics. However, when a real object is moving, the distribution of these characteristics continually change. The original ViBe algorithm has been modified in [65] to be able to detect slow-moving objects without appearing as a ghost. This method is called VIBeF and it creates the foreground model with an adaptive approach, which is proposed to support ViBe with a blind update. The authors concluded that the ViBeF method can adapt to complex scenarios including illumination changes and variations in background objects.

The authors of [66] tested the ViBe algorithm under low lighting conditions (it was reduced approximately by half). They found ViBe algorithm failed to detect a moving object in this condition while GMM and edge-based [67] methods in the same conditions could still detect the moving object with a higher amount of false positives. After the lights were turned off, the accuracy of both ViBe and GMM became utterly unreliable. Although the method they proposed is capable of detecting the moving object in this condition, the output result is still weak and it requires further improvements.

Detecting low light or the shadow of a moving object on the floor or wall is a challenging task and can significantly reduce object detection accuracy and can also increase the possibility of tracking failure. Therefore, a shadow is an important factor in benchmarks to measure the accuracy of object detection algorithms. In the last few years, many researchers have investigated shadow removal algorithms to improve the detection rate. An example of these algorithms is gradient amendment, edge-based, histogram, etc..[68][69][70]. These approaches typically are faced with some complex circumstances such as an area with a low light condition, lack of data in the darkness or entirely a change of chromatic properties of the context. Although experimental results show some improvement in detecting shadow, the average rate of shadow detection is not able to fully meet the standards in real-world practical requirements. Major issues of current shadow detection approaches are detecting only a fraction of moving objects. This mainly applies to pixel-based methods such as texture-based and chromaticity-based methods [70]. An alternative way to improve the detection accuracy in colour camouflage, shadow and illumination changes is to use depth information as chromatic changes don't affect depth data. Therefore, in the next section, the effectiveness of RGB-D camera and state-of-the-art algorithms using depth and colour will be investigated in detail.

2.7 RGB-D Sensors

The current background subtraction methods have achieved substantial success in many tasks. However, these methods only perform well under stable situations and could fail in some changes such as fast-moving objects in the cluttered background (e.g. moving curtains), a sudden illumination (etc. sunlight or change of light) and changes in background objects (e.g. moving a table from one side to another side).

In the last decade, various object detection methods have been proposed to solve the problem in sudden illumination changes [26][71][72][73]. Generally, these methods have a training stage after each change and these trainings are usually computationally high cost. One of the main possible solution to reduce the impact of the illumination changes is to use physical information of the scene. For example, geometrical descriptions of the building can be added to the model to detect the shadows on the wall or the floor [74].

Depth sensors produce geometrical information of the scene’s area which observes by the sensor. This could help to resolve the mentioned problems. Depth sensors create depth frames and every pixel in the depth frame contains a depth value related to the approximate distance from the corresponding point in the real world to the device. Depth information of the scene can be obtained from stereo vision devices [75], camera networks [76], time-of-Flight (ToF) [77] and structured light [78]. At present, the creation of low-cost RGB-D sensors such as Microsoft Kinect which combine an RGB colour camera with a depth sensor has completely affected the computer vision projects and attracted many researchers to detect and recognise objects and behaviours such as motion [79], suicide [80], drone [3] and obstacle detection [81].

These devices are able to produce well-calibrated RGB and depth frames and normally capture 30 frames per second which is appropriate for motion detection algorithms.

RGB-D devices are using different technology, for instance, Microsoft Kinect version 1 and Asus Xtion Pro Live are using structured light to produce depth frames. However, in the new version of Kinect, ToF sensor has been used to capture the depth frame [82]. Figure 2.4 illustrates the details of Kinect version 1 described in [83].

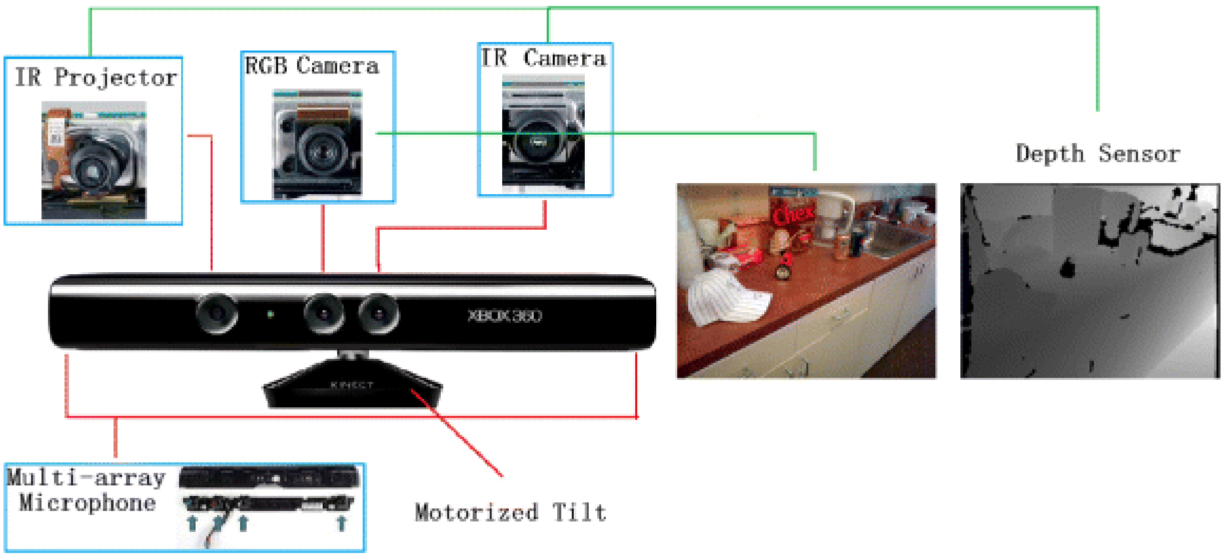
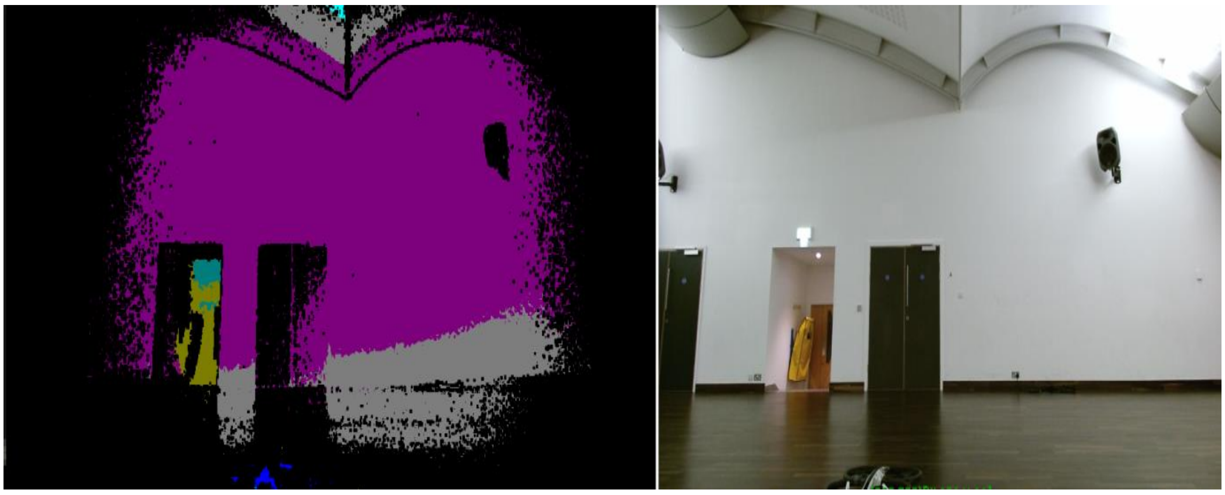


Figure 2.4. *Detail of Microsoft Kinect version one presented in* [83].

Many researchers in recent years have implemented these devices in their system which can capture colour and depth frames at the same time at 30 frames per second. Depth data as part of these devices are very attractive and suitable for systems based on detecting a moving object.

In the last decade, many researchers have proposed different systems in video surveillance to segment background of the scene from the foreground by using depth data and colour information [7][45][84][85][86][87][88][48]. Generally, depth data allow us to capture the shapes of objects which are not affected by shadows, interreflections, and illumination changes. Therefore, depth information could provide helpful information in such a phenomenon. On the other hand, depth sensors have their own limitations. Consequently, the results from background subtraction methods built on only depth data are often invalid due to the limitations of depth data [89][90]. Depth data is generally noisy and have restrictions for some surfaces which in the literature mainly is known as “holes” [91] or “Absent Depth Observations (ADO)” [7]. These limitations are mainly based on some physical phenomena such as depth shadows, the production of depth camouflage, absorption by black objects, absent observations, lower sensitivity at longer distances. Figure 2.5 demonstrates the amount of noise that could occur in a single depth frame and any black pixel in this image showing holes (ADO).

For example, the black speaker absorbed the signals because of its colour and therefore it is defined as ADO or in the cavity, some depth pixels are not available due to the physical appearance of the scene. Meanwhile, the size of depth frame is usually smaller than the colour frame. The black pixels on the edge of the depth frame are part of the outer boundary. Additionally, some pixels reached the maximum distance of the sensor or the IR signals are not back to the sensor due to the reflection of the surface. Consequently, the sensor is not able to show any value for those pixels. For these limitations, it has been assumed that it is useful to have both colour and depth measurements to cover each other's weaknesses in different challenging situations.



(a) (b)

Figure 2.5. *the amount of noise that could occur in a single depth frame.*
(a) Black points in the image show holes (ADO) and other colour coded shows the distance in depth frame, (b) Colour image.

2.8 RGB-D Methods

In the rest of this section, a brief description of the reviewed algorithms will be presented in subject order. The review will mainly concentrate on RGB-D background subtraction methods and will not be dealing with researches in other high-level systems (e.g., motion recognition, object recognition, human tracking or etc).

Many researchers in recent years have tried to improve the accuracy of background subtraction results by a combination of colour and depth information captured by RGB-D sensors. These methods are mainly divided into two different approaches. In the first type, two independent segmentations are carried out. One on the colour image and the other one on the depth data and the two results will be merged. Those methods which used this approach mainly extend the well-known RGB moving object detection methods which are originally designed for only colour frames. The second approach fuses the RGB-D data (colour and depth) before undertaking a joint segmentation [86].

Gordon et al. in [45] proposed a method which is an adaptation of the MoG algorithm to depth and colour information captured with a stereo camera. This method is known as $\mathrm{MOG}_{\mathrm{RGB-D}}$ which is a mixture of four-dimensional Gaussian distributions modelled for every background pixel. This consists of three channels which belong to the colour information (the YUV colour space) and the fourth channel for the depth data. The same update method of the original MoG is applied to update distribution parameters. Also, depth and colour independently are considered in this method. The system relies on depth-based decisions, when depth data is accurate, decreasing the colour camouflage misclassification. On the other hand, once the depth matching is not accurate, the colour-based decision has difficulties in dealing with some challenges such as illumination changes or shadows.

MOG is also adopted in the method proposed by Stormer et al. [47], where depth data is obtained by a ToF camera with the combination of the infrared and the range data channels to detect foreground objects. Two separate background models are constructed, and every pixel is identified as foreground when both models agree with this condition. Additionally, near or overlapping foreground objects are also divided by a depth gradient-based method. The author suggests that a combination of this method with a tracking system could increase the robustness.

Some other techniques are also introduced based on a Mixture of Gaussians and a combination of RGB and depth data. For example, a 4D version of MoG (MOG4D) [46] is introduced. However, this method also suffers from production of ghost and could not solve the problem of moving object detection.

Some other researches have investigated other types of background subtraction methods in RGB-D systems. For example, the author of [54] studied the codebook method in RGB-D system called Depth-Extended Codebook (DECB). This method was also not successful due to the production of the ghost. Also, a new background subtraction method known as BGSNet-D is introduced in [92]. This method is based on convolutional neural networks (CNN) and only use depth data. It designed mainly to deal with scenarios where the color frame is unavailable such as poor lighting conditions or darkness. In order to be effective in all scenarios, it needs to combine with existing RGB background subtraction methods. However, the outcome of these combinations is unclear.

On the other hand, in the recent years, various background subtraction algorithms based on multiscale convolutional such as MFCN [34] or using classifier such as [93] are proposed where these methods have shown great accuracy but are not capable of being implemented in a live application which reduces the applicability of these methods.

Maddalena and Petrosino introduced a new segmentation method based on RGB-D called RGBD-SOBS [53] and extensive experimental results on this method have been accomplished in [94]. Two different background models are built for colour and depth data. It uses a self-organizing neural background model which was implemented earlier for RGB sequences in [95]. The final detection mask will be accomplished by combining the results of colour and depth segmentation masks. The authors believed RGBD-SOBS could improve the results and achieve high accuracy compared to only using colour frames in scenarios which colour camouflage occur and also other colour and depth background challenges such as bootstrapping, out of sensor range data and intermittent motion. However, this method requires further work to handle depth camouflage scenarios. Bootstrapping sequences are defined as those sequences that foreground objects exist in all frames from the beginning [12].

Similar to colour methods, many researchers start investigating non-parametric approaches in RGB-D systems. For example, the authors of [51], introduced a new ViBe strategy based on using ToF (Time-of-Flight*)* sensor and colour frame. Individually for each sensor, a model is built and foreground result from each model combines with logical operations. Segmentation results showed that the colour and depth data could improve each other's limitations. For example, depth can help in the areas which colour masks mainly fail. This could be illumination changes, shadow, or when the foreground has a similar colour to the backgrounds. Instead, the colour segmentation result can produce more valid background mask when the foreground is very close to the scene or the depth frame is too noisy. However, a combination of colour and depth information has a few downsides which usually cause background pixels to be incorrectly identified as foreground. An example of these problems is infrared shadows created by the sensor. Resolving these issues was the main aim of the authors in [96]. They have introduced a method to effectively improve the ViBE algorithm using both depth data and colour frames. Despite more precise results, they complained about the challenging alignment between the PMD (Photonic Mixer Device) camera and RGB camera in this system.

ViBe also extended in [97] to introduce a background subtraction algorithm based on RGB-D information. At the initialisation stage, two background models are created individually for colour and depth data. Then, for foreground detection, a two-step procedure strategy of adaptive multi-cue combination has been introduced to accomplish this task. Also, weighted fusion applied to make the coarse outcome in the first step and apply adaptive refinement with spatiotemporal consistency in the second step to receive the final result.

In [52], a new method proposed to deal with challenging scenarios such as depth camouflage and colour camouflage known as a simple combination of appearance and depth information (SCAD). This has been implemented by adding the two likelihoods of the background using background subtraction based on the appearance and depth information. Then, optimisation-based function on the two likelihoods of the background using graph cuts was applied to get the segmentation mask. To improve the detection of ghost phenomenal and shadows, the authors of [98] proposed a new RGB-D background subtraction method based on ViBe algorithm which combines the colour and depth information to segment foreground mask. First, a depth model and a background model are created. A new update strategy of these models has been introduced to efficiently eliminate ghost phenomenal by the help of depth data which can obtain the distance of any object in front of the sensor. However, the accuracy of the method in other challenging scenarios is unclear.

In [48], Camplani and Salgado introduced a per-pixel background modelling approach based on colour and depth data. The background model is built on a mixture of Gaussian distributions. This method merges different statistical classifiers which enable the system to improve the accuracy of the detected foreground mask. The mixture of the two classifiers is joint together by using a weighted average from the result to consider the characteristics of colour and depth information for each pixel. We will refer to this method as CL_W in the rest of this thesis. The last stage of their segmentation is assembled by using the edges of colour and depth frames and also the previous foreground mask. Canny edge detection algorithm [99] added to the method to find the edges in colour and depth data. The colour classifier has a higher influence on the segmentation of those pixels belonging to the object boundaries as depth sensors are not reliable around object boundaries. This will help the system to decrease the effect of noise in pixels around object boundaries in depth frames. Alternatively, the depth-based classifier has more effect in low gradient pixels on the final segmentation. The authors stated using depth data allows solid foreground detection and decreases the amount of errors, especially in illumination changes and shadow scenarios. These authors in [100] presented another method based on the combination of several region-based classifiers. They found that the accuracy of this method is not in standard level in some cases such as a very fast-moving object as it is using the previous segmentation result to find the areas where the foreground can exist.

Generic Scene Modelling (GSM) is another nonparametric approach based on colour and depth data introduced in [7]. For an individual pixel of the scene, background model erected by Kernel Density Estimation (KDE) process and a Gaussian Kernel. Unlike the model in GMM, KDE doesn't require estimating the mixture parameters. This allows estimating the density function without any guesses or assumptions for density model. Accordingly, this model is only based on recent information observed from the scene. One-dimension model built for a depth data and a three-dimensional kernel for colour information and two other models for normalised chromaticity. In this model, the oldest sample is discarded, and a new sample is added to the background model. This update mechanism is called first-in-first-out.

In this section, we have studied the advantages of using a depth camera to produce the physical information of the scene without the effect of illumination changes. However, depth data also have some limitations such as depth camouflage and absent of depth value in some surfaces or distances. Therefore, it has been assumed that the combination of RGB and depth camera is necessary to cover each other weaknesses. Most of the current RGB-D methods have extended the well-known RGB moving object detection methods which are originally designed for only colour frames by producing one result on the colour image and the other one on the depth data and the two results are merged by a logical operation. In the next section, challenging scenarios which background detection algorithms typically have difficulties to fully detect the foreground objects will be investigated.

2.9 Challenging scenarios in RGB-D

Most of background subtraction algorithms are able to accurately detect a moving object in a steady situation where the position of the camera is fixed, the background is static and illumination is constant. However, in real-world, we may face some challenging situations which could significantly affect the performance of motion detection algorithms. Thierry Bouwmans [35] identified 13 challenging situations in the field of video surveillance which are the following:

- **Noisy image:** Occurs when the quality of frames are low. This could be due to images acquired by a webcam or low-quality cameras.
- **Camera jitter:** The wind or other factors could cause the camera to move. Consequently, it can cause movement in the sequences. Due to the motion, foreground mask could contain false detections.

- **Camera automatic adjustments:** Nowadays, the majority of digital cameras automatically adjust the colour levels of the frames in the sequence. An example of these technologies is automatic white balance, brightness control, automatic gain control, autofocus.

- **Illumination changes:** Sudden change in the light or sometimes a gradual change such as sunrise and sunset.

- **Bootstrapping:** The moving object exists in the background from the beginning including the training frames. Therefore, it is challenging to compute a representative background.

- **Camouflage:** The pixels of the foreground have similar pixel values compared to the background. Therefore, it is challenging for algorithms to detect foreground pixels.

- **Foreground aperture:** The foreground has uniform coloured regions and changes may not be identified in these regions. Consequently, fraction of the foreground could be detected by motion detection methods.

- **Moved background objects:** The position of the objects in the background could be moved. However, this movement should not be considered as part of the foreground.

- **Inserted background objects:** An object can be inserted in the scene which is part of the background region.

- **Dynamic backgrounds:** Some background objects are vacillating in the scene and should not be considered as part of the foreground.

- **Beginning moving object:** The primary movement of the object in the background may cause the motion detection algorithm to detect both the real object and the previous location of the object which is known as “ghost” in the literature.

- **Sleeping foreground object:** The moving object could become stationary for some period of time which is challenging for motion detection algorithms to distinguish it from the background.

- **Shadows:** Shadows could be identified as part of the foreground as it can completely change the pixel values. Extensive research has been carried out to detect the shadow from the foreground [68][69][70].

In addition to the mentioned challenges, recently a few challenging scenarios have been identified for the depth camera. According to the authors of [38], these challenges are as follow:

- **Depth Camouflage**: The moving object in depth is very close to the background. Consequently, the sensor returns the same values for background and foreground pixels which will be challenging to produce a foreground mask based on only depth data.

- **Out of Sensor Range**: Objects in the scene are too close or too far from the sensor. Consequently, the sensor cannot measure depth values as it is more or less than the maximum and minimum specifications of the sensor.

- **Depth Shadows**: Alike the colour, depth shadows are produced by a moving object which blocks the IR light emitted by the device from going to the background.

- **Specular Materials:** Specular objects exist in the scene which reflected back the IR rays from incoming direction to a single outgoing direction.

Testing the object detection algorithms under these challenging scenarios is the best way to measure the consistency and real accuracy of these methods. Some methods are able to handle some of these scenarios very well and totally fail in other situations. Therefore, in this thesis, we are going to measure the accuracy of the proposed methods and other state-of-the-art algorithms under a selection of these challenging scenarios.

Figure 2.6 illustrates an example of depth data problems highlighted in [38]. In (a), the hand of the person is very close to the cabinet and this causes depth camouflage to occur. The depth value of pixels in the cabinet is almost the same as the pixels on the hand. Therefore, it is difficult with only a depth frame to detect the hand. In (b), a depth shadow occurred behind the box as the sensor couldn't measure the depth values in that area. In (c), the sensor cannot measure the distance of the window and therefore created ADO in that area. In (d), some region in the room has a longer length than the maximum range of the sensor. Thus, the region remains as ADO. An ideal moving object detection would be a method which is able to fully detect the foreground in all these scenarios in a live application. However, such a method doesn't exist yet. Typically, most of the current methods perform well only in some of these challenging scenarios and have difficulties in detecting the full foreground silhouette in other scenarios. Thus, it is crucial to discover the weakness of current methods which have been discussed in this chapter and also will be shown in the next chapters in the evaluation and results section. Then we are going to propose new techniques in the next chapters to improve the weakness of current methods. The proposed method should increase the accuracy in these challenging scenarios. In order to validate this, publicly available datasets containing these challenging scenarios will be used for comparison and evaluation.

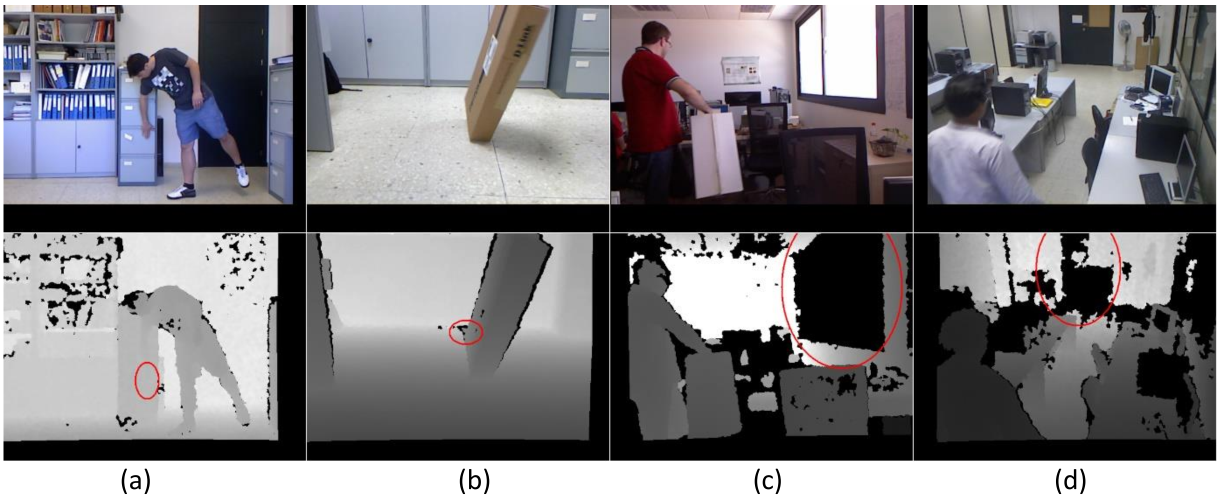


Figure 2.6. *Related depth data problems (highlighted by red ellipses)identified in* [38]. *(a) Depth camouflage. (b) Depth shadows. (c) Specular materials. (d) Out of sensor range.*

2.10 Conclusion and discussion

In this chapter, different algorithms to detect motion and changes in the scene have been explored. Also, the related and similar work to our proposed methods in this thesis is reviewed based on nonchronological order. Among these techniques, background subtraction is the most common type of motion detection. This approach generally requires a model to store observed data. Different variety of background subtraction methods are introduced to implement the model and segmentation procedure. One of the most popular background subtraction methods is nonparametric approaches such as ViBe and PBAS. Many previously stated studies in this chapter have shown growing interests in these methods for the advantages and accuracy. These methods are robust and can achieve high accuracy in most scenarios apart from a few conditions such as poor lighting, sudden illumination changes and production of ghost phenomenon. All these conditions could be addressed by adding depth information of the scene which potentially can be captured by depth sensors. Depth data could help these methods to significantly improve mentioned weakness and potentially increase the overall accuracy of the moving object detection. However, depth data also suffers from some weaknesses. The combination of colour and depth data could compensate for each other’s weakness. This hypothesis motivated us to use a depth sensor as well as a colour camera in our research in order to improve accuracy.

We have discovered in the literature review that the current state-of-the-art RGB-D algorithms have some weaknesses and they are still unable to detect moving objects in all challenging scenarios. Therefore, further work is required in this area. As colour cameras are significantly affected by illuminations and nonparametric methods have shown great accuracy, in the next chapter, we are going to investigate nonparametric methods alongside using RGB-D sensors. In fact, we are going to address the following research question, how pixel-wise nonparametric RGB-D approach can improve the accuracy of moving object detection?

To address this question, we need a method based on colour and depth data to cover each other weaknesses and consequently be able to tackle this problem in more challenging scenarios than current methods. Therefore, in the next section, a new nonparametric method will be introduced based on colour and depth frames to detect the moving objects in more challenging scenarios including a fast-moving object such as drones, illumination changes, colour camouflage, as well as improving the weaknesses of previous methods such as ghost phenomenon. This method is original, and it is the first contribution to this thesis (described in section 1.2). Results show this method outperform other state of the art algorithms in these challenging scenarios.

Then in chapter 4, we are going to expand the proposed method to increase the accuracy in some other challenging scenarios which known as Stationary Foreground Objects (SFOs), bootstrapping and depth Camouflage in the literature. In bootstrapping, normally the moving object exists in the scene from the first frame (during the initialisation) therefore, the foreground will exist in the background model and only a fraction of the moving object will be detected by the algorithm. In SFOs scenarios, the stationary or slow-moving object slowly absorbs to the model by the update procedure and gradually it will disappear from the detection mask. We are going to tackle these issues by proposing an adaptive blind update policy, detection of bootstrapping and changes in the background, and shadow detection method. In particular, we are going to investigate the following research question, what is the effect of using adaptive blind update policy on pixel-wise moving object detection? Addressing this question helps to resolve the weakness of the original method in scenarios where stationary and slow-moving objects exist and assists the method to deal with a variety of objects such as stationary, slow and fast-moving objects. The adaptive blind update policy tracks the moving object and adjusts the frequency of the blind update based on the speed of moving object. In addition, if the system detects any moving object during initialisation, then bootstrapping has occurred, and the system needs to find the whole moving objects. Also, illumination changes and shadow detection method has been proposed for this method in order to improve detection accuracy in case of shadow, illumination changes and depth camouflage. To achieve this, CIE L*a*b* colour coordinate for the colour frames has been used. Unlike the RGB space, L*a*b* colour is intended to approximate human vision where the L element closely indicates the

human perception of lightness. Thus, it can be used to check the colour value of pixels (a and b) without interfering of illumination. This method will be extensively evaluated in SBM-RGBD datasets which has 33 sequences under seven different challenging categories. The main disadvantage of the proposed method is that the system has a higher computational cost compared to the original method. Therefore, in chapter 5, instead of using a simple update mechanism, we have implemented a more sophisticated method using EC optimisation algorithms such as Hill Climbing and Genetic Algorithms to search and find the optimal background model. This improved the detection accuracy and efficiency of our method. In particular, in chapter 5 we are going to investigate the following research question, what is the effect of using heuristic searches in pixel-wise moving object detection?
The results and evaluation indicate improvement in the accuracy and efficiency over the original method.

Overall, in this chapter, we have discovered that current methods are unable to detect moving objects in all challenging scenarios and therefore the aim of this thesis and these three research questions will be improving the accuracy of moving object detection in more challenging scenarios in live applications.

Chapter 3
Nonparametric background modelling and segmentation to detect Micro Air Vehicles (MAV) using RGB-D Sensor

3.1 Introduction

In the last decade, the Autonomous Unmanned Aerial Vehicles (UAVs) have seen rapid progress by development in hardware and miniaturisation of microchips such as low-power micro radio devices, economical airframes, microprocessors, motors, and high-power density batteries. Small and micro UAVs are now able to perform different tasks in an indoor and outdoor environments such as item delivery, traffic monitoring, search and rescue, remote sensing, mapping [101], gesture-based control [102] and etc. Initially, UAVs were developed for military purposes as it is able to do surveillance and penetration into enemy zones without any risk of losing men in case of attacks [103]. In addition, Micro UAVs are relatively fast, lightweight and planned to fly at low altitude.

Currently, many researchers in the field of computer science and robotics are investigating UAVs to improve the functionality and enable these vehicles for more advanced tasks. Among these vehicles, quadcopter or also called a quadrotor is one kind of UAV that is often used as a research object. This kind of UAV has four motors and propellers which allows having a simple control mechanism, high manoeuvrability, take off, hover and land in a narrow area [104]. Figure 3.1 illustrates an example of two types of popular quadcopter (Parrot AR.Drone and Crazyflie).

One of the main basic requirements of these vehicles is localisation and tracking which is still challenging in GPS-denied environments. These vehicles are usually controlled and evaluated by external motion tracking system such as the VICON motion tracking system [105][106][17] or onboard visual sensors [107][108][109][110].

Recently, using external sensors in GPS denied environments has attracted many researchers interest [17][16]. In these methods, localisation of detected UAV is crucial for collision-free path planning. While different techniques have been proposed for static objects, localisation and detecting of dynamic objects such as UAVs are still challenging and hard to implement due to the limitations of sensors.

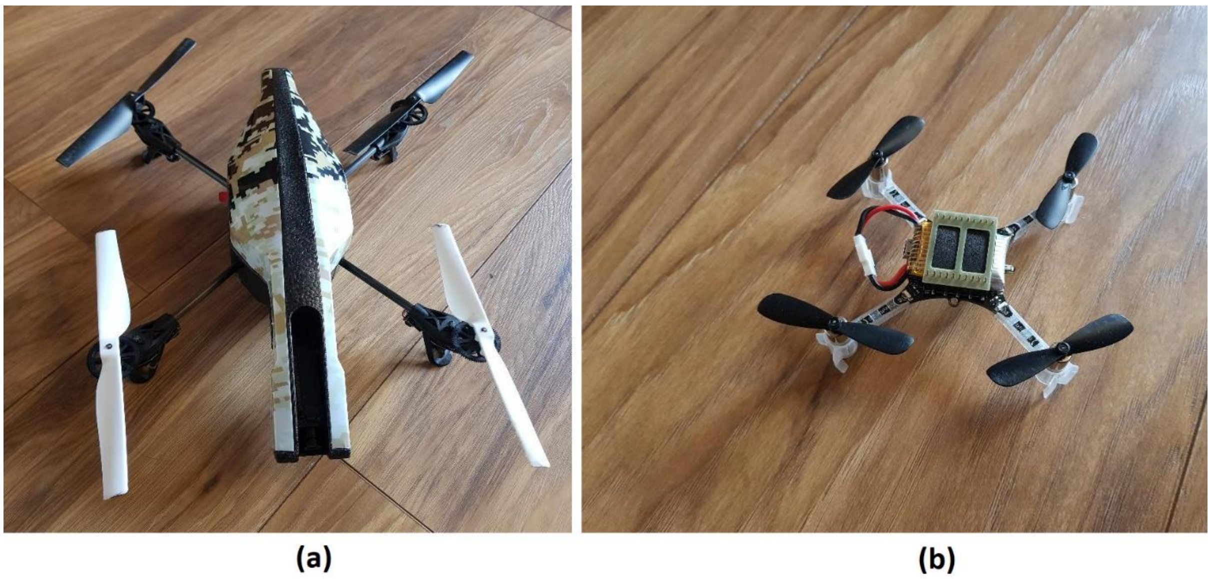


Figure 3.1. Two types of different quadcopter.
(a) Parrot AR.Drone, (b) Micro quadcopter(Crazyflie).

A new evaluation system has been introduced in [16] where they have used RGB-D Kinect sensor for 3D measurements instead of Vicon motion capture system which commonly used for verifying algorithm of UAV to control and self-localisation in the indoor area. For evaluation purpose, they have also applied a marker in order to recognise the object. Figure 3.2 illustrates their proposed method which is based on three steps. First, the Gaussian Mixture Model (GMM) is used to subtract background and foreground in the colour frame in order to find the Region of Interest (ROI). In the second step, subtracted foreground from RGB coordinates converted to depth coordinates. In the last phase, labelling filter applied to identify a marker in the region of interest (obtain real 3D position). The feasibility of this approach has been validated in a real experiment of using two kinds of UAVs. The authors have proved that position tracking for the horizontal and vertical movement of a quadcopter is possible. However, the authors complained about some issues and limitation of the proposed method such as fluorescent lights, the accuracy of the position tracking and limit of the recognition range of quadcopter’s location at distances of 1 to 3 meters.

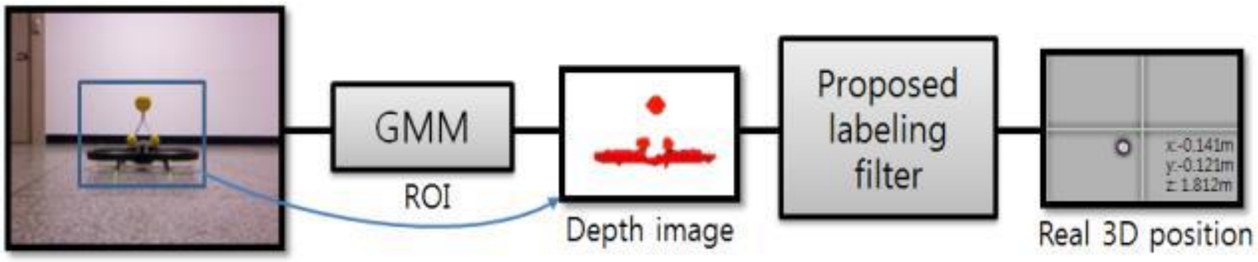


***Figure 3.2**. Proposed position tracking/marker recognition algorithm in* [16].

In this chapter, the goal is to propose a new method to address some of these limitations. In particular, we are going to investigate the accuracy of the proposed method in micro air vehicles (MAVs) detection and position tracking in challenging scenarios such as illumination changes in high speed moving MAVs.

In order to achieve this goal, we have introduced a new object detection method based on motion detection algorithm using colour and depth data to produce the segmentation result. Our proposed method stores some colour and depth frame as a model. It then compares each pixel from the new frames with the models in the same pixel location to identify the pixel as part of the background or foreground. When the models have been created, they regularly need an update to adapt to the changes in the scene.

To perform these updates, once the pixel is found as part of the background, the system updates the model by finding the closest pixel to the camera and substitute it with the current pixel if the new pixel is in the same or further location. To the best of the author's knowledge, this segmentation method has never been tested before. The approach to update the background model discussed in this chapter is different from other classical methods which are updating the samples in the model with the new frames based on different criteria. For examples using randomly substitutions, mean or first in first out.

In addition to the regular update, a blind update is also added to the model, in order for the system to adapt to the sudden changes in the background by updating the background as well as the foreground pixels. After a sufficient number of sequences, for each pixel, the background model swaps one of the samples in the model randomly with a pixel from the current frame in the same location regardless of being foreground or background. Then, the proposed method is compared to other state-of-the-art methods. Results have shown that this method is more accurate in object boundaries and it can tolerate more illumination changes. Our proposed method in this chapter is published in [3] and some contents of this chapter are from this paper. In the remaining part of this chapter, in section 3.2, a detail description of the proposed background subtraction method will be discussed. In section 3.3, the proposed method will be compared with the other state-of-the-art algorithms. In section 3.4, experimental results in the live application have been described in detail and the summary of this chapter is in section 3.5.

3.2 Proposed background subtraction

Our proposed method follows a nonparametric background modelling pattern, similar to previous works such as ViBe [8] and PBAS [14]. Consequently, the background model obtained by a history of previously observed pixel values and the foreground segmentation depends on a threshold amount.

Using nonparametric methods such as ViBe algorithm with both colour and depth data is not totally new since this approach has been already applied for moving object detection applications [51][96][86]. However, in [51] and [96] their vision system is made up with an RGB camera and separate TOF (Time of flight) sensor. These systems need experimental calibration and align both frames which is computationally heavy and difficult. Instead, some authors like [86] used a more straightforward way of having only a standard stereo camera.

Recently with the rise of low-cost RGB-D camera, researchers have started to use these sensors as it can produce better calibrated RGB and depth frames. These devices can capture up to 30 frames per second which would be beneficial for motion detection algorithms. These great benefits encourage us to start using RGB-D camera for the proposed background subtraction technique. Our method is original and it is using new segmentation rules consisting of different steps to be able to successfully apply in live applications and to cope with the changes in the background. This is the first contribution to this thesis. Figure 3.3 illustrates the flow chart diagram of the proposed algorithm.

The system stores the first N number of frames in “system initialisation” step to create colour and depth background models. “ADO Removal” will apply to individual depth frames before going to the model to eliminate all the unknown values in the depth frames. Once the initialisation has been completed (after N frames), the “Background models” will be ready and the system moves to the main loop. Each pixel of the new frame will be compared with the models to identify as foreground or background, this step is called “ Bg/Fg segmentation”.

Those pixels identified as a background will be guided to receive an update in “ background update” stage. Additionally, after α number of frames, the system will use a blind update to randomly swap foreground as well as the background pixels with the models in

"Blind update". In the remainder of this section, the key steps of the below diagram (Figure 3.3) are discussed in more detail.

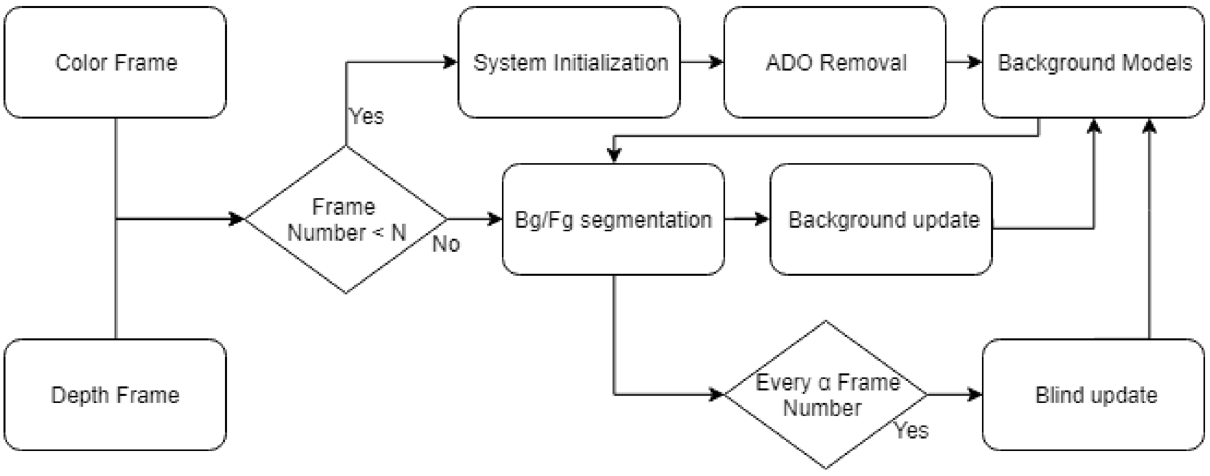


***Figure 3.3**. Flow chart of the proposed object detection method.*

3.2.1 System initialisation

Background subtraction methods usually need a scene model to enable the system to compare and segment the regions of the new frames as a background or foreground. Meanwhile, every model requires an initialisation process which has enhanced the importance of numerous popular methods described in publications, such as [29] which need various frames to initialise their model. These approaches can gather various amount of data which enable us to estimate the temporal distribution of the background pixels. However, these methods require a specific large number of frames to initialise the method before starting the segmentation. Therefore, these methods have difficulties in detecting the foreground in videos with a small number of frames [7]. On the other hand, other methods such as [8] need plenty of time to complete the stored model.

The possible answer to these issues could be introducing an update model process which adapts the models to the different lighting conditions. However, sudden illumination could completely change the chromatic properties of the context and even using such a dedicated update process could fail.

The authors of [8] introduced an appropriate technique for this issue which initialises the background model from a single frame and gradually building more sample models. Even this technique is not able to cope with sudden illumination changes such as the shadow of a moving object. A more convenient solution to these issues is to use depth images which helps us to understand a change in the physical position of each pixel in the real world. Therefore, in order for the system to be able to handle the sudden illumination changes, depth data added to the RGB in the proposed method.

Depth information is supposed to represent a steady long-term description of the scene. Therefore, theoretically storing one model of the scene may be sufficient for the background model. However, we experienced that cheap sensors like Microsoft Kinect have a considerable amount of noise. To find the most accurate depth measure, we store the same number of depth frames as colour frames. This allows the system to minimise the effect of noise by taking more depth samples. On the other hand, this amount of frames can be handled by the current hardware in a live application.

Unlike other approaches which need plenty of time and frame for initialisation, our method required to finish initialisation very quickly and start tracking the moving object as soon as possible. Therefore, the system blindly stores the first *N* number of colour and depth images as a model and then gradually modify the model during the update stages. This will allow us to start tracking our object rapidly. Different number of N from 10 to 40 has been tested for this method during the research. We recommend *N*= 20 samples as experimentally we have learned this is a reasonable amount of sample for this method. A lower amount of N will reduce the accuracy of the proposed method and a larger amount of N with the current hardware may cause a delay in live applications.

3.2.2 ADO Removal

As mentioned before, depth data could be very noisy and contain many ADO pixels. In order to reduce these noises, we need a hole filling strategy. The main goal of this strategy is to filter the unknown depth value and refining object boundaries. Each filtered frame is then used as a sample to build the depth model which help to prevent with falling of the temporal variations of the depth distances. This will help to have more accurate depth pixel values in the model to identify the foreground and background.

Recently many researchers have been investigating inpainting depth frames to remove holes from the depth frames [111][112]. However, these methods are very expensive in terms of computation. One of the most common ways to remove the ADO pixels is to fill them by the neighbouring depth data [112]. We have used this idea and made an assumption that neighbouring depth pixels are most likely to have a similar value. We have used this assumption to remove the ADO pixels by replacing with the randomly nearest pixel values.

This fast and straightforward method will significantly help to reduce the number of errors. However, this method could also lead to more error in rare scenarios. For example, an open area such as a corridor with a length of more than the maximum sensor range. However, these wrong values will be gradually corrected with more accurate values during the update process. Figure 3.4 illustrates an example of a depth sample in a model before (a) and after ADO removal (b). The black pixels on top of the desk and under the chair show the ADO pixels in (a). These pixels then removed and replaced by neighbouring pixels in (b).

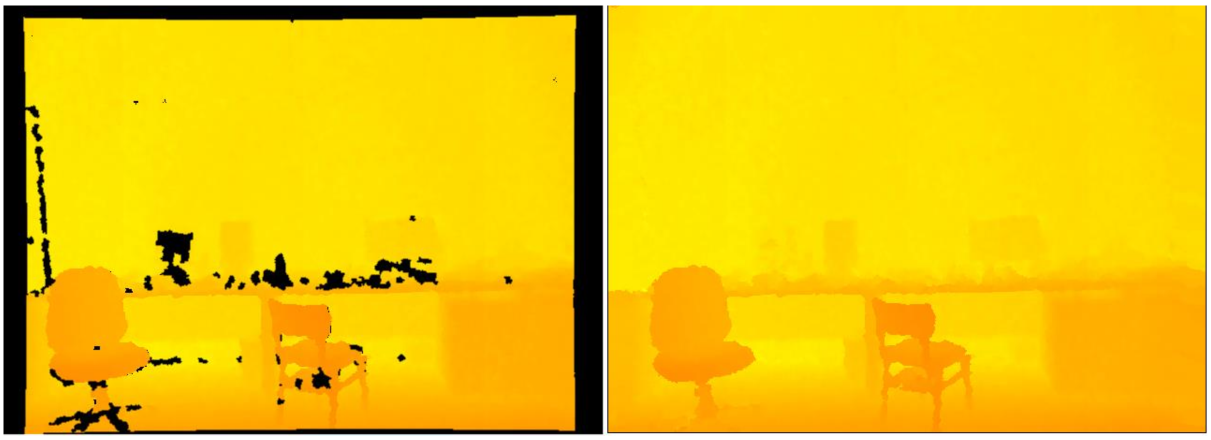


Figure 3.4. An example of depth image.
Black pixels show holes in depth frame, (a) depth frame before ADO removal, (b) post initialisation after ADO removal.

3.2.3 Bg/Fg Segmentation

Traditionally background subtraction techniques mainly rely on Probability Density Function (PDF) or statistical parameters such as variance or the mean. An alternative way is to consider statistical significance to build a model with previously observed real colour and depth data. This assumption is based on common sense that if the same pixel value has been observed many times in the same location, this pixel has a high probability of being background, compared to the pixel values that never come across.

As part of our background subtraction, we want to classify each pixel as foreground or background. In order to do this, we are fusing the results from colour and depth models to produce the final decision. Similar to the authors of [8], we create each background pixel with a set of samples instead of one background model. Accordingly, we have not used an estimation of the PDF for the background classification. Instead in each location, the current value of the colour pixel is compared to the collection of samples (colour model) to find out if the pixel value is close to some of the sample values instead of most of all samples in the same location. In a similar way, depth pixels will be compared to the depth model to check if the pixel has been in the same range or is closer to the camera.

In most cases, depth and RGB have the same individual segmentation outcome. In other words, both separately agree whether the pixel is part of the background or not. However, in some challenging scenarios, they are strongly against each other. An example of these situations could be colour camouflage such as foreground having the same colour of the background or depth camouflage such as moving the hand on the wall.

To make the final decision, we need to rely on the colour or depth model, one more than the other. Recently, with the production of new sensors such as time of flight which has been used in Kinect V2 sensor depth accuracy has been improved significantly [113][114]. On the other hand, illumination does not affect depth data. Therefore, we have relied more on the depth outcome to produce the result. Alternatively, if depth pixel is not available in any location, we will only rely on the decision of the colour model on that pixel location. Figure 3.6 illustrates the proposed classification in the flow chart diagram.

All non-ADO pixels from the new frame will be compared with the depth model first and if we could find enough close samples in the depth model, that pixel will be classified as background. This step called "Few similarities with depth model". The main reason we have added this condition is to detect shadows and illumination changes as part of the background. An example of this illustrated in figure 3.5. The shadow on the wall remarkably has different chromatic colour compared to the background. However, this wouldn't affect the depth data and the system could correctly identify the background without the effect of illumination (based on only depth data).

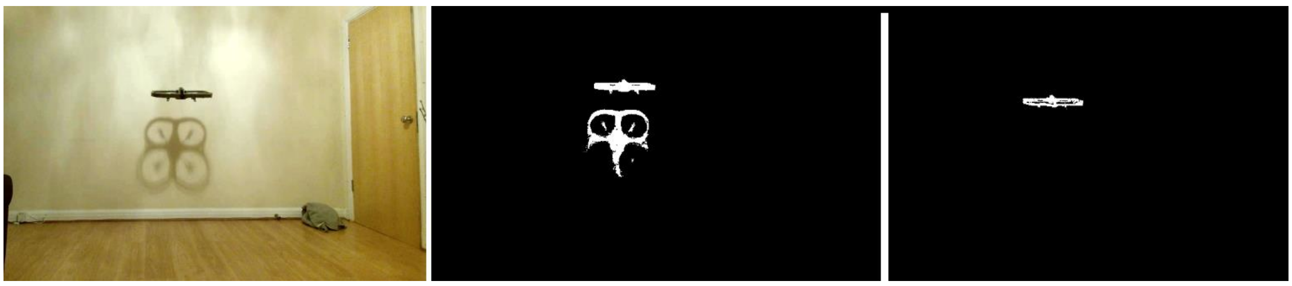

(a) (b) (c)

Figure 3.5. *An example of shadow.*
(a) Colour Image. (b) Original ViBe [8]. (c)Proposed method

If the system fails to find some similarity in the depth model, it would compare the new pixel with the depth model again. However, the system considers some tolerance this time. If it fails again to find some similarity with a depth model, that pixel will be classified as foreground regardless of the colour outcome. This step called "Few similarities with depth model +tolerance". It will allow the system to detect mainly the colour camouflage as the colour model is not able to detect these changes. All other pixels will be decided by the colour model in the same way. This means, the colour pixel will be compared with the model and if it could find some similarity in the colour model, it will be classified as a background otherwise, it will be classified as foreground.

Formally, let us denote a 3d point as an $X=(x,y,z) \in \mathbb{R}^3$, RGB-D camera produces a colour and depth images. We denote $v(X)$ the value in a given colour and $d(X)$ the value in the depth taken by the pixel located at X in the new image frame and with an index of i in a background sample value of v_i and d_i. Each background pixel located at X is modelled by a collection of N background colour and depth sample values taken before as:

$$\mathrm{M(X)_{RGB}} = \{\mathrm{v_1, v_2, v_3}, \ldots, \mathrm{v_n}\} \tag{3.1}$$

$$\mathrm{M(X)}_D = \{\mathrm{d_1}, d_2, d_3, \ldots, d_n\} \tag{3.2}$$

In this thesis, we refer to $M(X)_{\mathrm{RGB}}$ as a background colour model and $M(X)_D$ as a background depth model. In order to classify each pixel of the new frame as a background, we compare each depth pixel with the depth model $M(X)_D$ at location X. If the difference is greater or equal than Th_D (acceptable depth threshold which is close to 0), we count the pixel is similar to that sample. Every pixel from the new frame will be compared with all samples from the model in the same location. If the new pixel could find at least $\#_{\mathrm{Min}}$ (we recommend the value as $N/4$) similar pixels will be assigned as part of the background. The $\#_{\mathrm{Min}}$ should depend on the number of samples (N) and therefore, a higher amount of N requires larger $\#_{\mathrm{Min}}$. If we could not find at least $\#_{\mathrm{Min}}$ similar sample out of N number of samples at location X, the system will increase the Th_D and do the last process again. This time if the system could not find $\#_{\mathrm{Min}}$ similar sample, it will be count as a foreground. This formally defines as:

$$\#_{Min-Depth}\{\, d(X) + Th_D \geq d_i\} \tag{3.3}$$

Where $d(X)$ is the depth pixel value at location X. The pixel value will be identified as foreground if the number of cardinality denoted as $\#_{\mathrm{Min-depth}}$ reach to $\#_{\mathrm{Min}}$. Any depth pixel in the position of X in the model belongs to this set ($\#_{\mathrm{Min-depth}}$), if the difference of the new pixel $d(X)$ and the ith sample in the depth model (d_i) with considering some tolerance (Th_D) is greater than or equal to zero.

All other pixels will be decided by comparing the colour value and the samples in the colour model $M(X)_{\mathrm{RGB}}$. Where the colour pixel value is $v(x)$ and the threshold value is Th_{RGB}. The pixel value will be identified as background if the number of cardinality denoted as $\#_{\mathrm{Min-Colour}}$ will reach to $\#_{\mathrm{Min}}$. Any pixel in the X position in the model will be included in the $\#_{\mathrm{Min-Colour}}$ if the absolute value of deference in the new pixel value and the samples in the colour model are smaller than or equal to the given RGB threshold amount (Th_{RGB}).

$$\#_{Min-Colour}\{|v(X) - v_i| \leq Th_{RGB}\} \tag{3.4}$$

Appendix A show full C-like pseudo-code for the main part of the proposed method. Default values for all the parameters of the algorithm are also given.

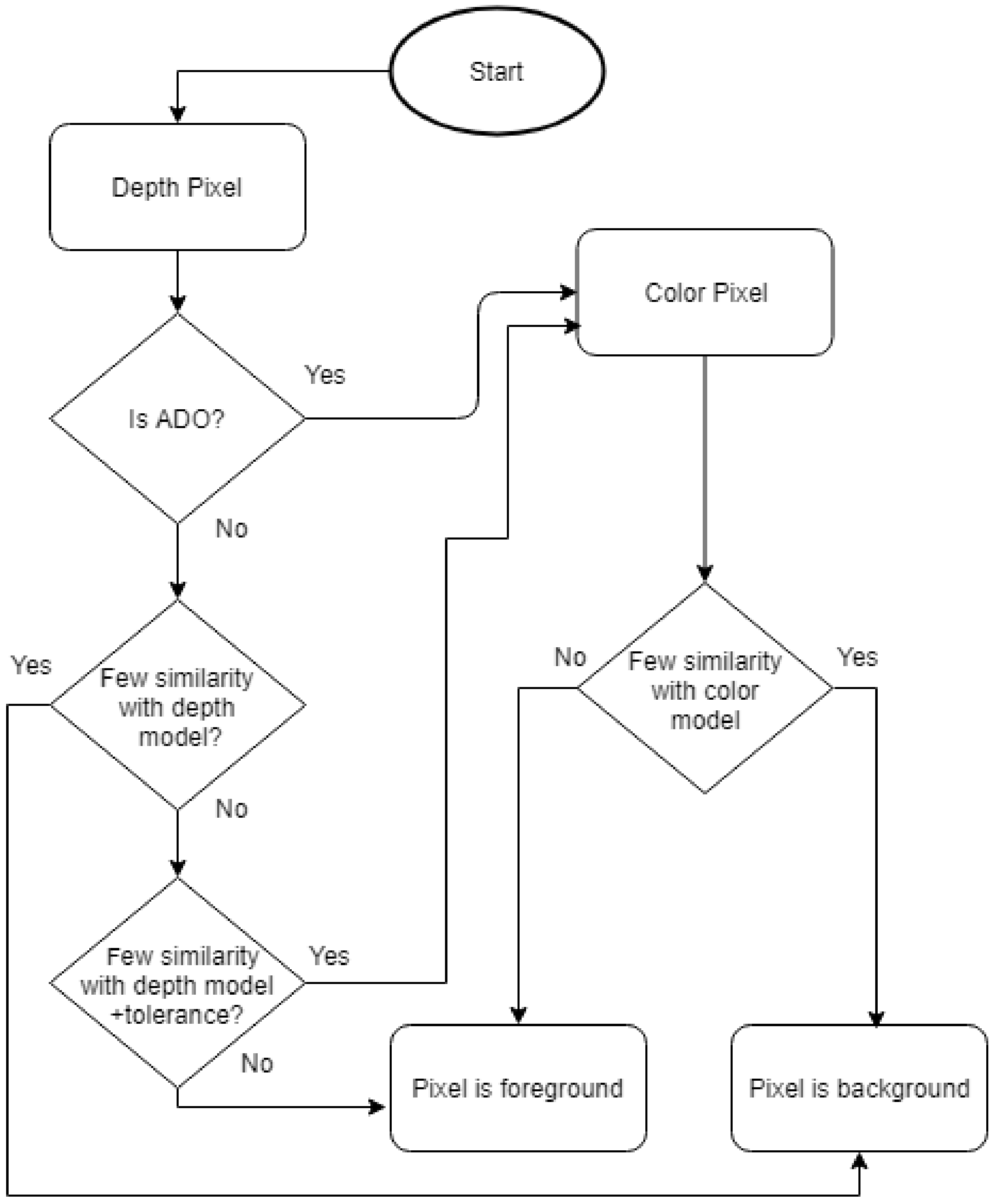

Start
Depth Pixel
Is ADO?
Yes
No
Color Pixel
Few similarity with depth model?
Yes
No
Few similarity with color model
No
Yes
Few similarity with depth model +tolerance?
Yes
No
Pixel is foreground
Pixel is background

Figure 3.6. Flow chart of the proposed classification method.

3.2.4 Background Update

In this section, we will explain how to continuously update the background model with the new frames over the time. The reason we have added this stage is that the system adapts to the changes in the background over the time. These changes could be illumination changes or moving an object in the background completely to a different position.

When a pixel is classified as a background, the system will randomly swap the value of a new colour pixel with one of the samples in the colour model in the same location. However, in the depth model, we will check the distance of the new pixel (in the depth image) with the background model distances. In order to compare the distance, the system finds the smallest sample in the depth model and compares it with the value of the new depth frame in the same location. If it has the same distance or longer, we see it as a good sample and will swap it with the smallest previous samples, otherwise, it is a bad sample and it will not change the samples in the model.

Figure 3.7 demonstrates an example of a good and bad sample pixel. We have defined the good and bad samples based on the fact that, if we assume point A and B are different samples of the background depth model in the same location, point A will be a bad sample as previously point B has been observed behind this point. This means, in a 3D space these two points have the same location of x and y with different d (dimension) and the point with the absolute smaller number is closer to the camera. Therefore, it cannot be part of the background. An exception of this rule could be the physical change in the background of the scene. For example, this could be moving forward a table in the background. This means some pixel values which belong to the model (it is part of the previous frames) doesn't exist anymore. However, in this case, the system will be able to cope with the changes from time to time in the blind update stage.

Formally, when pixel $v(X)$ is identified as a background, the system will swap the $v(X)$ with randomly one of the samples in the colour model $M(X)_{\text{RGB}}$. On the other side, the system finds the smallest distance value (d_{smallest}) in $M(X)_D$ and compare it with $d(X)$. If $d(X)$ is bigger than d_{smallest} then we can accept that as a good background pixel and replace it with d_{smallest} otherwise, it is not a good sample and we will not change our model. This is one of the most significant differences with the currently available methods which those usually modify the background samples according to old replace with the new one, mean or random number. This enables us to improve our model and update with the latest changes during the time as well as keeping the valid samples in the models. The only disadvantage of this selection is that if we move the background forward such as moving a table in the middle of the room, the system will always identify this as a foreground and will not change the model. For this reason, we have added a blind update step into our algorithm which will allow the system to adapt to the changes in the background during the time.

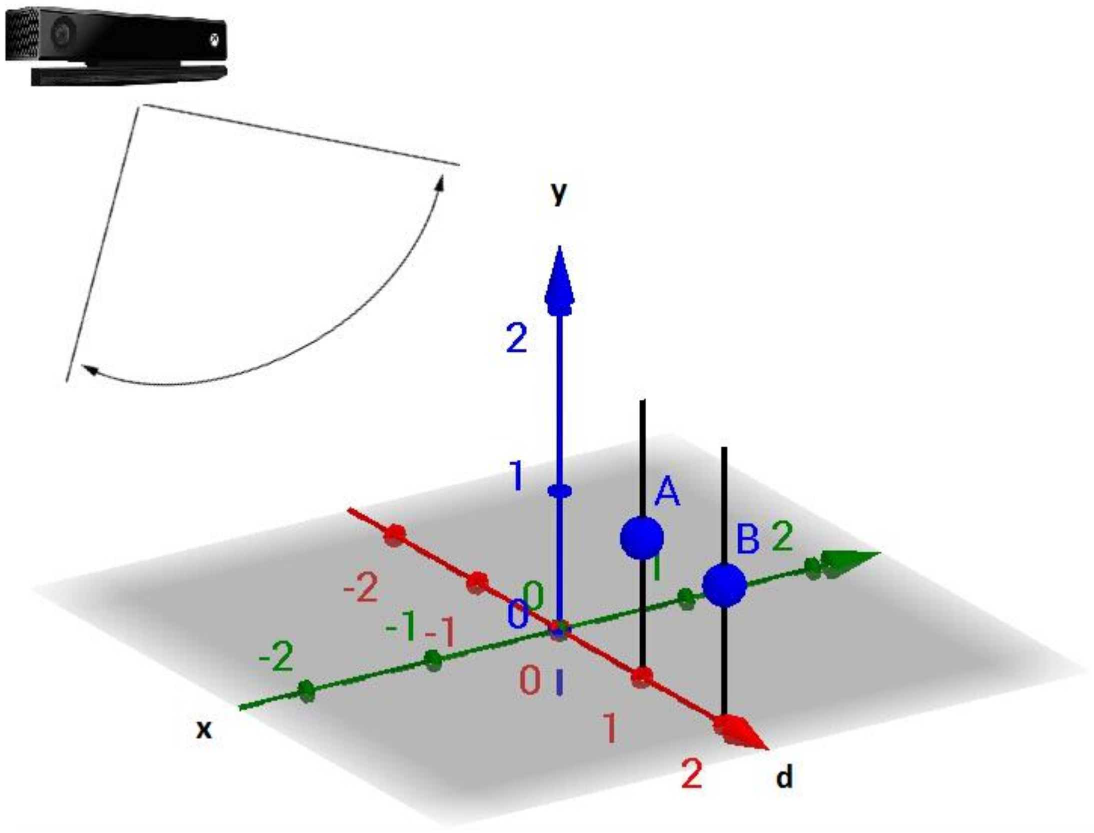


***Figure 3.7**. an example of a good and bad sample pixel.*

Point A and B are two points in the background depth model which they have identical X and Y with the different d. Point A is closer to the camera, therefore, it cannot be the background because another point (B) observed behind this point in this location.

3.2.5 Blind Update

In the classification step of our method, we update our background model by comparing the pixel of the new frame with the background model and replacing this with the new pixels if they are more valid. However, this will only allow us to replace those pixels which have already been identified as a background. Consequently, if we introduce a new object in the scene (as part of the background) because it has a smaller distance from the camera compared to all previous pixels in the depth model and different colour to the colour model, it will never be recognised as a background. Therefore, it will never be part of the background samples.

The term which referred in the literature as background history or background memory has always raised a question in subtraction techniques that which sample we can keep in the model and for how long we can use that. For instance, one of the classical approaches for updating the background model is that discard the old pixel model and replace with the new pixel after a period of time or number of a given frame (Usually after a couple of frames or seconds). These classical methods will update all the old pixels in the model where it is not always necessary to update the valid samples.

On the other hand, updating the model only by those pixels which identified as a background or including foreground pixels is always raised in background subtraction algorithms. In the literature, it has been described as a blind and conservative update procedure. A conservative approach only updates the model by pixels which are identified as a background and it never uses the pixel which belongs to the foreground. Conservative approach could help the system to detect the moving objects sharply over the time in a steady situation. However, this approach which has been used in our background update stage (as illustrated in diagram 3.3) only update the background pixels. This could cause a permanent misclassification, contribute to creating a ghost and failure in dynamic background scenarios. Most of the practical scenarios could reach to these situations.

Despite all the effort made by the previous researches, developing a fast approach to eliminate the ghost in the dynamic background situations is still challenging for background detection techniques. For these reasons, as illustrated in the diagram in figure 3.3, we have added another background update phase for the colour and depth models which called “blind update”.

A blind update will allow us to use any pixel for an update, whether it is classified as a background or foreground. The main downside of this method is poor detection of the slow-moving objects which are becoming part of the background model during the time. Several solutions have been introduced to solve this issue such as using a background model of large size or first-in-first-out. However, these solutions have negative impacts, such as higher computational and memory usage or time limiting.

Those pixels classified as part of the background in the scene, automatically will be used to update the background model. The method will swap the pixel from the new frame with the shortest in the depth model if these pixels have better values (longer distance compare to the model). However, if our background will be dynamic, the system will permanently identify the background as part of the foreground. For each pixel, the system will swap the value of current depth (only if its non-ADO) and colour frame with the model randomly after every α number of frames (we recommend this value as 30). As most of the current RGB-D camera capture 30 frames per second, the blind update will be applied every second. Updating the models blindly every second will allow the system to quickly adapt to the changes as well as tolerating slow-moving objects. This method has the advantage of a memoryless update strategy, producing a fast and efficient update. Moreover, random sampling increases the time gaps and allows the adaptation of the background models that are classified as a foreground.

3.3 Results

In this section, the results achieved by the proposed method are compared with alternative background/foreground subtraction algorithms based on colour and depth data. We have evaluated the presented system in two different ways. First, we have evaluated the proposed moving object detection method with two datasets and then the entire system is tested via a live demonstration in an indoor environment. In the rest of this thesis, we refer to the proposed method in this chapter as Modelling Background and Nonparametric Segmentation method (MBNS).

We have used two different indoor benchmark datasets. The first dataset contains sequences from two different types of MAVs. In the first sequence, we have evaluated the detection accuracy using an AR.Drone [11] and in the second sequences, we have used a Crazyflies [12], a smaller size quadcopter. Generally, small and fast-moving objects are challenging for moving object detection algorithms. Small moving objects are often filtered by algorithms as they are detected as noise. Fast-moving objects could register as a blurred area in the image captured by a low-cost camera. Also, synchronisation in RGB-D camera could fail in a fast-moving object. This means the position of moving object in the colour frame could be different in the depth frame.

To collect these two sequences, a Microsoft RGB-D Kinect V2 sensor has been used. The goal of this test is to measure the ability of the proposed method to detect a small and fast-moving object such as the microdrones tested under different indoor challenging scenarios as detailed in table 3.1. We have also generated hand-labelled ground truth for these sequences to measure the accuracy of each method used in the comparisons. In particular, we have compared the proposed method in this section with $\mathrm{CL_W}$ [48], $\mathrm{MOG_{RGB-D}}$ [45], $\mathrm{GSM_{UF}}$ and $\mathrm{GSM_{UB}}$ [7], PBAS [14] and $\mathrm{Vib_{bin}}$[51].

It is worth mentioning that the original PBAS is using only colour frames and it is not optimised for RGB-D data. In this thesis, PBAS has been extended to use colour and depth (RGB-D) images in order to enable us to have the same input for all methods. This has been done similar to the [51] by fusing the result of colour and depth binary mask using a logical "OR" (non-exclusive) and the same parameters used as original PBAS. We refer to this method as PBAS_{bin}. Further research can be conducted in the future to optimise the parameters for PBAS_{bin}.

We should state that all results for the proposed algorithm have been evaluated without using any post-filtering to compare the accuracy of the method. Clearly, the amount of noise may be reduced and the results will improve with post-filtering techniques such as morphological filtering. Therefore, using post-filtering is frequently adopted in some background subtraction methods [91].

Table 3.1. *Details of our dataset used for measuring the accuracy of the UAVs detection at 30fps.*

Sequence Name	**Number of Frames**	**Frequency of frames used for ground truth**	**Number of Ground truth**	**Number of Frames where moving object is present**	**Objective**
AR.Drone	350	Every 30 frames	12	220	Accuracy of UAV detection
Crazyflie	275	Every 30 frames	10	230	Accuracy of small UAV detection

For qualitative evaluation, a video is available on [115] to show the accuracy of the proposed method in some challenging scenarios such as a change in the background, removed an object from the background (intermittent motion), a change in the light and sunlight (illumination changes), micro UAV and appearance of shadow on the wall and floor. The following figures illustrate examples of this evaluation. The system automatically draws a rectangle around the foreground (drone) in these sequences. Figure 3.8 shows the proposed method could successfully detect the drone in a cluttered background.

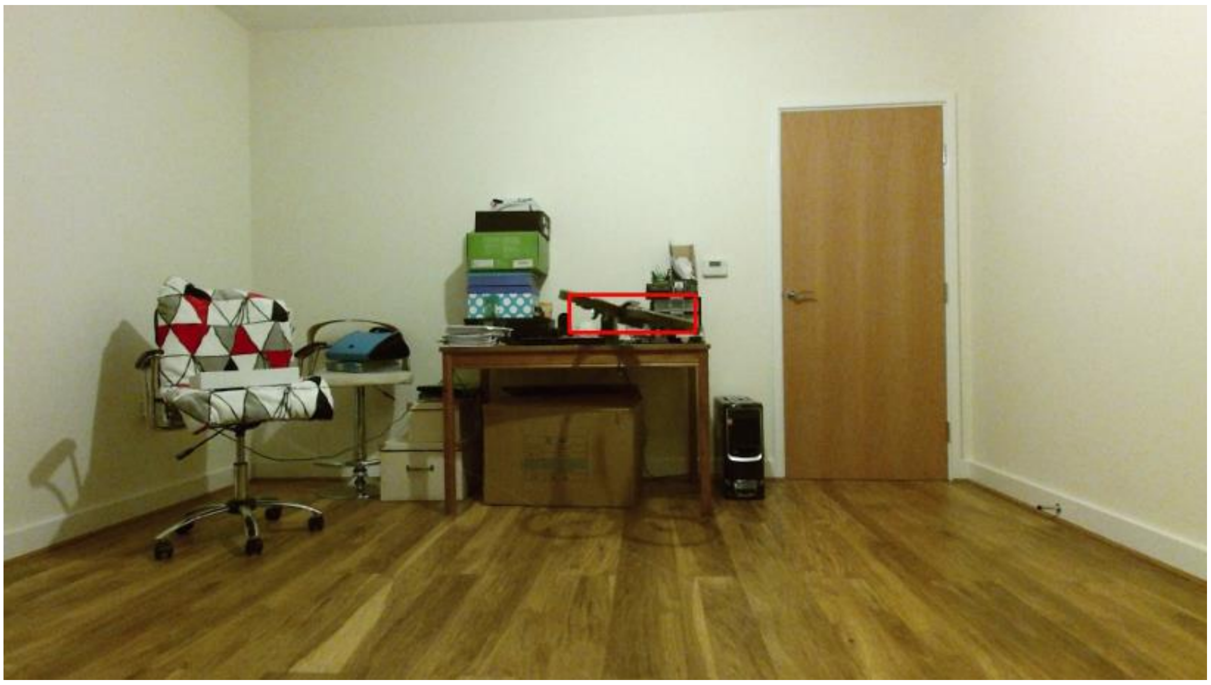


***Figure 3.8**. Detection accuracy in a cluttered background.*
Red rectangle illustrates the detected foreground region.

Figure 3.9 demonstrates the accuracy of the proposed method in a change of illumination (sharp sunlight) scenario. In this sequence, at first there is no sunlight and then suddenly sharp sunlight will shine on some parts of the room. Despite this interruption, the proposed method could fully detect the drone with high accuracy in this scenario.

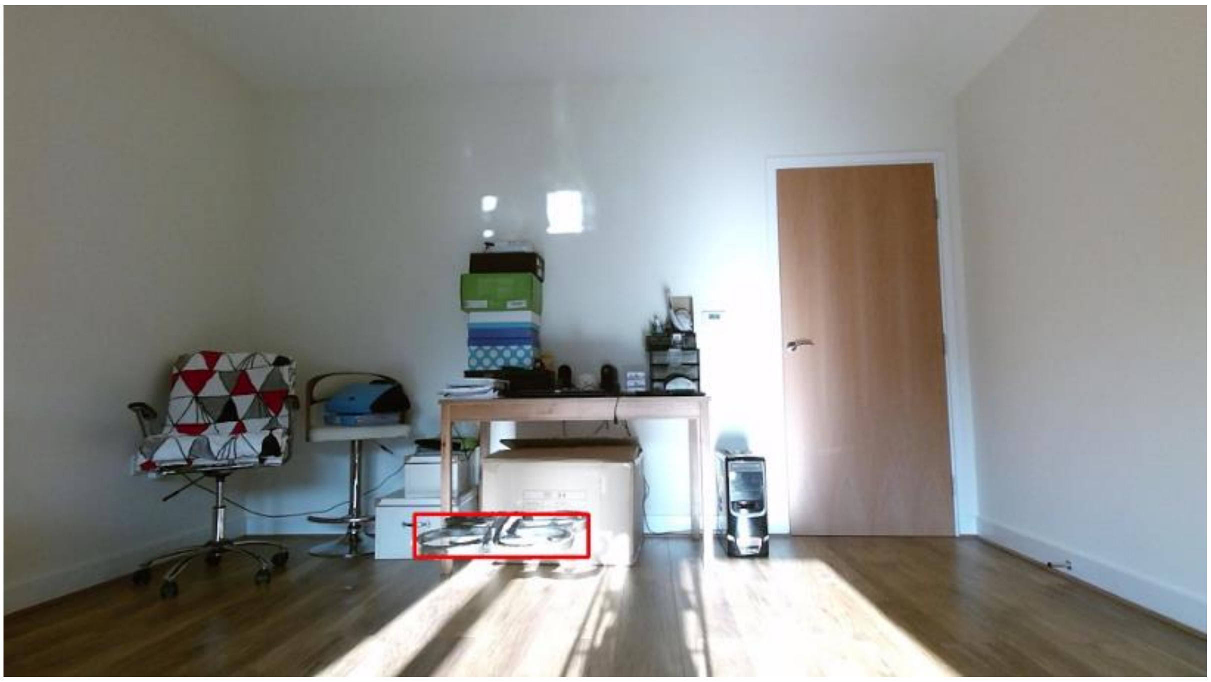


***Figure 3.9**. Detection accuracy in sharp illumination changes.*
Red rectangle illustrates the detected foreground region.

Figure 3.10 shows a large shadow on the wall and the papers on the floor. However, the system could successfully recognise the foreground (drone) from the shadow.

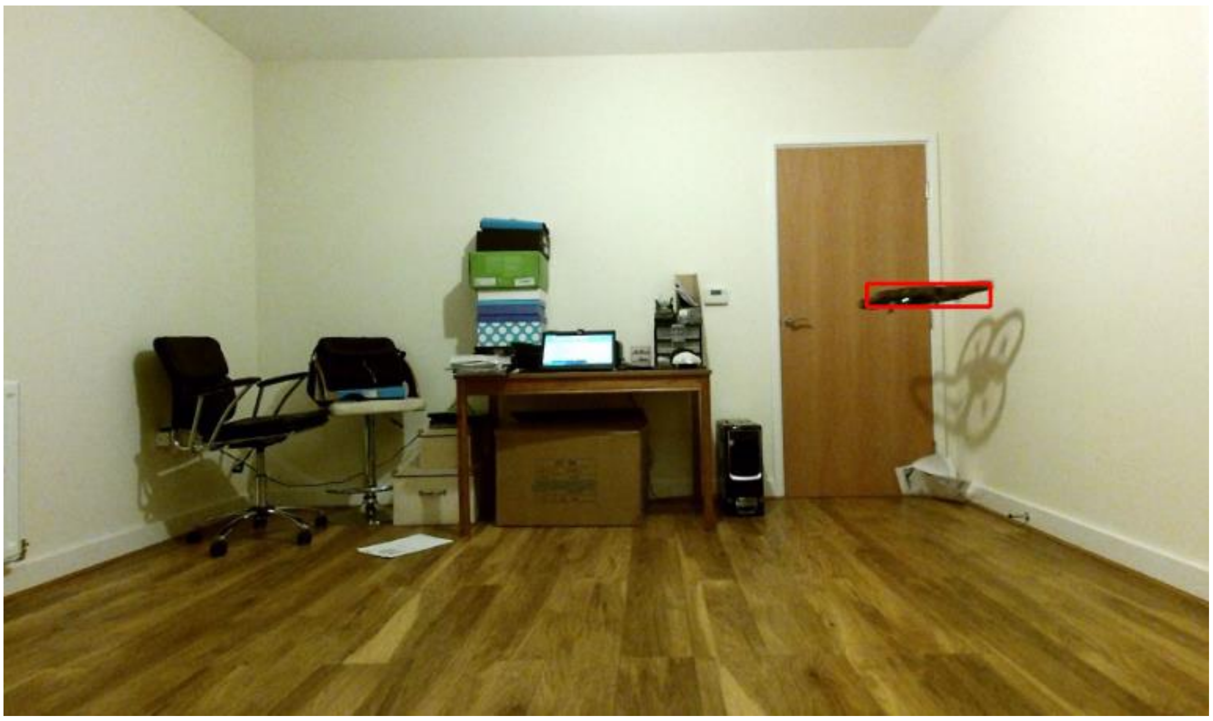
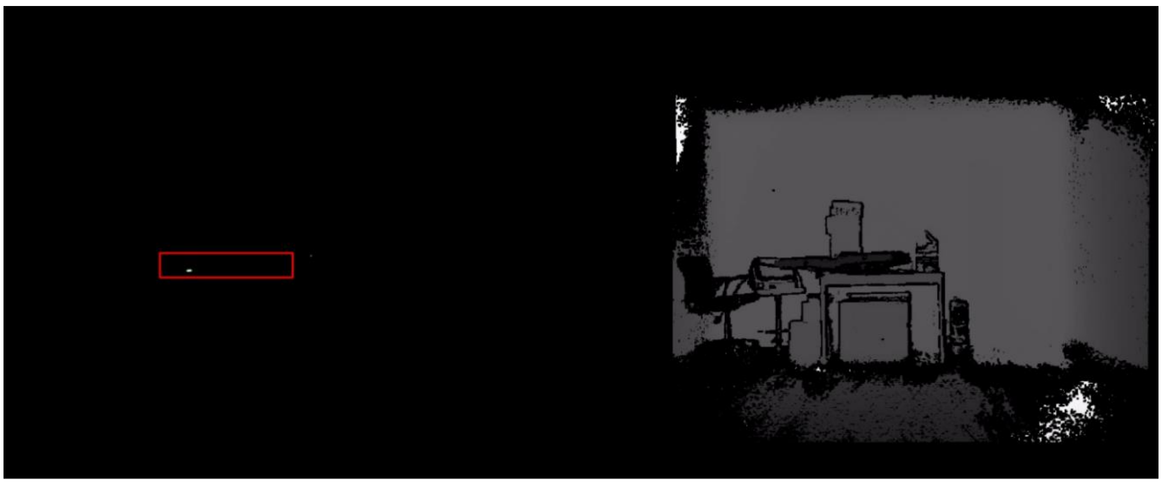


***Figure 3.10**. Detection accuracy in the appearance of a shadow.*
Red rectangle illustrates the detected foreground region.

Figure 3.11 demonstrates the accuracy of the proposed method when the light is switched off. In this scenario, the light is on at the beginning and then suddenly it will be switched off. The foreground is fully detected even though the colour frames are completely black.



(a) (b)

***Figure 3.11**. Detection accuracy when the light is switched off.*
Red rectangle illustrates the detected foreground region. (a) the detected region in the red rectangle, (b) depth data.

Additionally, we have tested the proposed method with the benchmark RGB-D dataset introduced in [48] to compare and rank the algorithms to ensure our proposed algorithm performs well among other currently available methods in different challenging scenarios. We have used the ground-truth provided with these datasets in order to measure the performances. This dataset has four different sequences and each sequence has been made to test the accuracy of the method in a specific challenging scenario. DCamSeq and ColCamSeq ground-truth has been produced to test the accuracy of individual method only in those sections in the images where every single problem is existing. This process guarantees that other challenging scenarios do not interrupt the algorithms segmentation. However, using these ground-truths for evaluation can produce inaccurate results. This is because some parts of the foreground (human body) is shown as a background (black pixels) and this can influence the results. To prevent this, the ground-truth for DCamSeq2 and colorCam2 from SBM-RGBD dataset [116] has been used for DCamSeq and ColCamSeq evaluation. The ground-truths are the same as the original and the only difference is that the area outside of the problem is coloured grey. This means the grey area is not part of the evaluation. This will help to produce more accurate results. For $\mathrm{CL_W}$ [48], $\mathrm{MOG_{RGB-D}}$ [45] methods, the results are taken from [48] due to source codes and parameters are not publicly available.

Generally, in the ground-truth, any moving object whether it is stationary for a short time such as humans, a car stopped at a traffic light, drone or any other moving object mark as white colour. However, small movements that continuously occur in the background classify as part of the background. For example, a wave on a water surface, tree branches or shakes of objects by wind (e.g. chandelier, curtains, light poles, flags, etc.). These background pixels are shown in black colour in the ground-truth. Also, the grey area demonstrates outside of the region of interest which is not either part of background or foreground. Table 3.2 shows the details of the dataset introduced in [48].

Table 3.2. *Details of the dataset in* [48] *which used for evaluation in this section.*

Sequence Name	**Number of Frames**	**Frequency of frames used for ground truth**	**Number of Ground truth**	**Number of Frame where moving object is present**	**Objective**	**Papers which also used these datasets**
GenSeq	300	Every 8 frames	39	115	Overall Performance	[7][88]
DCamSeq	670	Every 7 frames	102	400	Depth Camouflage	
ColCamSeq	360	Every 8 frames	45	240	Colour Camouflage	
ShSeq	250	Every 10 frames	25	120	Shadows Impact	

We have used the following metrics to measure the performance of the proposed algorithm to be able to compare and rank the results. These metrics are as followed:

False Positive (FP): Part of the Bg pixels which are classified as Fg.

$$FP = \frac{\text{Bg pixels classified as Fg}}{Total\ \text{Bg}\ pixels} \times 100 \tag{3.5}$$

False Negative (FN): Part of Fg pixels which are classified as Bg.

$$FN = \frac{Fg\ pixels\ classified\ as\ Bg}{Total\ Fg\ pixels} \times 100 \tag{3.6}$$

Total Error (TE): The full number of misclassified Bg/Fg pixels inside the region of interest which normalised according to the image size. This has been calculated as:

$$TE = \frac{(\text{FN} + \text{FP})}{Total\ pixels} \times 100 \tag{3.7}$$

Where total pixels are all the pixels inside the region of interest. FP, FN and TE are all dependants to the total pixels. Therefore, a higher amount of FN and FP increase the value of TE. The range of these metrics is at a maximum of 100 and a minimum of 0.
Similarity measure (S): Is non-linear metric that combines FN and FP which publicly known as Jaccard's index [117] and has been used in [118] as:

$$S(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{3.8}$$

Where A denoted as detected region and B is ground-truth. Result closer to 1 shows Fg correctly identified similar to the ground-truth, otherwise will be closer to 0.

similarity measure (S_B): To investigate the misclassified pixels near to the boundaries of moving objects. It is measured similar to S, but only considering the regions of 10 pixels surrounding the ground-truth boundaries.

Additionally, we used the proposed evaluation method in [62] to calculate the average ranking of method (RM) which combine the performance of each method across different metrics in each sequence and use overall ranking across category (RC) which shows in general how well an algorithm performs with respect to the other techniques by calculating an average (RM).

Let us denote the ranking of the ith technique for the metric m in the sequence sq as $rank_i(m, sq)$. Then the average ranking of technique i for the N_m number of metrics in the sequence sq is given as:

$$RM_i = \frac{1}{N_m}\sum_{i=1}^{m} rank_i(m, sq) \tag{3.9}$$

Accordingly, the overall ranking among all the categories ($\mathrm{RC_i}$) for $\mathrm{N_i}$ number of techniques is calculating by taking the mean of average ranking across all the sequence as:

$$RC_i = \frac{1}{N_{sq}}\sum_{i=1}^{sq} RM_i \tag{3.10}$$

N_{sq} defined as the number of sequences which is 4 in the dataset demonstrated in table 3.2. In general RM, RC, TE, FP and FN the lower amount demonstrate better performance and higher S and S_B demonstrate more similarity with ground-truth and therefore better performance.

3.3.1 MAVs Sequences

Figure 3.12 shows an example of this sequence and the binary mask of each method. (a) is colour frame and (b) is a depth frame and (c) ground truth for this frame. In this scenario, the moving object (Crazyflie) is fast and small. This will cause the sensor to frequently capture unmatched colour and depth images. Also, the colour frame is blurred due to the speed of the drone and some part of the drone has some unknown pixel's value in the depth frames. These conditions create a very difficult and challenging scenario for the algorithms to correctly detect the pixels of the moving object. For these reasons, all tested methods have shown weak performance in this sequence. Also, due to the small size of the drone and therefore a small number of misclassified pixels compared to the total pixels of the frame, the TE shows small values but this doesn't necessarily indicate that the methods are accurate. Table 3.3 shows the proposed algorithm could obtain the lowest TE and FP. This shows the system had less false detection in the whole area among other algorithms. However, the FN for the proposed method is very high which means the system could not successfully detect part of the foreground. Other methods also have the same problem except $\mathrm{Pbas_{bin}}$, but, it has also weak results in FP and S. On the other hand, the proposed method could achieve the highest similarity measure (S) which shows the closest result to the ground truth. Average ranking of the proposed method (RM) is the lowest in this sequence which means it could achieve the best performance overall.

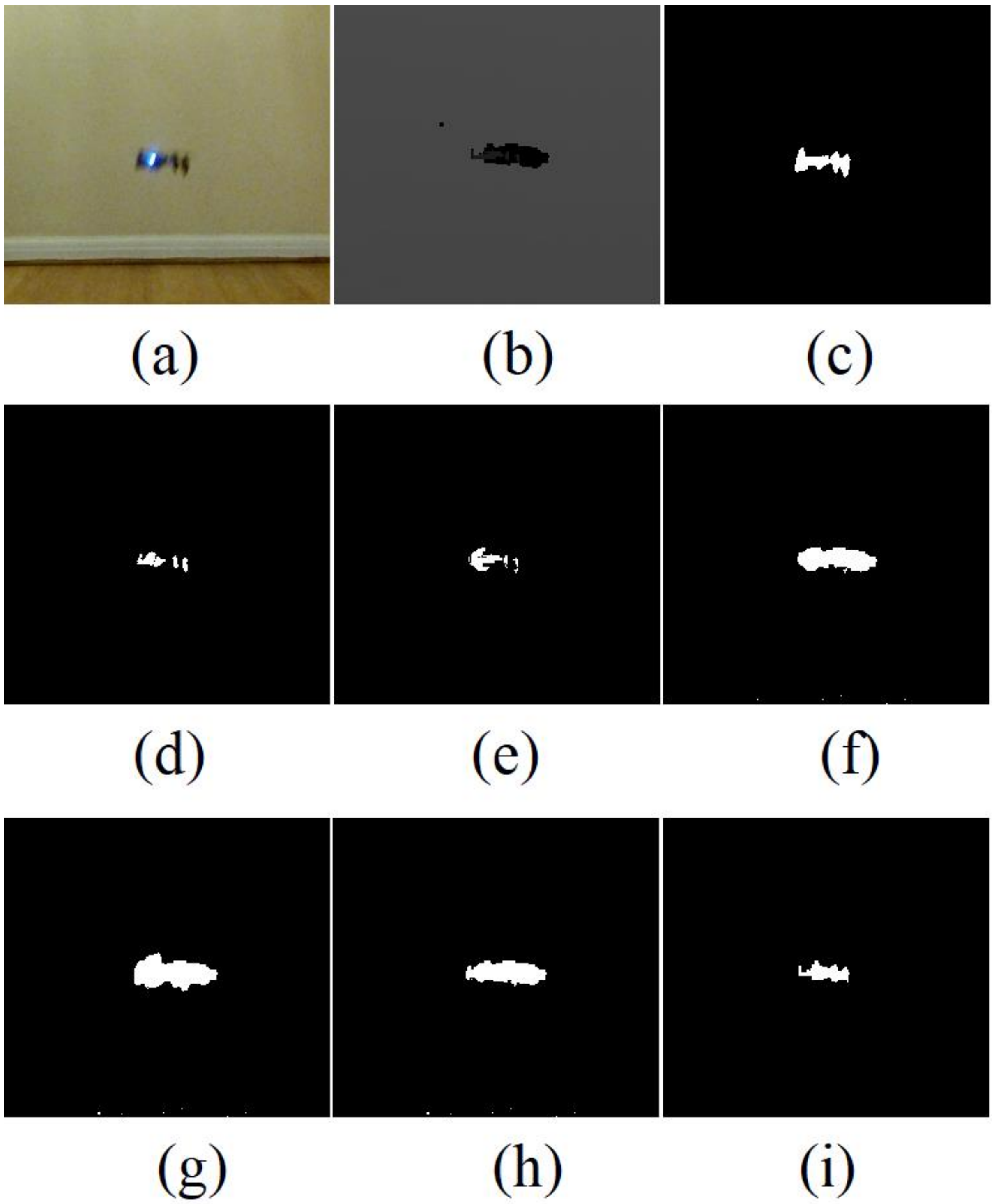


***Figure 3.12**. The result of micro UAV sequence.*
(a) Colour frame, (b) Depth frame, (c) Ground truth, (d) MOG_{RGB-D}output, (e) GSM_{UB}output, (f) GSM_{UF}output, (g) $PBAS_{bin}$ output, (h) $Vibe_{bin}$ output, (i) Proposed method output.

Table 3.3. *Craziflies sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG_{RGB-D}}$	0.63	0.17	51.25	13.56	0.02	0.02	0.34	0.17	0.37	0.14	3.2
$\mathrm{GSM_{UB}}$	0.08	0.01	55.96	23.49	0.03	0.03	0.27	0.13	0.29	0.09	3.8
$\mathrm{GSM_{UF}}$	0.12	0.27	37.59	18.25	0.09	0.03	0.27	0.11	0.34	0.08	3.4
$\mathrm{PBAS_{Bin}}$	0.63	0.06	0.20	0.57	0.63	0.06	0.11	0.04	0.44	0.07	3.2
$\mathrm{VIBE_{Bin}}$	1.45	0.19	19.12	10.71	1.43	0.19	0.04	0.02	0.42	0.06	3.8
Proposed Method	0.05	0.01	42.63	17.85	0.01	0.02	0.42	0.18	0.42	0.11	1.8

Table 3.4 demonstrates the result for the AR.Drone sequences. This table shows that the accuracy of detection in AR.Drone is higher than smaller drones such as the Crazyflie. The main reason for this is AR.Drone has a bigger surface and therefore, it receives more accurate depth data from the sensor. Table 3.4 shows that the proposed method could achieve the lowest TE and FP which shows this method has the lowest error compare to the other algorithms. Moreover, the similarity measurer in the images (S) and around object boundaries ($\mathrm{S_B}$) are the most similar to the ground truth. Similarly, the average ranking (RM) shows the best performance is for the proposed algorithm in overall of these metrics by achieving the lowest amount among all other methods.

Table 3.4. *AR.Drone sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG_{RGB-D}}$	0.15	0.16	14.79	12.02	0.04	0.06	0.79	0.13	0.80	0.11	2.6
$\mathrm{GSM_{UB}}$	0.49	0.29	74.97	11.66	0.01	0.01	0.25	0.12	0.26	0.12	4.2
$\mathrm{GSM_{UF}}$	0.31	0.26	9.59	6.52	0.25	0.21	0.69	0.11	0.74	0.08	3.2
$\mathrm{PBAS_{Bin}}$	1.25	0.33	0.25	0.80	1.25	0.33	0.33	0.09	0.66	0.10	4.2
$\mathrm{VIBE_{Bin}}$	1.02	0.26	2.73	3.41	1.18	0.25	0.33	0.09	0.74	0.09	3.8
Proposed Method	0.13	0.11	11.84	6.02	0.05	0.07	0.82	0.11	0.83	0.08	2.0

Figure 3.13 illustrates an example of this sequence and the output for all the compared methods. $\mathrm{GSM_{UB}}$(e) could not detect many foreground pixels and had a poor accuracy in this sequence. Instead, $\mathrm{PBAS_{bin}}$ has detected many background pixels as a foreground. The proposed method could achieve the best result in this sequence.

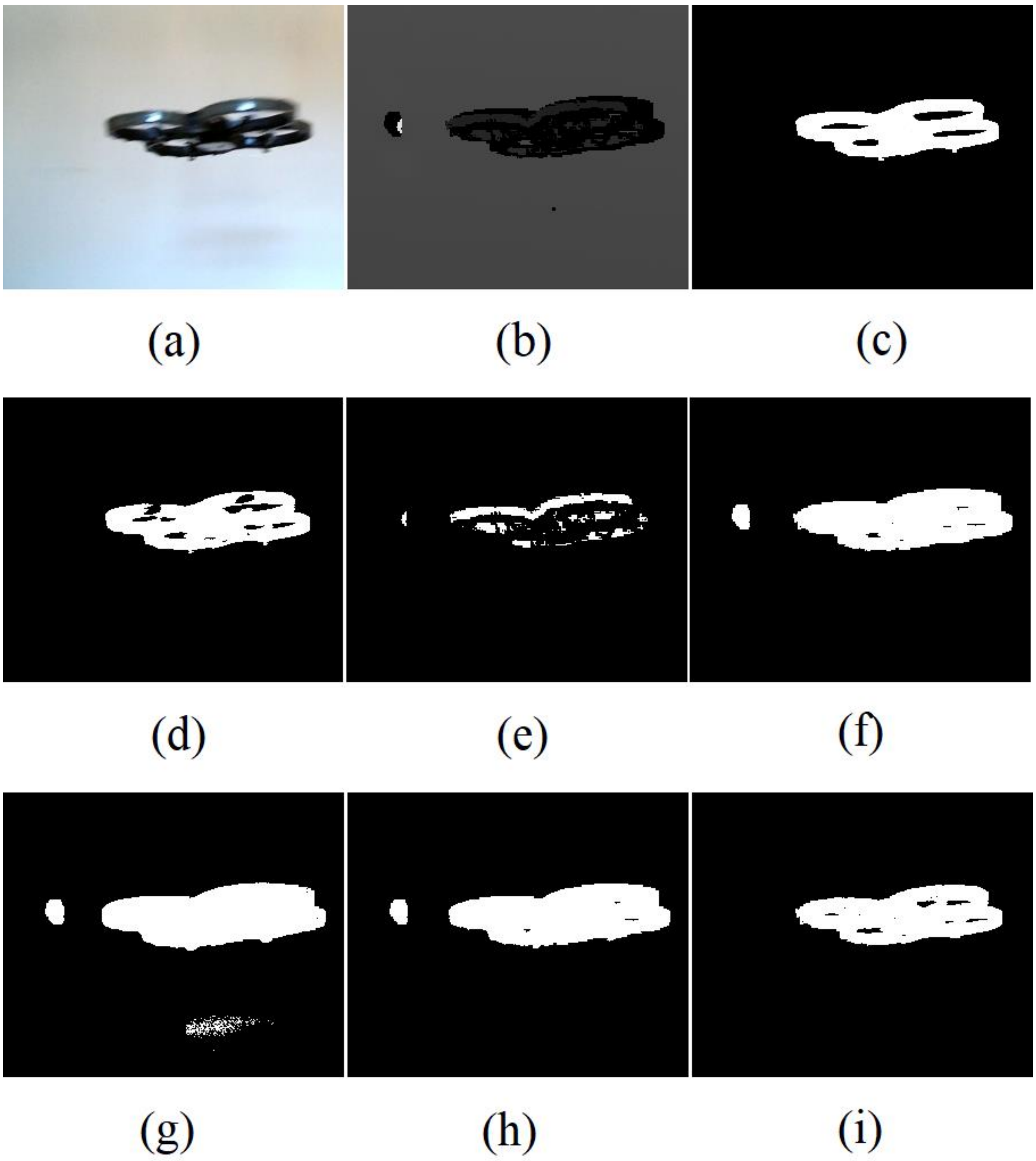


Figure 3.13. *The result of AR.Drone sequence.*
(a) Colour frame, (b) Depth frame, (c) Ground truth, (d) MOG_{RGB-D}output, (e) GSM_{UB}output, (f) GSM_{UF}output, (g) $PBAS_{bin}$ output, (h) $Vibe_{bin}$ output, (i) Proposed method output.

3.3.2 Benchmark Dataset

In this section, the benchmark RGB-D dataset introduced in [48] is briefly discussed and then results for each method shown.

GenSeq sequences

This sequence has been designed in [48] to test the overall performance of the method in case of several possible errors that may occur in one scene. This sequence contains a scene with an individual person moving. Figure 3.14 illustrates an example of this sequence which clearly shows most of the moving object has been successfully detected by the proposed method. However, some false positive and false negative detection are visible in the subtraction result. Additionally, Table 3.5 shows the full results for all frames of this sequence. The proposed method has the lowest amount of total error (TE) and the highest similarity with the ground truth (S and S_B). Consequently, it has the lowest average ranking of the method (RM) which shows it has the best performance in this sequence among all other methods.

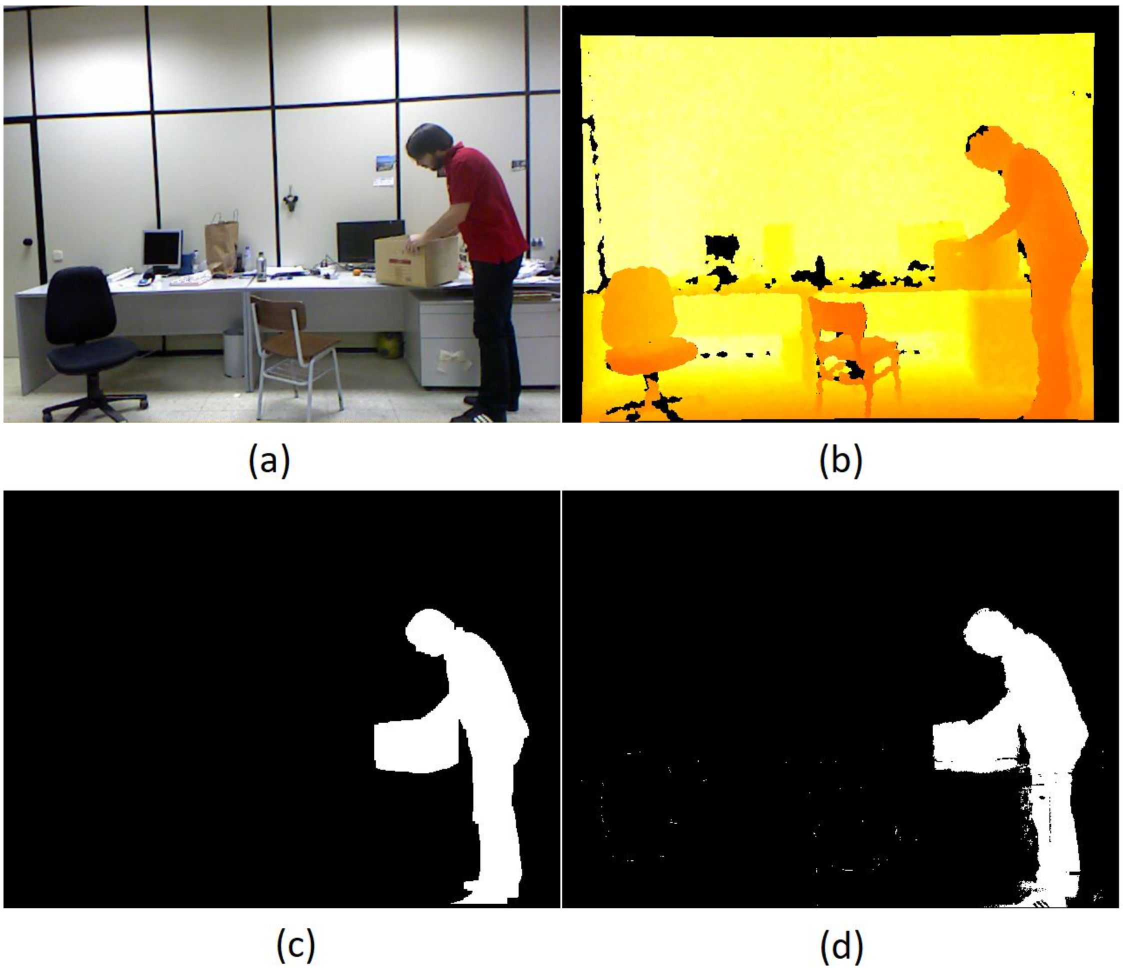


Figure 3.14. *Result of the proposed method in frame 1026 of GenSeq sequence. (a) Colour frame, (b) Depth frame, (c) Ground truth, (d) Proposed method.*

Table 3.5. *GenSeq sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG_{RGB-D}}$	1.93	0.66	0.63	0.01	2.09	0.02	0.79	0.20	0.45	0.13	4.0
$\mathrm{CL_W}$	1.30	0.42	1.49	0.02	1.27	0.01	0.83	0.21	0.53	0.14	3.0
$\mathrm{GSM_{UB}}$	1.38	0.56	1.04	0.78	1.44	0.66	0.83	0.20	0.78	0.11	2.8
$\mathrm{GSM_{UF}}$	1.30	0.52	4.08	15.38	1.30	0.60	0.83	0.20	0.78	0.14	3.2
$\mathrm{PBAS_{Bin}}$	8.24	13.78	0.33	0.53	9.36	15.97	0.66	0.21	0.71	0.10	4.6
$\mathrm{VIBE_{Bin}}$	2.32	0.58	1.59	1.52	2.43	0.56	0.77	0.16	0.75	0.09	4.6
Proposed Method	1.09	0.46	2.85	7.43	1.02	0.56	0.88	0.14	0.79	0.12	2.0

DCamSeq sequences

The goal is to investigate the tolerance of the algorithms in case of depth camouflage existence. Figure 3.15 illustrates an example of this sequence. In order to have an accurate measurement on the depth camouflage, the ground-truth and ROI of this dataset are only around the cupboard and hand of the person in the scene. Therefore, other parts of the moving object (other part of that person) are out of the measurement area.

Table 3.6 illustrates the result of this sequence, the total error (TE) and false negative (FN) of the proposed method is very high which shows weak detection in this sequence. Accordingly, after PBAS, the proposed method has the highest RM compare to other methods which demonstrate a weakness of our method. The main reason is that the depth model is not able to detect the entire hand when it is on top of the cupboard as the depth values of the hand and the cupboard is very similar. This problem can be solved by relying more on the colour model to produce the segmentation result. However, in that case, the system would be significantly less accurate in the shadows and colour camouflage scenarios. In this sequence $\mathrm{GSM_{UB}}$ and $\mathrm{GSM_{UF}}$ has achieved the lowest RM and shown the best result.

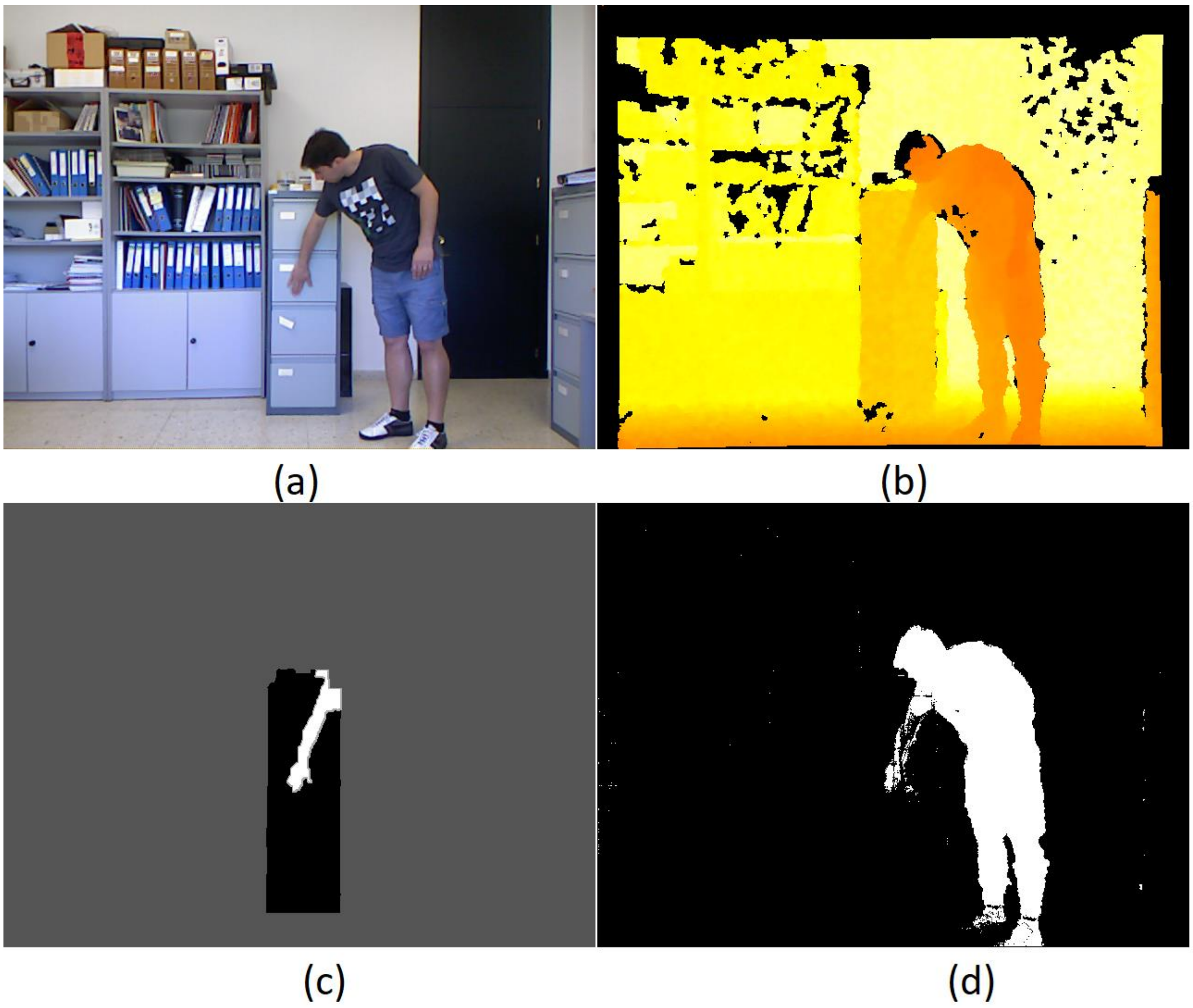


Figure 3.15. *Result of the proposed method in frame 1073 of DCamSeq sequence. (a) Colour frame, (b) Depth frame, (c) Ground-truth, (d) Proposed method.*

Table 3.6. *DCamSeq sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG}_{\mathrm{RGB-D}}$	2.11	1.29	15.25	0.09	1.31	0.02	0.61	0.14	0.61	0.11	2.0
CL_{W}	2.46	1.82	32.21	0.26	0.66	0.01	0.55	0.14	0.51	0.12	3.0
$\mathrm{GSM}_{\mathrm{UB}}$	2.42	1.70	49.84	10.73	0.55	1.57	0.37	0.17	0.40	0.14	4.0
$\mathrm{GSM}_{\mathrm{UF}}$	2.40	2.23	47.62	32.39	0.72	1.61	0.37	0.27	0.41	0.28	4.0
$\mathrm{PBAS}_{\mathrm{Bin}}$	5.06	12.38	46.23	32.46	3.59	13.09	0.30	0.21	0.38	0.23	6.2
$\mathrm{VIBE}_{\mathrm{Bin}}$	2.20	1.86	40.58	23.51	1.29	2.15	0.41	0.22	0.48	0.22	3.2
Proposed Method	2.43	2.21	52.44	30.45	0.56	1.64	0.36	0.25	0.38	0.25	5.2

ColCamSeq sequences

It has been made to investigate the possible error of the algorithms in the case of colour camouflage. Figure 3.16 illustrates an example of colour camouflage. In order to have an accurate measurement of the colour camouflage, the ground-truth and ROI of this dataset are only around the board and white box in the scene. Therefore, the other part of the moving object (the person) is out of the measurement area.

Table 3.7 illustrates the result of ColCamSeq sequence. The proposed method could achieve the highest similarity measure and average in FP, FN and TE which lead to getting the lowest RM. This means the proposed method performed well in this scenario and could almost completely detect the whiteboard from the same background colour. The reason behind this is our depth model recognised the board is not part of the background and consequently the system detect it as a foreground.

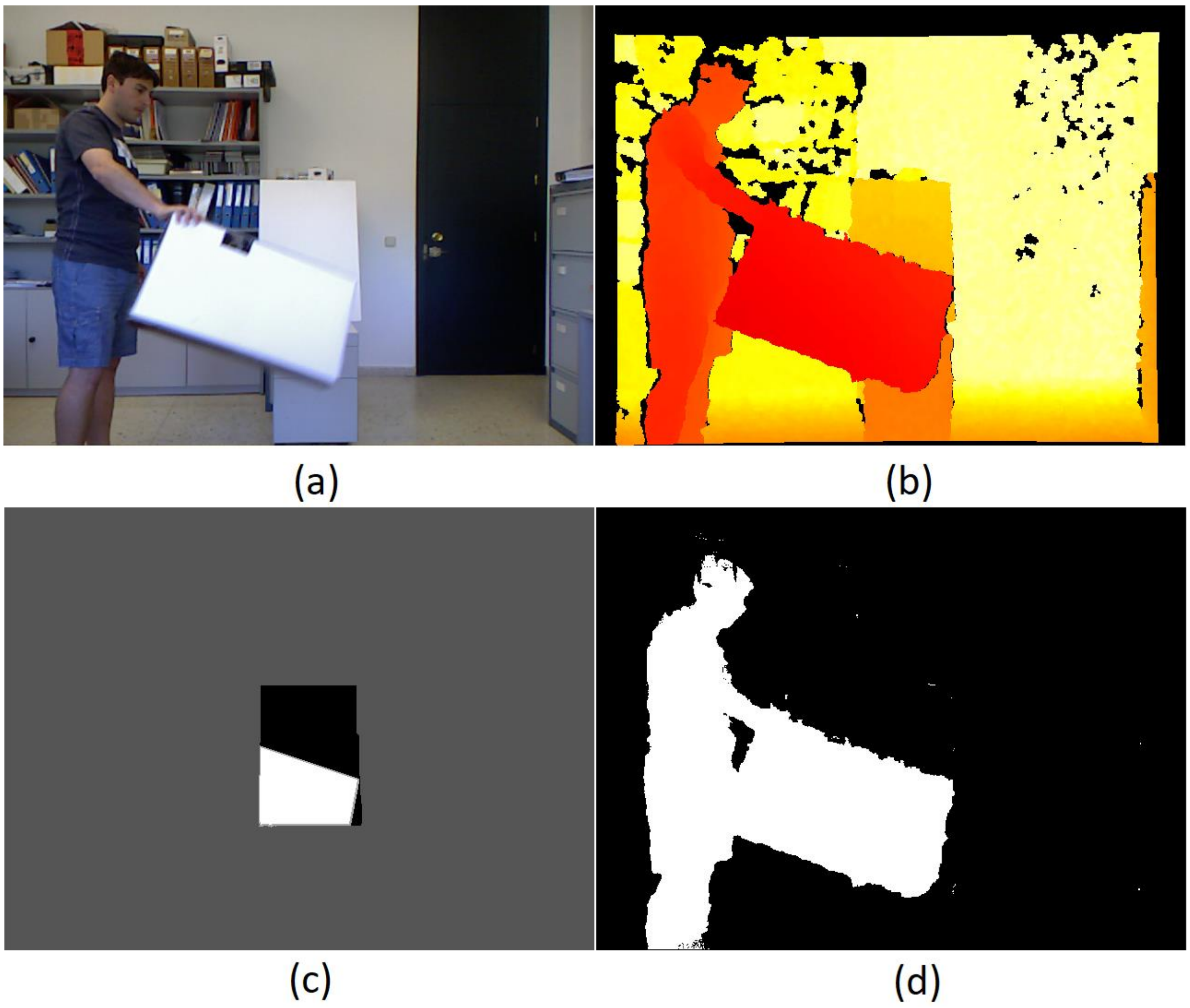


Figure 3.16. *Result of the proposed method in frame 1078 of ColCamSeq sequence. (a) Colour frame, (b) Depth frame, (c) Ground-truth, (d) Proposed method.*

Table 3.7. *ColCamSeq sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG_{RGB-D}}$	3.49	3.40	3.38	0.02	6.13	0.14	0.91	0.09	0.81	0.08	4.8
$\mathrm{CL_W}$	3.20	2.77	3.52	0.09	2.92	0.10	0.89	0.15	0.77	0.16	4.8
$\mathrm{GSM_{UB}}$	1.72	2.66	2.70	6.83	3.50	7.39	0.93	0.09	0.92	0.08	2.0
$\mathrm{GSM_{UF}}$	2.04	2.66	0.65	1.39	4.74	7.43	0.91	0.11	0.90	0.08	3.2
$\mathrm{PBAS_{Bin}}$	8.60	8.99	0.13	0.36	17.32	20.20	0.76	0.26	0.81	0.14	5.4
$\mathrm{VIBE_{Bin}}$	4.47	7.02	0.39	1.25	10.68	19.22	0.86	0.13	0.86	0.09	4.8
Proposed Method	1.94	2.68	0.75	1.74	4.37	7.30	0.92	0.09	0.91	0.07	2.6

ShSeq sequences

This sequence considered to test the impact of shadows in the scene. As table 3.8 illustrates the result of this sequence, the proposed method could successfully detect the foreground object and avoid the shadow of the box on the floor. The total error shows that the proposed method has the lowest amount of errors and the highest similarity measures (S and S_B) compared to the other methods. Accordingly, this allowed the proposed method to achieve the lowest RM which demonstrates the best performance between all other techniques. Figure 3.17 illustrates an example from ShSeq sequences which has been also demonstrated in the authors' paper [48]. This qualitative comparison shows the accuracy of the proposed method with other state of the art algorithms. These algorithms are (c) $\mathrm{MOG_{RGB-D}}$, (d) $\mathrm{CL_W}$ output, (e) $\mathrm{GSM_{UB}}$ output, (f) $\mathrm{GSM_{UF}}$ output, (g) PBAS output, (h) $\mathrm{Vibe_{bin}}$ output, (i) Proposed method output. Undoubtedly, $\mathrm{MOG_{RGB-D}}$, PBAS and $\mathrm{Vibe_{bin}}$ methods incorrectly detect the shadow area as part of the foreground region. $\mathrm{GSM_{UB}}$, $\mathrm{GSM_{UF}}$ and the proposed method performed well in this frame. The proposed method only failed to detect trivial part of the box. However, the proposed method has shown great accuracy around object boundaries.

***Table 3.8**. ShSeq sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG_{RGB-D}}$	3.94	1.54	0.59	0.02	4.50	0.07	0.77	0.09	0.66	0.05	5.6
$\mathrm{CL_W}$	0.81	0.35	1.60	0.05	0.68	0.02	0.94	0.04	0.71	0.07	3.0
$\mathrm{GSM_{UB}}$	0.87	0.33	0.98	0.88	0.88	0.42	0.93	0.03	0.76	0.06	3.4
$\mathrm{GSM_{UF}}$	1.66	0.38	0.14	0.19	1.92	0.44	0.89	0.04	0.65	0.05	3.8
$\mathrm{PBAS_{Bin}}$	3.92	2.73	0.35	0.31	4.48	0.10	0.78	0.11	0.60	0.03	5.4
$\mathrm{VIBE_{Bin}}$	3.72	0.99	0.06	0.15	4.31	1.17	0.78	0.07	0.64	0.03	4.4
Proposed Method	0.80	0.41	0.88	0.70	0.81	0.48	0.95	0.03	0.82	0.06	2.0

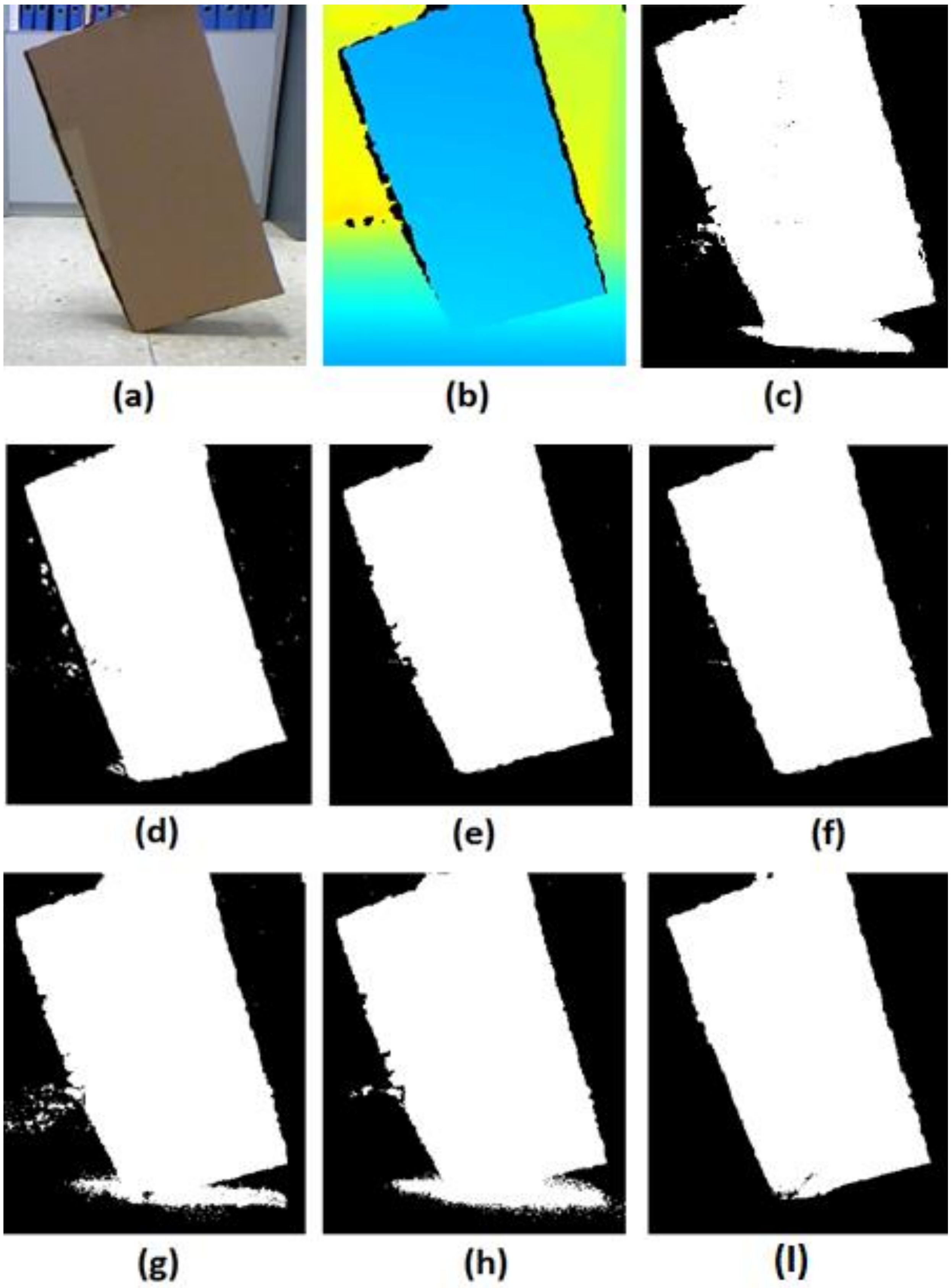


Figure 3.17. *An example of ShSeq sequences.*
(a) Colour data, (b) depth data codified in colour,(c) MOG_{RGB-D} (d) CL_W output, (e) GSM_{UB} output, (f) GSM_{UF} output, (g) PBAS output, (h) $Vibe_{bin}$ output, (i) Proposed method output.

The RM chart which illustrated in figure 3.18 summarises the results shown in Tables 3.5-8 by representing the RC and RM for an individual method in each sequence. The lower amount for RM and RC shows better result. As illustrated in this figure our method could achieve the lowest RM in GenSeq, ColCamSeq and ShSeq compared to the other five algorithms. This shows the best overall performance in all these benchmark datasets. However, in the DCamSeq the proposed method has the highest RM value. This means the proposed method is not able to perform well in case of depth camouflage but in all the other scenarios, it is able to demonstrate the best result. Indeed, according to RC values which has calculate the overall performance of the algorithms in these four sequences, the proposed method outperforms among these six methods as it could achieve one of the lowest amounts of RC. Despite the positive result PBAS previously achieved in colour only datasets, in these RGB-D sequences, it presented the weakest performance in all four scenarios by attaining the highest RM and RC. The main reason for this failure is that PBAS was originally introduced for colour frames only and it cannot tolerate the noise of depth frames. We have used the original parameters in this comparison. However, it might be possible to achieve better results by changing the parameters.

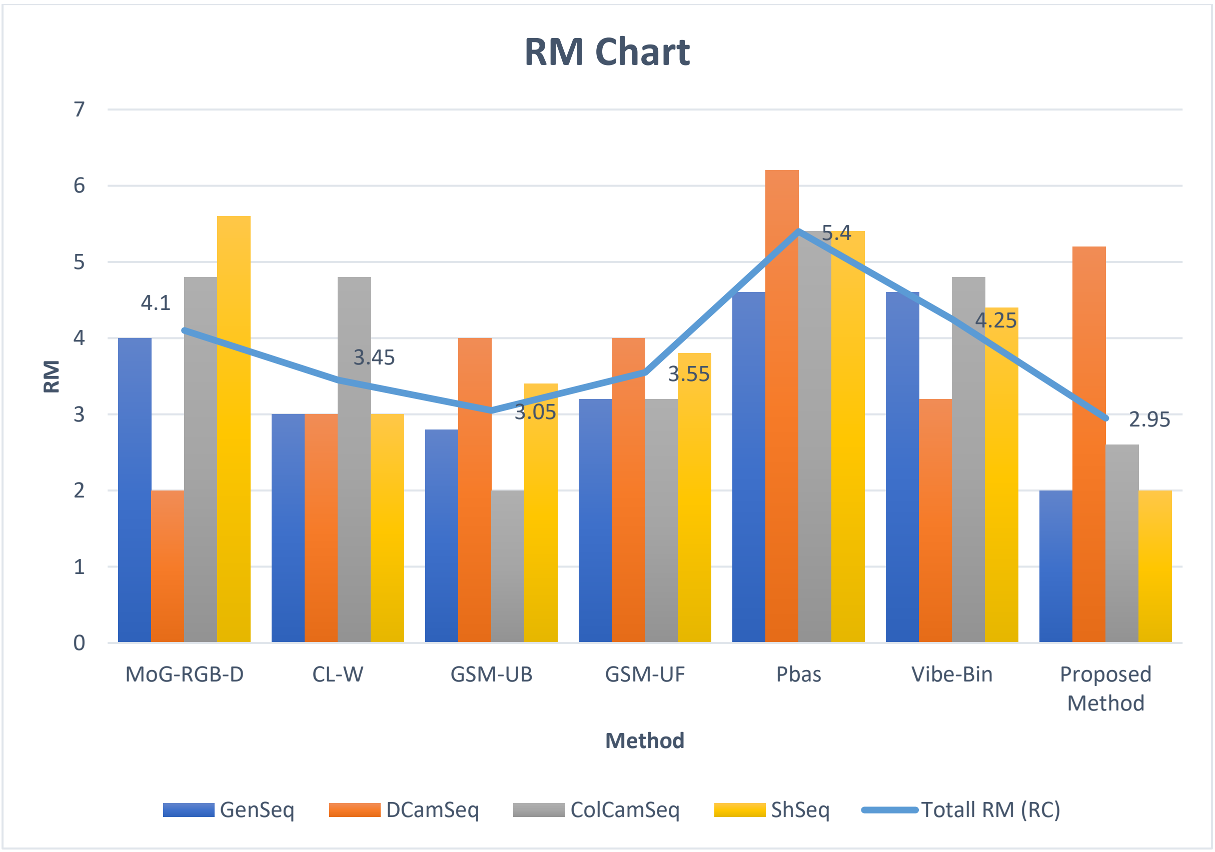


***Figure 3.18**. RM chart.*
shows the overall performance of CL_W, GSM_{UF}, GSM_{UB} , Mog_{RGB-D}.PBAS and Vib_{bin} and proposed method in GenseqSeq, DCamSeq, ColCamSeq and Shseq sequences. Each bar represents the RM and the blue line shows RC. The lower RM and RC values are the better the performance is.

The results in figure 3.18 show the system is able to cope with the dynamic background by using the blind update which randomly exchanges pixels regardless of being background or foreground in every couple of frames. This will help the system to have a more valid and accurate model. However, it can also lead to weak detection in the case of a stationary or very slow-moving object which the moving object will be absorbed to the model after a while. In addition, non-parametric algorithms normally have difficulties in bootstrapping scenarios as these methods require some frames to built and update the model over time. To verify these weaknesses, three sequences which stationary moving object occurs in the scene from SBM-RGBD [116] has been illustrated in figure 3.19. Where (a) is a colour image, (b) depth image, (c) ground-truth, (d) original method and (e) proposed method. And also the following sequences have been used for qualitative evaluation.

1. **Abandoned1:** The scenarios in this video is made to test the tolerance of the method in the creation of the ghost phenomenon. A box will move from the stationary position to the top of the cupboard and stay stationary for some frames.
2. **Abandoned2:** This sequence is made to test the tolerance of the method for a stationary foreground object. A bag will be dropped in the scene and remain there for a while.
3. **MovedBackground1:** It aims to measure the accuracy of the method in case of background changes. A box drops from a cupboard and stays on the floor.
4. **adl24cam0:** This sequence is made to test the tolerance of the method in the bootstrapping scenario. A man standing in the room from the beginning and he only sits and stands up again.

The sequences are dynamic and each row in the figure represents one of the last few frames of the sequence. These sequences are abandoned1, moveBackground1, abandoned2 sequences where a stationary moving object exist. Segmentation results which are closer to the ground truth (white area) show more accurate detection. The grey area shows outside of the region of interest which is not part of the evaluation.

In the first sequence (abandoned1), the brown box moves from the floor to the top of the cabinet and stays stationary. As the foreground stays stationary, it will gradually absorb to the model in the method.

In moved background1, the brown box drops from the top of the cabinet to the floor and stays stationary. The foreground gradually absorbs to the model due to the blind update.

In abandoned2 sequences, a black bag drops in the scene and stays stationary. The foreground starts absorbing to the model and the foreground will gradually disappear in segmentation mask. This starts from the bottom of the bag as it is closer to the background.

In adl24cam0 sequences, the person standing in the middle of the room and he exist there from the beginning frames. Therefore, foreground pixels will be in the background model during the initialisation which can cause poor detection accuracy. The background model gradually adapts to the background by replacing the real background pixels in the model.

The proposed method clearly demonstrated poor accuracy in the results in these bootstrapping and stationary moving object sequences. Therefore, an investigation will be initiated in the next chapter to improve the weakness of the proposed method in detecting the bootstrapping, stationary moving objects and resolve the issue of depth camouflage scenarios with the current hardware.

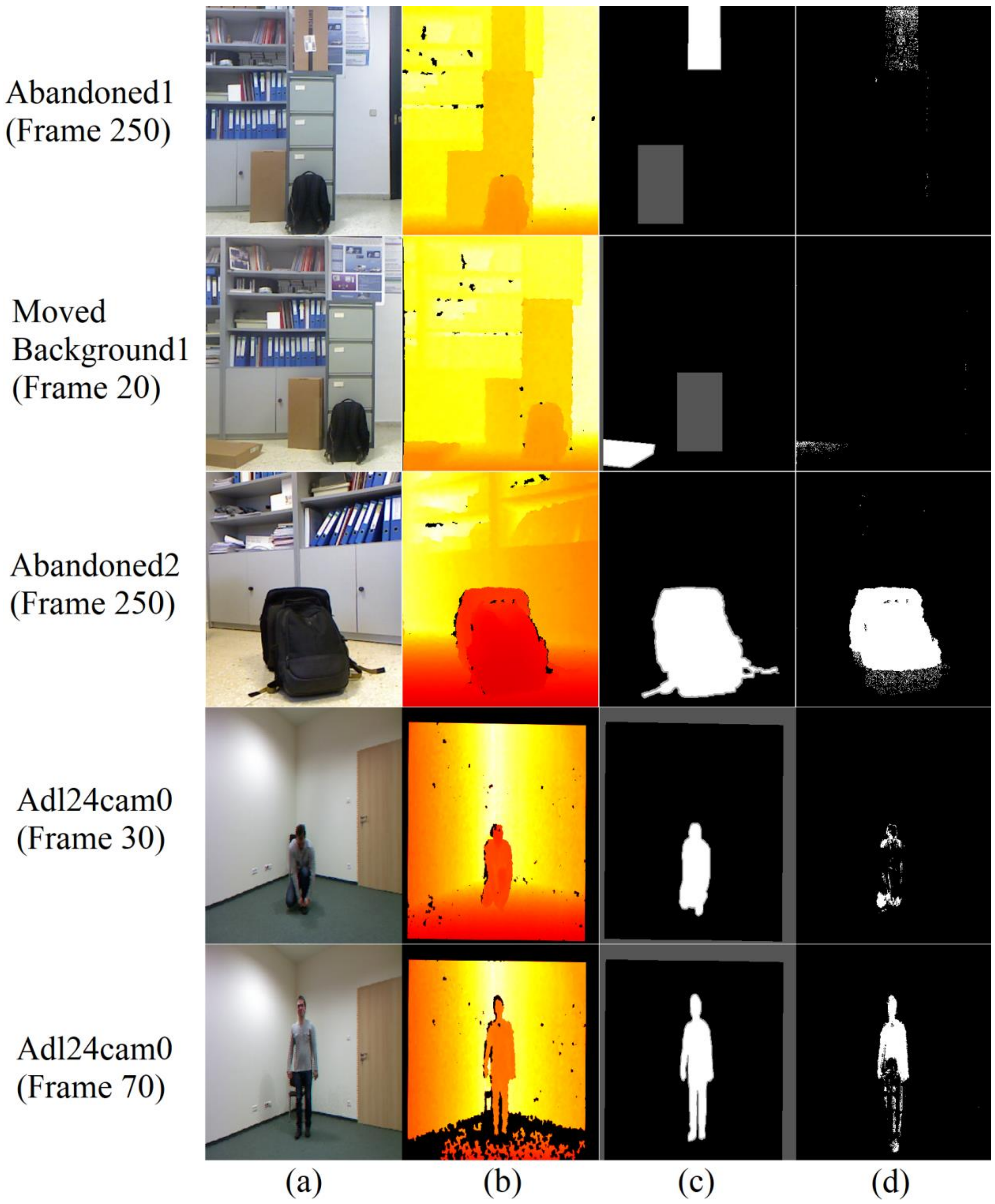


Figure 3.19. *Example of qualitative comparison in a stationary moving object dataset. The grey area shows outside of the region of interest. (a) Colour image, (b) Depth image, (c) Ground-truth, (d) Proposed method.*

3.4 Real-time Experiment

The proposed system has been tested in a live application within an indoor environment to test the accuracy of the system in a live application. A basic control system based on the proposed approach has been implemented to controlling a MAV and navigate it around the area. The computational cost of the algorithm is calculated as the mean rate of the processing time of the algorithm. The test was performed on a laptop with an Intel (R) Core (TM) i7-6700HQ CPU @2.6 GHz and 8 GB RAM along with Microsoft Kinect v2 sensor and a parrot AR.Drone [11]. As illustrated in figure 3.20, the coloured cover has been added to the front of the drone for more accurate depth data on the drone body and recognition of the front of the drone for navigation by the control system. The drone body was originally black and it absorbs the IR signals which creates ADO on surfaces of the drone.

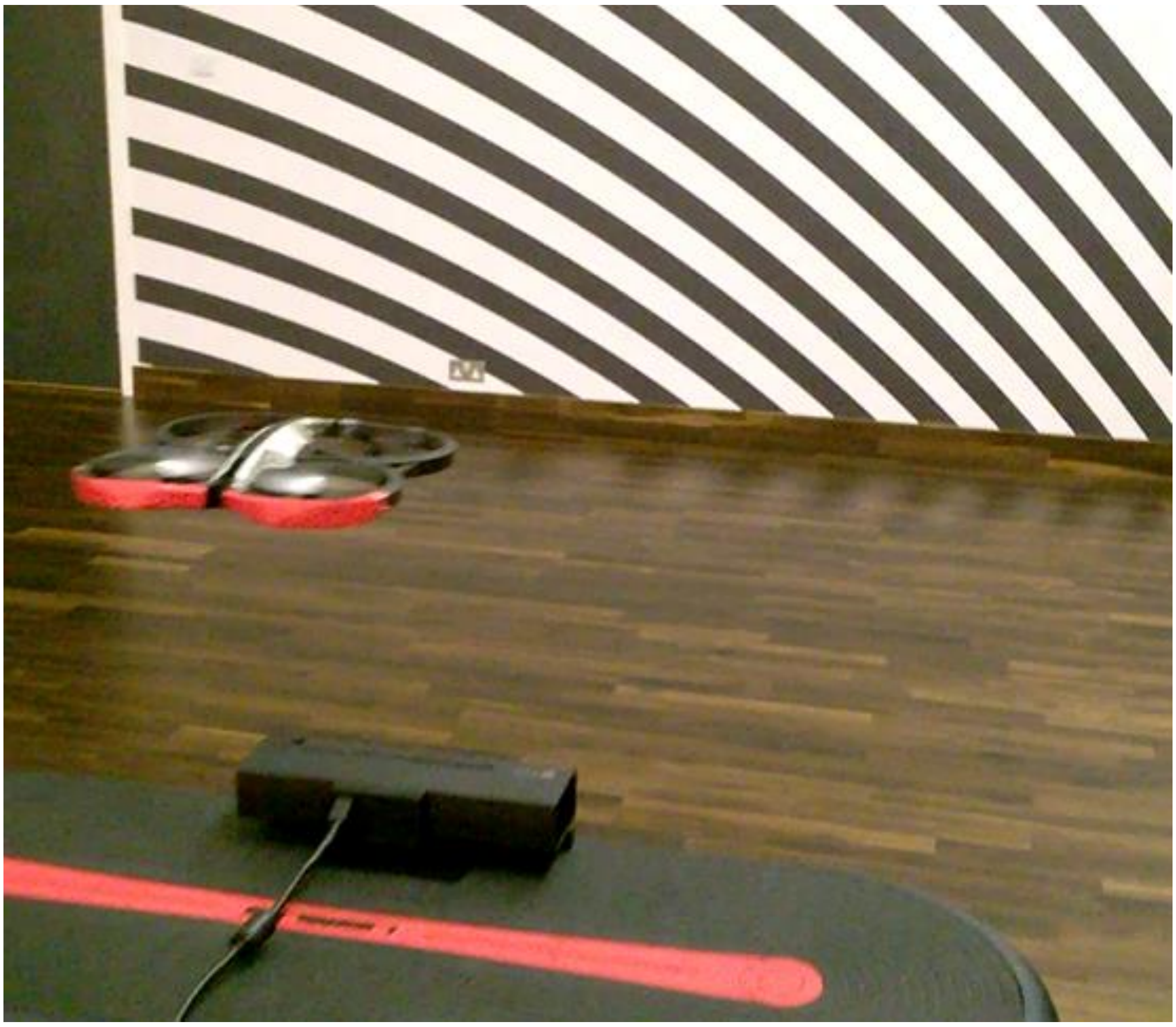


Figure 3.20*. An AR.Drone was used to testing the proposed method in real-time.*

It is worth mentioning that the proposed algorithm used during these tests has been implemented in Linux with AR. Drone SDK in C++ and OpenCV library [119] without any specific code optimisation as the aim of this experiment is to show that the proposed algorithm can be successfully run at real-time frame rates and therefore no effort has been made to optimise the code/set-up.

The quadcopter could successfully land on the floor and the total flying time was 210 seconds with a mean processing time of 68.8 ms and looking at the frame rate, it is about 15fps. The demonstration showed the proposed system could safely control the behaviour of the quadcopter (staying in the external camera frame and not collision to other objects such as a wall) in typical processing time required by other state-of-the-art systems [120][121] for each frame at real-time. Figure 3.21 shows the total time obtained by the system for the first 200 frames of the test. The minimum time for a single frame on this test was 55 ms and the maximum was 82 ms. A flat pattern at the beginning of this figure demonstrates the initialisation stage of the proposed method and the rest of oscillations belongs to the main loop of the algorithm including segmentation and background model update process.

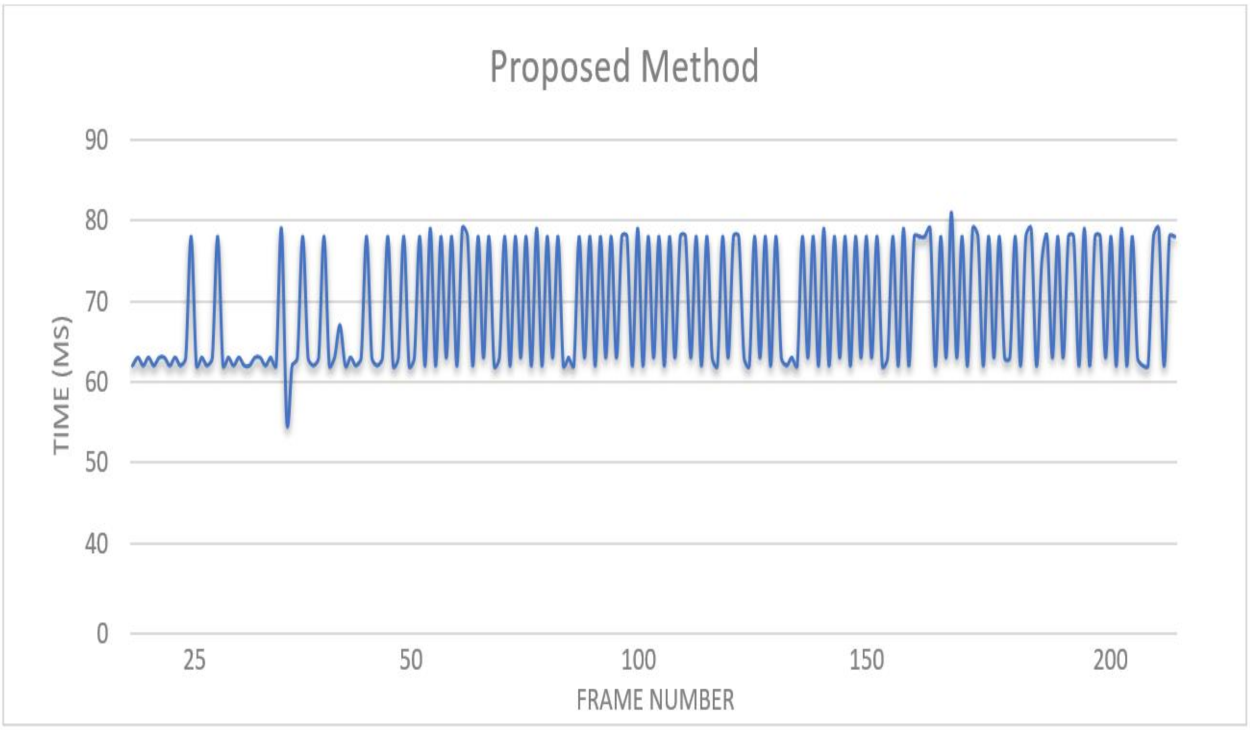


Figure 3.21. *The computational time for the system to control the quadcopter. The time for the first 200 frames is shown.*

3.5 Summary

In this chapter, we have presented a novel nonparametric approach for detection of different type of moving objects such as fast MAVs using background modelling and segmentation by the history of previously observed pixel values similar to the previous work of ViBe and PBAS algorithm. The proposed method has four steps which are: initialisation, post initialisation filtering, classification and update. Our system produces one model for colour and one for depth. By combining colour and depth model together to produce the final classification, we could improve the background segmentation accuracy in some challenging scenarios. These models get updated for each pixel which identified as a background. This update process is based on the distance of new pixel and existing pixels in the model. In fact, the lowest depth pixel in the model will be exchanged with the new pixel. This update method has never been implemented in this way before. Additionally, after α frames, the system blindly updates the models regardless of the pixel being background or foreground to prevent permanent misclassification and production of ghost phenomenon. These updates allow creating a more accurate depth model regardless of noisy depth frames. For this reason, the depth model has a greater influence on the final segmentation mask. In particular, the system relies more on the colour model when depth is not available or it cannot surly decide the foreground/background (e.g. near object boundaries). This helped the system to significantly reduce the amount of false detection in case of sudden illumination changes and appearing of shadow on the floor.

The results and evaluation section demonstrated that the proposed algorithm in the two UAVs sequences that we have captured achieved the best performance by having the lowest RM. However, the FN is high in both sequences which indicate some part of the foreground has been identified as background due to the size, speed and surface of the UAVs.

In the other four public datasets, the proposed method has the most accurate and reliable outcomes in comparison with other state-of-the-art methods. Furthermore, we have shown that the $\mathrm{GSM}_{\mathrm{UF}}$ and the proposed method achieved the best overall results by having the lowest RC as illustrated in figure 3.18. This system also improved the overall performance of the detection of high-speed moving MAVs by combining the depth and colour model to produce the segmentation and update models to make it more accurate time after time. These outcomes are also supported by the table 3.5-3.8, where it has been highlighted the robustness of the proposed method by achieving the lowest value of RM in three sequences and only a poor performance in one sequences (DCamSeq) as the proposed method has difficulty in detecting the moving object in the occurrence of depth camouflage. The accuracy of the proposed algorithm in the depth camouflage could not meet the standard and it requires further improvements. This problem can be improved by reducing the acceptable threshold amount and use a more accurate depth sensor.

On the other hand, the system is able to cope with the dynamic background by using blind update which randomly exchanges pixels regardless of being background or foreground in every couple of frames. This will help the system to have a more valid and accurate model. However, it can also lead to weak detection in the case of a stationary or very slow-moving object which the moving object will be absorbed to the model after a while. Therefore, in the next chapter, we are going to investigate to improve the weakness of the proposed method in detecting the stationary moving objects and resolve the issue of depth camouflage scenarios with the current hardware.

Chapter 4
Moving object detection using adaptive blind update policy and RGB-D camera

4.1 Introduction

Background subtraction is one of the main phases of many computer vision applications. The main goal of these algorithms is to separate the moving objects (known as foreground) in the scene from a robust model of the static environment (known as background)[100]. Traditionally a scene model requires a regular update to adapt to the changes over the time in the real-world scenarios. These changes could be appearing as a new object in the scene, moving a scene's object or illumination changes.

In the previous chapter, a nonparametric method has introduced which stores the previously observed pixel values in the models and the system use update methods to keep the model updated to the changes in the environment. However, how to update the background models efficiently and effectively is still challenging for the proposed method in the previous chapter and other state-of-the-art methods. The type of samples we can choose to create the scene model and for how long it is valid is important for any model. For example, first-in-first-out is one of the classical approaches for updating the background model which discards the old samples in the model and substitute them with the new pixels after several frames or seconds. These traditional approaches update all the old pixels in the model which often is not essential as those pixels may still be valid samples. On the other hand, updating the scene model only by those pixels that are recognised as background or also involving the foreground pixels has always been discussed. These procedures are known as a blind and conservative update policy [8]. Conservative update scheme never incorporates those pixels classified as part of foreground region. Theoretically, this policy is a suitable choice which can produce a sharp detection of the moving objects. However, in most practical scenarios, it can lead to deadlock situations and production of ghosts' phenomenon. For instance, a change to the background of the scene can cause the background model to incorrectly contain foreground samples. This prevents to update background model and therefore cause a permanent misclassification which many real-world scenarios lead to these situations.

On the other side, blind update such as the method used in ViBe [8], MoG [9] and proposed method in the previous chapter incorporates all sample values into the background model update regardless of being identified as foreground or background. In the previous chapter, we have realised that the main drawback of the blind update is a weak detection of a slow-moving object or those foregrounds that stopes and remain static for some period of time. These regions which contain the foreground area, gradually will become part of the scene model. In the literature, these motionless objects are typically called Stationary Foreground Objects (SFOs) [6]. Detection of SFOs is one of the well-known topics of background subtraction techniques which attracted the attention of many researchers in the last few years [10]. In SFO scenarios, pixels of the stationary object gradually absorb to the background model and eventually the model will adapt to the motionless objects. In this chapter, we are going to tackle this issue and a new blind update scheme will be proposed which is able to improve the detection accuracy of SFOs as well as fast-moving objects. In particular, in this chapter, we are going to address and investigate the following research question that what is the effect of using adaptive blind update policy on pixel-wise moving object detection. This method is new and has never been tested before. This method is part of our contribution to this thesis. Also, we have published some content of this chapter earlier in an IEEE journal [122].

In the remaining of this chapter, section 4.2 will be a discussion about the background of stationary foreground objects, section 4.3 will be in detail structure and explanation of the proposed method, section 4.4 experimental results, and comparison of the proposed method with other state-of-the-art algorithms in publicly available datasets and real-time evaluation has been described in detail, section 4.5 is the summary of this chapter.

4.2 Background

In order for the scene model to continuously adapt to the changes in the background, the model requires a continuous update. An example of these changes is illumination changes, appearing or removing an object in the scene or moving a scene's object. On the other hand, if the model rapidly receives an update, then the system will face difficulties to detect the stationary foreground objects as the foreground will gradually absorb to the model. This is one of the main weaknesses of the method introduced in the previous chapter [3]. We will call this method as NBMS in the rest of this thesis. To overcome these challenges identified in the previous chapter, it is important how the samples were collected to create the scene model and for how long these samples are useful. This problem has been broadly discussed in the literature.

The authors of [10] proposed a multi-object tracker adapted for conveying systems which is based on a feedback loop from tracking to detection. The main goal of their work was to stop the adaptation of the regions which belongs to stationary objects. They have used state-of-the-art background subtraction techniques ViBe [8] and the Gaussian Mixture Model (GMM) [13] to implement their ideas. The neighbouring update process of ViBe has been disabled for pixels of stationary objects to prevent the absorption of SFOs into the background model. On the other hand, the learning rate of α is considered to incorporate pixel samples in the GMM model. For the stationary objects, the α is set to zero. According to the authors of [10], significant improvements in tracking results have been achieved in real video sequences.

The new method proposed in [41], attempted to efficiently recognise the SFOs based on three nonparametric background models (long term, medium term, and short term). The goal of this method was to improve the detection quality of classical moving object detections and using novel Finite State Machine in scenarios featuring moving objects that become motionless (e.g. people in offices or vehicles on urban roads).

Recently, a new nonparametric method based on a self-adapting parameter called DTNBM has been introduced [50]. This method uses a dual target updating approach which can simultaneously tackle the background and foreground pixels. In addition, it can use different updating tactic for different types of pixels. A controlling threshold adaptation procedure is intended to help DTNBM to be applied in different scenarios. This shows adapting update policies could significantly help background subtraction methods to increase the application of these methods in various challenging situations. This ability inspired us to investigate in more detail an adaptive updating strategy into our moving object detection method to be able to deal with a different type of moving object such as fast, slow and stationary moving object.

Another way to increase the accuracy of the background subtraction method in SFOs is to have more information about the scene in the past by increasing the number of samples in the model. However, this approach is not practical in a live application due to the high computational cost. Besides, some foreground pixels could be stored in the background model which leads to misclassification. Another solution to this problem is to track the moving object and update the model based on tracking of the moving object. This is not new and has been previously applied differently. For example, recently the authors of [123] introduced a non-parametric method based on KDE [29] which models the motion of every foreground object and predicts the position of each object in the current frame. This helps to decrease the search zone for the foreground object and increases the probability of the data to match the model for each pixel.

All previously stated researches indicate that the best way to improve the detection accuracy of a stationary moving object is through tracking the object and using adaptive tactics in order to change the updating parameters based on the speed of moving object. Although previous researchers could improve the detection accuracy of slow and stationary moving objects, this issue is still challenging as there is no universal method to be able to detect the moving object in all challenging scenarios including SFO and therefore this topic requires further research. In this chapter, we are going to investigate adaptive blind update with the aid of depth data and colour frames in pixel-wise moving object detection.

4.3 Proposed method

The proposed method in this section is an improvement of the method introduced in the previous chapter known as NBMS [3]. This method is more complex and it has significantly improved the accuracy of the original method to deal with more challenging situations such as stationary moving objects, bootstrapping and shadows. The main novelties and contributions of this research are as follows; 1) Adding adaptive blind update method which changes the frequency of blind update based on the speed of moving object. 2) The segmentation rules are more effective and sophisticated compared to the original method. 3) Bootstrapping detection and segmentation proposed to improve the accuracy in bootstrapping scenarios. Bootstrapping sequences are generally defined as those sequences that foreground objects exist in all frames from the beginning [124]. Shadow detection method proposed based on CIE L*a*b* colour space [125]. The method is also thoroughly evaluated in all sequences of SBM-RGBD challenge dataset [116].

Our method creates background models by a history of previously observed pixel values. Then current pixels will be compared to the model for foreground segmentation. The proposed method involves different phases to cope with the changes in the background and effectively works in a live application. Figure 4.1 demonstrates the flowchart diagram of the proposed method.

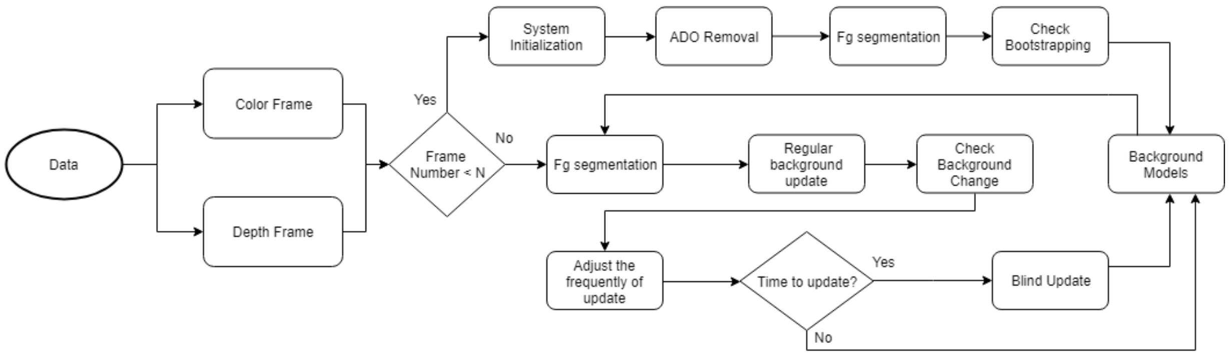


***Figure 4.1**. Flow chart of the proposed object detection method.*

The proposed method similar by the original method, stores the first *N* frames to produces one model for colour and another model for depth frames. This step called the "System Initialisation" which require to remove ADO pixels in "ADO Removal" before creating the "background Model". These steps are explained in detail in the previous chapters (Section 3.2.1 and 3.2.2). However, in this method, the segmentation process starts after the minimum number of samples stored in the model. If any moving object has been detected during the initialisation stage, then bootstrapping had occurred. We called this stage "Check for Bootstrapping". When bootstrapping is occurring, some pixels of the foreground stores in the background model. Therefore, the segmentation result (foreground detection) will be only a fraction of the foreground. To solve this problem, during initialisation the system needs to find the shape of each object existing in the scene by using edge detection techniques. This has been reached by combining the results of the colour and depth frames achieved by canny edges detection algorithm [99]. The system checks each object exists in the scene with the segmentation result (foreground). If a high percentage of any object in the scene is identified as a foreground, then the whole object will be added to the foreground mask. This significantly aids the system to increase the detection rate in the bootstrapping scenarios compared to the original method. However, in some rare cases, this could also increase misclassification.

Once the initialisation has finished and the background model has been completed, the system moves to the main loop to start comparing individual pixels to the model to begin the segmentation of the pixels as a foreground or background and produce the result which is called "Fg segmentation".

After segmentation, the background model will be updated with two methods "regular background update" and "blind update". First, only those pixels marked as a background will be updated. In addition, after α number of frames, the system blindly swaps the new pixels with the background model regardless of being marked as foreground or background. The rate of α was fixed in the NBMS and defined before running the application (explained in section 3.2.5). However, in the proposed method, the frequency of blind update (α) will be changed based on the speed of the moving object. This significantly helps the system to improve the weakness of the original method and increase the detection rate of a slow and stationary moving object. The main reason behind this is the rate of α will be significantly reduced in the case of a stationary moving object and consequently, the blind update will less frequently apply to the background model. This prevents the moving objects from absorbing into the background model for much longer than the original method. Also, another step has been added to the proposed method called “check the background change”. When a change in the background is occurring, the system will significantly increase the frequency of blind update in order for the model to adapt to the changes in the environment. The system identifies a change in the background by comparing the depth frame with a sample depth image (usually with an initial frame when the moving object doesn’t exist in the scene). If an area of a scene has a longer depth compared to the samples, it demonstrates a change in the background has occurred.

As figure 4.1 demonstrates, the proposed method has three main steps of initialisation, segmentation, and update. In the remaining of this section, the new steps of this figure will be described in more detail.

4.3.1 Fg Segmentation

Traditionally Probability Density Functions (PDF) and statistical parameters such as mean or variance are the most common components of the background subtraction algorithms. Alternatively, statistical significance can be used to build a model based on previously observed colour pixel values and depth data. The hypothesis is that, if the same pixel value has been observed a number of times in the same location, the pixel has a high chance of being part of the background.

The process of background subtraction will require classification of each pixel as background or foreground. The value of the current pixel in the colour frame will be compared with the colour model in each location to discover if it is close to some of the samples in the model. In parallel, depth pixels will be compared to the depth model to determine if the pixel is at the same distance or further to the camera. In some cases, RGB and depth have the same individual segmentation result. In other words, by comparing the pixel to the models, both (colour and depth) individually agree whether the pixel is part of the background or not. However, in some other challenging scenarios, they are strongly against each other. This means one of the sensors (colour or depth) has been affected by the noise or limitation of the sensor. An example of these situations could be colour camouflage such as foreground having the same colour as the background, change of illumination, shadow or depth camouflage such as moving the hand on the wall.

To produce the final decision (foreground mask), the system should find out from which sensor the noise is coming from and which sensor is more reliable. We have introduced a set of rules based on the facts that the system follows in the segmentation process to reduce the effect of these noises for producing the foreground masks. Figure 4.2 demonstrates the flowchart diagram of the rules which the system follows for the segmentation of each pixel. Thus, this diagram shows in detail the whole "Fg segmentation" step in figure 4.1.

Unlike RGB cameras, depth sensors are resistant to illumination effects. Besides, the depth accuracy has been significantly increased in the last few years with the creation of new sensors such as Microsoft Kinect V2 sensor [113][114]. Consequently, our method has relied more on depth outcome to produce the result. However, depth sensors still have some limitation such as depth camouflage. Generally, these limitations can be eliminated by using colour frames to identify the moving object. In some cases, both sensors can fail to detect the moving object or by error detect the background as part of the foreground, for example, a shadow on the ground or moving a hand on the wall.

To solve this issue in the dynamic background and improve the detection accuracy, some researchers used CIELa*b* colour space, where the chrominance components are totally unlinked from the luminance component [126]. For example, the authors of [127], applied random homogeneous region-based background modelling in the CIELa*b* colour space to detect swimmers in swimming pool environments, or in [85] CIELa*b* colour along with depth data used for a real-time segmentation application. The main reason the authors choose this colour space was consistency in perception which concluded to better performance.

CIE L*a*b* colour coordinate for the colour frames is defined by the Commission Internationalede L'Eclairage (CIE) in 1978 and then officially introduced by Hunter and Harold in 1987 [125]. One of the most significant characteristics of the L*a*b* space is device independence. L*a*b* colour space is built on one channel for Luminance (lightness) (L) and two other channels for colour (green-red and blue-yellow) (a *and* b). $L^* = 0$ represents the darkest black, and $L = 100$ the brightest white. a^* and b^*, will represent true neutral grey values at $a^* = 0$ and $b^* = 0$. The a^* axis represents green at negative values and red at positive values. The b^* axis represents blue at negative values and yellow at positive values.

Unlike the RGB space, L*a*b* colour is intended to approximate human vision where the L element closely indicates the human perception of lightness [128]. Thus, it can be used to check the colour value of pixels (a *and* b) without interfering of illumination (L) component.

In the proposed method in this chapter L*a*b* colour coordinate have been implemented instead of RGB colour space which significantly helped the system to improve the segmentation accuracy and detection of the shadow area. This method identifies an area as a shadow where the L component is low (close to 0) and the depth frame on that location shows no change compared to the depth model.

Formally, if we denote a 3d point as an $X=(x,y,z) \in R^3$, $d(X)$ the value in the depth and $v(X)$ the value in a given colour at location X in the new frames. v_i and $\mathrm{d_i}$ shows an index of i in a background sample value of each background pixel located at X which demonstrated by a collection of N background depth and colour sample values taken before as:

$$M(X)_{Lab} = \{v_1, v_2, v_3, \ldots, v_n\} \tag{4.1}$$

$$M(X)_D = \{d_1, d_2, d_3, \ldots, d_n\} \tag{4.2}$$

We refer to fg as a foreground pixel, bg as a background pixel, $M(X)_D$ as a background depth model and $M(X)_{Lab}$ as a background colour model at location X.

At first, to identify each pixel of the new frame as a bg or fg, the system checks the depth data. If the depth is ADO (an unknown value) at that pixel location, it only compares the colour value with the $M(X)_{Lab}$. This mean, the system will solely rely on the decision of the colour if depth value is not available in any pixel location. If the difference in colour value is smaller than $\mathrm{Th_{RGB}}$ (acceptable threshold), we count the pixel as a similar colour. Each pixel location which finds at least cardinality amount denoted by $\#_{Min}$ similar pixels will be classified as a bg. Although, the $\#_{\mathrm{Min}}$ is insensitive, it has a positive correlation with the number of samples. This means higher samples in the model require higher $\#_{\mathrm{Min}}$ value. Therefore, as a default we recommend $\#_{Min} = N/5$. This number of samples is reasonably enough to cover the noises and on the other hand, it is sufficient observed samples to decide whether the pixel is part of the background or foreground.

It then checks if bootstrapping has occurred in the sequence. If the system detects any moving object during the initialisation stage, it will categorise the sequence as a bootstrapping. Normally when bootstrapping is occurring, the moving object (foreground) will exist in the background model. This will significantly reduce the detection rate. To improve the detection in bootstrapping sequences, we have set up a special rule.

We attempt to find the shape of each object existing in the scene by combining the results of colour and depth canny edge detection. Then extracts all edges of the scene as an object. The system checks each object exists in the scene with the segmentation result (foreground). If a high percentage of any object in the scene is identified as a foreground, then the whole object will be added to the foreground mask. In the next step, we compare all the non-ADO depth pixels with $M(X)_D$. If the difference is larger than Th_D (acceptable depth tolerance threshold), we count the pixel as having longer depth value. Each pixel location which finds at least $\#_{Min}$ similar pixels, will be classified as bg. Then, pixels which are on the edges and around object boundaries will be only classified based on the colour result as depth value is not accurate around object boundaries [88][48]. All other pixels will be compared with the $M(X)_D$.Those pixels who cannot find $\#_{Min}$ in the same or longer range by considering some tolerance Th_D, will be classified as fg. for the remaining pixels, the colour value of L component will be then checked. If it represents a dark value, then we count this area as a shadow and the pixel will be classified as bg. The remaining pixels will be compared with the depth model again without any tolerance this time, if it could find enough similarity, the pixel will be classified as bg. All other pixels will be decided by comparing to the $M(X)_{Lab}$. Note the value of tolerance in this system is different from other methods. Experimentally, we have realised that depth sensors are more accurate in closer areas. Therefore, the depth tolerance value should depend on the distance of the pixels to the sensor. In fact, the larger the value of depth pixel is, the higher the tolerance needs to be.

As illustrated in figure 4.2, the system first checks whether the depth pixel is available or is ADO. If the pixel is ADO, the segmentation will be only based on the colour result. If the pixel is non-ADO, then it compares the value of depth pixel with the depth model. If the pixel is larger than some pixels ($\#_{Min}$) in the model, the pixel will be identified as a background pixel. This step is shown as “Longer depth compare to the model”. If the pixel is not larger than samples in the model, then the system in “Is bootstrapping occurring” step checks if bootstrapping happened in the scene. In the next step, it checks whether the pixel is on the object boundaries in “is pixel on the edges”. If the pixel is on the boundaries, then the depth pixel is not reliable and therefore it only relies on the result of colour. After that, it compares the depth pixel with the model and considering some tolerance in “Few similarity depth+tolerance samples”. If it couldn’t find $\#_{Min}$ similarity with the model, the pixel will be identified as foreground otherwise, the system checks if shadow occurs in “Is it shadow”. In the last step “Few similarity depth with samples?”, it again checks the pixel against the model without tolerance and if cannot find some similarity, it will segment the pixel based on the decision of colour.

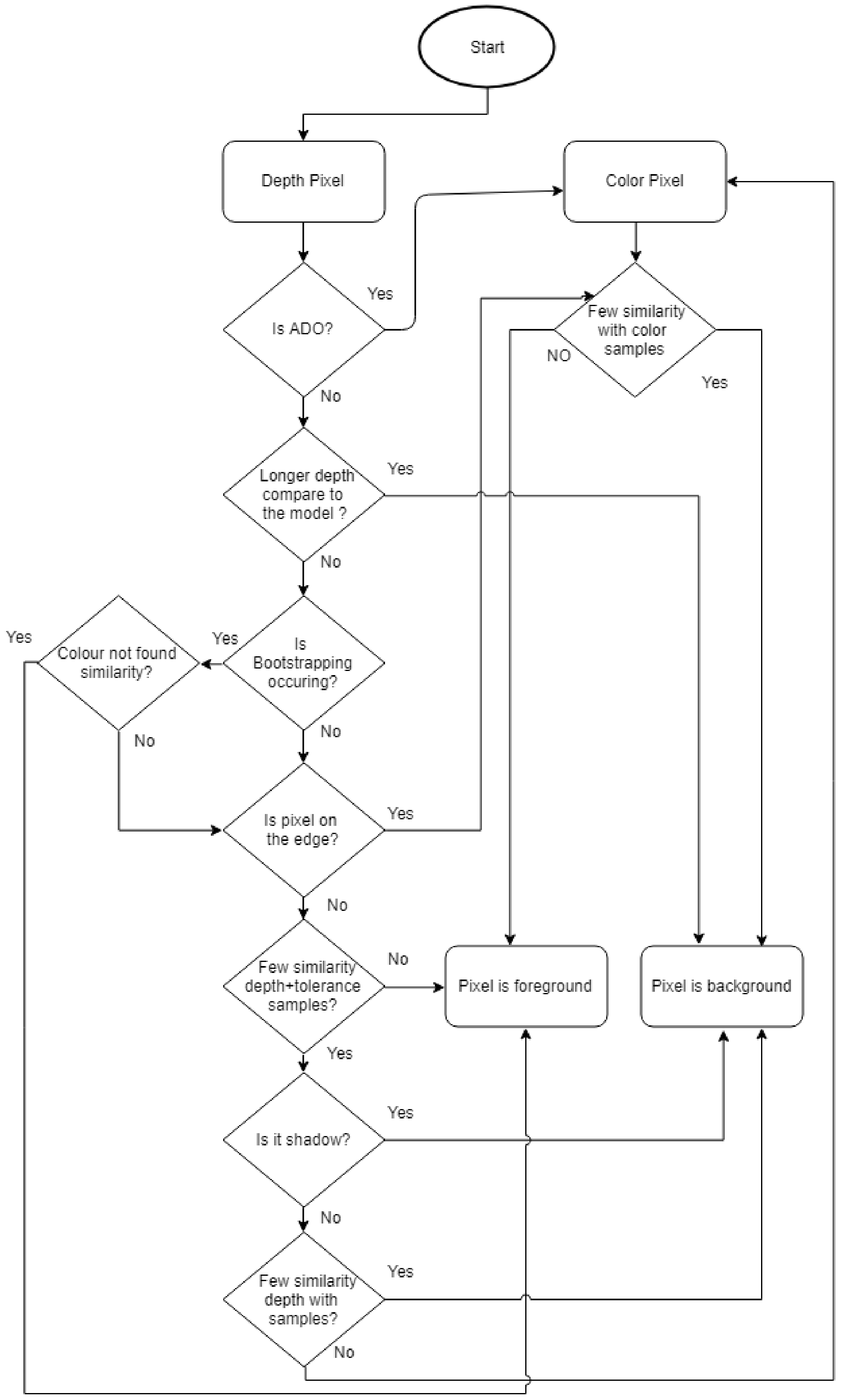

Start
Depth Pixel
Color Pixel
Is ADO?
Yes
No
Few similarity with color samples
NO
Yes
Longer depth compare to the model ?
Yes
No
Colour not found similarity?
Yes
No
Is Bootstrapping occuring?
Yes
No
Is pixel on the edge?
Yes
No
Few similarity depth+tolerance samples?
No
Yes
Pixel is foreground
Pixel is background
Is it shadow?
Yes
No
Few similarity depth with samples?
Yes
No

Figure 4.2. Flowchart of the proposed segmentation method.

4.3.2 Background Model Update

The proposed method is using two schemes to continuously update the background model with the new frames. Regular and blind update aids the models to adapt to the changes in the background over the time. An example of these changes is a new object appearing in the scene, illumination changes and change to the location of an object completely in a different area to the background. Similar to the original method [3], the regular update will only update those pixels identified as a background. This has been discussed in detail in the previous chapter (section 3.2.4).

4.3.3 Blind update policy

Conservative scheme that has been used in the regular update will only affect those pixels that have been identified as part of the background. The main drawback of this policy is that misclassified foreground pixels will be held in the background model. Then real background pixels will permanently classify as foreground and never enter the scene model. This will lead to a deadlock situation and creation of ghost phenomenon. On the other side, blind policy updates the scene model with any pixel regardless of being part of the foreground or background. This policy has some advantages such as being able to adapt to the changes in the background and prevent the production of ghost phenomenon. The weakness of a blind update is that slow and stationary objects will gradually become part of the background. Moreover, the frequency of blind update has also been discussed in the literature. Using a high rate of blind update could cause some foreground pixels misclassified as background. However, the background pixels are rarely misclassified.

In this chapter, we are proposing an adaptive blind update for background model which reduce the weakness of blind update and allows the model to adapt to complex scenarios. The scene with fast-moving foreground objects can tolerate more frequent blind update than slow-moving objects. This is because slow-moving objects will gradually absorb to the model. Therefore, the frequency of a blind update is fundamental and depends on the type of moving object (fast, slow or stationary). Consequently, by tracking the moving object, the frequency of blind update can be changed based on the speed of the object. We have defined three speeds categories for the moving object as fast, slow and stationary. When the moving object is fast, the blind update could occur more often. In the case of slow-moving, we will reduce the frequency and once it is stationary, the blind update rate should be near zero. The proposed method is based on three main stages: 1) Constantly detection of moving object, 2) Track the moving object, 3) calculate the frequency of blind update.

In this research, RGB-D sensor has been used which allows us to produce the 3D position of any pixel in the scene. A simple tracking method has been used in this research to reduce the cost of computation, although it is possible to employ more complex tracking methods.

If we denote the central position of the moving object as $X_t = (x_1, y_1, z_1)$ at the time *t*, a second after (*t+1*), the moving object will be at $X_{t+1} = (x_2, y_2, z_2)$. Then the distance taken by the moving object can be calculated as:

$$D_m(X_t, X_{t+1}) = \sqrt{(x_{2-}x_1)^2 + (y_{2-}y_1)^2 + (z_{2-}z_1)^2} \tag{4.3}$$

The distance taken by the foreground in a second can be used to change the frequency of blind update as the following:

$$\alpha = \begin{cases} High\,frequent\,update & D_m > d_{threshold} \\ Low\,frequent\,update & D_m < d_{threshold} \\ \approx 0 & D_m \approx 0 \end{cases} \tag{4.4}$$

Where α is the frequency of blind update and $d_{threshold}$ is the amount of distance that defines the fast or slow-moving object. If the moving object travel more than $d_{threshold}$ will be considered as fast-moving and lower than that will be considered as a slow-moving object.

4.4 Results and evaluation

To evaluate the performance of the proposed method, three experiments have been carried out in this chapter. At first, the proposed method has been compared in Total Error (TE), Similarity measure (S) and Similarity measure in object boundaries (S_B) to the original method introduced in the previous chapter [3]. Then the result of the proposed method has been compared with the other state-of-the-art algorithm to measure the accuracy of our approach. To achieve this, the proposed method has been tested on the publicly available dataset called SBM-RGBD introduced in [116]. The ground-truth, ROI (region of interest) and a MATLAB code to evaluate the results has been provided with this dataset to compare the accuracy of an individual method.

According to the SBM-RGBD organisation [116], to have a more precise evaluation, the parts around foregrounds are categorised as an unknown motion due to motion blur and semitransparency that do not allow an accurate segmentation. Thus, these parts are defined as outside of evaluation. Figure 4.3 illustrates an example of ground truth in this dataset where the outside in region of interest is assigned to a pixel value of 85, background pixels 0, foreground pixels 255 and unknown motion as 170.

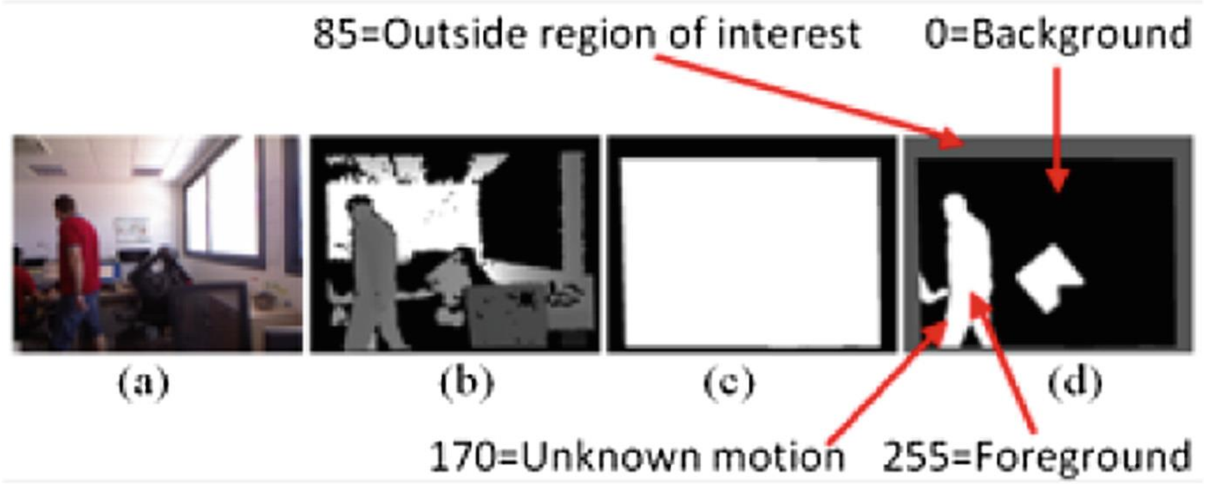


Figure 4.3. *Sequence ChairBox in* [116].
(a) colour, (b) depth images, (c) ROI, (d) ground truth

In particular, the evaluation is based on the following metrics: Recall, Specificity, False Positive Rate, False Negative Rate, Percentage of Wrong Classifications, Precision and F-Measure. This dataset has 33 sequences under seven different challenging categories in Illumination Changes, Colour Camouflage, Depth Camouflage, Intermittent Motion, Out of Sensor Range, Shadows and Bootstrapping. Table 4.1 illustrates the summary of this dataset. We should state that the proposed method has been evaluated with the entire dataset and only one set of tuning parameters have been used to produce the segmentation result for all the sequences. A detailed result of the proposed method (BSABU) including a qualitative segmentation result for each sequence is available on [129].

At last, we have tested the algorithm in the real-time application to measure the computational cost of our method. This test shows that the system can successfully run the application in real-time with approximately 12 frames per second.

Category	Sequence Name	Number of Frames	Number of Ground truth	Frame number where moving object occur	Main effected sensor	Test Objective
Bootst rapping	adl24cam0	70	4	1	Colour Depth	Sequences containing foreground objects in all their frames
	bear_front	290	15	1		
	Bootstrapping_ds	399	11	1		
	fall01cam0	160	9	1		
	fall20cam0	110	6	1		
Colour Camouflage	Cespatx_ds	429	11	133	Colour	Sequences containing foreground objects that are having similar colour to the background
	Hallway	618	18	48		
	colour Cam1	300	29	57		
	colour Cam2	360	26	61		
Depth Camouflage	DCamSeq1	600	46	1	Depth	Sequences containing foreground objects that are having a similar depth to the background
	DCamSeq2	670	52	98		
	Despatx_ds	465	12	145		
	Wall	218	80	60		
Illumination Changes	ChairBox	529	62	1	Colour	Sequences including strong and mild illumination
	Ls_ds	408	2	0		
	TimeOfDay_ds	1232	2	0		
	genSeq1	410	25	103		
Intermittent Motion	Shelves	554	134	183	Colour Depth	Sequences with scenarios known for making "ghosting" artifacts in the detected motion. Removed foreground objects or abandoned foreground objects.
	Sleeping_ds	300	9	85		
	abandoned1	250	47	1		
	abandoned2	250	72	52		
	movedBackground1	250	78	1		
	movedBackground2	250	37	1		
OutOfRange	MultiPeople1	1190	39	118	Depth	Sequences including foreground or background objects that are too far or close from the sensor
	MultiPeople2	1400	53	91		
	TopViewLab1	670	33	110		
	TopViewLab2	650	33	120		
	TopViewLab3	700	39	135		
Shadows	Shadows_ds	331	9	150	Colour Depth	Sequences that foreground objects caused creation of shadows. These are visible-light shadows in the colour frames or IR shadows in the depth frames
	fall01cam1	160	7	62		
	genSeq2	300	26	119		
	shadows1	260	28	65		
	shadows2	250	26	104		

Table 4.1. *Full details of SBM-RGBD datasets introduced in* [116].

4.4.1 Comparison to the original Method

In this section, the results achieved by the proposed method will be compared with the original algorithm. For more precise comparison, four more sequences which stationary moving object occurs in the scene from SBM-RGBD has been added to the publicly available sequences used in the previous chapter [3]. The following sequences have been used for comparison.

5. **SHSQ**: This sequence aims to measure the effect of shadows in the scene by moving a box on the floor.
6. **Genseq:** This sequence has been made to measure the overall performance of the method by having several challenging scenarios that occur in one scene. This sequence is a scene with an individual person moving a box containing strong and mild illumination changes.
7. **Colcam:** The moving object (the board) in this sequence has the same colour as the background to explore the possible errors of the method in colour camouflage.
8. **DCamSeq:** In this sequence, moving object (the hand) is moving around the cupboard and therefore, it has the same depth as the background to explore the possible errors of the method in depth camouflage.
9. **Abandoned1:** The scenarios in this video is made to test the tolerance of the method in creation of ghost phenomenon. A box will move from the stationary position to the top of the cupboard and stay stationary for some frames.
10. **Abandoned2:** This sequence is made to test the tolerance of the method for a stationary foreground object. A bag will be dropped in the scene and remain there for a while.
11. **MovedBackground1:** It aims to measure the accuracy of the method in case of background changes. A box drops from a cupboard and stays on the floor.
12. **MovedBackground2:** The video starts with a bag on the floor and will be removed from the scene. Similar to moved Background1 the goal is to measure the algorithm in background changes.

To quantitatively evaluate the accuracy of each method, the following metrics have been used to compare the outcomes. These metrics are explained in detail in the previous chapter (section 3.3).

False Positive (FP): Pixels belong to the background which are classified as foreground.

False Negative (FN): Pixels belong to the foreground which are classified as background.

Total Error (TE): The total amount of misclassified foreground and background pixels which normalised to the image size.

Similarity measure (S): This non-linear metric previously has been used in [118] and called Jaccard's index [117]. It combines the FN and FP. This explained thoroughly in the previous chapter as equation 3.3.

$$S(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{4.5}$$

Similarity measure (S_B): It only explores the misclassified pixels surrounding the foreground objects boundaries. It is calculating as S, but only covers the regions of 10 pixels around the ground-truth boundaries.

Figure 4.4 illustrates an example of qualitative comparison in a stationary moving object dataset. Where (a) is colour image, (b) depth image, (c) ground-truth, (d) original method and (e) proposed method. The sequences are dynamic and each row in the figure represents one of the last few frames of a sequence. These sequences are *abandoned1*, *moveBackground1*, *abandoned2*, *movedBackground2* and *shseq* sequences where a stationary moving object exist. Segmentation results which are closer to the ground truth (white area) show more accurate detection. The grey area shows outside of the region of interest which are not part of the evaluation. The proposed method clearly could achieve more accurate results in these stationary moving object sequences.

In the first sequence (abandoned1), the brown box moves from the floor to the top of the cabinet and stays stationary. As the foreground stays stationary, it will gradually absorb to the model in the original method. However, in the proposed method, the system significantly reduces the blind update as the item is stationary and consequently the foreground detection remains sharp and more accurate.

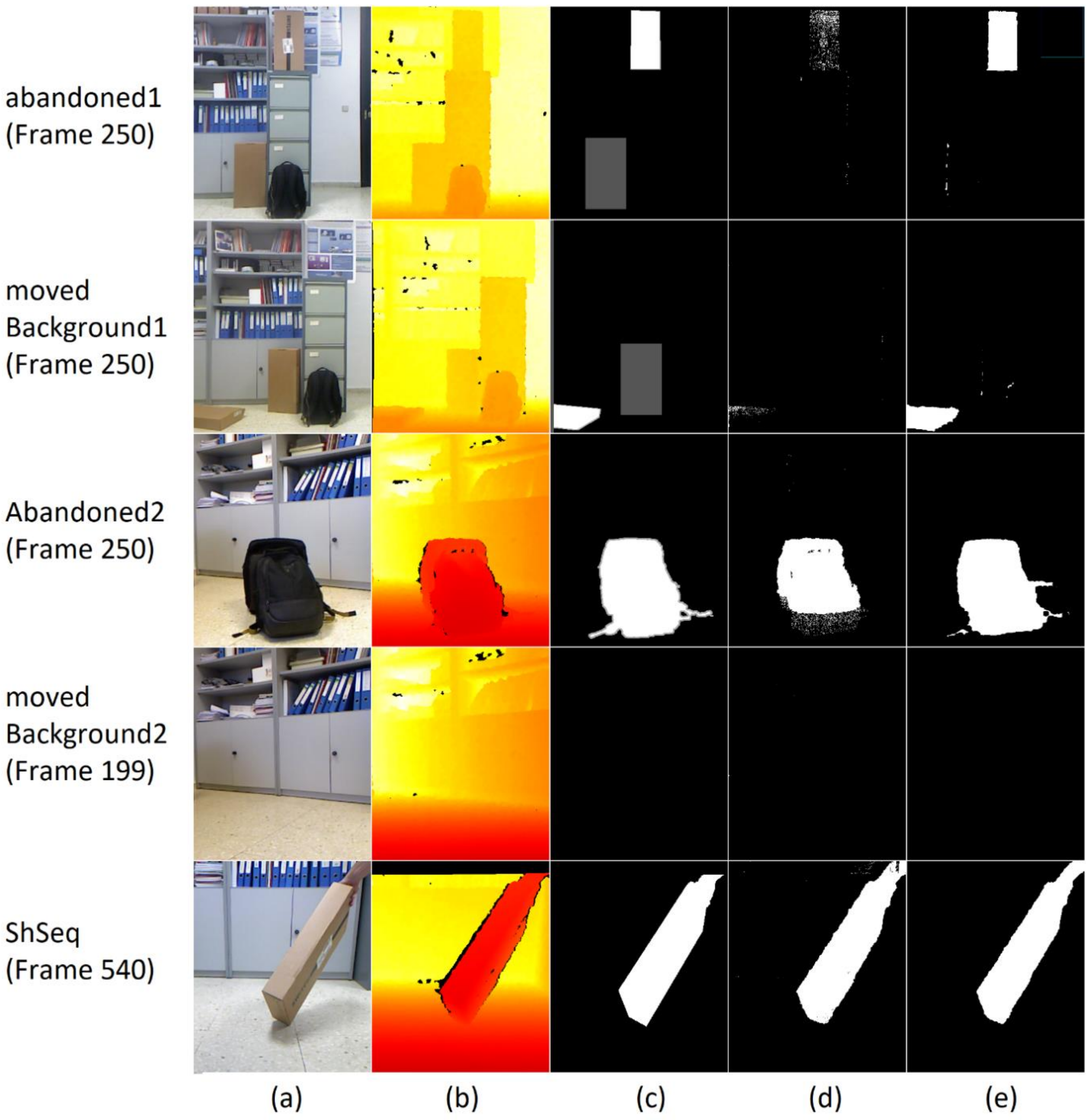


Figure 4.4. *Example of qualitative comparison in a stationary moving object dataset. The grey area shows outside of the region of interest (a) Colour image, (b) Depth image, (c) Ground-truth, (d) Original method, (e) Proposed method.*

In moved background1, the brown box drops from the top of the cabinet to the floor and stays stationary there. The foreground gradually absorbs to the model due to the blind update. However, the proposed method can keep the sharp detection of the foreground due to the reduction of frequently in the blind update.

In abandoned2 sequence, a black bag will be dropped in the scene and stay stationary there. The foreground starts absorbing to the model in the original method and the foreground will gradually disappear in segmentation mask. This starts from the bottom of the bag as it is closer to the background.

In moved background2 sequence, the bag will be removed from the scene and both methods could successfully maintain the background model.

In shseq, both methods could avoid the shadow. However, due to the advanced segmentation method, the proposed algorithm could produce a more accurate result in object boundaries, less noise and the hand is fully detected.

The results show that both methods could completely prevent ghost production in the movedBackground2 sequence which the bag has been removed from the scene to measure the tolerance of the algorithms in background changes. However, in stationary moving objects, the proposed method could achieve better results by preventing the absorption of the moving object into the background model.

Figures 4.5-7 demonstrate the accuracy of the proposed method in this chapter and the original method introduced in the previous chapter (NBMS) [3] in TE, S and S_B. The lower amount of TE and higher S and S_B shows better performance. As illustrated in the figures, the performance of both methods is very close in those sequences that the stationary object or slow-moving object doesn't exist for a while (*Genseq*, *Colcam*, *SHSQ*). However, in those sequences with a stationary moving object (abandoned1, *abandoned2*, *movedBackground1*, *movedBackground2*) the proposed method could have achieved better results in all sequences.

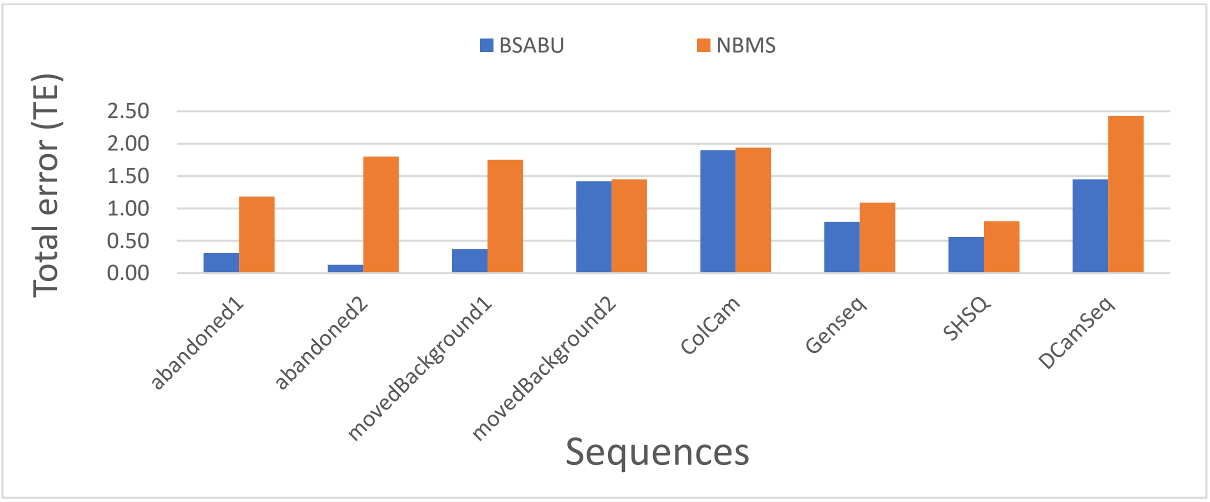


Figure 4.5. *Total error (TE). The lower amount shows better performance.*

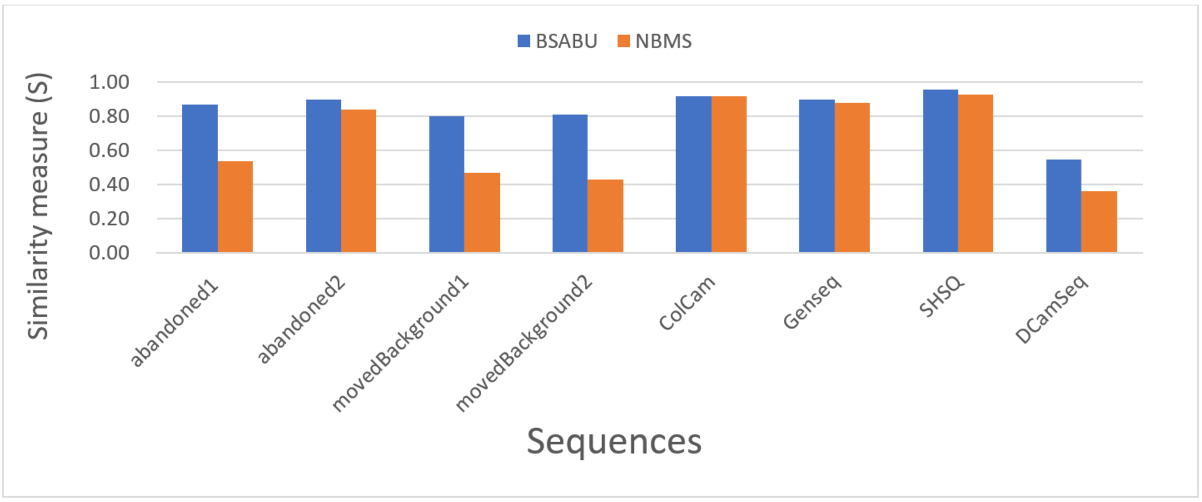


***Figure 4.6**. Similarity measure (S).*
The higher amount shows more similarity with the ground truth and therefore better performance.

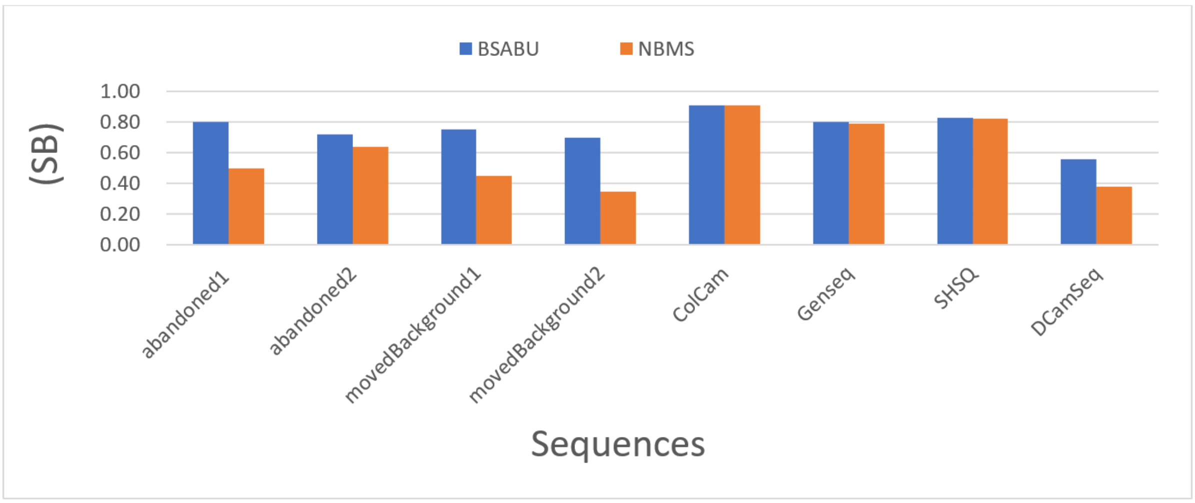


***Figure 4.7.** Similarity measure in object boundaries (S_B).*
The higher amount shows more similarity with the ground truth around the object boundaries and therefore better performance.

The proposed method in this chapter (BSABU) is also compared to the other methods in the drone datasets (Crazyflie and AR.Drone) which were introduced in chapter three (section 3.3.1). Table 4.2 illustrates the results of BSABU and other methods in Crazyflie sequences. The results show similar performance to the original method introduced in chapter 3. Also, Table 4.3 illustrates the results of BSABU and other methods in AR.Drone sequences. In this sequence, the proposed method demonstrated similar performance compared to the original method introduced in chapter 3. These results indicate the proposed method and the original method have very similar performance in scenarios which stationary moving object and bootstrapping doesn't exist.

Table 4.2. *Craziflies sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG}_{\mathrm{RGB-D}}$	0.63	0.17	51.25	13.56	0.02	0.02	0.34	0.17	0.37	0.14	4.2
$\mathrm{GSM}_{\mathrm{UB}}$	0.08	0.01	55.96	23.49	0.03	0.03	0.27	0.13	0.29	0.09	4.8
$\mathrm{GSM}_{\mathrm{UF}}$	0.12	0.27	37.59	18.25	0.09	0.03	0.27	0.11	0.34	0.08	4.4
$\mathrm{PBAS}_{\mathrm{Bin}}$	0.63	0.06	0.20	0.57	0.63	0.06	0.11	0.04	0.44	0.07	3.8
$\mathrm{VIBE}_{\mathrm{Bin}}$	1.45	0.19	19.12	10.71	1.43	0.19	0.04	0.02	0.42	0.06	5.0
MBNS	0.05	0.01	42.63	17.85	0.01	0.02	0.42	0.18	0.42	0.11	2.2
BSABU	0.04	0.02	40.73	15.05	0.04	0.03	0.40	0.20	0.41	0.14	3.0

Table 4.3. *AR.Drone sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG}_{\mathrm{RGB-D}}$	0.15	0.16	14.79	12.02	0.04	0.06	0.79	0.13	0.80	0.11	3.2
$\mathrm{GSM}_{\mathrm{UB}}$	0.49	0.29	74.97	11.66	0.01	0.01	0.25	0.12	0.26	0.12	5.4
$\mathrm{GSM}_{\mathrm{UF}}$	0.31	0.26	9.59	6.52	0.25	0.21	0.69	0.11	0.74	0.08	4.0
$\mathrm{PBAS}_{\mathrm{Bin}}$	1.25	0.33	0.25	0.80	1.25	0.33	0.33	0.09	0.66	0.10	5.2
$\mathrm{VIBE}_{\mathrm{Bin}}$	1.02	0.26	2.73	3.41	1.18	0.25	0.33	0.09	0.74	0.09	4.6
MBNS	0.13	0.11	11.84	6.02	0.05	0.07	0.82	0.11	0.83	0.08	2.2
BSABU	0.15	0.17	14.03	7.80	0.04	0.08	0.82	0.15	0.82	0.15	2.4

4.4.2 SBM-RGBD Dataset

In this section, we have evaluated the proposed method with the entire SBM-RGBD datasets and compared our results with other state-of-the-art methods in SBM-RGBD challenge. These methods are RGBD-SOBS [53], RGB-SOBS [95], SRPCA [124], SCAD [52], cwisardH+[130], AvgM-D, Kim and MFCN [34]. Table 4.4 demonstrates the average results of the entire dataset. A detailed result of the proposed method (BSABU) for each sequence is available on [129]. The SBM-RGBD dataset comes with a MATLAB code to measure the accuracy of the moving object detection across various challenges. Formally TP, TN, FP, and FN show the total number of True Positive, True Negative, False Positive and False Negative for each video. The seven metrics used in this challenge for evaluating the results of moving object detection are:

1. *Recall* $Rec = \frac{TP}{TP+FN}$ (4.6)
2. *Specificity* $Sp = \frac{TN}{TN+FP}$ (4.7)
3. *False Positive Rate* $FPR = \frac{FP}{FP+TN}$ (4.8)
4. *False Negative Rate* $FNR = \frac{FN}{TP+FN}$ (4.9)
5. *Percentage of Wrong Classifications* $PWC = 100 \times \frac{FN + FP}{TP + FN + FP + TN}$ (4.10)
6. *Precision* $Prec = \frac{TP}{TP+FP}$ (4.11)
7. *F-Measure* $F = \frac{2 \times Prec \times Rec}{Prec + Rec}$ (4.12)

***Table 4.4**. Average results of all available methods in SBM-RGBD datasets.*

Name	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure
RGBD-SOBS	0.8391	0.9958	0.0042	0.0895	1.0828	0.8796	0.8557
RGB-SOBS	0.7707	0.9708	0.0292	0.1578	5.4010	0.7247	0.7068
SRPCA	0.7787	0.9738	0.0262	0.1499	3.2243	0.7477	0.7474
SCAD	0.8847	0.9932	0.0068	0.0439	0.9088	0.8698	0.8757
cwisardH+	0.7622	0.9817	0.0183	0.1664	2.8806	0.7556	0.7470
AvgM-D	0.7065	0.9869	0.0131	0.2221	2.8848	0.7498	0.7157
Kim	0.8493	0.9947	0.0053	0.0793	1.0292	0.8764	0.8606
MFCN	0.9186	0.9984	0.0016	0.0100	0.2373	0.9103	0.9143
Proposed-Method(BSABU)	0.8211	0.9955	0.0045	0.1075	1.0854	0.8795	0.8477

In addition, for more precise comparison the evaluation method in [62] is used to calculate the average ranking of a method (RM) which combines the performance of each method across different metrics in each sequence for whole datasets illustrated in figure 4.9.

Formally, let us denote the $N_m$as number of metrics in sequence sq and ranking of the ith method for the metric m as $rank_i(m, sq)$. The average ranking of the method i is given as:

$$RM_i = \frac{1}{N_m} \sum_{i=1}^{m} rank_i(m, sq) \tag{4.13}$$

Figure 4.8 illustrates qualitative comparison in 7 different challenging sequences in SBM-RGBD datasets with more than 15000 frames. These sequences are; bear_front (Bootstrapping), colorCam1 (Colour Camouflage), Wall (DepthCamouflage), ChairBox (Illumination Changes), TopViewLab2 (Out of Range), movedBackground2 (IntermittentMotion) and shadows2 (Shadows). All methods which participate in SBM-RGBD challenge and report their results are listed in this comparison. These methods are; (a) Colour frame, (b) Colour coded depth frame, (c) ground-truth, (d) RGBD-SOBS, (e) RGB-SOBS, (f) SRPCA, (g) SCAD, (h) cwisardH+, (i) AvgM-D , (j) Kim, (k) MFCN, (l) Proposed-Method (BSABU). The ground truth (c) is also available with this dataset and results which are more similar to the ground truth (c) are better. Figure 4.8 shows some methods are not consistent which leads them to achieve poor detection result in some sequences and accurate on others. For example, RGBD-SOBS (d) only detect some part of the foreground in Bootstrapping and intermittent motion sequences, RGB-SOBS (e) method totally failed to detect the foreground in colour camouflage and intermittent motion sequences. SRPCA (f) method incorrectly detected a large number of background pixels as a foreground in bootstrapping, colour camouflage, depth camouflage, and out of range sequences. cwisardH+ (h) method also failed in bootstrapping, depth camouflage, intermittent motion sequences. AvgM-D (i) method totally failed to detect the foreground in illumination changes and out of range sequences. However, four methods (SCAD, Kim, MFCN, Proposed-Method (BSABU)) could show more consistent detection in all sequences. SCAD (g) and Kim (j) also couldn’t detect the entire foreground in the bootstrapping and illumination changes sequences. Therefore, according to the figure of 4.8, the proposed method and MFCN could achieve the most reliable and consistent results among all these sequences.

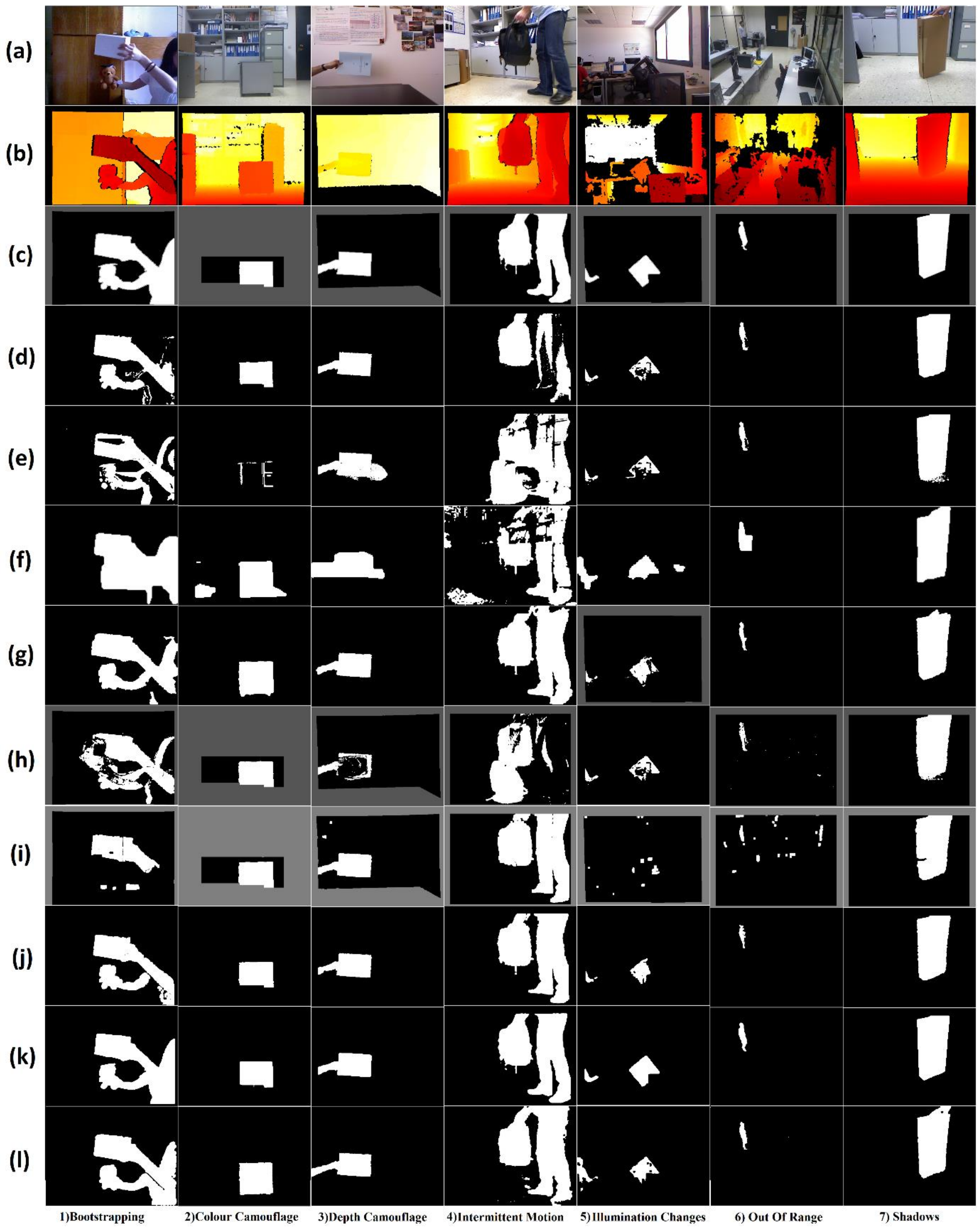
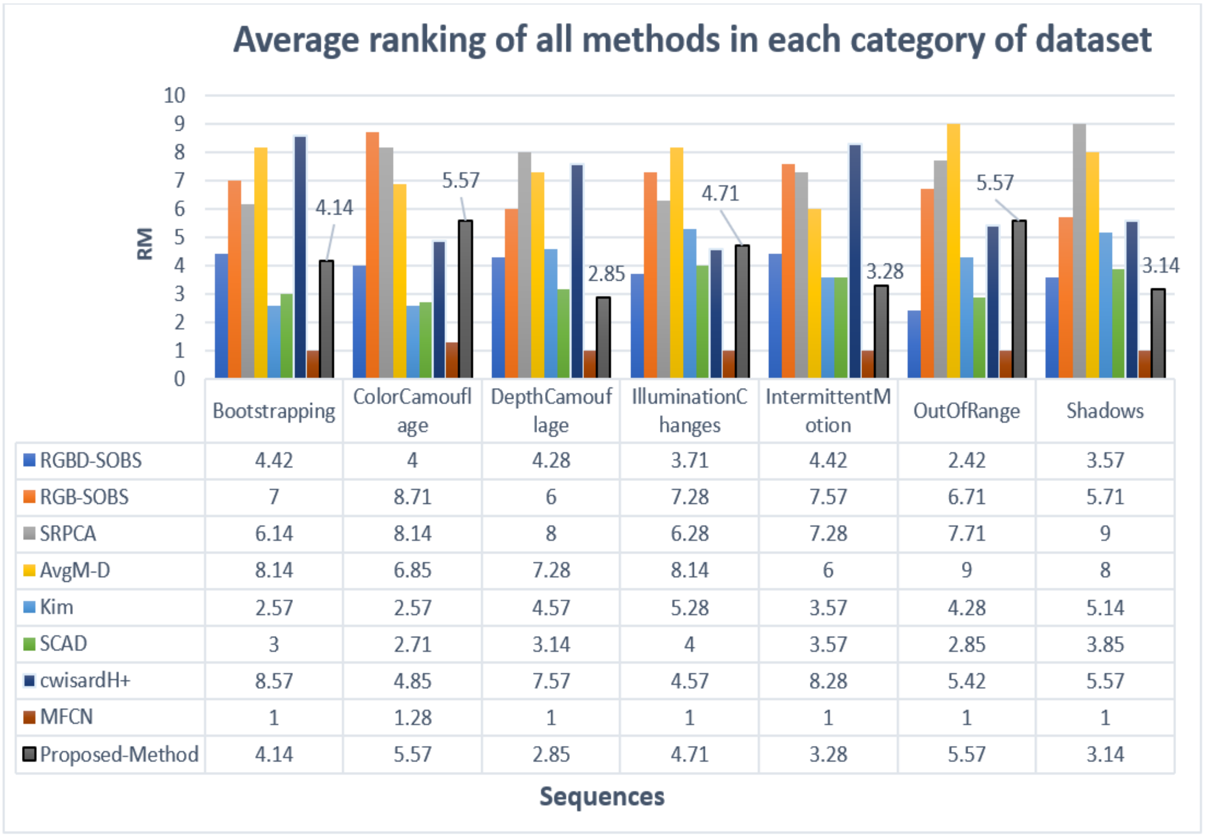


***Figure 4.8**. Qualitative comparison in 7 challenging scenarios.*
(a) Colour frame, (b) Colour coded depth frame, (c) Ground-truth, (d) RGBD-SOBS, (e) RGB-SOBS, (f) SRPCA, (g) SCAD, (h) cwisardH+, (i) AvgM-D, (j) Kim, (k) MFCN, (l) Proposed-Method (BSABU)

In order to have more accurate detection comparison of mentioned methods, figure 4.9 illustrates the quantitative comparison of mentioned methods in all 33 sequences of SBM-RGBD datasets. the MFCN method could detect the moving objects accurately in almost all categories and achieve the best results in all average results. However, this method requires many hours of training process for this dataset which is an extensive process compared to our method which requires only a few seconds to create the model. After MFCN, the proposed method could achieve the best results in *DepthCam*, *Intermittent motion*, and *shadows* sequences. Accurate detection in *DepthCam* sequences shows the method could manage the weakness of the depth sensors (absence of depth value, similar depth and sensor error in the depth value) by using colour frames. On the other hand, it also managed the appearance of shadows by using L*a*b* Colour and shadow detection method. The adaptive blind update significantly aided the proposed method to manage intermittent motion sequences which normally is based on the appearance of a new object or removing an object from the background.



	Bootstrapping	ColorCamouflage	DepthCamouflage	IlluminationChanges	IntermittentMotion	OutOfRange	Shadows
🟦 RGBD-SOBS	4.42	4	4.28	3.71	4.42	2.42	3.57
🟧 RGB-SOBS	7	8.71	6	7.28	7.57	6.71	5.71
⬜ SRPCA	6.14	8.14	8	6.28	7.28	7.71	9
🟨 AvgM-D	8.14	6.85	7.28	8.14	6	9	8
🟦 Kim	2.57	2.57	4.57	5.28	3.57	4.28	5.14
🟩 SCAD	3	2.71	3.14	4	3.57	2.85	3.85
🟦 cwisardH+	8.57	4.85	7.57	4.57	8.28	5.42	5.57
🟫 MFCN	1	1.28	1	1	1	1	1
⬛ Proposed-Method	4.14	5.57	2.85	4.71	3.28	5.57	3.14

Figure 4.9. *Average ranking of all existing methods in each category of the dataset. A lower amount shows better performance in the category.*

The weakest detection of the proposed method and most of the other methods belong to *Bootstrapping* sequences. The reason for this is that the moving object is included in the background from the beginning and consequently, it will be added to the background model. Therefore, the moving object will be assumed as a background. In some bootstrapping sequences, the ground truth starts from sequence 1 which is impossible for our method to detect the moving object (person) as the background model has not yet been created. Therefore, we have added histograms of oriented gradient [131] to help the detection of people at the beginning of each sequence.

On other sequences, the results are acceptable compared to the other methods which shows the stability of the algorithm in all sequences. In other words, it is proven that it is able to detect most moving objects in these challenging scenarios and does not completely fail in any scenario.

4.4.3 Real-time Experiment

The proposed system has been tested in a live application with the processing time of approximately 12 fps (frame per second) or 80 ms for a 640x480 video. The computational cost of the algorithm is calculated as the mean rate of the processing time of the algorithm. The test was performed on a computer with an Intel (R) Core (TM) i7-6700HQ CPU @2.6 GHz and 8 GB RAM along with Microsoft Kinect v2 sensor.

It is worth mentioning that the proposed algorithm used during these tests has been implemented in C++ and OpenCV library [119] without any specific code optimisation as the aim of this experiment is to show that the proposed algorithm can successfully run at real-time rates and therefore no effort has been made to optimise the code/set-up.

4.5 Summary

In this chapter, a moving object detection using adaptive blind updating is proposed. The main contribution of this chapter is adding adaptive blind update method, more complex segmentation, proposed bootstrapping detection, proposed shadow detection method based on L*a*b* colour space and complete evaluation of this method.

The proposed method in this chapter is able to deal and adapt to the complex scenarios such as environmental changes, illumination changes, bootstrapping, shadow, as well as detecting the fast, slow and stationary objects while reducing the creation of ghost phenomenon.

By tracking the moving object, the frequency of blind update will be changed according to the speed of the moving object. This strategy will significantly help the scene model to adapt to moving objects with different speed. A simple tracking method has been used in this work to reduce computational costs. In addition, the proposed shadow detection method based on L*a*b* colour space helps the system to increase detection accuracy in shadow and depth camouflage scenarios. Also, this method is able to detect bootstrapping sequences and then use an edge detection technique to improve the detection accuracy in bootstrapping scenarios.

Experimental results show that the proposed method can significantly improve the accuracy of the algorithm when stationary objects exist. On the other hand, on fast-moving sequences, the algorithm achieved slightly improved or equal results to the original method. The best performance of this method is only achievable when one moving object exists.

In general, the main advantages of the proposed method are to improve segmentation accuracy in stationary moving objects, bootstrapping, shadow and depth camouflage scenarios. Overall the proposed method proved it is more consistent in all situations. The main disadvantage of the proposed method is that the system has a higher computational cost compared to the original method. However, this method can still be applied in live applications.

Although this method performed well compared to the other existing methods, the regular update in this method is still using a basic updating mechanism. Methods introduced so far in this thesis are storing the pixels in the model and the pixels of the new frames will be compared with the models in order to identify them as background or foreground. These models are requiring a constant update to be able to adapt to the changes occurring in the background such as illumination changes. The traditional optimization approaches such as linear programming which has been implemented in the proposed methods are often cannot discover the optimal solution for problems with various peaks in a short amount of time. Therefore, in the next chapter, we are going to investigate the use of artificial intelligence in the update method to improve the weakness of the proposed methods. In particular, we are going to use AI to increase accuracy and efficiency by reducing the computation costs of nonparametric background subtraction methods.

Chapter 5
Background modelling and segmentation using a Genetic Algorithm and Hill Climbing

5.1 Introduction

Nonparametric background modelling and segmentation which are introduced in the previous chapters are storing the pixels in the model and then the new frames will be compared with the models in order to identify as background or foreground. These models are requiring a constant update to be able to adapt to the changes occurring in the background such as illumination changes.

The traditional optimization approaches such as linear programming and dynamic programming often cannot discover the optimal solution for problems with various peaks. In these cases, related researchers are using Evolutionary Computation (EC) methods in image segmentation. These approaches combine the classical segmentation methods with the EC techniques such as Genetic Algorithm [132][133]. EC is an umbrella term that combines Genetic Programming (GP), Genetic Algorithms (GA) and evolution strategies [134]. Typically, EC is responsible for optimising parameters and maximizing or minimizing objective functions in these hybrid systems [135]. These methods help to solve complex problems by discovering the optimal solution through simulating a natural system of biological evolution via using the processes of selection, mutation, and reproduction. This allows individuals in a population to compete to reach the same goal. Among these methods typically GA is a universal optimisation method in large space to search and find the solutions. GA has been introduced based on Darwin's theory of the evolution and survival of the fittest. This theory is discussed in detail in [136].

EAs methods have been effectively applied to an extensive range of problems such as parameters estimation [137], classification [138], optimization [139], pose estimation [140], motion estimation [141], and adaptation to environmental variations in computer vision applications [142].

In this chapter, a new update method based on Hill Climbing (HC) and Genetic Algorithm has been proposed and added to the object detection algorithm introduced in chapter three [3] based on colour and depth frames. This method generates an individual model for colour frames and additional model for depth frames. These models store the previously observed pixels. Then, the system will label each new pixel as foreground or background by comparing them to these models. Due to the number of stored samples, the size of involved search space is large and consequently update process is usually an expensive task in terms of computation. For example, in 800 × 600 image size with 20 samples the search size would be 9,600,000. Therefore, in live applications, a powerful search method is required to efficiently find the optimal background model instead of comparing all pixels one by one in the loop. The optimal background model can be calculated by a fitness function based on the distance of depth data. This will be discussed in more detail in section 5.3. First Hill-Climbing algorithm has been tested to check if it can efficiently find the optimal model and then a separate Genetic Algorithm has been investigated to search for the optimal background model. Using the Hill-Climbing algorithm in the field of computer vision is not completely new and has been previously applied differently in some other research [143][144]. Results show that HC is more accurate in the overall detection rate, around object boundaries and also is able to tolerate more illumination changes compared to the GA algorithm. In addition, the proposed method based on HC is more efficient than the original and GA methods in terms of computational cost. It is worth mentioning that our proposed method in this chapter belongs to a draft paper[145] and some contents of this chapter are from this journal paper.

In the remainder of this chapter, in section 5.2 related work will be discussed, section 5.3 the moving object detection algorithm, background modelling and update process proposed in this chapter will be explained in detail. In section 5.4, the result of the proposed method will be compared to the other state-of-the-art methods. Section 5.5 will be a real-time evaluation. Section 5.6 will be comparisons of all proposed methods in this thesis and section 5.7 is a summary of the entire chapter.

5.2 Related Work

In chapter three a novel sample-based method using colour and depth data was proposed [3]. This approach creates separate models for colour and depth information. This model used a fast initialisation process by storing the first N number of colour and depth frames to produce the models according to the background of the scene. The ADO removal strategy reduced the noises in depth frames by filling the unknown pixel values before storing to the depth model. In order to do this, the system exchanged all the unknown values with the estimated values. This method is based on the assumption that neighbouring depth pixels are most likely to have a similar value. This assumption is used to remove the unknown pixels by replacing them with one of the nearest pixel values. The advantage of this effective method is being fast and simple which can significantly decrease the number of errors by having more accurate depth pixel values in the depth model. To produce the final foreground mask image, this method merges the results from colour and depth models. In order to do this in each location the value of the colour pixel will be compared to the colour model to check if the pixel value is close to some of the sample values in the model or not.

Depth pixels were similarly compared to the depth model to find out if the pixel was in the same distance or was closer to the camera. On the other hand, if the value of the pixel were unknown (ADO), the system would only rely on the colour model outcome in that pixel location. In particular, those pixels which have a close or greater distance to some of the depth sample values in the depth model would be considered as a background. Other pixels were compared with the depth model again while the threshold has been increased. Those pixels which failed this condition were identified as a foreground. All the other pixels were compared with the colour model. Accordingly, if the remaining pixels could find some similarity with the colour model, they were identified as a background, otherwise, they were classified as a foreground.

Figure 5.1 demonstrates a flow chart diagram of the segmentation method. Those pixels which were classified as the background would be compared to the model. If they have better values (higher distance), they were then be exchanged by the lowest value in the model. The amount of tolerance the system can accept is fixed in this method. However, a closer point to the camera was more sensitive compared to the pixels located further from the camera [121]. In addition to the regular update, after some frames, the system would replace the new pixel with one of the samples in both models regardless of segmentation result. One of the main drawbacks in this system was that the update mechanism was based on a simple logical method which compared all pixels in the model after every new frame. This makes this method inefficient to deal with the noise of the sensors because the noise could have high value and therefore stay in the model for a long period of time. For this simple update mechanism, the system was unable to store reliable pixels to accurately detect the moving object in depth camouflage scenarios. Also, this update mechanism is comparing all pixels in the model after every new frame in the loop which is significantly inefficient in terms of computational cost.

Therefore, the proposed method in this chapter is instead to use advanced optimisation method such as GA and Hill Climbing to improve the accuracy and efficiency of this segmentation method.

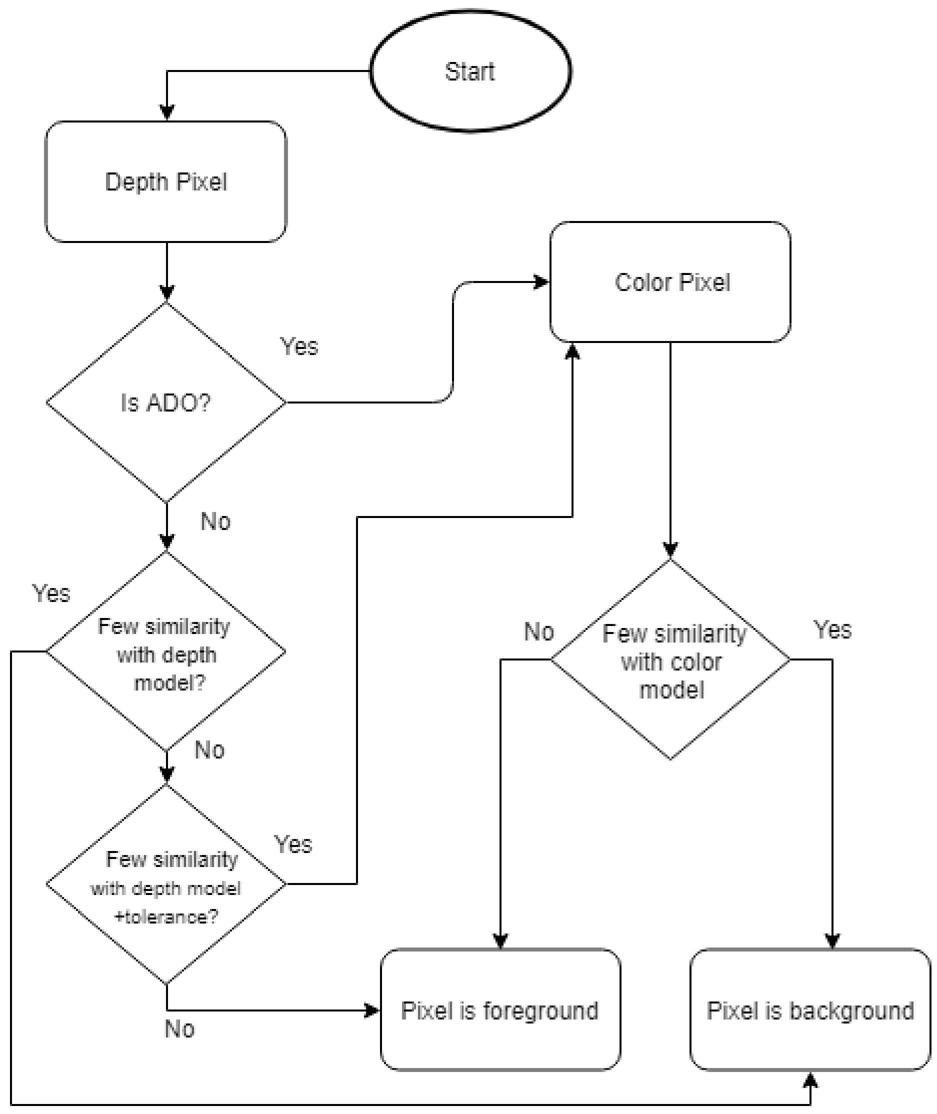

Start
Depth Pixel
Is ADO?
Yes
No
Color Pixel
Yes
Few similarity with depth model?
No
No
Few similarity with color model
Yes
Yes
Few similarity with depth model +tolerance?
No
Pixel is foreground
Pixel is background

Figure 5.1. *Flow chart of the proposed classification method introduced in chapter 3 [3].*

5.3 Proposed Method

The method proposed in this section uses a nonparametric background modelling and segmentation which has been introduced in chapter three [3]. The main contribution of this chapter is adding a new update method to efficiently find the optimal background model and adapt the model to the changes in the background such as illumination changes. The background model update has always raised a question about which sample we should store in the model and for how long we can use them [8]. Classical approaches typically discard the old pixels in the model and exchange them with the new pixel after a few frames or seconds. These approaches exchange all the old values regardless of whether they are valid samples or not. Recently, The ViBe [8] algorithm introduced random substitution with the model which might allow the valid sample pixel to stay longer in the model. In [3], a new model update method has been introduced which replaces the pixel by the smallest distance in the model with the pixel from the new frame in the same location. This helps to maintain an updated model with the new changes as well as keeping the valid samples in the models. However, the new samples are not necessarily always the best samples of the scene. On the other hand, traditional methods update the model pixel by pixel, instead, in this chapter, the model will be updated by each sample frame in the model in order to prevent permanent wrong background pixel allocation as it could be added to the model by error or noise of the sensor. In addition, the method is more efficient as it doesn’t need to compare the new pixel with all other pixels after every frame. In order to find the optimal background model, we need a powerful search method to search and update the background model. To do this, we have implemented two different type of optimization methods (Hill-climbing and Genetic Algorithm) to discover which one is more suitable for this problem. One (The Genetic Algorithm) is a global search strategy and the other one (Hill-Climbing) is a local search optimisation algorithm. First, we have implemented our update method based on Random Mutation Hill Climbing (RMHC) algorithm which is a type of heuristic search and it increases efficiency to find a solution in a reasonable amount of time. Typically, heuristic search methods do not necessarily always find an optimal solution but alternatively may discover an acceptable solution in a short period of time which is an important factor in live applications. Hill climbing is also referred to as a greedy local search since it goes ahead without thinking where to go next [146]. However, greedy algorithms in some cases are able to obtain decent results [147]. The goal of using Hill Climbing is to find out if this method is capable of selecting the true background pixels from previously observed pixels and store them in the background model in a shorter period of time. As Hill-Climbing algorithm sometimes gets stuck in local optima, separately this method is also implemented with Genetic Algorithm to assess if this update strategy can reach to the global optima and outperform the Hill Climbing method. We called these methods nonparametric background modelling using GA (NBM-GA) and nonparametric background modelling using Random Mutation Hill Climbing (NBM-HC). In the remaining of this section, in 2.1 the NBM-GA will be explained and in 2.2 NBM-HC will be discussed in more detail.

5.3.1 Background Modelling using a Genetic Algorithm (NBM-GA)

The proposed background modelling and segmentation method involve several steps to be able to successfully create the background model and then detect the moving objects by comparing the new frames with this model in the live application. Figure 5.2 illustrates the flowchart of the proposed Genetic Algorithm.

To implement this method, we have defined each gene as a pixel value and the number of genes in each chromosome is equal to the number of pixels in an image frame. The first N number of colour and depth frames will be stored as a vector of pixel value (integer) to create the background models which generate an initial population. This step is called “Generate Initial Population”. Depth data is typically noisy and has restrictions in measurement for some materials which are normally referred to as “Absent Depth Observations (ADO)”[7] or “Holes”[91]. Therefore, to remove all unknown values, the “ADO removal” step will be applied to all depth frames before storing them in the model. Then, the “Background Model/ Population” step is used to complete the background model and the algorithm starts in the main loop. In the “Bg/Fg Segmentation” step, every pixel from the new frame will be compared with the models to be labelled as background or foreground. This step follows the MBNS method proposed in chapter three [3].

As soon as the segmentation mask has been produced, the system starts to validate the results by comparing it to the previous result. Experimentally, we have found that around 1% of depth frames are extremely noisy which could lead to a segmentation result with a high amount of false positive. Therefore, the foreground mask will be compared with the previous one and if the number of foreground pixels significantly increased, that segmentation is invalid, and the system will skip the update process. This stage is called “Valid Segmentation”. This can reduce some noise from the detection mask. After validating the foreground/background segmentation, updating the models will commence. In order to update the models, the fitness of individual chromosome needs to be calculated and a parent required to be selected from the population to mate with the new chromosome (the new frame). In the remaining of this section, some important steps of this algorithm will be discussed in more detail.

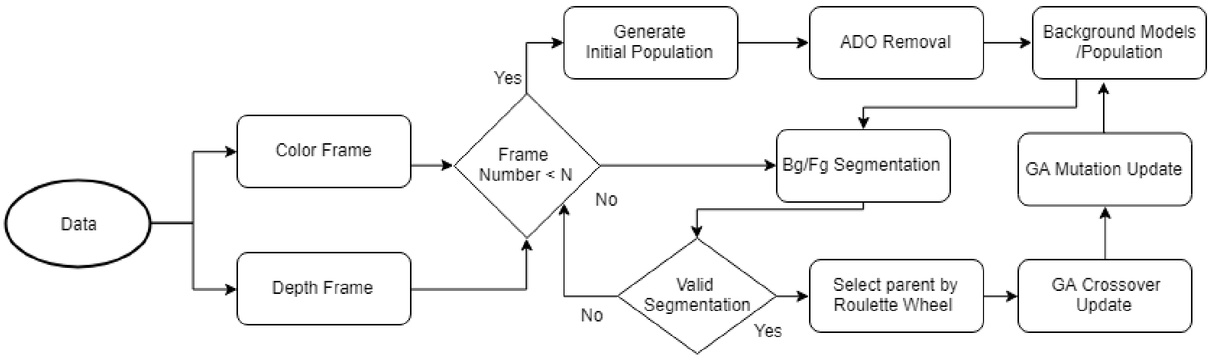


Figure 5.2. *The flow chart of proposed background modelling and object detection method based on Genetic Algorithm.*

5.3.1.1 Fitness Function

Each image frame is represented as a chromosome and encoded as a vector of pixel value (integer). A chromosome is a single string of genes which means that the individual pixel value in this method represents a gene. The GA needs to compare the fitness of each chromosome and rate how good a chromosome is to perform reproduction. Our fitness function calculates the total distance of the background pixels from the maximum distance for the individual chromosome in the depth model. This will allow the system to rank the depth samples. This rule could be incorrect in some scenarios such as a sensor error. In other words, those pixels belong to the model do not exist in the scene. However, GA crossover and mutation update stage is added to enable the system to cope with these errors from time to time.

Formally, if we denote a M_{RGB-d} as a model to store n number of RGB images and M_D to store n number of depth images (chromosomes),f_i and x_i shows an index of *i* in a background image samples.

$$M_{RGB-d}=\{f_1, f_2, f_3, \ldots, f_n\} \tag{5.1}$$

let us denote the following function F to measure the fitness of chromosome *f*:

$$F(f)= \sum_{i=0}^{n}(d_{Max} - d_i) \tag{5.3}$$

Where d_{Max} is the maximum distances the sensor can capture, $d_i$is the distance of the ith background pixel in image f and n is the total number of stored background pixel in the image f. The rationale behind the calculation of fitness function is that depth pixels further from the camera have more possibilities to be the background pixels as explained in figure 3.7. Once the fitness function is calculated, a Roulette Wheel will be used to choose the sample to receive an update.

5.3.1.2 Roulette Wheel Selection

Pixels with the lowest depth value have more chance to be picked in the roulette wheel. The chance of each chromosome to be selected is proportional to its fitness compared to the total fitness of the other chromosomes.

Typically, Depth pixels are noisy and sometimes the largest pixel value doesn't mean the best pixel for the model. Therefore, the proposed method in this chapter is using GA and Roulette Wheel which is a powerful tool for performing a search and optimization. GA is responsible for searching and finding the optimal background models over the time instead of other approaches. The main advantage of using a Roulette Wheel is that the updated model is able to avoid being trapped in a local optimal solution as depth data is noisy and stochastic.

This will begin by choosing a random number from 0 to $\sum F(fitness)$. Then reduce each fitness from the random number until it reaches 0. The last fitness which the random number became 0 will be chosen to receive an update. This allows all samples (chromosomes) in the model to have a chance to receive an update. However, chromosomes with better fitness will have more probability to receive an update. This allows the update procedure to not get trapped in the local optimal which could lead to constant misclassification and reduce some noises in the results such as sensor errors or wrong background detection. Formally the probability of individual chromosome i is calculated as follows:

$$P(i) = \frac{fitness(i)}{\sum_{j=1}^{n} fitness\ (j)} \tag{5.2}$$

Where *P(i)* is the probability of solution *i*, *fitness(i)* is the fitness of solution i and n is the number of solutions (chromosomes). A uniform crossover will be applied, once the parents are selected (the first parent will be the new frame and the other parent for mating is selected by the Roulette Wheel from the model).

5.3.1.3 Uniform Crossover

A uniform crossover will be applied by flipping a coin for each chromosome (50% chance each) to decide the genes from which parent will be included in the offspring. A random crossover increases time gaps and allows the background models to adapt to the changes in the background. A random number will be chosen from 0 and 1 for all the genes in the chromosome. For each gene, if the random number is 0 and the identified background pixel is better, it will be included in the off-spring, otherwise, the gene from the other parent will be in the off-spring. Then, in "GA mutation update" stage, a mutation will be applied.

5.3.1.4 GA Mutation Update

Mutation update will help to update the pixels from other observed pixel values in the pool instead of the genes from the two parents. This update allows all the pixels in the pool have a chance without considering them as either part of the foreground or background. This update has been added into our algorithm to enable the system to cope with the changes in the background during the time by exchanging the foreground as well as the background pixels. To achieve this, some genes from random chromosomes will be included in the same location in the offspring. Naturally, the mutation is a very slow phenomenon and used for a random exploration, but it should not lead the GA to a pure random search as it would be the case if many genes suddenly mutated. Therefore, low mutation rate ensures that not too many mutations are considered at once. Consequently, the mutation probability (MP) in this work considered as $\frac{1}{N\times 2}$. The GA pseudocode proposed in this chapter is as follows:

Algorithm 5.1: The proposed Genetic Algorithm Pseudo Code

Input:

μ	Population size (μ = 20)
NG	Number of Generations (30 generations per second)
CP	Crossover Probability (50% each parent)
MP	Mutation Probability (2.5%)
n	The number of integers (genes) making up each Chromosome

```
1: Initialize:
       Generate initial population μ with chromosomes of length n from beginning
   frames

2: while (not end of sequence)
3: for i = 0 to NG-1
4:         Let compute the fitness of all chromosomes
5:         By Roulette Wheel select a chromosome X. (The chance of a chromosome
           surviving is based on proportional of its fitness to the total of the others)
6:         Crossover background pixels in X and the new chromosomes with a chance of
           CP per chromosome to create offspring X′ .
```

```
 7:      Mutate X′ with random chromosomes from the pool and a chance of MP per
         gene.
 8:      Replace X′ with the X.
 9:      end for
10:      end while
Output:  Population of best background models
```

5.3.2 Background Modelling using a Random Mutation Hill Climbing (NBM-HC)

The Random Mutation Hill Climbing method (RMHC) is an offshoot of Hill Climbing algorithm. The goals of RMHC is to find a point in the search space that maximises some objective function (fitness). The movement starts off at some random area in the search space. It then looks randomly at its close neighbours until it finds a place with better fitness. This move continues searching for progress from this new point until the final goal is achieved. The Random Mutation Hill Climbing algorithm used in this chapter due to the nature of the problem we are going to solve. The depth pixel produced by RGB-D camera is not always accurate and it produces some noisy pixels. Thus, the segmentation algorithm in this method needs at least cardinality denoted by $\#_{\mathrm{Min}}$ similar samples in the model in order to identify the pixel as background. Therefore, even finding the best solution with Hill climbing will not help to segment the pixels, as only one sample in the model can be created from the noise/fault of the sensor. Consequently, we need to constantly update the entire model. The Random Mutation Hill Climbing method in this chapter is as follows:

NBM-HC method has some advantages over NBM-GA such as being simpler to implement, faster and more efficient in terms of computational costs. However, Hill climbing algorithms may get stuck in the local optima. In the next section, both methods will be evaluated and the results are discussed in detail.

Algorithm 5.2: The proposed Random Mutation Hill Climbing Pseudo Code

```
 1: Initialize:
    Create the $N$ initial background model solutions with the beginning frames.
 2: Choose a random point in the search space (N) called $S$ and calculate the fitness of this
    point named $F$
 3: for i = 0   to iteration -1
 4:             Then choose another random point close to $S$ called $S'$ with the fitness of $F'$
 5:             Compare the $F'$ and $F$
 6:             if ($F > F'$)
 7:             continue the search
 8:             end if
 9:             if ($F < F'$)
10:             move from $S$ to $S'$
11:             Randomly exchange some background pixels in $S$ with the better pixels (higher
                distance) in the new frame
12:             end if
13:  end for
Output:         Best solutions in Background Model
```

5.4 Results

In this section, the results obtained from the proposed methods (NBM-GA and NBM-HC) will be compared with other object detection algorithms using RGB-D dataset. As both NBM-GA and NBM-HC are based on random numbers, we ran the algorithm for twenty times and the average results has been used in this section. Additionally, the standard deviation of these results also added to the table 1-4. The methods compared in this chapter are CL_{W} [48] , $\mathrm{GSM}_{\mathrm{UF}}$ and $\mathrm{GSM}_{\mathrm{UB}}$ [7], $\mathrm{PBAS}_{\mathrm{bin}}$[14], $\mathrm{Vib}_{\mathrm{bin}}$[51] and MBNS [3]. The original PBAS algorithm is based on colour frames only. In this chapter, it has been extended to use colour and depth (RGB-D) frames similar to [51], by merging the outcome of colour and depth binary mask using a logical "OR" (non-exclusive). We refer to this technique as $\mathrm{PBAS}_{\mathrm{bin}}$. It is worth mentioning that all output for the proposed method is evaluated without any morphological filtering to help us measure the accuracy of the proposed method. Undoubtedly, post filtering functions will reduce the noises and the results will be improved.

To compare and rank the performance of the algorithms, the following metrics are applied to measure the accuracy of the results. False Positive (FP), False Negative (FN), Total Error (TE) , Similarity measure (S), $S(A,B)$, similarity measure (S_B), Average ranking (RM), Average ranking among all the sequence. Please note these metrics are explained in detail in chapter 3 (section 3.3).

The proposed methods in this chapter (NBM-GA and NBM-HC) first evaluated in the drone datasets (Crazyflie and AR.Drone) were introduced in chapter three (section 3.3.1). Table 5.1 illustrates the results of the proposed methods and other methods in Crazyflie sequences. NBM-HC achieved the lowest RM which shows the best performance compared to the other methods in this challenging sequence. However, still the false negative is very high, S and S_B are low which shows despite the improvements, it is possible to enhance the detection accuracy even further in this sequence.

Table 5.1. *Craziflies sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG_{RGB-D}}$	0.63	0.17	51.25	13.56	0.02	0.02	0.34	0.17	0.37	0.14	6.2
$\mathrm{GSM_{UB}}$	0.08	0.01	55.96	23.49	0.03	0.03	0.27	0.13	0.29	0.09	6.8
$\mathrm{GSM_{UF}}$	0.12	0.27	37.59	18.25	0.09	0.03	0.27	0.11	0.34	0.08	6.0
$\mathrm{PBAS_{Bin}}$	0.63	0.06	0.20	0.57	0.63	0.06	0.11	0.04	0.44	0.07	5.0
$\mathrm{VIBE_{Bin}}$	1.45	0.19	19.12	10.71	1.43	0.19	0.04	0.02	0.42	0.06	6.2
MBNS	0.05	0.01	42.63	17.85	0.01	0.02	0.42	0.18	0.42	0.11	2.8
BSABU	0.04	0.02	40.73	15.05	0.04	0.03	0.40	0.20	0.41	0.14	4.0
NBM-GA	0.05	0.01	42.05	16.62	0.01	0.02	0.41	0.20	0.41	0.13	3.6
NBM-HC	0.04	0.01	41.93	15.55	0.01	0.02	0.42	0.19	0.42	0.12	2.0

Table 5.2 illustrates the results of the proposed methods and other methods in AR.Drone sequences. NBM-HC achieved the lowest RM which shows the best performance compared to the other methods in this scenario. However, the improvements compared to the original method and NBM-GA is not significant.

Table 5.2. *AR.Drone sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FN and FP show better result and higher S and S_B demonstrate higher similarity to the ground truth.

Method	TE		FN		FP		*S*		*SB*		RM
	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	
$\mathrm{MOG}_{\mathrm{RGB-D}}$	0.15	0.16	14.79	12.02	0.04	0.06	0.79	0.13	0.80	0.11	4.8
$\mathrm{GSM}_{\mathrm{UB}}$	0.49	0.29	74.97	11.66	0.01	0.01	0.25	0.12	0.26	0.12	7.0
$\mathrm{GSM}_{\mathrm{UF}}$	0.31	0.26	9.59	6.52	0.25	0.21	0.69	0.11	0.74	0.08	5.6
$\mathrm{PBAS}_{\mathrm{Bin}}$	1.25	0.33	0.25	0.80	1.25	0.33	0.33	0.09	0.66	0.10	6.8
$\mathrm{VIBE}_{\mathrm{Bin}}$	1.02	0.26	2.73	3.41	1.18	0.25	0.33	0.09	0.74	0.09	6.2
MBNS	0.13	0.11	11.84	6.02	0.05	0.07	0.82	0.11	0.83	0.08	3.0
BSABU	0.15	0.17	14.03	7.80	0.04	0.08	0.82	0.15	0.82	0.15	3.8
NBM-GA	0.13	0.11	12.51	6.48	0.04	0.09	0.82	0.14	0.83	0.12	2.6
NBM-HC	0.12	0.10	12.29	5.95	0.04	0.08	0.83	0.13	0.83	0.09	2.0

The proposed method in this chapter and other algorithms have been tested with the benchmark RGB-D dataset introduced in [48]. We compare these results and rank all the methods to confirm our proposed algorithms achieves the best result between the existing methods in various challenging scenarios. To measure the accuracy of each method, the results have been compared to the ground-truth provided with this dataset. This dataset is made from four sequences and every sequence is designed to measure the accuracy of the methods in one of the challenging scenarios. Figure 5.3 illustrates an example of this dataset.

GenSeq sequence consists of a person who is moving a box in an office environment and can test the algorithm in a general condition when several possible errors may occur in one scene. This sequence has 300 frames in total and contains 39 hand-labelled ground-truth. Table 5.3 illustrates the full outcome for all the frames in this sequence. NBM-HC outperforms by achieving the lowest TE (total error), the highest similarity with the ground-truth and around the object boundaries (S and S_B) which helped the method to receive the best average ranking (RM) among all other methods. However, the original method performed to some extent better than NBM-GA in this sequence such as achieving lower FN.

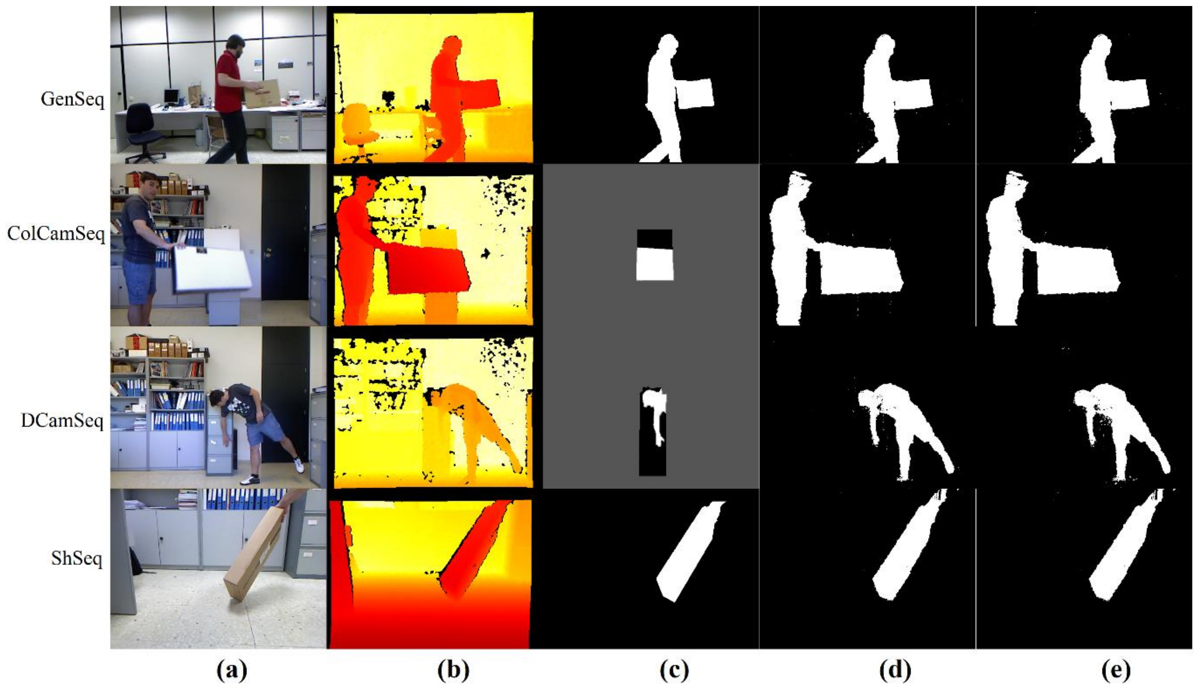


Figure 5.3. *An example of dataset introduced in* [48].
(a) Colour frame, (b) Depth frame, (c) Ground truth, (d) NBM-GA, (e) NBM-HC

Table 5.3. *GenSeq sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FP and FN illustrate better outcome and higher S and S_B shows more similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\text{MOG}_{\text{RGB-D}}$	1.93	0.66	0.63	0.01	2.09	0.02	0.79	0.20	0.45	0.13	6.4
CL_{W}	1.30	0.42	1.49	0.02	1.27	0.01	0.83	0.21	0.53	0.14	4.8
GSM_{UB}	1.38	0.56	1.04	0.78	1.44	0.66	0.83	0.20	0.78	0.11	4.6
GSM_{UF}	1.30	0.52	4.08	15.38	1.30	0.60	0.83	0.20	0.78	0.14	5.2
PBAS_{Bin}	8.24	13.78	0.33	0.53	9.36	15.97	0.66	0.21	0.71	0.10	7.0
VIBE_{Bin}	2.32	0.58	1.59	1.52	2.43	0.56	0.77	0.16	0.75	0.09	7.0
MBNS	1.09	0.46	2.85	7.43	1.02	0.56	0.88	0.14	0.79	0.12	2.2
NBM-GA	1.21	0.79	3.72	1.06	1.23	0.94	0.87	0.15	0.79	0.13	3.4
NBM-HC	1.08	0.55	3.72	3.51	1.07	0.70	0.88	0.07	0.79	0.08	2.4

DCamSeq has been created to measure the tolerance of the algorithm in case of depth camouflage by moving a hand around the file cabinet which is part of the background. This sequence has 670 frames and contains 102 ground-truth. Table 5.4 shows the result of all the methods in this sequence. Both proposed methods could significantly improve the accuracy in TE, FN, S and S_B compared to the original method. In fact, NBM-HC outperformed in this sequence and could achieve the best results by receiving the lowest RM in this sequence while MBNS achieved one of the weakest results in this sequence. Figure 5.4 illustrates the qualitative comparison of this sequence among NBM-HC, NBM-GA and MBNS. MBNS disclosed poor performance in this sequence as most parts of the hand was not detected by this algorithm. NBM-HC and NBM-GA demonstrated the highest accuracy in this frame by detecting most pixels of the foreground correctly. However, both methods contain a high number of false positive.

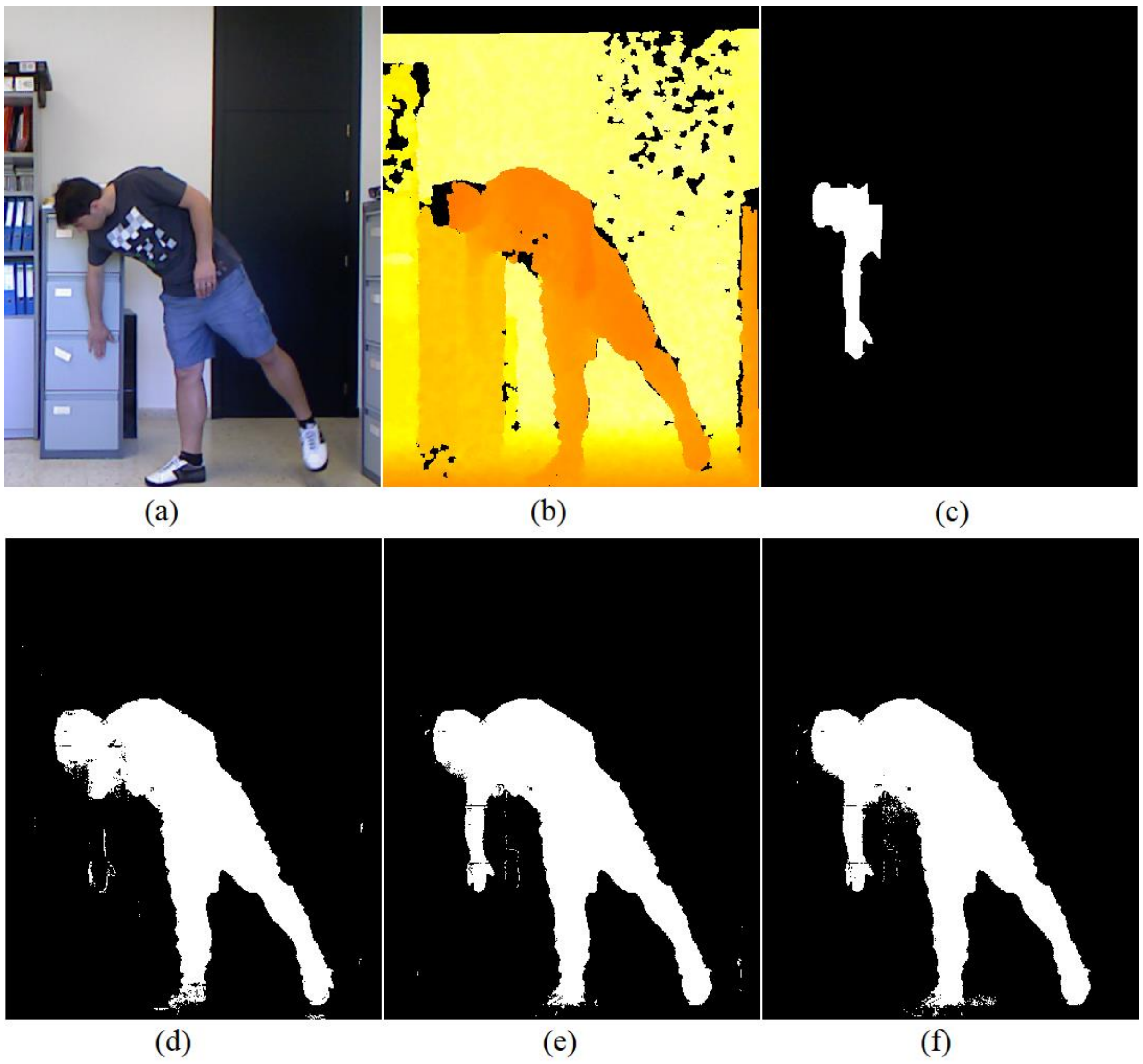


Figure 5.4. *Frame 1061 of DCamSeq sequence introduced in* [48].
(a) Colour frame, (b) Depth frame, (c) Ground truth, (d) MBNS, (e) NBM-GA, (f) NBM-HC.

***Table 5.4**. DCamSeq sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FP and FN illustrate better outcome and higher S and S_B shows more similarity to the ground truth.

Method	**TE**		**FN**		**FP**		***S***		***SB***		**RM**
	Avg.	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	**Avg.**	**St.Dev**	
$\mathrm{MOG}_{\mathrm{RGB-D}}$	2.11	1.29	15.25	0.09	1.31	0.02	0.61	0.14	0.61	0.11	3.6
CL_{W}	2.46	1.82	32.21	0.26	0.66	0.01	0.55	0.14	0.51	0.12	4.6
$\mathrm{GSM}_{\mathrm{UB}}$	2.42	1.70	49.84	10.73	0.55	1.57	0.37	0.17	0.40	0.14	5.6
$\mathrm{GSM}_{\mathrm{UF}}$	2.40	2.23	47.62	32.39	0.72	1.61	0.37	0.27	0.41	0.28	5.6
$\mathrm{PBAS}_{\mathrm{Bin}}$	5.06	12.38	46.23	32.46	3.59	13.09	0.30	0.21	0.38	0.23	8.2
$\mathrm{VIBE}_{\mathrm{Bin}}$	2.20	1.86	40.58	23.51	1.29	2.15	0.41	0.22	0.48	0.22	5.0
MBNS	2.43	2.21	52.44	30.45	0.56	1.64	0.36	0.25	0.38	0.25	6.8
NBM-GA	1.21	1.20	16.19	12.03	1.01	1.88	0.63	0.21	0.68	0.19	2.6
NBM-HC	1.39	1.63	9.84	7.84	1.36	2.26	0.65	0.21	0.70	0.19	2.6

ColCamSeq has been designed to measure the accuracy of the algorithms when colour camouflage has occurred by moving a white box to the white background. This sequence contains 360 frames and 45 ground-truth. Table 5.5 shows the result of this sequence. Overall, both proposed methods performed well and received the best average ranking in this sequence. However, NBM-HC performed slightly better by achieving lower TE and FP. The main reason behind the success of this scenario is the depth model will not be able to find any similarity with the depth pixels. Therefore, the object will be recognized as part of the foreground.

ShSeq sequence has been created to measure the effect of shadows in the scene. Table 5.6 demonstrates the outcome of this sequence. Although NBM-GA and NBM-HC performed well in this sequence, MBNS presented slightly higher accuracy in FN and similarity with the ground-truth (S and S_B) which helped this method to receive the lowest RM and consequently the best overall outcome between all methods. Both NBM-GA and NBM-HC have a high rate of FN compare to the MBNS due to the technique this method store the samples in the model. Figure 5.5 demonstrates frame number 446 from ShSeq sequences which have been also demonstrated in the author's papers in [48] and [3].

Table 5.5. *ColCamSeq sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FP and FN illustrate better outcome and higher S and S_B shows more similarity to the ground truth.

Method	TE		FN		FP		*S*		*SB*		RM
	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	
MOG_{RGB-D}	3.49	3.40	3.38	0.02	6.13	0.14	0.91	0.09	0.81	0.08	6.6
CL_W	3.20	2.77	3.52	0.09	2.92	0.10	0.89	0.15	0.77	0.16	6.2
GSM_{UB}	1.72	2.66	2.70	6.83	3.50	7.39	0.93	0.09	0.92	0.08	2.4
GSM_{UF}	2.04	2.66	0.65	1.39	4.74	7.43	0.91	0.11	0.90	0.08	4.0
$PBAS_{Bin}$	8.60	8.99	0.13	0.36	17.32	20.20	0.76	0.26	0.81	0.14	7.0
$VIBE_{Bin}$	4.47	7.02	0.39	1.25	10.68	19.22	0.86	0.13	0.86	0.09	6.4
MBNS	1.94	2.68	0.75	1.74	4.37	7.30	0.92	0.09	0.91	0.07	2.8
NBM-GA	2.40	3.09	0.58	1.72	5.57	8.38	0.91	0.11	0.90	0.07	4.2
NBM-HC	3.32	7.22	0.76	1.78	4.55	19.74	0.92	0.10	0.91	0.07	4.0

Table 5.6*. ShSeq sequence results.*
False positives (FP), False negatives (FN). Total error (TE). Similarity measure (S), Similarity measure in object boundaries (S_B). Lower TE, FP and FN illustrate better outcome and higher S and S_B shows more similarity to the ground truth.

Method	TE		FN		FP		*S*		*SB*		RM
	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	Avg.	St.Dev	
MOG_{RGB-D}	3.94	1.54	0.59	0.02	4.50	0.07	0.77	0.09	0.66	0.05	7.4
CL_W	0.81	0.35	1.60	0.05	0.68	0.02	0.94	0.04	0.71	0.07	4.2
GSM_{UB}	0.87	0.33	0.98	0.88	0.88	0.42	0.93	0.03	0.76	0.06	5.0
GSM_{UF}	1.66	0.38	0.14	0.19	1.92	0.44	0.89	0.04	0.65	0.05	5.4
$PBAS_{Bin}$	3.92	2.73	0.35	0.31	4.48	0.10	0.78	0.11	0.60	0.03	7.0
$VIBE_{Bin}$	3.72	0.99	0.06	0.15	4.31	1.17	0.78	0.07	0.64	0.03	6.0
MBNS	0.80	0.41	0.88	0.70	0.81	0.48	0.95	0.03	0.82	0.06	2.6
NBM-GA	0.80	0.34	1.39	0.80	0.70	0.42	0.94	0.03	0.81	0.06	3.2
NBM-HC	0.79	0.33	1.23	0.85	0.73	0.42	0.94	0.03	0.81	0.06	3.0

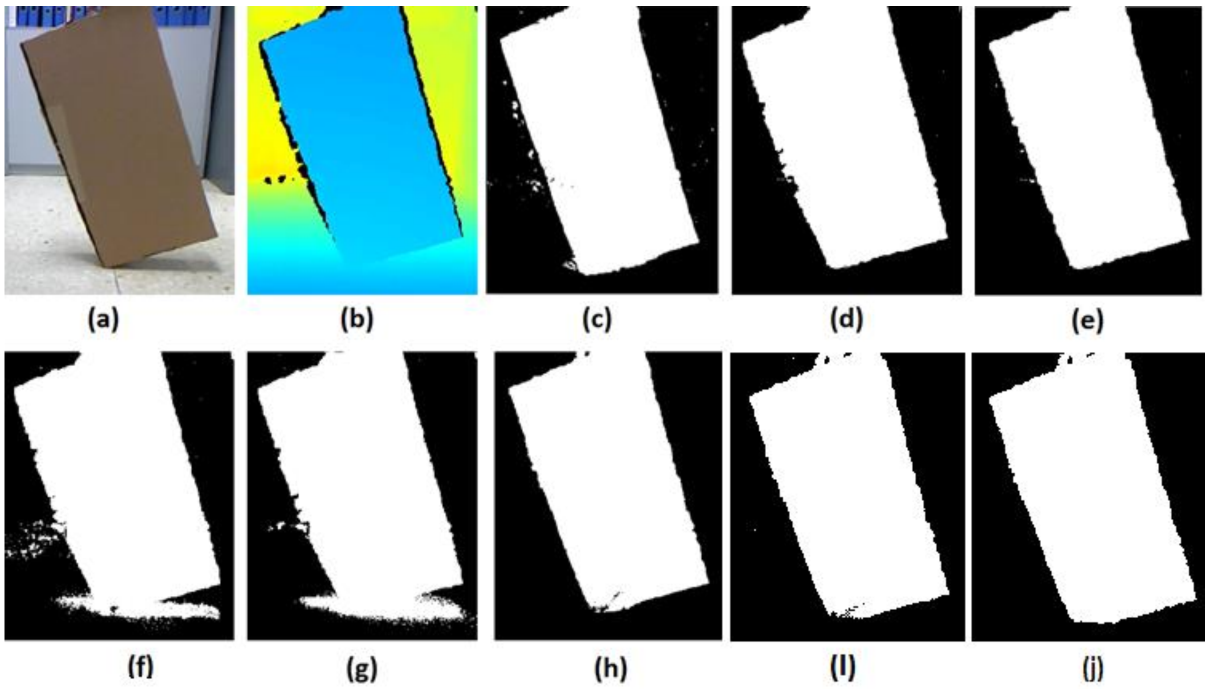


Figure 5.5. *Frame 446 of ShSeq sequences.*
The proposed methods could not perform as good as MBNS (a) Colour image, (b) Colour coded depth data, (c) CL_W output, (d) GSM_{UB} output, (e) GSM_{UF} output, (f) PBAS output, (g) $Vibe_{bin}$ output, (h) MBNS output, (i) Hill climbing, (j) Proposed method output.

Figure 5.6 illustrates the RC and RM for each individual method in each sequence. The lower amount shows a better result. According to this figure, NBM-HC outperformed in accuracy compared to the other state-of-the-art algorithms by achieving the lowest RM in ColCamSeq and GenseqSeq and DCamSeq. This verifies that our method achieved the best overall performance in these sequences. Also, in Shseq, it has performed moderately well. In the DCamSeq, MBNS obtained a high RM value which demonstrated poor performance. However, this weakness has improved significantly in both NBM-HC and NBM-GA by only changing the approach to update the background models. This confirms that these updating strategies completely change the performance of these methods. Consequently, the overall performance of our algorithms in these four sequences outperforms among these other seven methods by achieving the lowest amount of RC. NBM-GA received second place after NBM-HC as this method obtained slightly lower performance in some of the sequences such as GenseqSeq. Figure. 5.7 illustrate convergence graph from NBM-HC and NBM-GA methods for an average of twenty runs and best fitnesses. It can be seen that converged in the methods NBM-HC and NBM-GA are close to each other. The average fitness at iteration 580 in NBM-GA is to some extent lower than NBM-HC. However, the average of the best fitness is contradictory. Consequently, NBM-HC method is helpful in speeding up the best convergence.

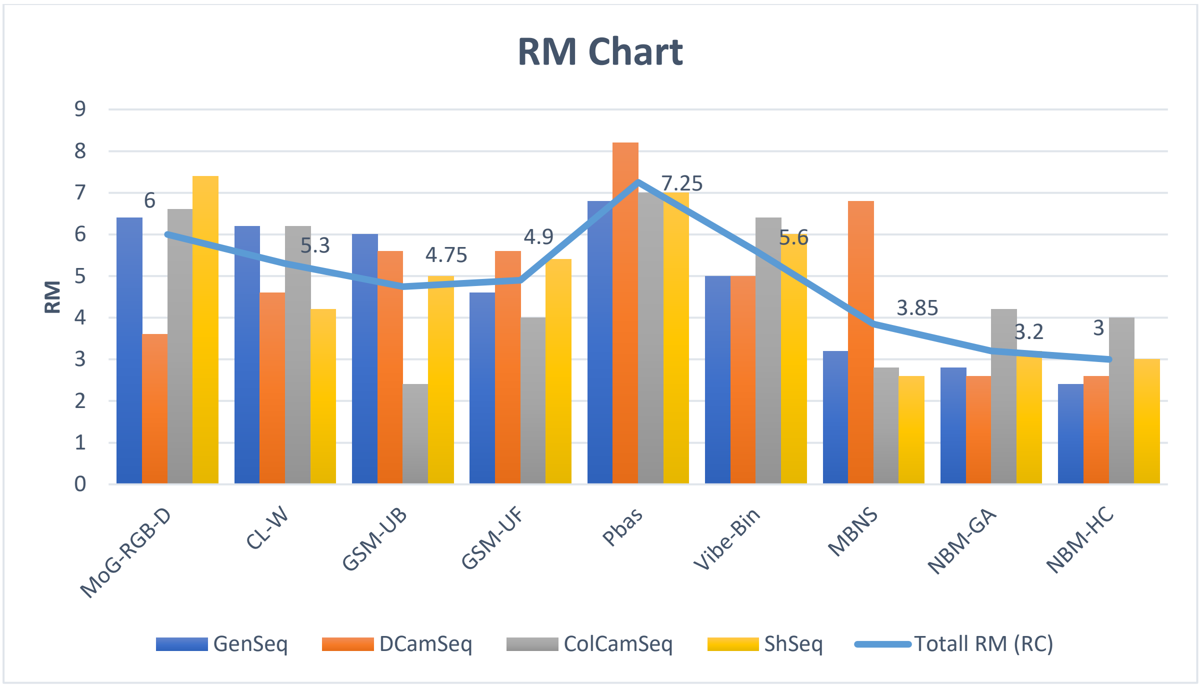


Figure 5.6. *RM chart illustrates the overall performance. CL_W, GSM_{UF}, GSM_{UB} , PBAS and Vib_{bin}, MBNS, NBM-GA and NBM-HC (proposed method) in GenseqSeq, DCamSeq, ColCamSeq and Shseq sequences. The lower RM and RC demonstrate better performance.*

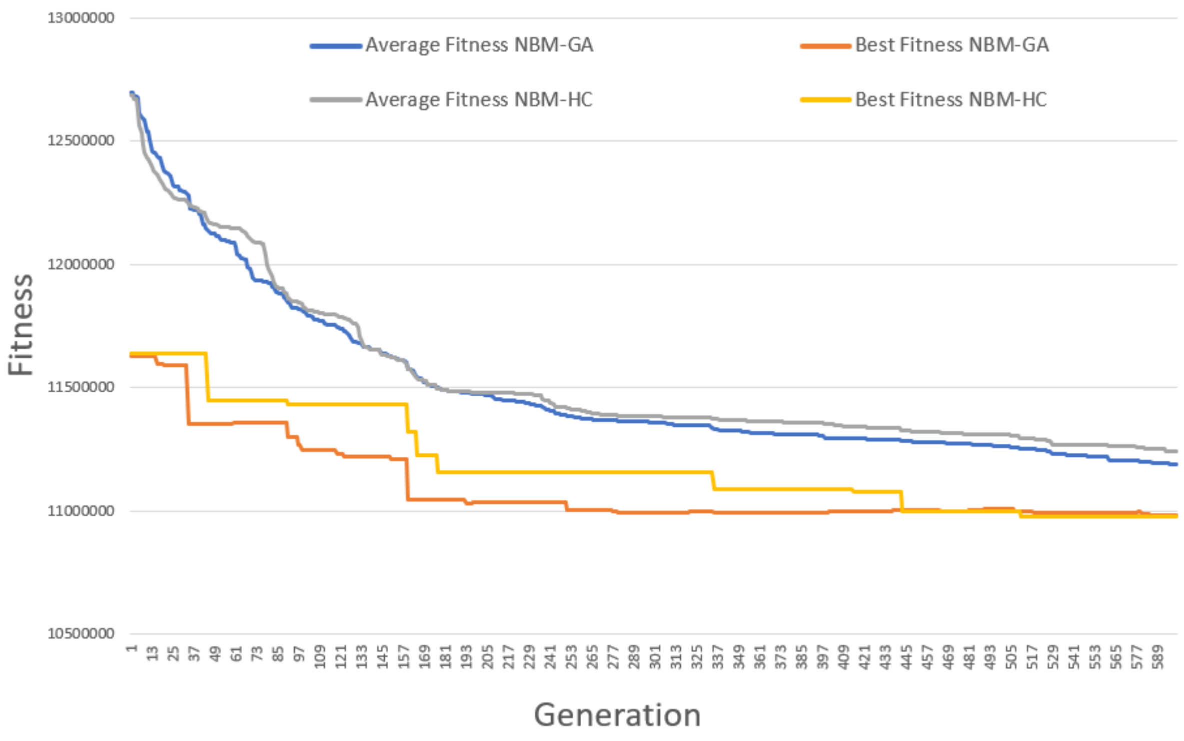


Figure 5.7. *Convergence graph from NBM-HC and NBM-GA methods for twenty times runs on dataset introduced in [48].*

In order to have a better understanding about the accuracy of NBM-HC method, a qualitative comparison of NBM-HC with other state of the art algorithms in SBM-RGBD Challenge [116] including the proposed method in the previous chapter known as (BSABU) is shown in figure 5.8. This figure belongs to frame number 534 of DCamSeq2 dataset. Although this sequence (Depth Camouflage) was the main weakness of the original method (MBNS), this evaluation proof this is not the case for BSABU and NBM-HC. In fact, it is opposite for these two methods. This sequence is very challenging and many background subtraction algorithms have difficulties to detect the foreground in this sequence. Clearly, RGBD-SOBS [53], AvgM-D, CwisardH+[130] and Kim totally failed to detect the fingers in this sequence. RGB-SOBS [95] detected the hand with high accuracy because it is not using depth data and therefore this method will have many other weaknesses in other situations such as colour Camouflage, illumination changes and shadow sequences as we have seen in the previous chapter. SRPCA [124] could detect the fingers but the accuracy is low, and all fingers are connected to each other. In fact, only SCAD [52], BSABU and NBM-HC could fully detect the hand in this sequence.

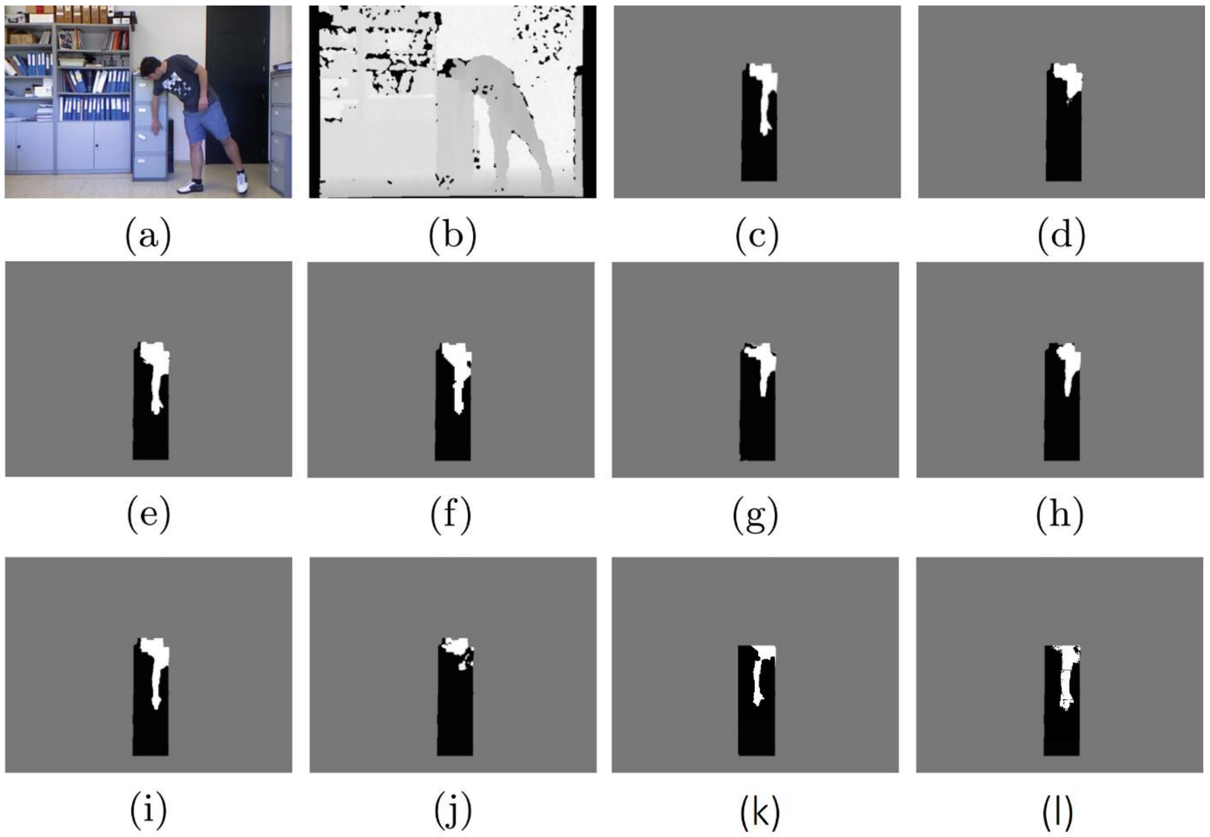


Figure 5.8. *Frame 534 of DCamSeq2 (Depth Camouflage) dataset.*
(a) colour frame, (b) depth frame, (c) ground truth, (d) RGBD-SOBS, (e) RGB-SOBS, (f) SRPCA, (g) AvgM-D, (h) Kim, (i) SCAD, (j) CwisardH+, (k) BSABU, (l) NBM-HC.

5.5 Real-time Experiment

Using an evolutionary algorithm as part of the loop sometimes is very expensive in terms of computation and could increase the time required to build a model or solution. In this section, both proposed methods (NBM-GA and NBM-HC) has been tested in a live application to measure the speed of these moving objects algorithms. The performance of each method is calculated as the mean rate of the processing time. Data captured by Microsoft Kinect sensor and the evaluation was performed on a computer with an Intel (R) Core (TM) i7-6700HQ CPU @ 2.6 GHz and 8 GB RAM and the. Figure 5.9 and figure 5.10 illustrate the speed of processing moving object detection using GA and HC per frame in second. Although both methods performed well and demonstrated that are capable of being used on real-time applications, NBM-HC with average processing of 0.0626 second (S) performed better than NBM-GA, looking at the frame rate, it is about 16 frames per second. In addition, NBM-HC performed better than the original method (MBNS) which shows using Hill climbing not only increased the accuracy but also increased the performance in terms of speed and computational costs. The maximum time to process the frame in NBM-HC is 0.097 S and the minimum is 0.047 S. In NBM-GA the maximum time taken to process a frame is 0.099 S and minimum is 0.056 S. It is important to note in this evaluation the proposed algorithms implemented in C++ and OpenCV library [119] without applying any specific code optimisation as the primary goal of this experiment was to prove the proposed algorithms are able to effectively run at real-time. Thus, not any effort has been made in implementation to optimise the code.

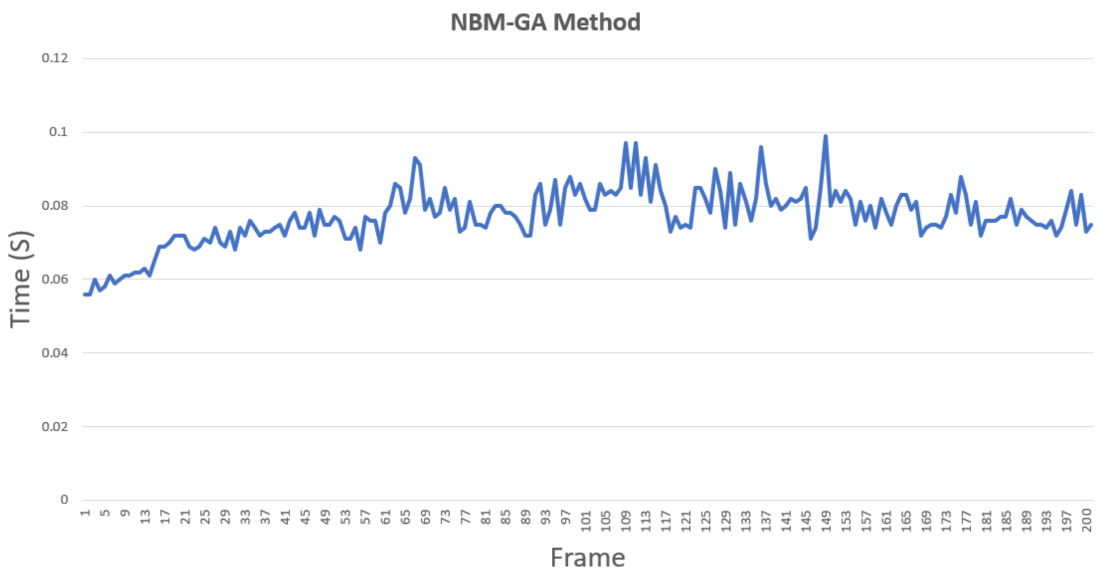


Figure 5.9. *The computational time for NVM-GA to process each frame. The time for the first 200 frames is shown in seconds.*

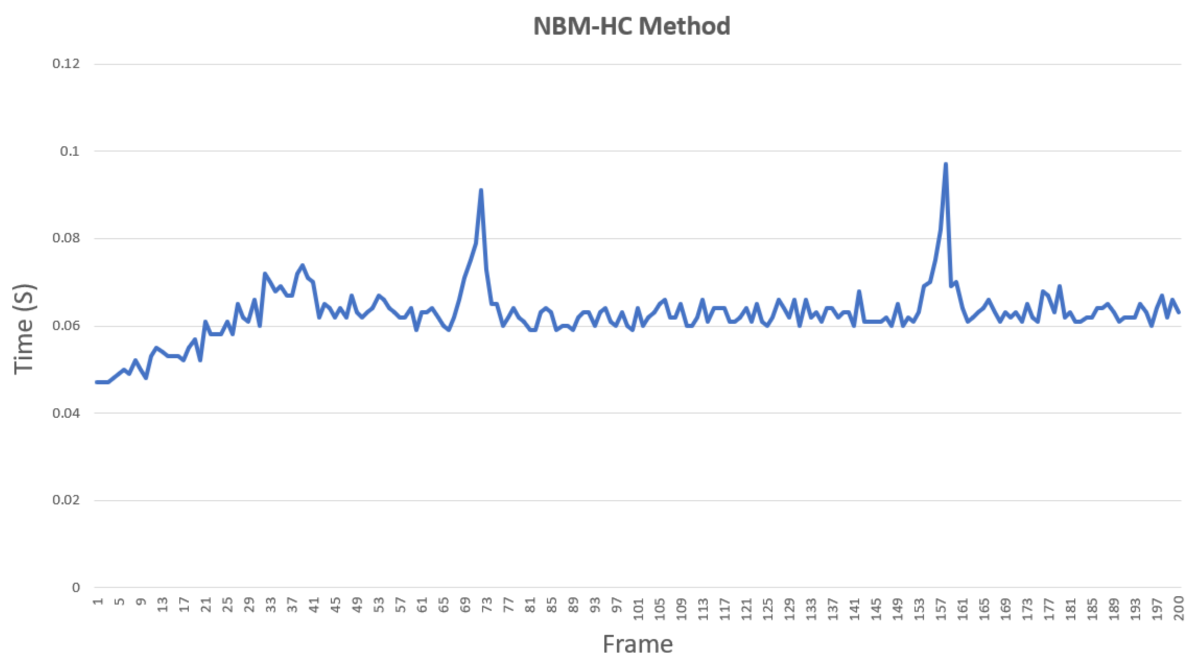


Figure 5.10. The computational time for NVM-HC to process each frame. The time for the first 200 frames is shown in seconds.

5.6 Summary

In this chapter, a novel method for moving object detection similar to the previous work of MBNS algorithm has been presented. The proposed method is built based on four major steps including; initialization, post initialization filtering, classification, and update.

The main contribution of this chapter is to use EC methods in the update stage instead of pixel-wised update. Hill climbing (NBM-HC) and GA (NBM-GA) have been implemented to search and find the optima background model. The proposed methods produce one model for colour and another for depth data. The system then combines the results from the colour and depth model to produce the final segmentation mask. The fitness function calculates the total distance of background pixels for each sample in the depth model and then models are updated by the Genetic Algorithm for pixels identified as a background. The crossover operation exchanges the pixels from the chosen parent in the models with the new frames. The chosen parent will be selected by Roulette Wheel which allows all the chromosomes to have a chance to be selected. This helps the system to avoid being trapped in the local minima and maxima. These local optima can be created by noise. This means, the pixel has a high depth value (it is probably the background pixel) however, it is not true in reality and it does not exist. This is only created by the fault of the sensor. The Roulette Wheel cause this method to sometimes remove such a pixel in the update procedure. This is not the case in the regular update of the original method. This is one of the main advantages of this method over MBNS method. Additionally, a mutation will be applied without considering they are classified as foreground or background in order to prevent permanent misclassification.

The results and evaluation section demonstrated that applying GA to solve the background modelling problem addressed in this chapter can improve the accuracy over the original method (MBNS). Even though the worst average NBM-GA run could accomplish more accurate detection than the MBNS, there is no evidence to prove that the GA on average perform better than the Hill-Climbing algorithm introduced in this chapter. In fact, the results indicate a contrary. It is clear that both proposed algorithms (NBM-GA NBM-HC) have the most accurate and reliable outcomes in comparison with MBNS and other state-of-the-art methods by achieving the lowest RC as demonstrated in figure 5.6. The results in table 1-6 proved that the proposed methods could improve the robustness in three different benchmark sequences by achieving the lowest value of RM. Also, these updating strategies could significantly improve the weakness of the original method and allow the proposed methods to achieve the best results in DCamSeq sequence.

This research investigated the first attempt to compare the performance of GAs against Hill-Climbing directly on this type of background modelling and moving object detection problem. This research started with the assumption that GAs do not get stuck in local optima and should obtain higher performance by giving a chance to all frame solutions to stay in the model. However, based on the evidence demonstrated in this chapter, being stuck in local optima is not a problem for NBM-HC as it updates the model with the new frames. This is interesting as usually GA perform better than Hill-Climbing algorithm as GA can reach to global optima [148]. However, this is not always the case and in some problems similar to the background modelling problem, HC outperformed GA [147][149]. Therefore, it would be interesting to investigate if this problem categories as elementary landscapes introduced in [150]. This would require further research as elementary landscapes are normally quite complex and tools required to recognize elementary landscapes are abstruse [151]. In addition, only a small percentage of optimisation problems have local search spaces that correspond to elementary landscapes [152].

Also, it would be interesting in further studies to implement other types of optimisation algorithms instead of Hill Climbing or Genetic Algorithm. Also, in some sequences, the amount of false negative is relatively high which need further investigation and improvement. A possible solution for this problem would be using a more accurate depth sensor.

Chapter 6
Conclusions and Future Research

In this chapter, a summary of the work presented in this thesis will be discussed and then indicate some directions for further research in the future which follow from this thesis.

6.1 Comparisons of all proposed methods in this thesis

In this section, all proposed methods in this thesis will be compared in terms of accuracy and efficiency. Figures 6.1,6.2 and 6.3 illustrates the accuracy of all methods in total error (TE), comparison to the ground truth (S) and comparison to the ground truth in object boundaries (S_B). These figures show that BSABU, NBM-GA and NBM-HC significantly improved the weakness of the original method (NBMS) in depth camouflage scenario (DCamSeq) which NBM-HC performed better than others. Instead, BSABU performed slightly better in shadows (SHSQ) and Genseq sequences. The main reason for this is using CIE L*a*b* colour space in BSABU which allows the colour pixels to separate the colour value without interfering of illumination (L) component. Also, BSABU has another advantage over other proposed methods which is high accuracy in stationary foreground objects and bootstrapping scenarios. On the other hand, according to figure 6.4 the NBM-HC method is faster and more efficient in terms of computation. Therefore, we would suggest using the BSABU where efficiency is not a priority and NBM-HC where speed is an important element and bootstrapping doesn't exist.

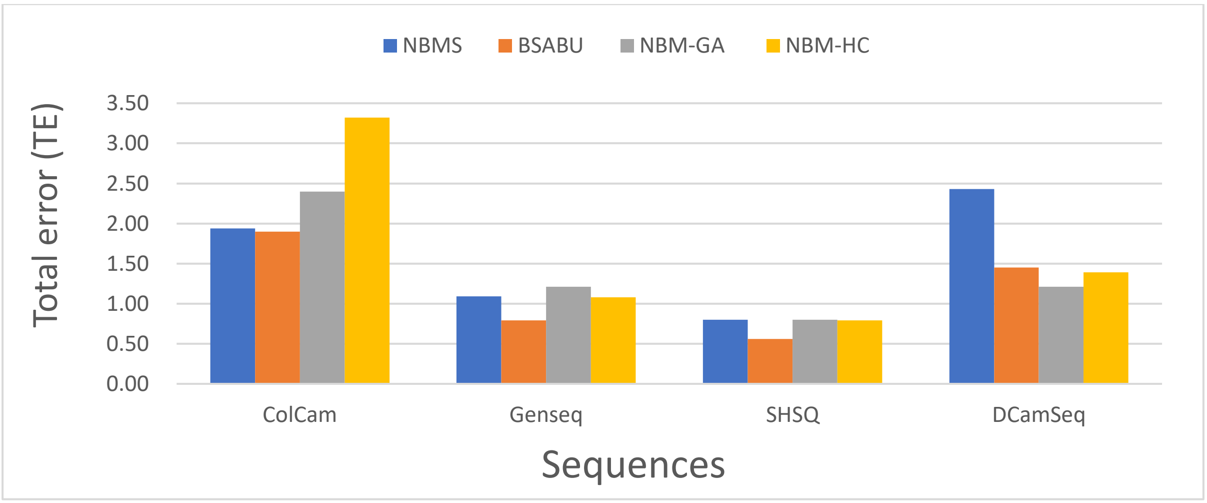


Figure 6.1. Total error (TE).

The lower amount shows better performance.

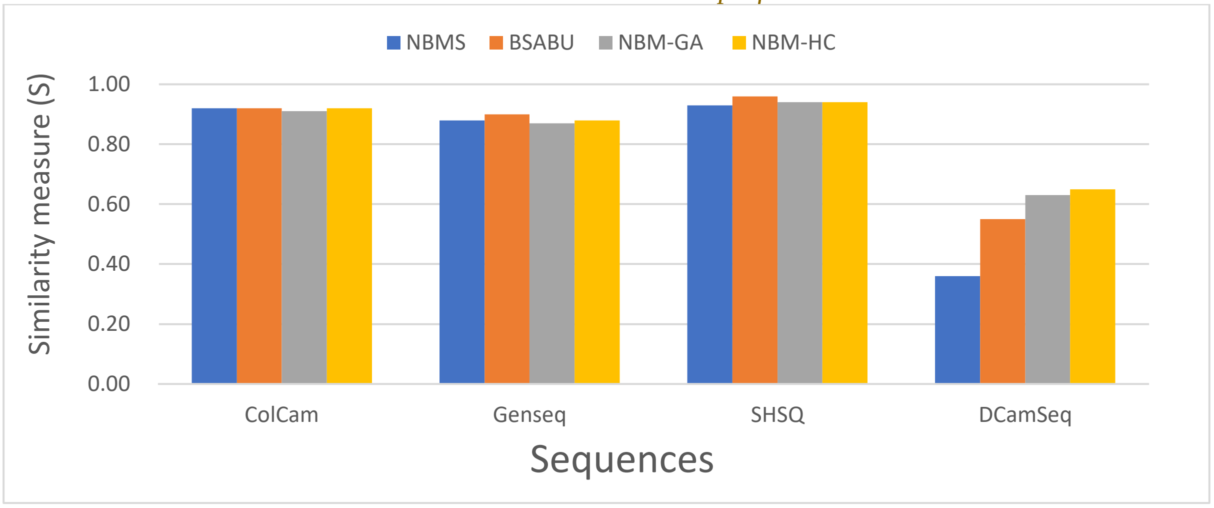


Figure 6.2. *Similarity measure (S).*
The higher value shows better performance.

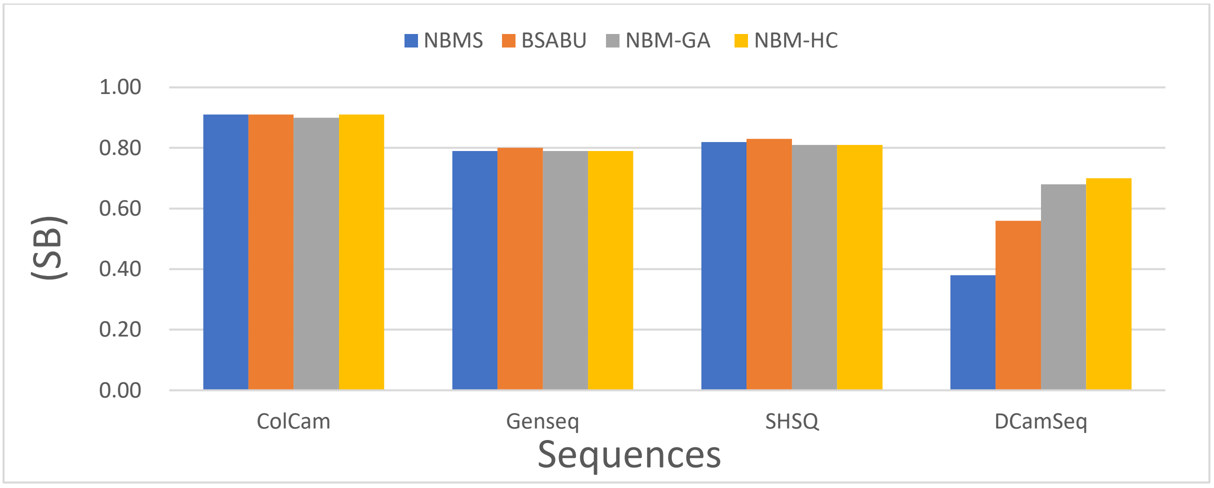


Figure 6.3. *Similarity measure around object boundaries (S_B).*
The higher value shows better performance.

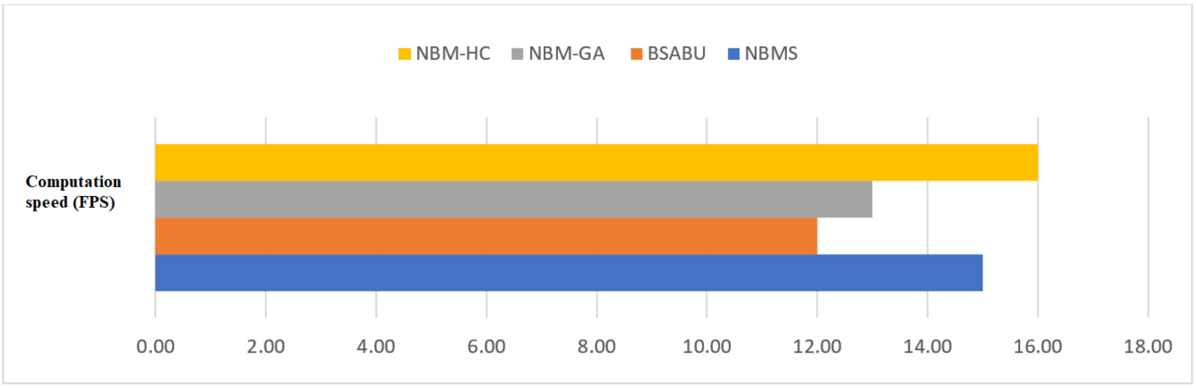


Figure 6.4. *The efficiency (FPS) of the proposed methods in this thesis.*

6.2 Concluding Remarks

Motion and change detection by a fixed camera still is a challenging task in the field of computer vision, especially for the real-time automated surveillance and object tracking applications. This is mainly due to some problems related to the moving objects and monitored environment such as the characteristics of the moving object, the same foreground colour as the background, complex dynamic environment (e.g. curtains or tree leaf movement), variations in illumination and weather conditions (e.g. sunshine), along with the distance from the camera. Therefore, in recent years, many researchers have introduced different strategies to overcome these problems. One of these motion detection strategies was nonparametric background subtraction algorithms such as ViBe and PBAS which in the last few years became popular among researchers due to robustness and being simple to implement. However, these methods suffer from some limitations such as the production of a ghost, weak detection of slow and stationary moving objects and detection under major illumination changes. By finding a new way to reduce these limitations and increasing the accuracy of the foreground masks, we could obtain more robust and consistent results. One of the strategies to overcome these issues is to use depth sensors such as Microsoft Kinect which illumination doesn’t affect depth sensors. However, these sensors also have some limitations such as depth shadows, depth camouflage, absorption by black objects, absent observations, and lower sensitivity at longer distances which require a powerful software to reduce the effect of these limitations by combining a colour and depth sensors. Therefore, the aim of this thesis was to improve the detection accuracy of moving object detection by achieving more precise and consistent detection in different challenging scenarios.

In chapter 3 of this thesis, we investigated how the pixel-wise nonparametric method can improve RGB-D moving object detection. To apply this, we have introduced a new nonparametric moving object detection method using an RGB-D camera in an indoor environment. This method is known as MBNS and worked by the history of previously observed pixel values based on four major steps; initialisation, post initialisation filtering, classification and update. It first creates one model for colour frames and another one for depth frames. It then compares each pixel of the new frame with the models in order to identify them as a foreground or background. After segmentation, the models will be updated in two ways. First, in the regular update, those pixels identified as a background will be updated based on their distance. In particular, the smallest sample (pixel) in the model will be replaced with the pixel in the new frame in the same location. Secondly, in order to prevent permanent misclassification, some pixels will randomly be swapped with the new frame regardless of being foreground or background. This updating strategy is called blind update. This process will be repeated for all pixels of every new frame. This method has been evaluated in four different sequences from benchmark dataset which outperformed other state-of-the-art methods. It has also been tested in two different fast-moving objects (quadcopters) in the sequences captured by us. In addition, it has been tested in a live application to measure the speed and computational cost. This method demonstrated that it is capable of being implemented in live applications.

During an extensive evaluation process, despite the excellent results achieved by the proposed method compared to the other state of the art algorithms, we have discovered this method has weaknesses in some challenging scenarios such as detection of slow and stationary moving objects, depth camouflage and bootstrapping scenarios. The main reason for the weak detection of a slow-moving object is that the moving object gradually absorbs to the model by the blind update and consequently, it will disappear from the detection mask. Another weakness of the proposed method is dealing with the bootstrapping sequences. In bootstrapping, generally the moving object exists in the scene from the first frame (during the initialisation) therefore, the foreground will exist in the background model and only a fraction of the moving object will be detected by the algorithm.

Thus, in chapter 4, we have tried to handle these issues (SFOs and bootstrapping) by proposing an adaptive blind update policy, shadow detection method, detection of bootstrapping and changes to the background. In particular, we investigated what is the effect of using adaptive blind update policy on pixel-wise moving object detection and if it can resolve the weakness of the original method in scenarios where stationary and slow-moving objects exist.

The adaptive blind update policy tracks the moving object and adjusts the frequency of the blind update based on the speed of moving object. This allows the method to be able to simply detect the stationary, slow and fast-moving objects. On the other hand, the proposed method detects any changes occurring in the background and increase the blind update in order the model adapt to the recent changes in the environment. If the system detects any moving object during initialisation, then bootstrapping has occurred and it starts finding the shape of the moving object by a combination of colour and depth edge detection result. The objects that some part of them has been identified as the moving object will be added to the foreground mask.

Additionally, illumination changes and shadow detection method has been proposed for this method in order to improve detection accuracy in case of shadow, illumination changes and depth camouflage. To achieve this, CIE L*a*b* colour coordinate for the colour frames has been used. Unlike the RGB space, L*a*b* colour is intended to approximate human vision where the L element closely indicates the human perception of lightness. Thus, it can be used to check the colour value of pixels (a and b) without interfering of illumination. In this system, shadow has been defined as an area where the lightness (L component) is very low (dark) and the depth model indicates no change in the depth values. This method has been extensively evaluated in SBM-RGBD datasets which has 33 sequences under seven different challenging categories in Illumination Changes, Colour Camouflage, Depth Camouflage, Intermittent Motion, Out of Sensor Range, Shadows and Bootstrapping. This method is also separately compared with the original method which results demonstrated significant improvement in the accuracy of those sequences containing a slow and stationary moving object.

Overall, the proposed method proved it is more consistent in most scenarios. The main disadvantage of the proposed method was the system has a higher computational speed compared to the original method. However, this object detection with a lower number of frames per second can be applied in live applications with the current hardware.

In chapter 5, instead of using a simple update mechanism, we have implemented a more sophisticated method using EC optimisation algorithms such as Hill Climbing and Genetic Algorithms to search and find the optimal background model. This improved the detection accuracy and efficiency of our method. In particular, we investigated what is the effect of using heuristic searches in pixel-wise moving object detection. To apply this, GA has been added to the method. A fitness function has been implemented to rank each solution based on the total distance of depth pixels in each frame in the depth model and then the crossover operation exchanges the pixels from the chosen parent in the models with the new frames. The chosen parent will be selected by the Roulette Wheel which allows the other chromosomes also to have a chance to be selected. This allows the system to prevent being trapped in a local optima solution. This is one of the main advantages of this method over the original method (MBNS). The results and the evaluation section demonstrated that applying GAs to solve the background modelling problem addressed in chapter five improved the accuracy over the original method (MBNS). However, there is no evidence to prove that the GA on average performed better than the Hill-Climbing algorithm introduced (NBM-HC). In fact, the results demonstrated the opposite. Both proposed algorithms (NBM-GA NBM-HC) have the most accurate and reliable results in comparison with MBNS and other state-of-the-art methods. Also, these updating strategies could significantly improve the weakness of the original method and helped NBM-GA and NBM-HC methods to achieve the best results in DCamSeq sequence. In addition, NBM-HC has the lowest computational cost compared to all other solutions proposed in this thesis. Therefore, we suggest NBM-HC method should be used where the efficiency is important and bootstrapping or the stationary moving object doesn't exist in the scene. On the other hand, BSABU can deal with all type of moving objects with a higher computational cost.

6.3 Research Achievements

In the literature review in chapter 2, we have noticed that despite the extensive research in the past, the problem of moving object detection remains challenging in many situations. Therefore, our aim during the research presented in this thesis was to improve the accuracy of moving object detection by achieving more precise and consistent detection in different challenging scenarios. To achieve this aim, first, we have identified the challenging scenarios which normally background subtraction algorithms have difficulties to detect the full silhouette.

Then the weaknesses and advantages of the current methods have been investigated such as accuracy of object detection, the accuracy of detection around object boundary, consistency and capability of applying the method in the live applications (efficiency). Based on these previous researches, new methods are proposed in this thesis to improve the weaknesses of previous methods and obtain a more reliable algorithm to tackle the moving object detection problem in farther challenging scenarios. These include detection of a fast and slow-moving object, detection. In particular, based on previous studies, we investigated how to address the below three research questions in this thesis. The first research question was investigated in chapter 3, the second research question was addressed in chapter 4 and the last question was investigated and implemented in chapter 5.

- How pixel-wise nonparametric RGB-D approach can improve the accuracy of moving object detection problem?
- What is the effect of using adaptive blind update policy on pixel-wise nonparametric moving object detection?
- What is the effect of using heuristic searches in nonparametric moving object detection?

The proposed methods in these chapters enhanced the accuracy and efficiency of RGB-D moving object detection in many challenging scenarios. The proposed methods in chapter 3,4 and 5 follow nonparametric approaches and created based on the following hypothesis:

- If any pixel value has been observed many times in the same location, the pixel has a high chance of being part of the background.
- Another hypothesis applied in our methods is to use a combination of colour and depth data to cover each other's weaknesses.
- The other hypothesis that has been introduced and implemented in this thesis is that if two separate pixels in the 3D dimensional space with the same length (x) and height (y) observed, the pixel with a higher depth (d) most likely is the background pixel.
- Besides, we made an assumption that neighbouring depth pixels are most probably to have a similar value. This assumption is used to remove the unknown pixels by replacing them with one of the nearest pixel values called ADO removal strategy. The advantage of this effective method is being fast and simple which helped to significantly decrease the number of errors by having more accurate depth pixel values in the depth model.

All these hypotheses and assumptions applied to our methods and more details will be discussed in the next section.

6.4 Discussion of proposed methods, limitations and parameter's optimisations

Figure 6.5 illustrates the summary of this thesis that how hypothesis and literature review could lead us to the research questions and how these questions have been addressed in this thesis. The first research question (how pixel-wise nonparametric RGB-D approach can improve the accuracy of moving object detection problem?) has been addressed in chapter 3 and the evaluation results in this chapter (table 3.3-3.8 and figure 3.18) demonstrated the proposed pixel-wise nonparametric RGB-D moving object detection method (MBNS) outperform in drones and four publicly available datasets by achieving the most accurate and reliable outcomes in comparison with other state-of-the-art methods. The proposed method created accurate depth model regardless of noisy depth frames and this helped the system to significantly reduce the amount of false detection in case of colour camouflage, sudden illumination changes and appearing of shadow on the floor. However, this method suffers in depth camouflage scenarios.

The second research question (What is the effect of using adaptive blind update policy on pixel-wise nonparametric moving object detection?) has been addressed in chapter 4. Extensive evaluation results on adaptive blind update policy method in SBM-RGBD challenge benchmark [129] illustrated in figure 4.9. This figure showed that the proposed method could outperform in several scenarios such as Depth-Camouflage, Intermittent motion, and Shadows sequences and satisfactory detection rate in others. In other words, the proposed method detected the moving objects in all different situations and it did not fail in any scenario. This means BSABU is a consistent and reliable method.

In chapter 5, the third research question (what is the effect of using heuristic searches in nonparametric moving object detection?) has been investigated by using optimization algorithms such as Genetic Algorithm and Hill climbing to update the models. According to the evaluation results in figure 5.6, these algorithms could help the method to update the models more efficiently and detection results have been improved with these methods. Hill climbing based method (NBM-HC) has the lowest computational costs compared to all other methods proposed in this thesis. This proved Hill-Climbing helped the original method to increase the accuracy as well as being more efficient in updating the models.

Overall, the methods we have introduced in this thesis improved moving object detection accuracy in many challenging scenarios. During extensive tests, we could not find any situation where proposed methods are unable to detect the moving objects. However, there are still situations such as depth-camouflage and bootstrapping that the results could improve further, and the foreground mask would be detected more accurately. In addition, according to section 6.1, NBM-HC slightly increased the accuracy and efficiency of the proposed original method (chapter 3) by using heuristics searches. However, this amount of optimisation is not enormous and can be achieved in other ways such as optimising the programming code (c++) or using GPU programming. Therefore, we would suggest using the BSABU method where efficiency is not a priority as this method is more consistent in many different challenging scenarios and NBM-HC where speed is an important element and bootstrapping doesn’t exist.

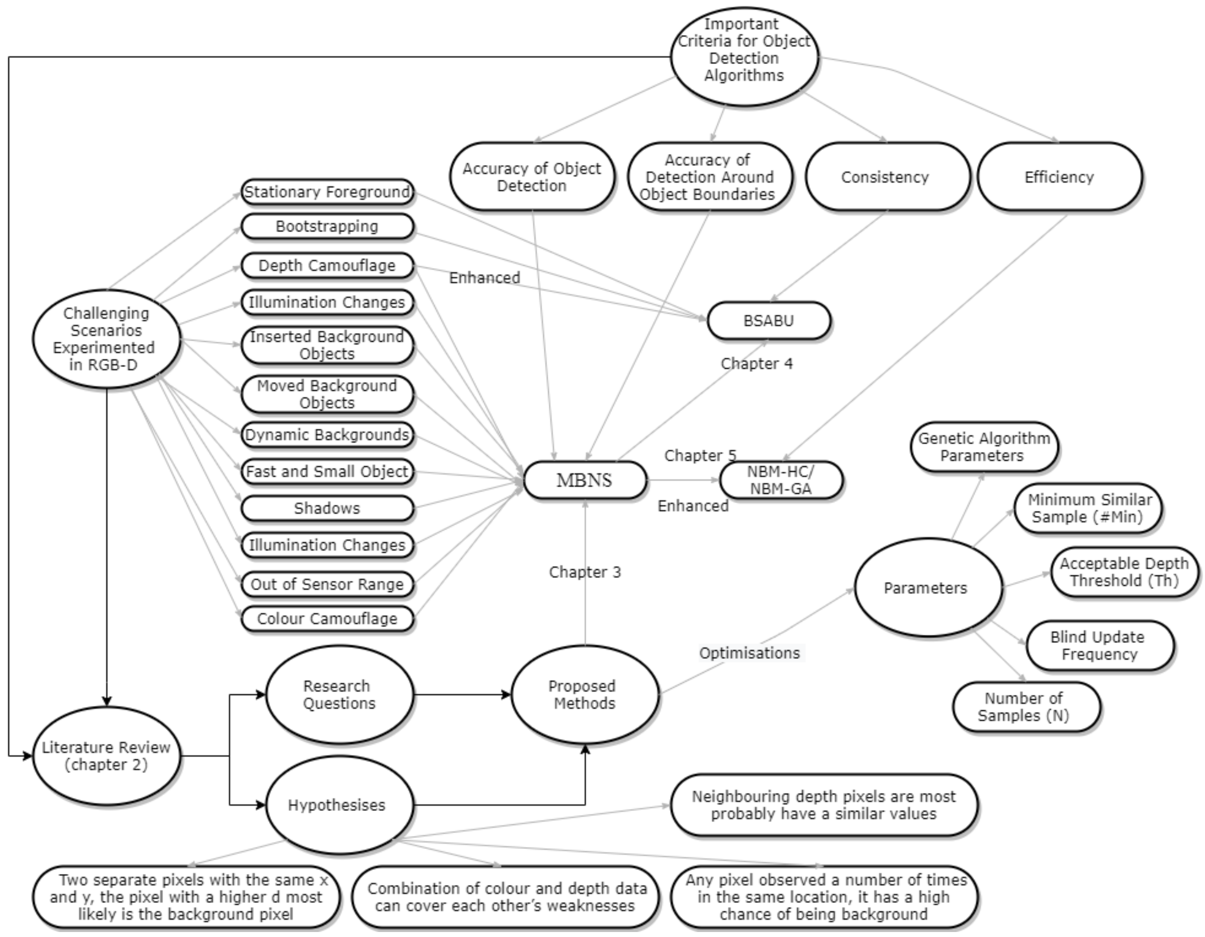

Important Criteria for Object Detection Algorithms
Accuracy of Object Detection
Accuracy of Detection Around Object Boundaries
Consistency
Efficiency
Stationary Foreground
Bootstrapping
Depth Camouflage
Illumination Changes
Inserted Background Objects
Moved Background Objects
Dynamic Backgrounds
Fast and Small Object
Shadows
Illumination Changes
Out of Sensor Range
Colour Camouflage
Challenging Scenarios Experimented in RGB-D
Enhanced
BSABU
Chapter 4
Chapter 5
MBNS
NBM-HC/ NBM-GA
Enhanced
Chapter 3
Genetic Algorithm Parameters
Minimum Similar Sample (#Min)
Acceptable Depth Threshold (Th)
Parameters
Blind Update Frequency
Number of Samples (N)
Optimisations
Research Questions
Proposed Methods
Literature Review (chapter 2)
Hypothesises
Neighbouring depth pixels are most probably have a similar values
Two separate pixels with the same x and y, the pixel with a higher d most likely is the background pixel
Combination of colour and depth data can cover each other's weaknesses
Any pixel observed a number of times in the same location, it has a high chance of being background

Figure 6.5. *Summary of this thesis.*

On the other hand, some other factors such as the type of depth sensor and parameter's optimisations can affect the performance of these methods. For example, the number of stored pixels in the models (N). The higher number of N can increase the accuracy of the methods. However, the method needs more frames for initialisation and more memory to store the pixel values. Therefore, the computational cost will be increased in compensate to increase the accuracy. During the research in this thesis, we have selected N= 20, as this number of samples in other methods (such as $ViBe_{bin}$[51]) proved that it is sufficient for moving object detection and also it is possible to keep these samples in the memory with the current hardware.

Th_D (acceptable depth) is a sensitive parameter and plays a key role in the proposed methods. This value needs to be accurately selected according to the faults of the sensor. Increasing the Th_D can increase the FP in depth camouflage scenarios and reduce TN in shadow and illumination changes. Generally, the Th_D needs to be greater for the sensors with higher noise. Therefore, this can be measured based on the noise of the sensor.

The $\#_{Min}$ is used as the minimum number of similar samples in the model to the new pixel. It should depend on the number of samples (N). A higher amount of N requires larger

$\#_{Min}$ and therefore it is recommended as N/5. This amount can tolerate a reasonable amount of noise in the background and also it is enough to not accept noise as the foreground.

Blind update frequency also plays an important role in the update procedure which generally more frequent update can increase the detection in high speed moving object and reduce accuracy in slow and stationary moving objects. This has been extensively discussed in chapter 4.

For GA the population size selected as the same as N ($\mu = 20$) is based on the same rationale. The number of generations is selected as 30 generations per second as this was the maximum generation we could run on the live application. Crossover probability of 50% each parent and the mutation probability (MP) of $\frac{1}{N\times 2}$. Naturally, the mutation is a very slow phenomenon and used for random exploration, but it should not lead the GA to a pure random search as it would be the case if many genes suddenly mutated. Therefore, low mutation rate ensures that not many mutations are considered at once. In the next section, we recommend several ways to improve our current research even further in the future.

6.5 Recommendations for Future Research

Even though we have introduced several new moving object detection methods that are capable to deal with the challenging scenarios caused by the moving object or complex background, the work possibly can be further developed in a number of ways:

- More complex decision-making system for the adaptive blind update can be developed to deal with more than one moving object. This enables the system to be able to adjust the frequency of the moving object based on more than one moving object. For example, if there are one slow-moving object and one fast-moving object in the scene, the system would be able to adjust the frequency in a way to be suitable for both objects or some area of the scene can have high frequency and another area could have a lower rate for the blind update.
- In this thesis, we have used a basic tracking system to track the moving object and adjust the frequency of the blind update based on the speed of the moving object. However, this possibly can be improved by implementing more accurate and advanced tracking techniques.

- One of the main issues of using depth sensors is the limitation in the distance these sensors can capture. For instance, Microsoft Kinect can only obtain a maximum length of 6-8 meters and after only 3 meters, the accuracy starts reducing. Therefore, in a large area such as a factory or long hallway, it would be practical to use several sensors together and implement a synchronisation algorithm to control all these sensors in one place.
- Due to the limitation we had during this research, we could only use low-cost depth sensors like Microsoft Kinect. Using more advanced depth sensors and reducing the depth tolerance in the algorithm would possibly solve the issue of depth camouflage, shadow and illumination changes.
- In the background modelling and moving object detection problem, the Hill-Climbing algorithm outperformed GA. It would be interesting to investigate if this problem categories as elementary landscapes as only a small percentage of optimisation problems have local search spaces that correspond to elementary landscapes. This would require further research as elementary landscapes are normally quite complex and tools are required to recognize elementary landscapes are abstruse.
- In chapter 4, we have introduced a method based on adaptive blind update policy (BSABU) and in chapter 5, we have introduced a method based on Hill Climbing algorithm (NBM-HC). The combination of these two methods would improve the detection accuracy of individuals. This can be achieved by some machine learning techniques such as Ensemble Classifier that can combine these two methods to produce one optimal solution. Such a system can rely more on BSABU when a stationary moving object, bootstrapping and shadow exist in the scene and rely more on NBM-HC on all other scenarios.

References

[1] D. G. T. Kinjal A Joshi, “A Survey on Moving Object Detection and Tracking in Video Surveillance System,” *Int. J. Soft Comput. Eng.*, vol. 2, no. 3, pp. 44–48, 2012.

[2] S. Bianco, G. Ciocca, and R. Schettini, “Combination of Video Change Detection Algorithms by Genetic Programming,” *IEEE Trans. Evol. Comput.*, vol. 21, no. 6, pp. 914–928, 2017.

[3] N. Dorudian, S. Lauria, and S. Swift, “Nonparametric background modelling and segmentation to detect micro air vehicles using RGB-D sensor,” *Int. J. Micro Air Veh.*, vol. 11, p. 1756829318822327, 2019.

[4] S. Yong, J. D. Deng, and M. K. Purvis, “Modelling semantic context for novelty detection in wildlife scenes,” in *2010 IEEE International Conference on Multimedia and Expo*, 2010, pp. 1254–1259.

[5] N. Friedman and S. J. Russell, “Image Segmentation in Video Sequences: {A} Probabilistic Approach,” *CoRR*, vol. abs/1302.1, 2013.

[6] C. Cuevas, R. Martínez, and N. García, “Detection of stationary foreground objects : A survey,” *Comput. Vis. Image Underst.*, vol. 152, pp. 41–57, 2016.

[7] G. Moyà-Alcover, A. Elgammal, A. Jaume-i-Capó, and J. Varona, “Modeling depth for nonparametric foreground segmentation using RGBD devices,” *Pattern Recognit. Lett.*, vol. 96, no. C, pp. 76–85, Sep. 2017.

[8] O. Barnich and M. Van Droogenbroeck, “ViBe : A Universal Background Subtraction Algorithm for Video Sequences,” *IEEE Trans. Image Process. vol. 20, no. 6*, vol. 20, no. JULY 2011, pp. 1709–1724.

[9] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, 1999, vol. 2, no. c, pp. 246-252 Vol. 2.

[10] A. Benamara, S. Miguet, and M. Scuturici, “Real-Time Multi-object Tracking with Occlusion and Stationary Objects Handling for Conveying Systems,” in *Advances in Visual Computing*, 2016, pp. 136–145.

[11] S. Piskorski, N. Brulez, P. Eline, and F. D’Haeyer, “Ar. drone developer guide,” *Parrot, sdk*, 2012.

[12] “Crazyflie 2.0 | Bitcraze.” [Online]. Available: https://www.bitcraze.io/crazyflie-2/. [Accessed: 15-May-2017].

[13] Z. Zivkovic, “Improved adaptive Gaussian mixture model for background subtraction,” *Proc. 17th Int. Conf. Pattern Recognition, 2004. ICPR 2004.*, vol. 2, pp. 28–31, 2004.

[14] M. Hofmann, P. Tiefenbacher, and G. Rigoll, “Background segmentation with feedback: The Pixel-Based Adaptive Segmenter,” in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 38–43.

[15] N. Goyette, P. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, “A Novel Video Dataset for Change Detection Benchmarking,” *IEEE Trans. Image Process.*, vol. 23, no. 11, pp. 4663–4679, Nov. 2014.

[16] J. Baek, S. Park, B. Cho, and M. Lee, “Position Tracking System using Single RGB-D Camera for Evaluation of Multi-Rotor UAV Control and Self-Localization,” vol. 735, pp. 1283–1288, 2015.

[17] A. Carrio, S. Vemprala, A. Ripoll, S. Saripalli, and P. Campoy, “Drone Detection Using Depth Maps,” *CoRR*, vol. abs/1808.0, Aug. 2018.

[18] Y.-L. Hou and G. K. H. Pang, “People Counting and Human Detection in a Challenging Situation,” *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 41, no. 1, pp. 24–33, Jan. 2011.

[19] S.-P. Yong, J. D. Deng, and M. K. Purvis, “Novelty detection in wildlife scenes through semantic context modelling,” *Pattern Recognit.*, vol. 45, no. 9, pp. 3439–3450, Sep. 2012.

[20] B. K. P. Horn and B. G. Schunck, “Determining optical flow,” *Artif. Intell.*, vol. 17, no. 1, pp. 185–203, 1981.

[21] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, “High Accuracy Optical Flow Estimation Based on a Theory for Warping,” Springer Berlin Heidelberg, 2004, pp. 25–36.

[22] Z. Tang, Z. Miao, and Y. Wan, “Background Subtraction Using Running Gaussian Average and Frame Difference,” in *Entertainment Computing – ICEC 2007*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 411–414.

[23] T. D. Papageorgiou, W. A. Curtis, M. McHenry, and S. M. LaConte, “Neurofeedback of two motor functions using supervised learning-based real-time functional magnetic resonance imaging,” in *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2009, pp. 5377–5380.

[24] S. S. Cheung and C. Kamath, “Robust techniques for background subtraction in urban traffic video,” in *Proc. SPIE 5308*, p. 881.

[25] C. Zhan, X. Duan, S. Xu, Z. Song, and M. Luo, “An Improved Moving Object Detection Algorithm Based on Frame Difference and Edge Detection,” in *Fourth International Conference on Image and Graphics (ICIG 2007)*, 2007, pp. 519–523.

[26] P. A. K. Chun-hyok, Z. Hai, Z. H. U. Hongbo, and P. A. N. Yilin, “A Novel Motion Detection Approach Based on the Improved ViBe Algorithm,” *2016 Chinese Control Decis. Conf. (CCDC), Yinchuan*, pp. 7081–7086, 2016.

[27] B. Stenger, V. Ramesh, N. Paragios, F. Coetzee, and J. M. Buhmann, “Topology free hidden Markov models: application to background modeling,” in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 1, pp. 294–301.

[28] D. Comaniciu and P. Meer, “Robust analysis of feature spaces: color image segmentation,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 750–755.

[29] A. Elgammal, D. Harwood, and L. Davis, “Non-parametric model for background subtraction,” *Comput. Vision—ECCV 2000*, vol. 1843, pp. 751–767, 2000.

[30] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, “Pfinder: real-time tracking of the human body,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 780–785, Jul. 1997.

[31] A. Cavallaro, O. Steiger, and T. Ebrahimi, “Semantic video analysis for adaptive content delivery and automatic description,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 10, pp. 1200–1209, Oct. 2005.

[32] Z. Zivkovic and F. van der Heijden, “Efficient adaptive density estimation per image pixel for the task of background subtraction,” *Pattern Recognit. Lett.*, vol. 27, no. 7, pp. 773–780, 2006.

[33] L. Sharma and P. K. Garg, *From Visual Surveillance to Internet of Things : Technology and Applications.* 2019.

[34] D. Zeng and M. Zhu, “Background Subtraction Using Multiscale Fully Convolutional Network,” *IEEE Access*, vol. 6, pp. 16010–16021, 2018.

[35] T. Bouwmans, “Traditional and Recent Approaches in Background Modeling for Foreground Detection : An Overview,” *Comput. Sci. Rev.*, vol. 11, no. May, pp. 31–66, 2014.

[36] R. Heras Evangelio and T. Sikora, “Static Object Detection Based on a Dual Background Model and a Finite-State Machine,” *EURASIP J. Image Video Process.*, vol. 2011, no. 1, p. 858502, Dec. 2010.

[37] A. Sobral and A. Vacavant, “A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos,” *Comput. Vis. Image Underst.*, vol. 122, pp. 4–21, 2014.

[38] L. Maddalena and A. Petrosino, “Background Subtraction for Moving Object Detection in RGBD Data: A Survey,” *J. Imaging*, vol. 4, no. 5, 2018.

[39] L. Maddalena and A. Petrosino, “A Self-Organizing Approach to Background Subtraction for Visual Surveillance Applications,” *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1168–1177, Jul. 2008.

[40] C. Cuevas and N. Garcia, “Efficient Moving Object Detection for Lightweight Applications on Smart Cameras,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 1, pp. 1–14, Jan. 2013.

[41] R. Martínez, C. Cuevas, D. Berjón, and N. García, “Detection of static moving objects using multiple nonparametric background models,” pp. 4–5, 2015.

[42] F. Porikli, Y. Ivanov, and T. Haga, “Robust Abandoned Object Detection Using Dual Foregrounds,” *EURASIP J. Adv. Signal Process.*, vol. 2008, no. 1, p. 197875, Oct. 2007.

[43] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, “Background modeling and subtraction by codebook construction,” in *Proceedings - International Conference on Image Processing, ICIP*, 2004, vol. 2, pp. 3061–3064.

[44] H. Wang and D. Suter, “A consensus-based method for tracking: Modelling background scenario and foreground appearance,” *Pattern Recognit.*, vol. 40, no. 3, pp. 1091–1105, 2007.

[45] G. Gordon, T. Darrell, M. Harville, and J. Woodfill, “Background estimation and removal based on range and color,” in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, 1999, vol. 2, pp. 459–464.

[46] I. Schiller and R. Koch, “Improved Video Segmentation by Adaptive Combination of Depth Keying and Mixture-of-Gaussians,” in *Image Analysis*, 2011, pp. 59–68.

[47] A. Stormer, M. Hofmann, and G. Rigoll, “Depth gradient based segmentation of overlapping foreground objects in range images,” in *2010 13th International Conference on Information Fusion*, 2010, pp. 1–4.

[48] M. Camplani and L. Salgado, “Background foreground segmentation with RGB-D Kinect data: An efficient combination of classifiers,” *J. Vis. Commun. Image Represent.*, vol. 25, no. 1, pp. 122–136, 2014.

[49] Y. Wang, P. Jodoin, F. Porikli, J. Konrad, Y. Benezeth, and P. Ishwar, “CDnet 2014: An Expanded Change Detection Benchmark Dataset,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 393–400.

[50] Z. Zhong, J. Wen, B. Zhang, and Y. Xu, “A general moving detection method using dual-target nonparametric background model,” *Knowledge-Based Syst.*, vol. 164, pp. 85–95, 2019.

[51] J. Leens, S. Piérard, O. Barnich, M. Van Droogenbroeck, and J.-M. Wagner, “Combining Color, Depth, and Motion for Video Segmentation,” Springer Berlin Heidelberg, 2009, pp. 104–113.

[52] T. Minematsu, A. Shimada, H. Uchiyama, and R. Taniguchi, “Simple Combination of Appearance and Depth for Foreground Segmentation,” Springer, Cham, 2017, pp. 266–277.

[53] L. Maddalena and A. Petrosino, “Exploiting Color and Depth for Background Subtraction,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2017, vol. 10590 LNCS, pp. 254–265.

[54] E. J. Fernandez-Sanchez, J. Diaz, and E. Ros, “Background subtraction based on color and depth using active sensors.,” *Sensors (Basel).*, vol. 13, no. 7, pp. 8895–8915, 2013.

[55] A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis, “Background and foreground modeling using nonparametric kernel density estimation for visual surveillance,” *Proc. IEEE*, vol. 90, no. 7, pp. 1151–1163, Jul. 2002.

[56] B. Han, D. Comaniciu, and L. Davis, “Sequential kernel density approximation through mode propagation: Applications to background modeling,” *Proc. ACCV, vol. 1*, 2004.

[57] Y. Sheikh and M. Shah, “Bayesian modeling of dynamic scenes for object detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 11, pp. 1778–1792, 2005.

[58] Y. Xu, J. Dong, B. Zhang, and D. Xu, “Background modeling methods in video analysis: A review and comparative evaluation,” *CAAI Trans. Intell. Technol.*, vol. 1, no. 1, pp. 43–60, 2016.

[59] T. Bouwmans, S. Javed, M. Sultana, and S. K. Jung, “Deep neural network concepts

for background subtraction:A systematic review and comparative evaluation," *Neural Networks*, vol. 117, pp. 8–66, Sep. 2019.

[60] J. Wen, Y. Xu, J. Tang, Y. Zhan, Z. Lai, and X. Guo, "Joint Video Frame Set Division and Low-Rank Decomposition for Background Subtraction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 12, pp. 2034–2048, Dec. 2014.

[61] M. Van Droogenbroeck and O. Paquot, "Background subtraction: Experiments and improvements for ViBe," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 32–37.

[62] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.net: A new change detection benchmark dataset," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 1–8.

[63] G. Bo, S. Kefeng, Q. Daoyin, and Z. Hongtao, "Moving Object Detection Based on Improved ViBe Algorithm," vol. 9, no. 12, pp. 225–232, 2015.

[64] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1337–1342, Oct. 2003.

[65] H. Wang and L. Shi, "Foreground Model for Background Subtraction with Blind Updating," pp. 74–78, 2016.

[66] B. Nyan and S. Grünwedel, "PhD Forum: illumination-robust foreground detection for multi-camera occupancy mapping," *6th ACM/IEEE …*, pp. 5–6, 2012.

[67] S. Gruenwedel, P. Van Hese, and W. Philips, "An Edge-Based Approach for Robust Foreground Detection," in *Advanced Concepts for Intelligent Vision Systems*, 2011, pp. 554–565.

[68] J. R. Rabha, "Background Modelling by Codebook Technique for Automated Video Surveillance with Shadow Removal," *2015 IEEE Int. Conf. Signal Image Process. Appl.*, pp. 584–589, 2015.

[69] I. Huerta, M. B. Holte, T. B. Moeslund, and J. Gonzàlez, "Chromatic shadow detection and tracking for moving foreground segmentation," *Image Vis. Comput.*, vol. 41, pp. 42–53, 2015.

[70] Z. Chen, Y. Zhao, X. Huang, X. Liang, Y. Yuan, and X. Hu, "An improved shadow removal algorithm based on gradient amendment," *2014 12th Int. Conf. Signal Process. (ICSP), Hangzhou*, vol. 2015-Janua, no. October, pp. 1190–1194, 2014.

[71] Y. Dong and G. N. Desouza, "Adaptive learning of multi-subspace for foreground detection under illumination changes," *Comput. Vis. Image Underst.*, vol. 115, no. 1, pp. 31–49, Jan. 2011.

[72] M. Shakeri and H. Zhang, "Object detection using a moving camera under sudden illumination change," *Proc. 32nd Chinese Control Conf.*, pp. 4001–4006, 2013.

[73] H. Wang, Q. Wang, Y. Li, Y. Liu, and Y. Dai, "An illumination-robust algorithm based on visual background extractor for moving object detection," *2015 10th Asian Control Conf. Emerg. Control Tech. a Sustain. World, ASCC 2015*, 2015.

[74] M. Rogez, L. Tougne, and L. Robinault, "A Prior-knowledge based Casted Shadows

Preciction Model Featuring OpenStreetMap Data," *VISAPP 2013 - Proc. Int. Conf. Comput. Vis. Theory Appl.*, vol. 1, pp. 602–607, 2013.

[75] X. Hu and P. Mordohai, "A Quantitative Evaluation of Confidence Measures for Stereo Vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2121–2133, 2012.

[76] M. Cristani, M. Farenzena, D. Bloisi, and V. Murino, "Background subtraction for automated multisensor surveillance: A comprehensive review," *EURASIP J. Adv. Signal Process.*, vol. 2010, p. 343057, 2010.

[77] A. Kolb, E. Barth, R. Koch, and R. Larsen, "Time-of-Flight Cameras in Computer Graphics," *Comput. Graph. Forum*, vol. 29, pp. 141–159, 2010.

[78] M. Daneshmand *et al.*, *3D Scanning: A Comprehensive Survey*. 2018.

[79] C. Sun, Y. Wang, and M. Sheu, "Fast Motion Object Detection Algorithm Using Complementary Depth Image on an RGB-D Camera," *IEEE Sens. J.*, vol. 17, no. 17, pp. 5728–5734, 2017.

[80] S. Lee *et al.*, "Detection of a Suicide by Hanging Based on a 3-D Image Analysis," *IEEE Sens. J.*, vol. 14, no. 9, pp. 2934–2935, 2014.

[81] J. Hernández-Aceituno, R. Arnay, J. Toledo, and L. Acosta, "Using Kinect on an Autonomous Vehicle for Outdoors Obstacle Detection," *IEEE Sens. J.*, vol. 16, no. 10, pp. 3603–3610, May 2016.

[82] H. Sarbolandi, D. Lefloch, and A. Kolb, "Kinect range sensing: Structured-light versus Time-of-Flight Kinect," *Comput. Vis. Image Underst.*, vol. 139, pp. 1–20, 2015.

[83] Z. Cai, J. Han, L. Liu, and L. Shao, "RGB-D datasets using microsoft kinect or similar sensors: a survey," *Multimed. Tools Appl.*, vol. 76, no. 3, pp. 4313–4355, Feb. 2017.

[84] M. J. Dahan, N. Chen, A. Shamir, and D. Cohen-Or, "Combining color and depth for enhanced image segmentation and retargeting," *Vis. Comput.*, vol. 28, no. 12, pp. 1181–1193, Dec. 2011.

[85] E. Mirante, M. Georgiev, and A. Gotchev, "A fast image segmentation algorithm using color and depth map," in *2011 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, 2011, pp. 1–4.

[86] S. Ottonelli, P. Spagnolo, P. L. Mazzeo, and M. Leo, "Improved video segmentation with color and depth using a stereo camera," *2013 IEEE Int. Conf. Ind. Technol.*, pp. 1134–1139, Feb. 2012.

[87] A. Bleiweiss and M. Werman, "Fusing time-of-flight depth and color for real-time segmentation and tracking," 2009, vol. 5742 LNCS, pp. 58–69.

[88] M. Camplani, C. R. Del Blanco, L. Salgado, F. Jaureguizar, and N. García, "Advanced background modeling with RGB-D sensors through classifiers combination and inter-frame foreground prediction," *Mach. Vis. Appl.*, vol. 25, no. 5, pp. 1197–1210, 2014.

[89] E. Francois and B. Chupeau, "Depth-based segmentation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 1, pp. 237–240, 1997.

[90] N. D. Doulamis, A. D. Doulamis, Y. S. Avrithis, K. S. Ntalianis, and S. D. Kollias, "Efficient summarization of stereoscopic video sequences," *IEEE Trans. Circuits Syst.*

Video Technol., vol. 10, no. 4, pp. 501–517, Jun. 2000.

[91] M. Braham, A. Lejeune, and M. Van Droogenbroeck, “A physically motivated pixel-based model for background subtraction in 3D images,” *2014 Int. Conf. 3D Imaging, IC3D 2014 - Proc.*, 2015.

[92] X. Wang, L. Liu, G. Li, X. Dong, P. Zhao, and X. Feng, “Background subtraction on depth videos with convolutional neural networks,” *CoRR*, vol. abs/1901.0, 2019.

[93] Y. Sun, M. Liu, and M. Q.-. Meng, “Active Perception for Foreground Segmentation: An RGB-D Data-Based Background Modeling Method,” *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 4, pp. 1596–1609, Oct. 2019.

[94] L. Maddalena and A. Petrosino, “Self-organizing background subtraction using color and depth data,” *Multimed. Tools Appl.*, vol. 78, no. 9, pp. 11927–11948, May 2019.

[95] L. Maddalena and A. Petrosino, “The SOBS algorithm: What are the limits?,” in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 21–26.

[96] S. Pierard and M. Van Droogenbroeck, “Techniques to improve the foreground segmentation with a 3D camera and a color camera,” *20th Annu. Work. Circuits, Syst. Signal Process.*, pp. 247–250, 2009.

[97] J. Huang, H. Wu, Y. Gong, and D. Gao, “Random sampling-based background subtraction with adaptive multi-cue fusion in RGBD videos,” in *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 2016, pp. 30–35.

[98] X. Zhou, X. Liu, A. Jiang, B. Yan, and C. Yang, “Improving Video Segmentation by Fusing Depth Cues and the Visual Background Extractor (ViBe) Algorithm,” *Sensors (Basel).*, vol. 17, no. 5, p. 1177, May 2017.

[99] J. Canny, “A Computational Approach to Edge Detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[100] M. Camplani, C. R. del Blanco, L. Salgado, F. Jaureguizar, and N. García, “Multi-sensor background subtraction by fusing multiple region-based probabilistic classifiers,” *Pattern Recognit. Lett.*, vol. 50, no. C, pp. 23–33, Dec. 2014.

[101] H. Chao, Y. Cao, and Y. Chen, “Autopilots for small unmanned aerial vehicles: A survey,” *Int. J. Control. Autom. Syst.*, vol. 8, no. 1, pp. 36–44, Feb. 2010.

[102] A. Mashood, H. Noura, I. Jawhar, and N. Mohamed, “A gesture based kinect for quadrotor control,” in *2015 International Conference on Information and Communication Technology Research (ICTRC)*, 2015, pp. 298–301.

[103] H. Eisenbeiss, “A MINI UNMANNED AERIAL VEHICLE (UAV) : SYSTEM OVERVIEW AND IMAGE ACQUISITION,” 2004.

[104] F. Arifin, R. A. Daniel, and D. Widiyanto, “Autonomous Detection and Tracking of an Object Autonomously Using,” *J. Ilmu Komput. dan Inf.*, pp. 11–17, 2014.

[105] M. Muller, S. Lupashin, and R. D’Andrea, “Quadrocopter ball juggling,” in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 5113–5120.

[106] H. F. Durrant-Whyte, N. Roy, and P. Abbeel, *Robotics : science and systems VII*. MIT Press, 2012.

[107] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, “RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments,” *Int. J. Rob. Res.*, vol. 31, no. 5, pp. 647–663, Apr. 2012.

[108] D. Li, Q. Li, N. Cheng, Q. Wu, J. Song, and L. Tang, “Combined RGBD-inertial based state estimation for MAV in GPS-denied indoor environments,” in *2013 9th Asian Control Conference (ASCC)*, 2013, pp. 1–8.

[109] M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza, “Autonomous, Vision-based Flight and Live Dense 3D Mapping with a Quadrotor Micro Aerial Vehicle,” *J. F. Robot.*, vol. 33, no. 4, pp. 431–450, Jun. 2016.

[110] L. V. Santana, M. Sarcinelli-Filho, and R. Carelli, “Estimation and control of the 3D position of a quadrotor in indoor environments,” in *2013 16th International Conference on Advanced Robotics (ICAR)*, 2013, pp. 1–6.

[111] M. Camplani and L. Salgado, “Efficient spatio-temporal hole filling strategy for Kinect depth maps,” in *Proc. SPIE 8290, Three-Dimensional Image Processing (3DIP) and Applications II*, 2012, p. 82900E.

[112] C.-T. Hsieh, Chieh-Fu;Yih, Chi-Hsiao;Hsieh, “An Improved Depth Image Inpainting,” in *Proceedings of International Research Conference on Information Technology and Computer Sciences (IRCITCS) held on 28-29 September 2013 in Kuala Lumpur].*, 2014, pp. 15–23.

[113] L. Yang, L. Zhang, H. Dong, A. Alelaiwi, and A. El Saddik, “Evaluating and Improving the Depth Accuracy of Kinect for Windows v2,” *IEEE Sens. J.*, vol. 15, no. 8, pp. 4275–4285, 2015.

[114] A. M. Pinto, P. Costa, A. P. Moreira, L. F. Rocha, E. Moreira, and G. Veiga, “Evaluation of depth sensors for robotic applications,” *Proc. - 2015 IEEE Int. Conf. Auton. Robot Syst. Compet. ICARSC 2015*, pp. 139–143, 2015.

[115] N. Dorudian, “Quadcopter Detection,” 2018. [Online]. Available: https://figshare.com/s/deed56cf5dd1c333fedf.

[116] M. Camplani, L. Maddalena, G. Moyá Alcover, A. Petrosino, and L. Salgado, “A Benchmarking Framework for Background Subtraction in RGBD videos, in S. Battiato, G. Gallo, G.M. Farinella, M. Leo (Eds), New Trends in Image Analysis and Processing-ICIAP 2017 Worksho,” *Lect. Notes Comput. Sci. Springer, 2017*, vol. 10590 LNCS, pp. 219–229, Sep. 2017.

[117] R. Real, J. M. Vargas, and R. Olmstead, “The Probabilistic Basis of Jaccard’s Index of Similarity,” *Syst. Biol.*, vol. 45, no. 3, pp. 380–385, Sep. 1996.

[118] L. Li, W. Huang, I. Y.-H. Gu, and Q. Tian, “Statistical Modeling of Complex Backgrounds for Foreground Object Detection,” *IEEE Trans. Image Process.*, vol. 13, no. 11, pp. 1459–1472, Nov. 2004.

[119] I. Intel Corporation, Willow Garage, “Open Source Computer Vision Library,” 2018. [Online]. Available: http://opencv.org.

[120] M. C. P. Santos, L. V. Santana, M. M. Martins, A. S. Brandao, and M. Sarcinelli-

Filho, "Estimating and controlling UAV position using RGB-D/IMU data fusion with decentralized information/Kalman filter," in *2015 IEEE International Conference on Industrial Technology (ICIT),* 2015, pp. 232–239.

[121] A. Gongora and J. Gonzalez-Jimenez, "Enhancement of a commercial multicopter for research in autonomous navigation," in *2015 23rd Mediterranean Conference on Control and Automation (MED),* 2015, pp. 1204–1209.

[122] N. Dorudian, S. Lauria, and S. Swift, "Moving Object Detection using Adaptive Blind Update and RGB-D Camera," *IEEE Sens. J.*, vol. 19, no. 18, pp. 8191–8201, 2019.

[123] D. Berjón, C. Cuevas, F. Morán, and N. García, "Real-time nonparametric background subtraction with tracking-based foreground update," *Pattern Recognit.,* vol. 74, pp. 156–170, 2018.

[124] S. Javed, T. Bouwmans, M. Sultana, and S. K. Jung, "Moving Object Detection on RGB-D Videos Using Graph Regularized Spatiotemporal RPCA," Springer, Cham, 2017, pp. 230–241.

[125] R. S. Hunter and R. W. Harold, *The measurement of appearance*. Wiley, 1987.

[126] P. R. Narkhede and A. V Gokhale, "Color image segmentation using edge detection and seeded region growing approach for CIELab and HSV color spaces," in *2015 International Conference on Industrial Instrumentation and Control (ICIC),* 2015, pp. 1214–1218.

[127] and A. H. K. S. Wah, "Robust human detection within a highly dynamic aquatic environment in real time," *IEEE Trans. Image Process.,* vol. 15, no. 6, pp. 1583–1600, Jun. 2006.

[128] Y. Wang, C. Hsieh, C. Han, and K. Fan, "The color identification of automobiles for video surveillance," in *2011 Carnahan Conference on Security Technology,* 2011, pp. 1–5.

[129] L. Maddalena, "Results of the SBM-RGBD Challenge," 2018. [Online]. Available: http://rgbd2017.na.icar.cnr.it/SBM-RGBDchallengeResults.html. [Accessed: 01-Aug-2018].

[130] M. De Gregorio and M. Giordano, "CwisarDH+ : Background Detection in RGBD Videos by Learning of Weightless Neural Networks," Springer, Cham, 2017, pp. 242–253.

[131] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05),* vol. 1, pp. 886–893.

[132] O. Banimelhem and Y. Ahmed Yahya, "Multi-Thresholding Image Segmentation Using Genetic Algorithm," *World Congr. Comput. Sci. Comput. Eng. Appl. Comput.,* no. April 2012, 2012.

[133] J. Dey and N. Praveen, "Moving object detection using genetic algorithm for traffic surveillance," in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT),* 2016, pp. 2289–2293.

[134] M. Negnevitsky, *Artificial intelligence: a guide to intelligent systems. Pearson Education.* Addison-Wesley, 2005.

[135] Y. Liang, M. Zhang, and W. N. Browne, “Image segmentation: A survey of methods based on evolutionary computation,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics),* vol. 8886, pp. 847–859, 2014.

[136] R. V. Rao and V. J. Savsani, *Mechanical Design Optimization Using Advanced Optimization Techniques*. London: Springer London, 2012.

[137] T. Bäck and H. Schwefel, “An Overview of Evolutionary Algorithms for Parameter Optimization,” *Evol. Comput.*, vol. 1, no. 1, pp. 1–23, 1993.

[138] M. I. Heywood, “Evolutionary model building under streaming data for classification tasks: opportunities and challenges,” *Genet. Program. Evolvable Mach.*, vol. 16, pp. 1–44, 2014.

[139] I. Fister jr, X.-S. Yang, I. Fister, J. Brest, and D. Fister, “A Brief Review of Nature-Inspired Algorithms for Optimization,” *Elektroteh. Vestnik/Electrotechnical Rev.*, vol. 80, 2013.

[140] K. Lwin *et al.*, “Sea Docking by Dual-eye Pose Estimation with Optimized Genetic Algorithm Parameters,” *J. Intell. Robot. Syst.*, 2019.

[141] J. Chae, Y. Jin, Y. Sung, and K. Cho, “Genetic Algorithm-Based Motion Estimation Method using Orientations and EMGs for Robot Controls,” *Sensors (Basel).*, vol. 18, 2018.

[142] B. Bhanu, S. Lee, and J. Ming, “Adaptive image segmentation using a genetic algorithm,” *IEEE Trans. Syst. Man. Cybern.*, vol. 25, no. 12, pp. 1543–1567, Dec. 1995.

[143] T. T. San and N. War, “Stereo matching algorithm by hill-climbing segmentation,” in *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)*, 2017, pp. 1–2.

[144] T. Ohashi, Z. Al Aghbari, and A. Makinouchi, “Hill-climbing Algorithm for Efficient Color-based Image Segmentation,” *Signal Process. Pattern Recognition, Appl.*, 2003.

[145] N. Dorudian, S. Lauria, and S. Swift, “Background modelling and segmentation using a Genetic Algorithm and Hill Climbing,” *(to be Submitt).*

[146] R. Chopra, *Artificial intelligence : a practical approach.* S Chand, 2012.

[147] S. K. N. Abdul Rahim, A. Bargiela, and R. Qu, “Hill Climbing versus genetic algorithm optimization in solving the examination timetabling problem,” *ICORES 2013 - Proc. 2nd Int. Conf. Oper. Res. Enterp. Syst.*, pp. 43–52, 2013.

[148] A. Prügel-Bennett, “When a genetic algorithm outperforms hill-climbing,” *Theor. Comput. Sci.*, vol. 320, no. 1, pp. 135–153, 2004.

[149] M. Mitchell and J. Holland, “When Will a Genetic Algorithm Outperform Hill-Climbing?,” *St. Fe Institute, Work. Pap.*, 1993.

[150] P. F. Stadler, “Towards a theory of landscapes,” 1995.

[151] D. Whitley, A. Sutton, and A. Howe, “Understanding elementary landscapes,” in *GECCO’08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation 2008*, 2008, pp. 585–592.

[152] F. Chicano, L. D. Whitley, and E. Alba, “A Methodology to Find the Elementary

Landscape Decomposition of Combinatorial Optimization Problems," *Evol. Comput.*, vol. 19, no. 4, pp. 597–637, 2011.

APPENDIX A

C-LIKE SOURCE CODE FOR PROPOSED METHOD (MBNS)

Pseudo-code for the main part of our algorithm for grayscale depth and colour images.

default values for all the parameters of the algorithm is also given in the below code.

```
int width, height;
// Total number of samples
int N=20;
// Random frame frequency (blind update frequency)
int M=40;
// Minimum number of close samples
int # Min= N/4;
// Input Current Colour Image
byte ColourImage[width][height];
// Input Current Depth Image
byte DepthImage[width][height];
// Background Colour Model
byte ColourModel[N][width][height];
// Background Depth Model
byte DepthModel[N][width][height];
// Output Bg/Fg segmentation Mask
byte segMask[width][height];

byte background=0;
byte foreground=255;
int NoTolerance=5;
int colourTolerance =DepthTolerance=30;
int ADO = 650; // or 0


// For each pixel
for ( int i=0;i < width; i++)
   {
      int ioff= step*i;
      //compare with all pixels Models
```

```
 for (int j=0; j< height; j++)
 {
    int countColor =0, index=0, countDepth=0,
    countDepthNoTolerance=0;


// 1. Compare color and depth pixel to the background models
 while(index<N)
    {
       //difference of two colour pixels
       int dist= ColourModel[index][i][j]- ColourImage[i][j];
       if (dist<= ColorTolerance && dist>= ColorTolerance)
          countColor ++;

       //difference of two depth pixels
       dist= DepthImage [i][j]- DepthModel [index][i][j];
       if(Depthsample != ADO)
       {
          If (dist+ DepthTolerance >= 0 )
             countDepth ++;

          if (dist2+ NoTolrance > 0)
             countDepthNoTolerance++;
       }

       index++;
    }

 // 2. Classification

   bool isBackground=false;
    //If depth is ADO, Only rely on color frame
    if (DepthImage[i][j]== ADO) // 0 or 650
     {
       if(countColor>=# Min )
                  isBackground=true;
        else
                  isBackground=false;
     }

   //If depth is strongly saying is background then the system will accept it
   else if (countDepthNoTolerance > # Min)
        isBackground=true;

   // If depth is strongly saying is not background then the system will accept it
    else if (countDepth < # Min)
       isBackground=false;

   //All remaining pxiels will be decided by color frame
  else if ( countColor >= # Min)
```

```
          isBackground=true;

 //3. Update the model by background pixels

       if (isBackground)
       {
          segMask[i][j] =0;
          int SmallestDepthAmount=0;
          int SmallestDepthNumber=0;
       //find the smallest depth amount and the position in the model
         findThesmallestDepth( SmallestDepthAmount, SmallestDepthNumber);

           //randon number (0-N)
          rand= GetRandomNumber(0,N);
          //randomly swap the pixel with the model
            ColourModel[rand][i][ j]= ColourImage[i][j];

         If ( (SmallestDepthAmount <  DepthImage[i][j])  && (DepthImage[j][j] !=  ADO) )
            DepthModel [SmallestDepthNumber][i][ j]= DepthImage [i][j];

       }
      else
          segMask[i][j] =255;

        //4. Blind randomly update the models

        //Update after N number of frame
       if (FrameNumber%N == 0)
       {
        // replace randomly chosen sample
        rand= GetRandomNumber(0,N);
        ColourModel[rand][i][ j]= ColourImage[i][j];

        If (DepthImage != ADO)
        DepthModel [rand][i][ j]= DepthImage [i][j];
        }
}
```