STATUS OF THESIS

Title of thesis | 3D VISUAL TRACKING USING A SINGLE CAMERA

I YASIR SALIH OSMAN ALI

hereby allow my thesis to be placed at the Information Resource Center (IRC) of Universiti Teknologi PETRONAS (UTP) with the following conditions:

1. The thesis becomes the property of UTP
2. The IRC of UTP may make copies of the thesis for academic purposes only.
3. This thesis is classified as

☐ Confidential

☑ Non-confidential

If this thesis is confidential, please state the reason:

_____
_____
_____
_____

The contents of the thesis will remain confidential for _____ years.

Remarks on disclosure:

_____
_____
_____
_____

Endorsed by

_____ _____

Signature of Author | Signature of Supervisor

Permanent address:
Alshabiya District, 23-159
Kassala East, Kassala state
11111, Sudan

Name of Supervisor
Dr. Aamir Saeed Malik

Date : _____ Date : _____

UNIVERSITI TEKNOLOGI PETRONAS

3D VISUAL TRACKING USING A SINGLE CAMERA

by

YASIR SALIH OSMAN ALI

The undersigned certify that they have read, and recommend to the Postgraduate Studies Programme for acceptance this thesis for the fulfillment of the requirements for the degree stated.

Signature: _____

Main Supervisor: Dr. Aamir Saeed Malik

Signature: _____

Co-Supervisor: Ms. Zazilah May

Signature: _____

Head of Department: Assoc. Prof. Dr. Nor Hisham Hamid

Date: _____

3D VISUAL TRACKING USING A SINGLE CAMERA

by

YASIR SALIH OSMAN ALI

A Thesis

Submitted to the Postgraduate Studies Programme

as a Requirement for the Degree of

MASTER OF SCIENCE

ELECTRICAL AND ELECTRONICS ENGINEERING DEPARTMENT

UNIVERSITI TEKNOLOGI PETRONAS

BANDAR SERI ISKANDAR,

PERAK

SEPTEMBER 2011

DECLARATION OF THESIS

Title of thesis | 3D VISUAL TRACKING USING A SINGLE CAMERA

I YASIR SALIH OSMAN ALI

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

Witnessed by

_____ _____

Signature of Author | Signature of Supervisor

Permanent address:
Alshabiya District, 23-159
Kassala East, Kassala state
11111, Sudan

Name of Supervisor
Dr. Aamir Saeed Malik

Date : _____ Date : _____

DEDICATION

To my family and friends

ACKNOWLEDGEMENTS

I start by the name of Allah the Almighty who have given the strength and determination to complete this job. My sincere prayers are upon his Messenger who urges his follower to seek knowledge from cradle to grave. Firstly, I would like to thank the Almighty God for blessing me to be at this stage and enabling me to complete this job despite the difficulties. My Allah accepts this work and makes it sincere to His face for the benefit of humanity.

I would like to express my utmost gratitude to my supervisor Dr. Aamir Malik for his continuous stream of ideas that kept me occupied. In addition, he was very instrumental in shaping my research directions. He has been very helpful and giving his students opportunities to explore the global space of research by acquiring state of the art equipments and initiating collaboration with industrial partners.

I also would like to thank my co-supervisor Ms. Zazilah May for her continuous support. The gratitude is extended to Dr. Nidal Kamel and Dr. Vijanth for their ideas. I would also like to thank my colleague in the CISIR for their support. Special thank is given to Jawad, Rauf, Linie and Roushanak. A special thank is also extended to my friends and country-mate; especially Khalid Izzudin, Tag, Huziafa and Hamada who were supportive throughout this research.

I would like to thank Univeristi Teknologi PETRNAS for supporting this research through the graduate assistantship scheme. This thesis was support by the Ministry of Science, Technology and Innovation (MOSTI), Government of Malaysia (Fund No: 01-02-02-SF0064). Part of this research was supported by the Short Term Internal Research Fund (STIRF No. 42/09.10) provided by Research Enterprise Office, Universiti Teknologi PETRONAS.

Finally, I would like to thank my parents and my grandmother for their continuous support despite the distance. Their prayer was instrumental for completing this work. I would also like to thank my brother and sister for cheering me up. A special gratitude is given to my uncle how was instrumental in my life; from cradle up to this moment. My Allah blesses them all.

ABSTRACT

Tracking is an important application in computer vision and it is used in automated surveillance and motion based recognition. 3D tracking address the localization of moving target is the 3D space. Therefore, 3D tracking requires 3D measurement of the moving object which cannot be obtained from 2D cameras. Existing 3D tracking systems use multiple cameras for computing the depth of field and it is only used in research laboratories. Millions of surveillance cameras are installed worldwide and all of them capture 2D images. Therefore, 3D tracking cannot be performed with these cameras unless multiple cameras are installed at each location in order to compute the depth. This means installing millions of new cameras which is not a feasible solution.

This work introduces a novel depth estimation method from a single 2D image using triangulation. This method computes the absolute depth of field for any object in the scene with high accuracy and short computational time. The developed method is used for performing 3D visual tracking using a single camera by providing the depth of field and ground coordinates of the moving object for each frame accurately and efficiently. Therefore, this technique can help in transforming existing 2D tracking and 2D video analytics into 3D without incurring additional costs. This makes video surveillance more efficient and increases its usage in human life.

The proposed methodology uses background subtraction process for detecting a moving object in the image. Then, the newly developed depth estimation method is used for computing the 3D measurement of the moving target. Finally, the unscented Kalman filter is used for tracking the moving object given the 3D measurement obtained by the triangulation method. This system has been test and validated using several video sequences and it shows good performance in term of accuracy and computational complexity.

ABSTRAK

Pengesanan ialah satu aplikasi penting dalam penglihatan komputer dan digunakan dalam pengawasan berautomasi serta pengecaman secara gerakan. Pengesanan 3D mengutarakan pertempatan objek bergerak di dalam ruangan 3D. Oleh itu, pengesanan 3D memerlukan ukuran 3D objek bergerak yang tidak boleh didapati daripada kamera 2D. Sistem pengesanan 3D sedia ada menggunakan sebilangan kamera untuk pengiraan kedalaman medan dan hanya digunakan di makmal-makmal penyelidikan. Berjuta-juta kamera pengawasan telah dipasang diseluruh dunia dan kesemuanya hanya mengambil gambar 2D. Oleh itu, pengesanan 3D tidak boleh dilakukan oleh kamera-kamera tersebut melainkan sebilangan kamera dipasangkan di setiap lokasi untuk mengira kedalaman. Ini bermakna, memasang berjuta-juta kamera baru yang merupakan penyelesaian yang tidak layak dilaksanakan.

Kajian ini memperkenalkan kaedah baru penganggaran kedalaman daripada satu gambar 2D menggunakan penyegitigaan. Kaedah ini mengira kedalaman medan mutlak untuk sebarang objek di dalam gambar dengan ketepatan yang tinggi dan masa pengiraan yang singkat. Kaedah yang dibangunkan ini digunakan untuk melaksanakan penglihatan pengesanan 3D dari satu kamera dengan memberikan kedalaman medan dan koordinat darat objek bergerak tersebut bagi setiap bingkai dengan tepat dan efisen. Oleh itu, teknik ini boleh membantu dalam mengubah pengesanan 2D sedia ada dan analitik video 2D ke dalam 3D tanpa melibatkan lebihan kos. Ini membuatkan pengawasan video lebih efisen dan menambah kepenggunaan dalam kehidupan manusia.

Kaedah yang diusulkan menggunakan proses pengurangan latarbelakang untuk mengesan objek bergerak di dalam gambar. Kemudian, kaedah penganggaran kedalaman yang baru dibangunkan telah digunakan untuk mengira ukuran 3D sasaran bergerak tersebut. Akhir sekali, unscented Kalman filter telah digunakan untuk

mengesan objek bergerak tersebut dengan syarat ukuran 3D telah diperolehi dengan kaedah penyegitigaan. Sistem ini telah diuji dan disahkan dengan menggunakan beberapa jujukan video dan menunjukkan prestasi yang bagus dalam erti ketepatan dan kerumitan pengiraan.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

LKF	Linear Kalman filter
EFK	Extended Kalman filter
UKF	Unscented Kalman filter
GPF	Generic particle filter
SIR	Sequential importance resampling
SIRPF	Sequential importance resampling particle filter
AuxPF	Auxiliary particle filter
SFF	Shape from focus
SFD	Shape from defocus
SFM	Structure from motion
SFT	Shape from texture
DfT	Depth from triangulation
SFS	Shape from shading
RBPF	Rao-Blackwellized particle filter
APF	Adaptive particle filer
ILW	Iterative likelihood weighting
GAP	Grayscale-arranging pairs
MRF	Markov random fields

LIST OF SYMBOLS

W_G	Global background modeling threshold using the GAP method
M	Number of frame used for background modeling
W_S	Neighboring pixel similarity threshold in the GAP algorithm
W_S	Probability of similar neighboring pixels
U	Total number of pixels in the image
W	Image width
H	Image height
$I(x, y)$	A pixel at location (x, y)
Ref^+	Reference points with intensity larger than target pixel
Ref^+	Reference points with intensity smaller than target pixel
$LumGain$	Luminance gain
$ChromaGain$	Chromaticity gain
$v(x, y)$	Velocity of the moving object at location in the direction of x
ϵ	Random noise for object velocity
θ	Pitch angle of the camera
h	Height of the camera
FOV_H	Horizontal field of view of the camera
FOV_V	Vertical field of view of the camera
ψ	Vertical angle of an object in the scene
ϕ	Horizontal angle of an object in the scene
X	X coordinate of an object in the scene
Y	Y coordinate of an object in the scene
Z	Depth of field of an object in the scene
x_{k+1}	Next state variable
z_k	Measurement variable
x_k	Current state of the variable

CHAPTER 1
INTRODUCTION

1.1 **Introduction**

Object tracking is an important component in computer vision and video analytics system. Recently, the need for an effective tracking system is driven by the availability of powerful and affordable computers and the increasing need for visual tracking applications for security and monitoring purposes. Tracking is the process of identifying the object trajectory in the scene while it undergoes motion. It provides information about the object location, shape, size orientation and the type of deformation it takes [1].

Normally, object tracking systems require measured inputs about the object status in every frame which is used to predict the status of the moving object in the next frame. The 3D tracking requires 3D measurement of the moving object parameters in every frame. This includes measuring the object location in the image as well as the depth of field which is not present in 2D images. Therefore most of the existing 3D tracking systems utilize multiple cameras for depth estimation which are only available in research laboratories but not as a commercial product.

In 2007, there were estimated around 21 million surveillance cameras installed worldwide which captures 2D images and they are mostly used for recording purposes [2]. Therefore, these cameras can only support 2D tracking system and 2D tracking does not provide sufficient information for video analytics. Thus multiple cameras are normally used for performing higher order video analytics such as behavior analysis and action recognition.

In this thesis, a new 3D tracking system is proposed that utilizes existing 2D surveillance camera installations. This system utilizes only 2D images from single camera to obtain 3D localization of moving object in real world coordinates. The

advantages of such system are that it does not require any additional cost and it can be added to the existing camera installations. Figure 1.1 shows a flow diagram for the proposed 3D tracking system. The flow starts with image acquisition from video sources which can be a 2D camera or even a video file. The moving object is detected in each image using detection algorithms. The depth of the moving object is computed next by extracting suitable depth cues from the image. Object detection information and the estimated depth map are combined together to give 3D localization of the moving object. Finally, a suitable tracking algorithm is used to estimate the new location of the moving object given its previous location and 3D measurements of current object location.

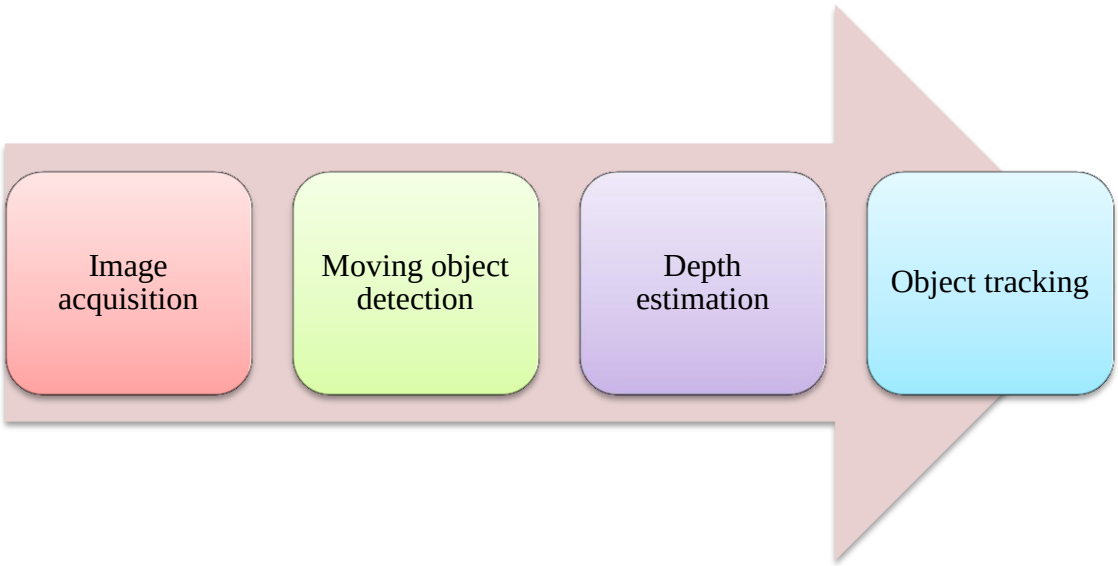


Figure 1.1: Flow diagram for 3D tracking system using single image.

1.2 **Applications of 3D Tracking**

Tracking is an important component in video analytics and it is used in many applications in computer vision. These applications range from security and entertainment to studying and analyzing human behaviors. In a survey conducted by Salih and Malik [3] (figure 1.2), security and monitoring dominates the research for 3D tracking and it represents more than 40% of the surveyed work in the period from 2006 – 2010. Human behavior analysis falls second (20%) based on the number of published work. Human tracking include pattern identification such as gait analysis

and abnormality detection. Traffic management, robot navigation and virtual reality applications each represents around 10% of the published works. The remaining of this section discusses some of the common applications of 3D visual tracking systems.

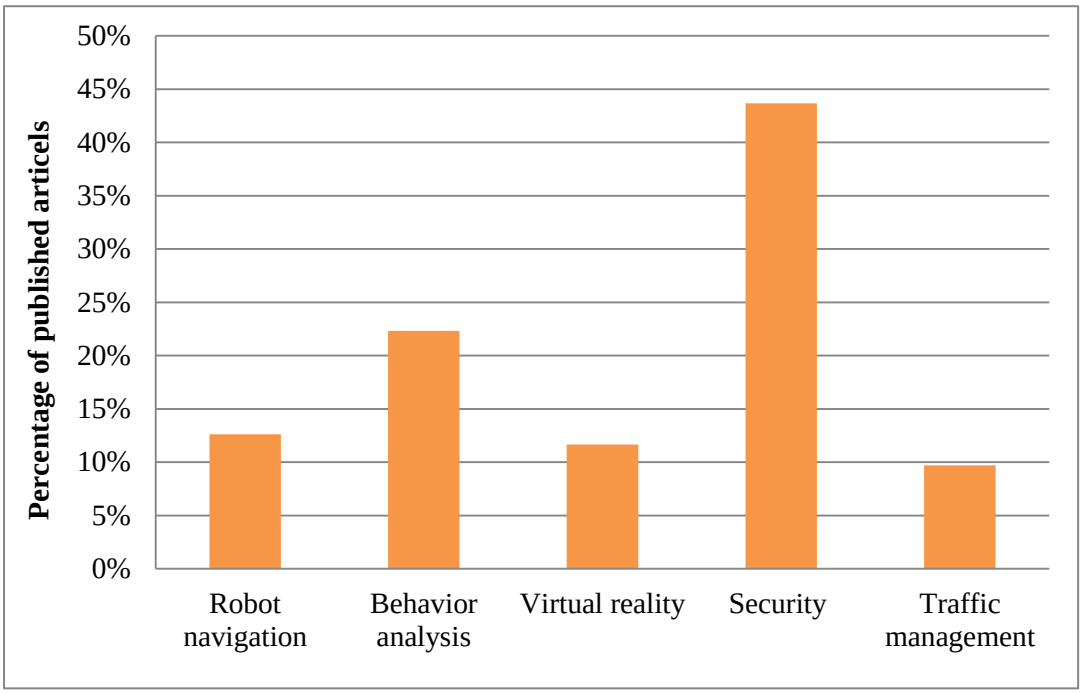


Figure 1.2 : Applications of object tracking in the published work for the period of (2006 – 2010).

Security and monitoring is the major application for 3D visual tracking. Tracking is used for identifying specious events such as unattended object, loitering and crowds in unwanted locations [4-6]. For example, Zigzag motion in a parking lot and stopping by many cars might give an indication to a possible car theft or someone who is lost and both cases require security attention. The second most common application for visual tracking is behavior analysis. Gait tracking is used as a biometrics which use for recognition. Gaze tracking can be used in shopping centers to identify products with high attractiveness without the need to conduct extensive surveys [7], [8].

Navigation of robots and unmanned vehicles are one of the important areas of 3D motion tracking. 3D tracking allows the machine to gauge the proximity of an obstacle and thus takes the correct decision for next movement [9]. Visual tracking is

used for monitoring of highways and transportation infrastructure is very useful for traffic management which reduces accident rates. An intelligent monitoring system is used to detect traffic jams and accordingly divert vehicles from congested roads to less crowded ones. In addition, vehicle tracking is used for accident prediction by identifying vehicle that moves in a wrong direction or a vehicle that suddenly stops in the middle of highway [10-13]. 3D tracking is used in computer games and virtual reality applications. For example, the Nintendo console Wii utilizes 3D tracking of IR light sources to localize the Wii Remote [14].

1.3 **Motivation**

In 2007, there were more than 21 million cameras installed worldwide and by this day this number might have doubled or tripled. Since most of these cameras capture 2D images, they cannot be used to perform 3D tracking using existing technologies because existing solution requires multiple cameras for 3D tracking [2]. As a result, developing a 3D visual tracking system that uses the existing 2D camera installations is more preferable than using the multiple cameras tracking system. In addition, 3D tracking system is required because of its demand in various applications including security and monitoring [15], traffic management and robot navigation. Moreover, providing a cheap 3D tracking system will surface as a gateway for new applications which expand to advertisement, medical diagnosis and health applications.

1.4 **Problem Statement**

Visual tracking of moving objects is an active area of research for a long time but it is all based on 2D imaging setup. In the recent years, 3D tracking has become popular because of the availability of computing power. However, existing 3D tracking systems are based on multiple cameras; mostly stereovision and it exist at research scale only. Tracking is used in surveillance industry for security and monitoring purposes. Almost all of the existing surveillance cameras capture 2D images. Therefore, 3D tracking system cannot be implemented on the existing surveillance

systems unless multiple cameras are used which require installing additional cameras in order to be able to extract 3D measurement of moving object.

1.5 **Research Objectives**

The objectives of this research can be summarized in the following points:

- To investigate the available 3D tracking algorithms and to select the most suitable one to be combined with proposed depth estimation method using a single camera.
- To develop a fast and efficient depth estimation method for moving objects using single image/camera.
- To compute the absolute 3D coordinate of a moving object in the scene given its location in the image.

1.6 **Research Scope**

This research can be categorized into three parts; the first part investigates the detection of moving objects in the scene and extracting the object of interest from the images. This section focuses on studying existing object detection schemes and selects the best one to be used in the proposed system. The second part focuses on developing a novel and fast depth estimation method for computing the depth of the detected object using geometrical relationship between the object and the camera model. The last part focuses on investigating the existing tracking algorithms specially the stochastic filters. Then the most appropriate algorithm is selected to be combined with the proposed depth estimation method.

1.7 **Thesis Outlines**

This thesis is organized as follows: chapter 2 discusses the related works on 3D tracking. Section 2.1 discusses object detection methods and tries to select the most suitable detection scheme for the proposed system. Section 2.2 discusses exiting depth estimation methods and evaluates the possibility of using them for real time

applications such as object tracking. Section 2.3 discusses the most commonly used algorithm in the field of visual tracking and how they were implemented in the published works. In addition, it lays the foundation of an objective comparison between the existing object tracking techniques.

Chapter 3 discusses the implementation of existing object detection techniques and object tracking algorithms. Section 3.1 focuses on object detection algorithms and focuses on object detection using background subtraction. In addition, this section implements some shadow removal methods and shows its effects. Section 3.2 presents an objective and subjective evaluation of six visual tracking algorithms. This section helps in selecting the best method to be used for 3D visual tracking using single camera.

Chapter 4 introduces a novel depth computation method from a single 2D image using triangulation. This section discusses how this method is implemented and how it is used for measuring distance in the image.

Chapter 5 is dedicated to present the results of testing the proposed 3D tracking system on various video sequences. Four experiments were presents in this section and the tracking results as well as the depth computation results are recorded and compared with ground truth measurement.

Finally, chapter 6 provides concluding remarks for this work and highlights some of the future improvements that can be made on this work.

CHAPTER 2
LITERATURE REVIEW

2.1 **Introduction**

As shown in Figure 1.1, the proposed 3D tracking system is made up of four major blocks. The first component is the image acquisition block which consists of a camera or a video file. In this thesis, the concentration is on the remaining three blocks; object detection, depth estimation and object tracking. This chapter scans through some of the earlier works on object detection methods with focus on background subtraction techniques. In addition, this section discusses existing depth estimation techniques and highlights their limitations. Finally, this section discusses the existing visual tracking methods and it is implemented in computer vision applications.

2.2 **Object detection**

Any tracking system requires a detection method in order to identify the location of the object of interest in image coordinates. An object can be detected using background differencing, color/shape matching or segmentation algorithms such as active contour or graph cuts. In addition, supervised classifiers are helpful in detecting a moving object that belongs to a known class. Background differencing is the most common method for detecting moving object. Background differencing does not require any prior knowledge about the scene [1]. Figure 2.1 shows a complete flow of object detection system using background subtraction. Firstly, the background is identified. Then, the background is subtracted from the new frame in order to extract regions with motion. After that, the foreground is filtered in order to remove noisy regions such as very small foreground regions. In addition, this step helps in removing cast shadows in the foreground image. Finally, suitable representation scheme is used in order to represents the moving objects.

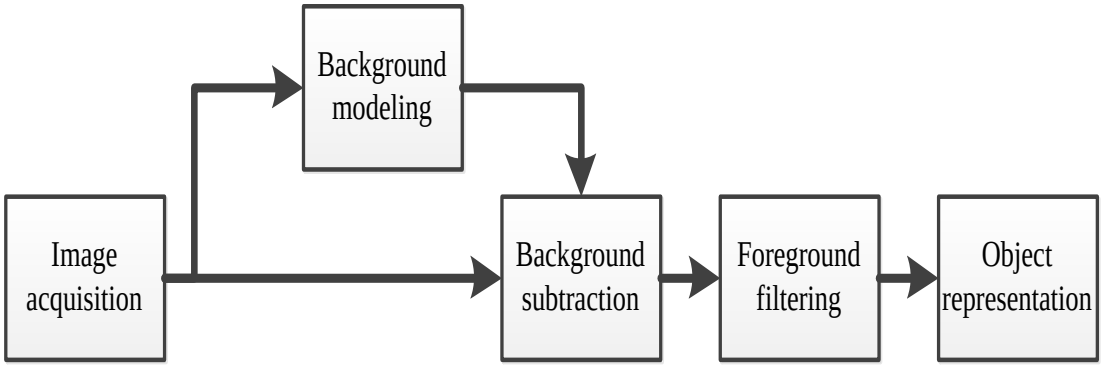


Figure 2.1: Object detection system using background subtraction.

2.2.1 Background Modeling Methods

There are two types of background modeling; non-adaptive and adaptive modeling. In the first type, the background is static and it is modeled one time only. This is suitable for indoor background scenarios where there are no environmental variations. Adaptive background modeling is needed in areas where there is a considerable illumination variation due to sunlight as well as shadows and reflections. Temporal averaging is considered to be the simplest type of background modeling. In temporal averaging, the mean intensity value is computing for a sequence of training frames. Sometimes mono-Gaussian model is used in in order to accommodate for slight variations in background. This is achieved by computing the mean value and the standard deviation for each pixel across the training frames [15]. Usually median filtering is used instead of temporal averaging because it removes outliers from the estimated background [16], [17].

Adaptive background models continuously update the background in order to accommodate for background variations. This is very useful for outdoor scenes where the same pixel exhibit different intensities at different times of the day. Multi-model adaptive background model is commonly used for modeling outdoor scenes. Stauffer and Grimson [18] implemented an adaptive background model using a mixture of Gaussians where each pixel in the new frame is compared with corresponding Gaussian models. If match is found, then the pixel is considered as a background otherwise a new Gaussian is created using the current frame. Multipoint-pair background model is based on assigning multiple reference points for each pixel in

the image where this point maintains a constant relationship with its reference points across a number of consecutive frames. During the detection step, if the target point maintains the same difference with its reference points, it is considered as background point. Otherwise it is considered as foreground point.

Satoh et al. [19] proposed the use of radial reach filter which uses reference points around the eight directions of the target point. All these points should maintain an intensity difference more than specified threshold with the target point. Satoh and Sakaue [20] used a bipolar radial reach filter in which two reference points are identified in each radial direction. Half of the reference points maintain positive difference with the target point and the other half maintain negative difference with the target point for a number of consecutive frames. Zhao et al. [21] introduced the concept of gray-scale arranging pair (GAP) for background modeling. This method uses multiple point pair that exhibit stable relationship across number of frames. In this method, the reference points are globally distributed in the image. In addition, Zhao et al. proposed an efficient implementation of GAP method and they compared it with [18], [20], [22], [23] and it shows higher detection accuracy than these methods.

2.2.2 Garyscale Arranging Pairs (GAP) Background Modeling

GAP method is adopted for background modeling because of its higher detection accuracy compared to other existing algorithms this method is based on consistent intensity difference between each point in the image and multiple reference points. GAP method computes N reference points for each image point $I(x, y)$. Half of these points maintain an intensity difference greater than a threshold (W) while the other half maintains intensity difference with the target point less than a threshold ($-W$). This algorithm searches for reference points from the whole image because neighboring pixels exhibit natural correlation. The search algorithm is optimized by two steps. Firstly, the search range is reduced by searching for reference points that maintain the mean intensity difference smaller than some threshold rather than searching for the same point in all frames. Secondly, the process is spatially sampled so that similar neighboring pixels have the same reference points. These two steps

reduce the computational complexity of the algorithm drastically. Despite these reduction steps, this algorithm still has high computational time compared to other background modeling techniques.

Table 2.1 shows detailed flow of the background modeling algorithm proposed by Zhao et al. [21]. The algorithm takes M frames for modeling the background of the scene. Less number of frames reduces the computational time but it produces less-robust model. W_G is the intensity difference threshold, large W_G makes it insensitive and small W_G makes it oversensitive to background changes. W_S is a threshold to determine similar neighboring pixels so same reference points can be assigned to them. W_B and W_P are similarity weights where the first one determine the number of frames at which two pixels are similar while the second one determines the number of frames at which two pixels exhibit similar intensity difference (usually $0.75 - 0.95$).

Initially, the image is converted into a vector format and three memory buffers are allocated. Ref^+ is the buffer for positive reference points for each pixel and Ref^- is for the negative ones. The third buffer is allocated for assigned pixels which tell whether a pixel has been allocated with reference points or not. For each pixel in the background image, the algorithm searches among the pixel neighbors for the most similar pixel. If this pixel satisfies the similarity threshold and it has been assigned a reference points before, the new pixel will take the reference points of its similar assigned neighbor and the new pixel will be added to the assigned pixels buffer. If there is no assigned neighbor exists among the pixel neighborhood, the algorithm searches for reference points in the whole image until $\frac{N}{2}$ reference points are allocated for Ref^+ and Ref^-.

Table 2.1: Flow diagram for GAP background modeling algorithm [21].

Input M frames, Threshold W_G, W_S, W_B, W_P
Initializations
Convert 2D image into 1D vector $\langle I_M(W \times H) \rightarrow I_M(U) | U = W \times H \rangle$
Compute global threshold $W = W_B \times W_P - 255 \times (1 - W_P)$
Allocate buffer for reference points Ref^+, Ref^- and assigned pixels set $S \leftarrow \emptyset$
Sort pixels $\{Q_1, Q_1, \dots, Q_U\}$ based on mean intensity $\overline{I(Q_u)}$ for training frames
For each Pixel $P; u \in \{1,2, \dots, U\}$ **Do**
$Ref^+(P) \leftarrow \emptyset,\ Ref^-(P) \leftarrow \emptyset, Cnt \leftarrow 0$
$k = \min_{p \in Nbr(P_u)} \left| \overline{I(p)} - \overline{I(P)} \right|$ // $\overline{I(p)}$ is the mean of p across M frames
$L = \# \left\langle t \middle| \frac{|I_t(k) - I_t(P)| < W_S}{M} \right\rangle; t = 1,2, \dots, M$
If $p \in S$ and $L > W_B$ **Then**
Add P to S // declare pixel P as assigned pixel
break // jump to the next point
End
// Search for positive difference reference points
For $u = U: 1$ **Do**
If $\overline{I(P)} - \overline{I(Q_u)} > W$ **Then**
$L = \# \left\langle t \middle| \frac{|I_t(P) - I_t(Q_u)| \geq W_G}{M} \right\rangle; t = 1,2, \dots, M$
If $L \geq W_P$ **Then**
Add Q_u to $Ref^+(P)$
$Cnt = Cnt + 1$
If $Cnt = \frac{N}{2}$ **Then**
$Cnt = 0$
break
End
End
End
End
For $u = U: 1$ **Do** // Search for negative difference reference points
If $\overline{I(P)} - \overline{I(Q_u)} < -W$ Then
$L = \# \left\langle t \middle| \frac{|I_t(P) - I_t(Q_u)| \leq -W_G}{M} \right\rangle; t = 1,2, \dots, M$
If $L \geq W_P$ **Then**
Add Q_u to $Ref^-(P)$
$Cnt = Cnt + 1$
If $Cnt = \frac{N}{2}$ **Then**
$Cnt = 0$ and **break**
End
End
End
End
Add P to S // declare pixel P as assigned pixel
End

During the detection step, the system compares each pixel in the test image with its reference points. If the test pixel intensity is larger than the intensity of Ref^{+} and smaller than the intensity of Ref^{-} then the pixel is classified as background, otherwise it is classified as foreground pixel. Due to noise, not all the reference points would satisfy this condition. If 70% ($W_H = 0.7$) or more of the reference points has larger or smaller intensity than the target pixel, then the pixel is considered as background. This detection scheme gives more accurate results compared to other methods existing in the literature. Table 2.2 shows a flow diagram for object detection algorithm using the multi-pair reference point background modeling.

Table 2.2: Object detection algorithm using reference points [21].

Input test Image J, Threshold $W_H$and reference points Ref^{+}, Ref^{-} , N
For each Pixel $P; u \in \{1,2, \ldots., U\}$ **Do**
    // compare with Ref^{+} reference points
    $Cnt = 0$
    For point in $P \in Ref^{+}(P)$ **Do**
        If $J(P) \geq J(Q)$ **Then**
            $Cnt = Cnt + 1$
        End
        $p^{+} = \frac{Cnt}{N/_2}$
    End
    // compare with Ref^{-} reference points
    $Cnt = 0$
    For point in $P \in Ref^{-}(P)$ **Do**
        If $J(P) \leq J(Q)$ **Then**
            $Cnt = Cnt + 1$
        End
        $p^{-} = \frac{Cnt}{N/_2}$
    End
    // classifying pixel P
    If $p^{+} > W_H$ and $p^{-} > W_H$ **Then**
        $P = 0$ // classifying pixel P as background pixel
    Else
        $P = 1$ // classifying pixel P as foreground pixel
    End
End

2.2.3 Shadow/highlights removing

Shadow areas are normally partially illuminated and it has less illumination compared to its surroundings while highlights are exactly the opposite where the highlighted point has higher illumination than the background. Shadow area has similar chromaticity but attenuated luminance compared to the non-shadowed image [24]. Xu et al. [24] assumed that shadowed regions maintain the same colors and texture properties as non-shadowed one. Thus shadow regions can be detected by comparing its color and texture with the non-shadowed background. Branca et al. [25] uses the photometric gain which is the ratio between the luminance of current frame and the background for detecting shadow regions in the foreground image. According to [25], shadow regions tend to have photometric gain less than 0.90. Cucchiara et al. [26] proposed a method for detection ghost and shadows in video streams. Shadow is characterized by certain attenuation in the hue, saturation and value components of the HSV image where the amount of attenuation is found empirically.

Figure 2.2 shows a flow diagram for shadow elimination algorithm. For the blob of a moving object candidate, the algorithm computes the average luminance and chromaticity of that area in both current frame and the background image. Then it computes the Luminance gain and chromaticity gain using equation (2.1) and (2.2). Finally if the blob has a Luminance gain less than 90% and it maintain the chromaticity gain near to 100% (0.9 – 1.10) then it is considered as shadow. Because shadow carries same color information as the background objects. Similarly if the Luminance gain is greater than 110% and the color is maintained, the blob is considered as highlights. Otherwise it will be considered as a true moving object blob provided that it passes the filtering constraints such as size limits and others. The threshold for this method has been set based on the work of Branca et al. [25] which shows less computational cost compared to the recent works on shadow removal algorithms.

$$LumaGain = \frac{Y_{Image}}{Y_{Background}} \tag{2.1}$$

$$ChromaGain = \frac{Cb_{Image} + Cr_{Image}}{Cb_{Background} + Cr_{Background}} \tag{2.2}$$

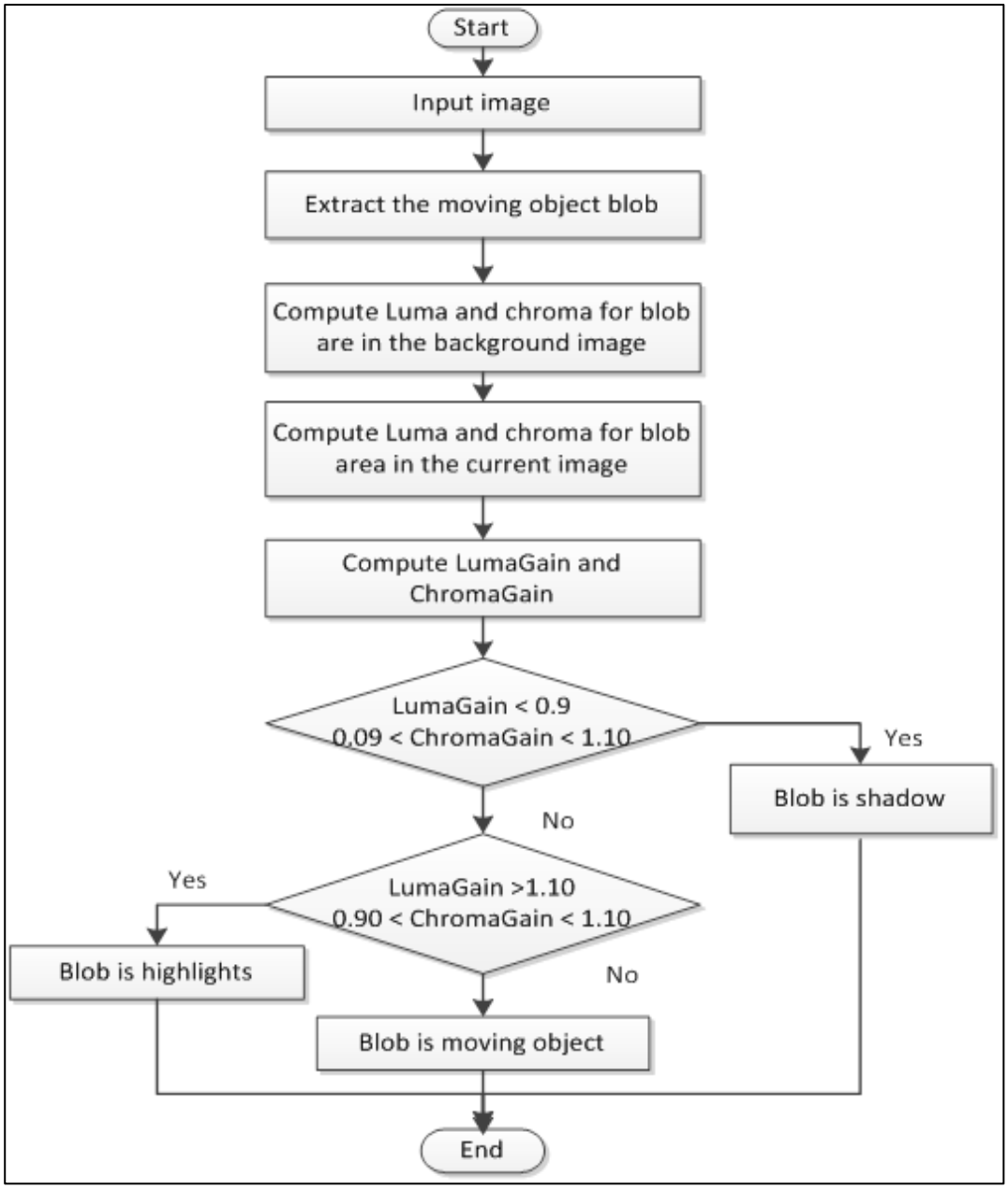


Figure 2.2: Flow diagram of shadow/highlights elimination algorithm in the foreground image.

2.2.4 Object representation

Moving object is represented based on its apparent shape. Yimlaz et al. [1] defined common object representation schemes as shown in figure 2.3:

- Point representation such as centroid point which is suitable for small objects.
- Primitive representation such as using bounding box or bounding ellipse [27].

- Articulated body representation: uses connected structures to represents human body or an object that is made-up of small sub-objects [7].
- Skeleton representation: represents the moving object with connected lines.
- Contour representation: mostly used for representing non-rigid object which cannot be represented by bounding box.

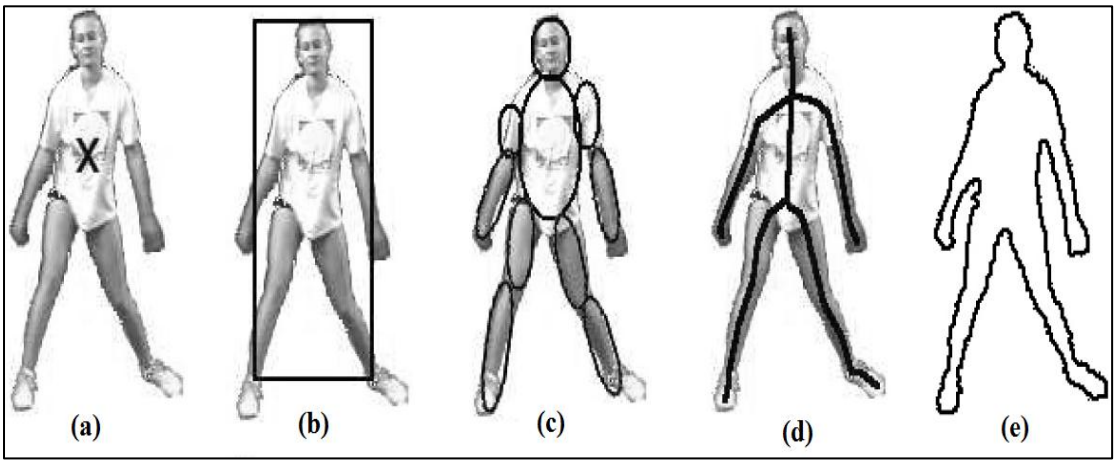


Figure 2.3: Different object representation schemes: (a) centroid representation, (b) primitive representation, (c) articulated body representation, (d) skeleton representation, and (e) contour representation. (*Images are reproduced with permission from* [1]).

Point representation conveys information about the location of the object. While primitive representation gives information about the location as well as size of the moving object. Articulated, skeleton and contour representation provides information about the detailed structure of the moving object.

2.3 **Depth Estimation**

In traditional imaging systems, 3D scene is projected on 2D image sensor, thus the depth of the field is lost due to this projection. Depth estimation is an important field in computer vision and many techniques have been proposed for depth estimation and 3D shape recovery from 2D images. There are two categories in optical depth estimation; active methods and passive method as it is shown in Figure 2.4. This section discusses existing depth estimation algorithm and their implementation in the published work.

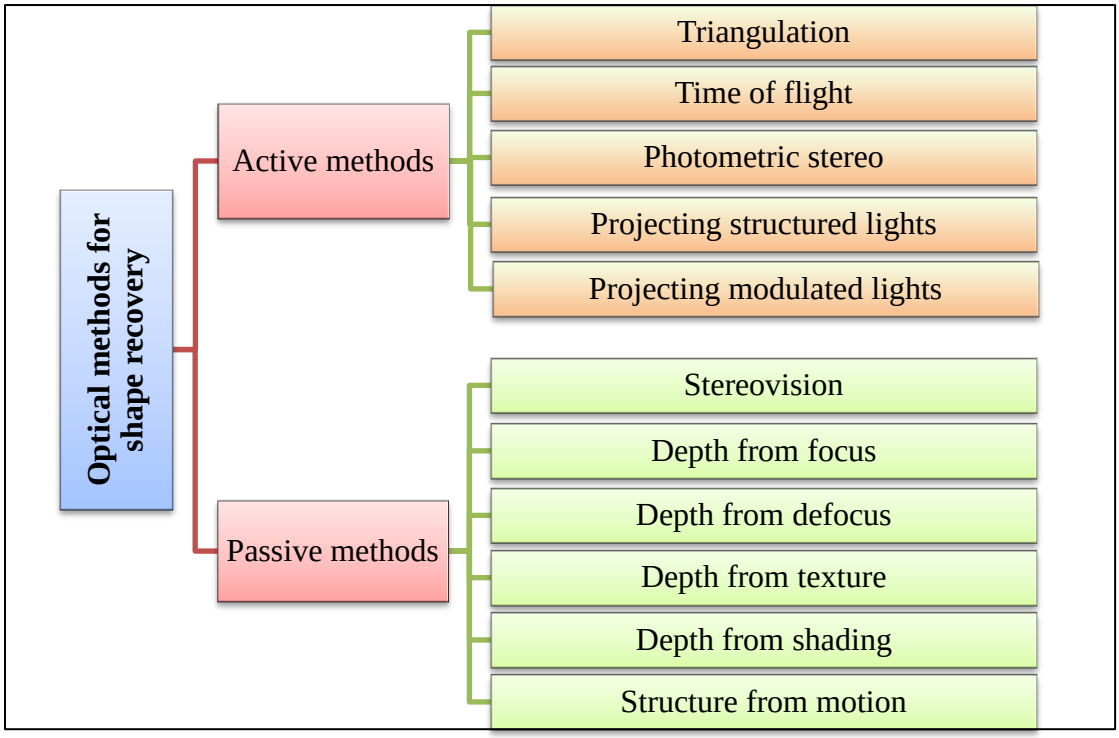


Figure 2.4: Taxonomy of optical depth estimation methods.

Figure 2.4 shows a classification for optical depth estimation techniques. There are two types of optical depth computation methods. The first group is the active shape recovery methods and the second ground is the passive shape recovery methods. In the active group, lighting plays an active role in the depth estimation process. This is clear in laser scanner which uses the time of flight for estimating the distance between the scanner and each point in the scene. There are various techniques for active depth estimation such as time of flight, triangulation, phase-shift, structured light and modulated light. In general, active depth estimation techniques require additional devices beside the camera such as laser source or projecting mirrors or light pattern generator. This makes it expensive and not suitable for real time applications. Moreover, this work addresses depth estimation using the existing camera installations which are 2D camera with no additional hardware. Therefore, active depth estimation algorithms cannot be used for the proposed 3D tracking system.

The second group is the passive shape recovery methods. In this group, light does not play an active role in the shape recovery process and the depth of field is computed using the image intensities only. Passive depth estimation methods use

various visual cues for depth estimation similar to the human visual system. These cues are stereoscopic parallax, motion parallax and monocular depth cues [28]. Stereovision uses two images captured from two parallel cameras displaced by a known distance [29]. In structure from motion, spatial disparity of consecutive frame is used to compute the depth of the scene. Objects which are closer to the camera undergo larger displacement than objects which are far from the camera for the same amount of translation between consecutive frames [30]. Depth from focus/defocus is derived from the fact that objects are focused only at the focus plane from the camera [31], [32]. Depth form texture uses texture gradient and texture energy for computing the depth of field [33]. Depth form shading uses the image reflectance map for computing the surface gradient which is related to the depth of field [34].

2.3.1 Depth from Vanishing Lines

Parallel lines in the scene vanish to one point in the image because of the perspective distortion of the imaging system. This method has been employed for estimating the depth from single image. Vanishing lines can be computed using Hough transform. Criminisi et al. [35] computed distance between objects in the scene using vanishing lines from single view. This method can compute the distance between two points in the image by employing the geometry of parallel lines in the scene. Rother et al. [36] uses casual people motion to compute the horizon of the scene using three different observations as in Figure 2.5. These different observations for the same object have been fused in one image in order to produce three horizon points. Then the size of the moving person is computed using L_1 minimization. Barinova et al. [37] computed the vanishing points of a cluttered scene using RANSAC algorithm.

Using Vanishing points for depth estimation involves low computational requirements compared to other depth estimation methods. However, it is dictated by the presence of vanishing lines in the image which is not always possible to compute. Hence, this method cannot be used in visual surveillance because the environment is cluttered and parallel lines are not available in all scenes.

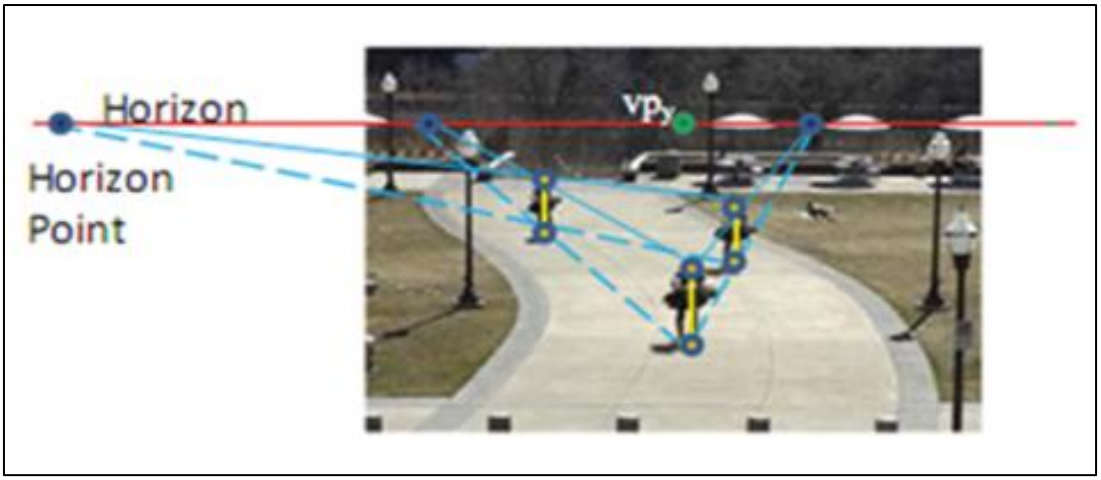


Figure 2.5: Combing three images in order to compute the horizon. (*Image reproduced from [37]*)

2.3.2 Combined Visual Cues for Depth Estimation

In all previous method only single visual cue is used for the depth computation. Therefore, each method works in specific scenarios where these cues are prevalent. Recent research move tends to use multiple visual cues in order to get better depth estimation. At each image point multiple depth cues are computed in order to extract the geometric structure at that point. Hoiem et al. [38] introduces a set of features for describing the 3D geometric structure of the scene which can be used to learn the appearance of the object in real images. This set includes shape features, location feature as well as color features to describe an object in the image. Lila et al. [39] proposed a depth estimation method using texture and focus cues. They have employed the wavelet decomposition in order to analyze texture variations and extract focused regions. Torralba and Oliva [40] computed the mean absolute depth of the scene using Fourier spectrum of the image. This method can be used to rescale relative depth estimation methods such as stereo and structure from motion in order to obtain the absolute depth of each point in the scene.

Due to the ambiguity nature of depth estimation, local features from a single image are not sufficient for estimating the depth for the scene. Therefore, more global information is utilized in order to obtain more accurate results. Machine learning tools were extensively used for learning the relationship between local/global depth cues

and the depth of the field. Jung and Ho [41] estimated the depth by classifying object into four categories (plane, cubic, sky and ground) using Bayesian learning scheme then it assigns a suitable depth value to each segment. Although this method works with specific type of images, the algorithm results are similar to human perception and it can be improved by include more object categories. Hoiem et al. [42] developed an automatic image pop-up system by classifying the image into geometric classes. Image pixels are labeled into sky, vertical and ground using a supervised training. Finally, the 3D model is created by using cut and fold operation in order to pop-up the vertical elements on the ground.

Saxena et al. [43] used Markov random field (MRF) to incorporate more global in learning depth of the field from multiple monocular cues. Multiple scale random fields were used to learn the depth of a regular size image segments (patches). In reality, image components tends to have irregular shapes, in [44] Saxena et al. employed a multi-scale MRF for learning the depth of an irregular patches (superpixels) which produces more realistic results. Das et al. [45] worked on improving the algorithm in [44] by designing a new omnidirectional high pass filter that can capture more depth features than the filters in [44]. Liu et al. [46] performed a semantic segmentation of the scene using multiple-classes MRF. The depth is computed for these classes using a second MRF.

2.3.3 Depth Estimation for Real Time Applications

Active depth estimation methods cannot be used in real time applications because it has long acquisition time and it requires additional hardware with the camera such as laser source or structured light sources. Stereovision uses two cameras and it requires large computational resources in order to compute the correspondence for the left and right images. Structure from motion requires identifying good feature to track and finding correspondence between them across frames. Thus SFM similar to stereovision are not applicable for real time depth estimation. Shape from focus requires large number of images in order to attain a good depth map which requires high computational resources. Again this is not suitable for surveillance system because the information flux is very fast.

Shape form texture and shape from shading both involve solving an ill-posed problem and require higher computational time. In addition, these methods work only when the texture/shading cue in the image is very obvious. Thus, these methods are not recommended for visual surveillance applications. Depth estimation using vanishing lines has a short computational time compared to other methods. However, there must be enough parallel lines in the scene in order to compute the required depth which is not always possible.

The last option is combining multiple visual cues in order to compute the depth at each point in the image such as combining focus information with texture gradient and texture energy in a global manner for each pixel in the image which involves large computational complexity. Most of the time, machine learning tools are used to learn the relationship between the combined depth cues and the depth of the field. Supervised learning requires large dataset with ground truth depth maps. Therefore, using multiple depth cues and machine learning tools is not favorable for real time applications because of the computational complexity and the need to perform learning.

As a conclusion, existing depth estimation methods are not suitable for real time application, because they require either multiple cameras (stereovision) or multiple images from the same camera (SFF, SFD and SFM). Moreover, most of the existing depth estimation methods have high computational complexity because of the need to find correspondence points between multiple images (stereovision and SFM) or solving an ill-posed problem to compute the reflectance (SFS). In addition, some depth estimation algorithms require learning depth cues extracted from the images which requires large datasets with corresponding ground truth depth maps. Because of all these limitations, there is a need for a new depth estimation technique that has real time performance and does not require using additional resources.

2.4 **Object Tracking Algorithms**

The objective of tracking is to predict the motion of the moving object and make decisions about its next move. There are various methods for tracking a moving object such as shape matching, mean-shift and stochastic filters. Figure 2.6 shows the percentage of published work for the period of 2006 - 2010. Stochastic filters (Kalman and Particle filters) accounts for more than 50% of the published work on object tracking. This is because stochastic filters have the ability to include noise in measurements. This means Kalman and particle filters assume that measurements might contain random noise which is very common in visual tracking. Other algorithms have less influence compared to stochastic filters. For example, the mean-shift algorithm has bad accuracy when it comes to abrupt motion. Therefore, it is mostly combined with other algorithms in order to improve its accuracy. Contour evaluation and shape matching are deterministic approach but they have large computational time compared to stochastic filters. Machine learning tools are less frequently used in object tracking because they require large datasets to learn their estimation coefficients.

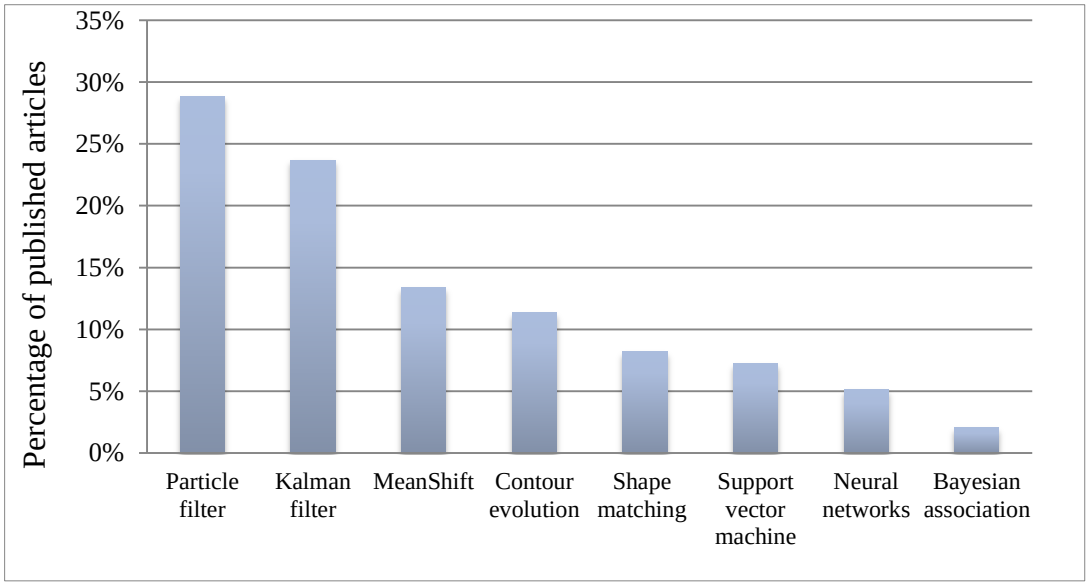


Figure 2.6: 3D tracking techniques published during the period of (2006- 2010).

This section discusses the first three algorithms in Figure 2.6 which represents around 70% of the published works. These algorithms are mean-shift method, Kalman filters and particle filters.

2.4.1 State Space Modeling

Filtering is an attempt to estimate the state of the system using measured observations. The system is represented by its state space model. At each new state, the filter tries to estimate the state of the system prior to measurements. When the measurements are available, the estimation is corrected and new estimation for the next state is computed [47]. Figure 2.7 illustrates a dynamic system representation in which the state *x* is paired with a measurement *y*, the measurements are dependent on the state{*y* = f(*x*)}.

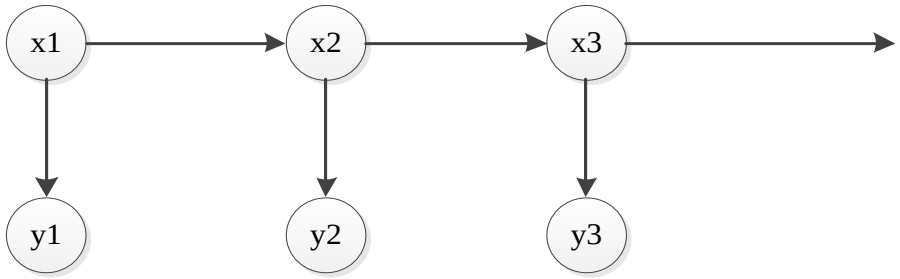


Figure 2.7: State space representation for system dynamics.

A dynamic system can be represented using state space model. State space is derived for any system represented by differential equations. Let's consider a dynamic system which is represented by an n-order differential equation:

$$y_{k+1} = a_{0,k}y_k + \cdots + a_{n-1,k}y_{k-n+1} + u_k \tag{2.3}$$

Where $\{u_k\}$ is the input. Equation (4.1) can be written in a matrix format.

$$\underbrace{\begin{bmatrix} y_{k+1} \\ y_k \\ \vdots \\ y_{k-n+2} \end{bmatrix}}_{x_{k+1}} = \underbrace{\begin{bmatrix} a_0 & \cdots & a_{n-2} & a_{n-1} \\ 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix}}_{F} \underbrace{\begin{bmatrix} y_k \\ y_{k-1} \\ \vdots \\ y_{k-n+1} \end{bmatrix}}_{x_k} + \underbrace{\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{G} u_k \tag{2.4}$$

$$z_k = \underbrace{[1 \quad 0 \quad 0 \quad 0]}_{H} x_k \tag{2.5}$$

Assuming the process is noise, random noise can be added to x_{k+1} and z_k. Equations (2.4) and (2.5) can be written in more general format as:

$$x_{k+1} = Fx_k + Gu_k + w_{k-1} \tag{2.6}$$

$$z_k = Hx_k + v_k \tag{2.7}$$

Where x_k is the state variable, z_k is the observations (measurements), w_k and v_k are state noise and measurement noise respectively. Matrix F is the state transition matrix, matrix H is the measurement matrix and k is the time sequences $k = 1,2,\cdots$ [48]. As special case of state space models when only the system outputs are measureable, the model is called observer model. The observer state space model is a "black box" modeling in which only the outputs of the systems are measureable while inputs are unknown (considered as noise). For observer design problems, the state model and the measurement model are written as:

$$x_k = Fx_{k-1} + w_k \tag{2.8}$$

$$z_k = Hx_k + v_k \tag{2.9}$$

2.4.2 Mean-Shift Algorithm

Mean-shift algorithm is a popular method for tracking because of its simplicity and low computational requirements [49]. The main limitation of mean-shift is its inability to cope with rapid object motion. Generally, mean-shift is combined with other tracking algorithms such as particle filter and Kalman filters in order to get more accurate estimation results.

Mean-shift is categorized as a kernel tracking algorithm where the tracked object is represented by an object region (rectangle or elliptical patch) [1]. The algorithm has three main steps; initialization, model creation and similarity calculation [50].
This step is performed at the initial stage of the tracking where the target is defined, number of histogram bins $\{m\}$ is specified and object area and center position$\{y\}$ are defined too. In addition, kernel function is also defined in this stage $\{k(x)\}$ [62, 64, 96].

In this step, two object models are created. At the initial stage of the tracking, a

reference model for the object is defined. Then for every new frame, a candidate target model is generated. If the candidate model doesn't match with the reference model using the similarity function, the current search window is shifted using the mean shift algorithm. The reference target model $\{\hat{q}_u\}_{u=1,\ldots,m}$ is given in equation (2.10), where n is the number of pixels (consider a 1-D representation of the image) in the search window, h is the width of the search window and $b(x_i)$ is the histogram bin at position x_i while the j of $b(x_i, j)$ in equations (2.10) and (2.12) represent the frame number. Equation (2.12) shows the candidate model for frame j. The model created in equation (2.13) is normalized using equation (2.11) [62, 96].

$$\hat{q}_u = C \sum_{i=1}^{n} k\left(\left\|\frac{y - x_i}{h}\right\|\right) \delta(b(x_i, 0) - u) \tag{2.10}$$

$$C = \left(\sum_{i=1}^{n} k\left(\left\|\frac{y - x_i}{h}\right\|\right)\right)^{-1} \tag{2.11}$$

$$\hat{p}_u(y_j) = \sum_{i=1}^{n} k\left(\left\|\frac{y - x_i}{h}\right\|\right) \delta(b(x_i, j) - u) \tag{2.12}$$

$$\hat{p}_u(y_j) = \frac{\hat{p}_u(y_j)}{\sum_{u=1}^{m} \hat{p}_u(y_j)} \tag{2.13}$$

After computing the candidate's histogram, the similarity between the target histogram and the reference histogram is computed. Usually, the similarity is defined using distance measures such as Bhattacharyya distance which is shown in equation (2.14) [62, 96].

$$\rho[\hat{p}(y_j), \hat{q}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(y_j)\hat{q}_u} \tag{2.14}$$

If similarity between current and reference target model does not match, the algorithm shifts the mean of the search window using equation (2.15) [96, 97]. This process is repeated until the match is attained. Table 2.3, illustrates the steps of the mean-shift tracking algorithm.

$$y_{j+1} = \frac{\sum_{i=1}^{n}\sum_{u=1}^{m} x_i \sqrt{\frac{\hat{q}_u}{\hat{p}_u(y_j)}}\,\delta(b(x_i,j)-u)}{\sum_{i=1}^{n}\sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\hat{p}_u(y_j)}}\,\delta(b(x_i,j)-u)} \tag{2.15}$$

Table 2.3: Pseudo-code for the mean-shift algorithm [53].

1) IF frame = 0THEN
   i. Detect the object
   ii. Create object reference model (y_{ref})
2) ELSEIF frame = 1 THEN
   i. Create a new candidate target model (y_0)
   ii. Compute the similarity degree of this candidate $S_0 = Similariy\ (y_0)$
3) ELSE THEN
   i. Create a new candidate target model (y_1)
   ii. Compute the similarity degree of this candidate $S_1 = Similariy\ (y_1)$
   iii. WHILE$S_1 < S_0$THEN
      a. $y_1 \leftarrow \frac{1}{2}(y_1 + y_0)$
      b. Compute the similarity degree of this candidate $S_1 = Similariy\ (y_1)$
   iv. END WHILE
   v. IF $\|y_1 - y_0\| < Thresh$ THEN
      STOP
   vi. ELSE THEN
      i. $y_0 \leftarrow y_1$
   vii. ENDIF
4) END ELSE

Main shift method is very popular in visual tracking because of its simplicity and low computational requirements. However, it is not robust to abrupt motion variations [54]. Hu et al. [49] used mean-shift for object tracking using multiple image fusion using aspect graph. Most of the time this algorithm is combined with additional constraints in order to improve its robustness for abrupt motions such as combing mean-shift with motion vectors [51], combing it with Scale Invariant Features (SIFT) [55] and also fusing it with Gabor wavelets [52]. For measuring the ground truth object location, mean-shift algorithm uses object color which is represented n-bin histogram vector [51], [52], [55] and in some cases shape descriptors been used to represent the true location such as Fourier descriptors [49].

2.4.3 Kalman Filters

Kalman filter is an iterative prediction-correction process to estimate the state of the system [47]. Let's consider a system represented by a state space model similar to the one in Equations (2.8-2.9). Let W_k and V_k have a Gaussian probability distribution function (PDF) as shown in equations (2.16) and (2.17) with Q and R being the covariance matrices for the state and measurement respectively [48].

$$p(w) = \mathcal{N}(0, Q) \tag{2.16}$$

$$p(v) = \mathcal{N}(0, R) \tag{2.17}$$

Equation (2.8) and (2.9) can be generalized to fit nonlinear systems where transition matrices are nonlinear $\{F \rightarrow f(x)\ and\ H \rightarrow h(x)\}$.

$$x_k = f(x_{k-1}) + Qv_k \tag{2.18}$$

$$z_k = h(x_k) + Re_k \tag{2.19}$$

v_k and e_k are random variables with zero mean. The functions $f()$ and $h()$ are state transition matrix and measurement matrix respectively in a generalized nonlinear form. An important assumption of Kalman filter is that x_k and z_k has a Gaussian distribution thus it can be represented by mean value and a covariance matrix as shown in equation (2.20) where $\hat{x}_k$ is the mean value of the state variable x_k and P_k is its covariance matrix [48].

$$p(x_k) = \mathcal{N}(\hat{x}_k, P_k) \tag{2.20}$$

There are three common types of Kalman filters. The first one is the linear Kalman filter which is used with linear systems. The second type is the extended Kalman filter which relies on linearizing nonlinear systems and after linearization it follows similar steps to the linear Kalman filter. The third type of Kalman filters is the unscented Kalman filter which uses the unscented transformation in order to linearize highly nonlinear systems.

2.4.3.1 *Kalman Filter*

Linear filter assumes that the process matrix F and the measurement matrix H are linear. The linear Kalman filter estimates the optimal value for the next state using linear equations provided that the above assumption holds. Kalman filter predicts the next state in two steps; firstly it projects the current state forward in the time which is known as the time update or forward projection. In the second step, it feedbacks the current measurement in order to correct the time update estimation and this is known as measurement update or feedback [48]. Time update and the measurement update are known as predictor-corrector cycle similar to what is shown in Figure 2.8. This cycle is continuous along the iterative process.

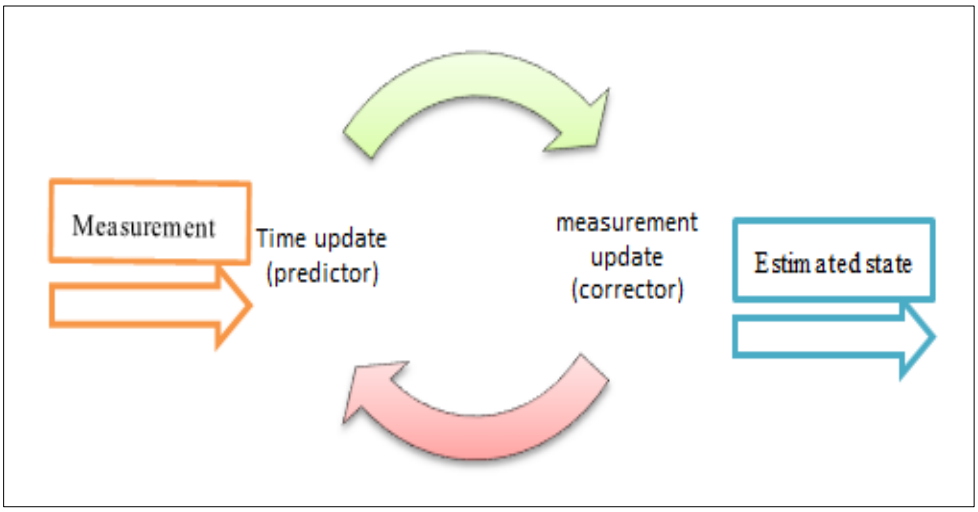


Figure 2.8: Prediction-correction loop of Kalman filters.

The prediction – correction algorithm of linear Kalman filter can be expressed mathematically as follows:

Prediction Equations:

- Projecting the estimate forward in time

$$\hat{x}_k^- = F\hat{x}_{k-1} \tag{2.21}$$

- Projecting the error covariance forward in time

$$P_k^- = FP_{k-1}F^T + Q \tag{2.22}$$

Correction Equations:

- Computing the Kalman gain

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \tag{2.23}$$

- Correcting the estimate using actual measurements

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \tag{2.24}$$

- Correcting the error covariance

$$P_k = (I - K_k H) P_k^- \tag{2.25}$$

K_k is known as Kalman gain which is chosen to minimize the covariance error. z_k is the actual measurement and I is the identity matrix. As mentioned earlier, Kalman filter strictly assumes the posterior is Gaussian and the system has linear dynamics. As long as these assumptions are correct, Kalman filter provides an optimal estimation for the next state of the process at low computational cost. The above assumptions are major limitation of linear Kalman filter because not all processes satisfy them.

Linear Kalman filter is very popular in the published work especially for robotic navigation. Weng et al. [56] used linear Kalman filter for object tracking based on its color features. The color feature is used to compare current color histogram of the region of interest with a reference color histogram of the tracked object. In this work the error covariance of the filter is adaptively updated using the ratio of object area in consecutive frames. Devarkota et al. [57] used linear Kalman filter with multiple hypothesis tracker (MHT) for tracking human head from range images. The MHT generate a number of head candidates and after the filtering step the algorithm computes the fittest candidate to be the new head location. In another work Du and Yuan [58] used linear Kalman filter with Gabor wavelets for vehicle detection and tracking. The two filters are combined in such a way that the Kalman filter predict the region of interest then the Gabor filter detect the vehicle with in the region of interest. Linear Kalman filter has been extensively used for robot navigation; Fernandez et al. [59] used it for mobile robot positioning in indoor environment. Suppes et al. [60] used linear Kalman filter for obstacle detection by mobile robot using stereoscopic camera as a proximity sensors.

2.4.3.2 *Extended Kalman Filter (EKF)*

When the process is nonlinear or the relationship between process and measurement is not linear, the linear Kalman filter will perform poorly because it assumes relationship to be linear with Gaussian PDF. This problem is addressed by considering a Kalman filter that linearizes around the mean and covariance of current state; this is known as Extended Kalman filter (EKF) [48]. Let's consider the generalized state space model shown in equations (2.18) and (2.19). $f(x_k)$ is a non-linear transition function that relates current state to previous state, and $h(x_k)$ relates the measurement to the current state. Now if the nonlinear functions are approximated using first order Taylor approximation, the prediction-correction cycle for the extended Kalman filter can be expressed as:

Prediction Equations:

- Projecting the estimate forward in time

$$\hat{x}_k^- = \hat{x}_{k-1} \tag{2.26}$$

- Projecting the error covariance forward in time

$$P_k^- = P_{k-1} + Q \tag{2.27}$$

Correction Equations:

- Computing the Kalman gain

$$K_k = P_k^-(P_k^- + R)^{-1} \tag{2.28}$$

- Correcting the estimate using actual measurements

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - \hat{x}_k^-) \tag{2.29}$$

- Correcting the error covariance

$$P_k = (I - K_k)P_k^- \tag{2.30}$$

It is clear from these equations that for the prediction stage, the EKF assumes that the current state is equal to the previous state. Then it will be corrected by the measurements in the corrector stage. The EKF linearizes the nonlinear system around the mean (the previous state estimate).

Extended Kalman filter enhances linear Kalman filter to suit slightly nonlinear

systems by linearizing it using around the mean. Jia et al. [61] used an extended Kalman filter using color and motion cues. Zhou and Aggarwal [62] used extended Kalman filter for object tracking in outdoor environment by fusing multi-view images. Ababsa [63] used an iterative extended Kalman filter for camera pose estimation based on matching 2D edges from image with 3D edge lines of the room. In this filter the correction step is repeated interactively until an error threshold is met. In another work, Gao et al. [64] used multiple extended Kalman filters for object tracking in cluttered scene. Each of these multiple filters improves the estimated results by the previous filter in order to get higher accuracy estimation. Lippiello et al. [65] used adaptive extended Kalman filter for navigating a robotic arm. This filter uses noise statistics of abrupt motion to update covariance matrices. In all previous works, the correction step of the extended Kalman filter is been improved either by adaptively tuning the covariance matrices or iteratively repeating the correction process until an error threshold is satisfied.

2.4.3.3 *Unscented Kalman Filter (UKF)*

Systems that are highly nonlinear and cannot be represented by first order linearization; they cannot be approximated using EKF. The Unscented Kalman filter can overcome this problem by representing the state with a minimal set of carefully chosen points, which are known as sigma points. Similar to LKF and EKF, the UKF uses the same predictor-corrector cycle with some additional steps [97, 98].The sigma points form a probability distribution with mean $\hat{x}_{k-1}$ and covariance P_{k-1}. This process is called an unscented transformation which is generally used in computing the statistics of random variables that undergo nonlinear transformations. Hence UKF algorithm have additional step beside the predictor and corrector which is the selection of $2n + 1$ sigma points where n represent the dimensions of the state space [67]. Again let's consider a system represented by the state space shown in equations (2.18) and (2.19). The correction-prediction cycle of the unscented Kalman filter can be written as follows:

Selection of sigma points

$$s_{k-1}^{0} = \hat{x}_{k-1} \qquad -1 < W^{0} < 1 \qquad (2.31)$$

$$s_{k-1}^{i} = \hat{x}_{k-1} + \left(\sqrt{\frac{n}{1-W^0}P_{k-1}}\right)_i \qquad i = 1,2,\dots,n \tag{2.32}$$

$$s_{k-1}^{i+n} = \hat{x}_{k-1} - \sqrt{\frac{n}{1-W^0}}\left(\sqrt{P_{k-1}}\right)_i \qquad i = 1,2,\dots,n \tag{2.33}$$

$$W^i = \frac{1-W^0}{2n} \qquad i = 1,2,\dots,2n \quad where \sum_i W^i = 1 \tag{2.34}$$

Sigma points are selected around the previous mean. $\left(\sqrt{\mathrm{P_{k-1}}}\right)_{\mathrm{i}}$ is the $\mathrm{i^{th}}$ column of matrix square root of the previous covariance. The weight of the first sigma point is set randomly, while other weights are proportional to it [98, 99].

Predictor Equations:

1. Projecting the estimate forward in time by firstly propagating the sigma point using the state transition function $\{f(s_{k-1}^{i})\}$

$$\hat{x}_k^- = \sum_{i=0}^{2n} W^i f(s_{k-1}^{i}) \tag{2.35}$$

2. Projecting the error covariance forward in time

$$P_k^- = \sum_{i=0}^{2n} W^i \left(f(s_{k-1}^{i}) - \hat{x}_k^-\right)\left(f(s_{k-1}^{i}) - \hat{x}_k^-\right)^T + Q \tag{2.36}$$

Corrector Equations:

- Propagating the observation through the nonlinear observation function

$$\hat{z}_{k-1} = \sum_{i=0}^{2n} W^i h(s_{k-1}^{i}) \tag{2.37}$$

- Auto-correlation of the previous measurement

$$P_z = \sum_{i=0}^{2n} W^i \left(h(s_{k-1}^{i}) - \hat{z}_{k-1}\right)\left(h(s_{k-1}^{i}) - \hat{z}_{k-1}\right)^T + R \tag{2.38}$$

- Cross correlation of previous measurement and estimated state

$$P_{xz} = \sum_{i=0}^{2n} W^i \left(f\left(s_{k-1}^i\right) - \hat{x}_k^-\right)\left(h\left(s_{k-1}^i\right) - \hat{z}_{k-1}\right)^T + R \qquad (2.39)$$

- Computing the Kalman gain

$$K_k = P_{xz} {P_z}^{-1} \qquad (2.40)$$

- Correcting the estimate using actual measurements

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - \hat{z}_{k-1}) \qquad (2.41)$$

- Correcting the error covariance

$$P_k = P_k^- - K_k P_z {K_k}^T \qquad (2.42)$$

Figure 2.8 illustrates the working principle of unscented Kalman filter and compares it with extended Kalman filter. UKF approximates the mean by a third order Taylor series while the EKF linearizes it by a first order Taylor series. For the covariance, both methods approximate it in the same manner. However, the covariance error in UKF is smaller than the one in EKF because its mean estimation is more accurate than EKF [68].

Although the unscented Kalman filter gives better approximation for nonlinear systems, it is still bounded by the Gaussian dynamics assumption. Thus if the systems has a non-Gaussian dynamics, Kalman filter fails to give satisfactory estimation.

In the published work, the unscented Kalman filter is associated with nonlinear systems that cannot be linearized using first order approximation such as articulated object tracking where the object is represented by a high dimension state space. Casuo et al. [69] used unscented Kalman filter for estimating hand pose where the hand is represented by voxel model extracted from silhouette image of the hand. Ziegler et al. [9] used unscented Kalman filter for tracking an articulated human body by estimating rotation angle of body joints using stereo images. Tsai et al. [70] used unscented Kalman filter for human robot interaction. This robot undergoes continuous rotation in order to compensate tracked object motion. This is a case where the system dynamics are highly nonlinear; thus it cannot be linearized using extended Kalman filter. Ponsa et al. [71] used unscented Kalman filter for vehicle tracking from a

moving vehicle. This is another example for nonlinear system that cannot be approximated with a linear one. The filter tries to estimate the motion of the tracked vehicle and the host vehicle simultaneously; this is done by change the projection angle that matches current image with a reference image for the tracked object.

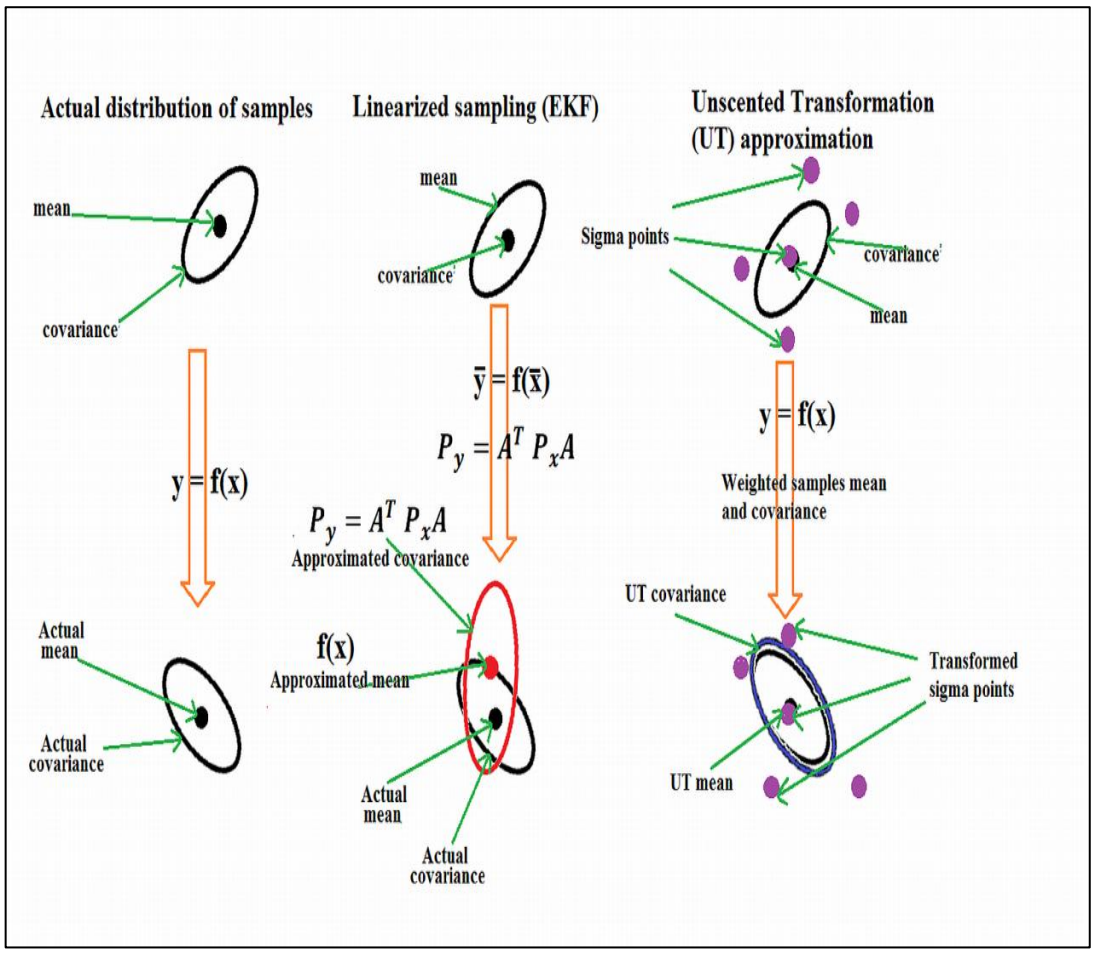


Figure 2.9: Comparison between EKF and UKF. The first part shows the actual nonlinear model [68].

2.4.4 Particle Filters

Particle filter is a technique for implementing a recursive Bayesian filter using Monte Carlo sampling [72]. Different particle filtering techniques exist in the literature. This section will explain the basic principles of particle filters, and then it discusses some of the common particle filters algorithms in the literature.

2.4.4.1 *Fundamentals of Particle Filters*

Particle filter is based on Bayes probability theorem and Monte Carlo sampling algorithm. Bayes' rule states that the posterior distribution of a stochastic process is a normalized value of the multiplication of the likelihood and the prior [47].

$$posterior\ probability \propto likelihood\ .prior \tag{2.43}$$

For a stochastic process with dynamic states x and observations y, the basic elements of Bayes' inference system are:

i. Likelihood probability$\{p(y|x)\}$: is the conditional probability of the observation *y* given the state *x*.

ii. Prior probability$\{p(x)\}$: Probability density functions of the state *x*.

iii. Posterior probability$\{p(x|y)\}$: is proportional to the product of the likelihood and the prior as expressed by the Bayes' rule.

$$p(x|y) = \frac{p(y|x)p(x)}{\int p(y|x)p(x)} \tag{2.44}$$

The term $\int \mathrm{p(y|x).p(x)}$ acts as a normalization factor. For dynamic system with states x_k and observations y_k, the elements of the Bayesian inference are written as:

$$p(x_k|y_k) = \frac{p(y_k|x_k)p(x_k|x_{k-1})}{\int p(y_k|x_k)p(x_k|x_{k-1})} \tag{2.45}$$

Monte Carlo method is a statistical sampling and estimation method. The most common type of Monte Carlo sampling is the importance sampling (IS); which aims to sample a distribution in the region of "importance" in order to achieve computational efficiency. The IS is best suited for high dimensional space where the data are sparse and the region of interest is relatively small compared to the data space [94, 102]. If there is a true probability distribution $p(x)$ which is hard to sample, IS method says that the true distribution can be replaced by a chosen proposal distribution $q(x)$. Figure 2.10 illustrates the concept of using a proposal distribution with importance weights (green circles) to approximate a true probability distribution of the system. Using the proposal distribution, the expectation value can be evaluated using equation (2.46).

$$E[f(x)] = \int f(x)p(x)dx = \int f(x)\frac{p(x)}{q(x)}q(x)dx \tag{2.46}$$

Monte Carlo importance sampling uses N number of independent samples, drawn from the proposed distribution $q(x)$ to obtain a weighted sum that approximates the above integral.

$$[f(x)] = \frac{1}{N}\sum_{i=1}^{N} W(x^{(i)})f(x^{(i)}) \tag{2.47}$$

Where $W(x^{(i)})$ in equation (2.47) is called the importance weight which is computed using equation (2.48). These weights have to be normalized to ensure that these weights sums to one. $\widetilde{W}(x^{(i)})$ is the normalized weight computed using equation (2.49).

$$W(x^{(i)}) = p(x^{(i)})/q(x^{(i)}) \tag{2.48}$$

$$\widetilde{W}(x^{(i)}) = W(x^{(i)})/\sum_{i=1}^{N} W(x^{(i)}) \tag{2.49}$$

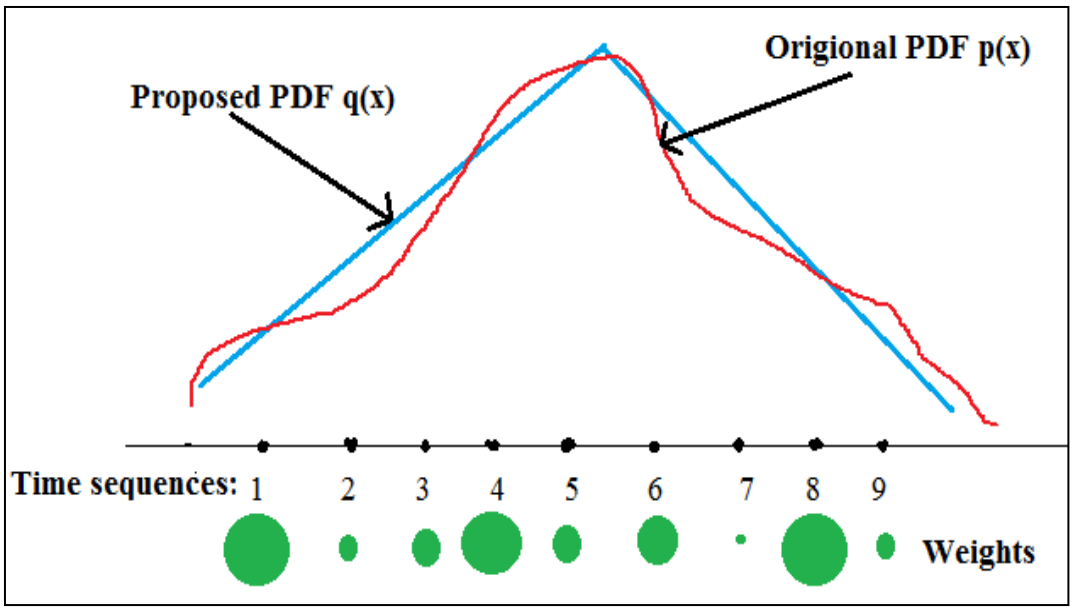


Figure 2.10: Illustration for the principle of weighted sampling.

2.4.4.2 *Generic particle filtering Algorithm (GPF)*

This is the original algorithm for particles filters which is based on Monte Carlo importance sampling and the Bayesian filters. Particle filter aims to approximate the posterior probability density function by selecting sample (particle) $x_k^{(i)}$ with a weight probability (weight) $w_k^{(i)}$ of x_k, k = 1…N [74]. This can be expressed mathematically as:

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{N} w_k^{(i)}\, \delta\left(x_k - x_k^{(i)}\right), \quad \sum_{i=1}^{N} w_k^{(i)} = 1, \tag{2.50}$$

The algorithm of the generic particle filter is shown in Table 2.4. It starts with drawing initial samples, then computing samples weights and finally computing the expectation value for the weighted samples.

Table 2.4: Pseudo-code for generic particle filtering algorithm.

1) IF k = 0 THEN
   Draw $x_0^{(i)} \mid i = [1, \cdots, N]$ from $p(x_0)$ using identical independent distribution and weights $w_k^{(i)} = \frac{1}{N}$
2) END IF
3) ELSE THEN
   i. inputs: $\langle x_{k-1}^{(i)}, w_{k-1}^{(i)} \rangle \mid i = [1, \cdots, N]$, observation z_k and number of samples N
   ii. FOR $i = 1, \cdots, N$DO
      a) Propagate new set of particles $x_k^{(i)}$by independently sampling the importance function $x_k^{(i)} \sim q(x_k \mid x_{k-1}^{(i)}, z_k)$
      b) Update the weight $\acute{w}_k^{(i)}$ of $x_k^{(i)}$ according to the likelihood $\acute{w}_k^{(i)} = w_{k-1}^{(i)} \frac{p\left(z_k \mid x_k^{(i)}\right) p\left(x_k^{(i)} \mid x_{k-1}^{(i)}\right)}{q\left(x_k^{(i)} \mid x_{k-1}^{(i)}, z_k\right)}$
   iii. END FOR
   iv. Weight normalization $w_k^{(i)} = \frac{\acute{w}_k^{(i)}}{\sum_i \acute{w}_t^{(i)}}$, $\sum_i w_k^{(i)} = 1$
   v. Compute the expectation value of x_k from the approximation $\sum_{i=1}^{N} w_k^{(i)} \delta(x_k - x_k^{(i)})$
4) END ELSE
5) At any time or depending on the efficiency criterion, resample $\langle x_k^{(i)}, w_k^{(i)} \rangle \mid i = [1, \cdots, N]$, by evenly weighted particles$\langle x_k^{(i)}, \frac{1}{N} \rangle \mid i = [1, \cdots, N]$,

As shown in Table 2.4, an initial distribution $q(x_k)$ is assumed for the state variables. This distribution has N equally spaced particles with equal weight of $1/N$. Then at every time instant a new set of particles $x_k^{(i)}$ are drawn by sampling from $q(x_k \mid x_{k-1}^{(i)}, z_k)$. Then the weights are updated using likelihood $p\left(z_k \mid x_k^{(i)}\right)$. After that, all weights are normalized to ensure that sum of weights is equal to one. It has to be noted that particles exhibit a degeneracy phenomenon; which means most of the particles tend to have zero weight and this reduces the number of effective particles used to simulate the distribution. Hence, it is important to resample the particles according to its weights wherever degeneracy is encountered; particles with high weights are duplicated while the ones with smaller weights are discarded [73]. This is known as particles resampling and it aims to prevent the degeneracy

The degeneracy phenomenon can be measured by calculating the number of effective particles [72]. Resampling enables the distribution to focus on effective regions and keep the number of particles fixed throughout the filtering process. Figure 2.11 illustrates the particles resampling process during which the particles with higher weights are duplicated while the ones with small weight are discarded. Excessive resampling could lead to sample impoverishment in which only certain points dominate the distribution and thus the distribution loses it diversity [72].

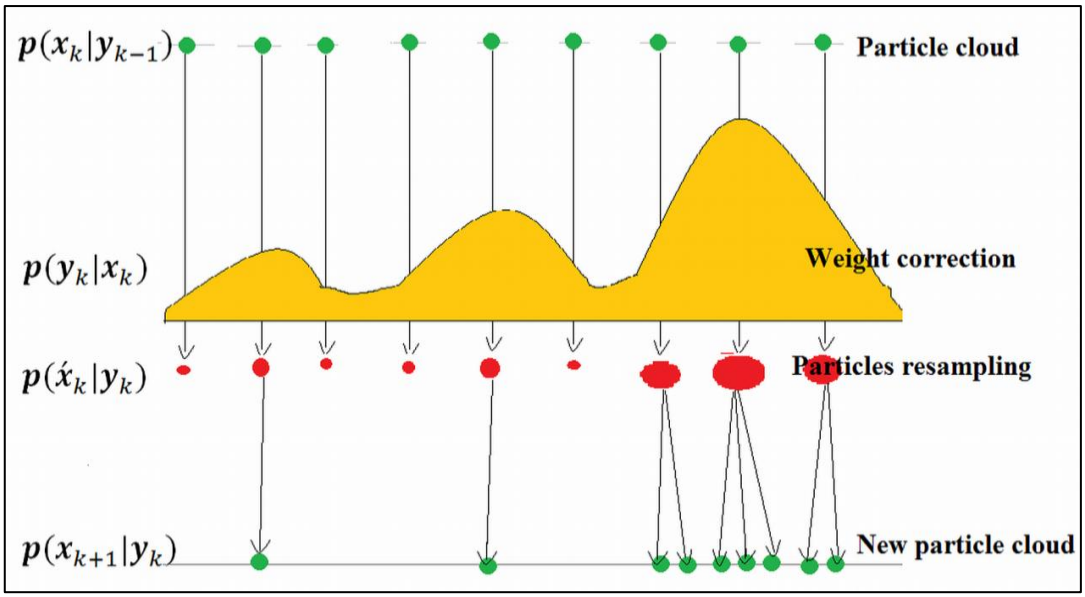


Figure 2.11: Particles resampling process.

This particle filter is considered to be the most commonly used particle filter. Taiana et al. [75] used particle filter for robot navigation using wide angle camera. Kim et al. [76] used particle filter for tracking an articulated body model where the body is represented by connected cylinders and the filter estimate the angles between these connected cylinders. In another work Catalin and Nedevschi [77] used particle filtering for vehicle tracking using stereo camera. Lao et al. [78] implemented particle filtering algorithm that generate particles sequentially. Sequential particle generation improves the sampling process and also reduces the number of particles needed. Ongkittikul et al. [79] used mean-shift method to improve the particles generating process by shifting lower weight particles to the neighboring higher weight ones. Shen et al. [80] combined particle filter with mean-shift method in order to enhance particle sampling and prevents the occurrence of particles degeneracy.

Feng et al. [81] reduces the computational time of particle filters by omitting the unchanged variables from the prediction cycle. This is done by comparing current and previous measurements and if there is no change then these variables and not predicted but rather replaced by previous estimation multiplied by measurement confidence while the changing variables are predicted using the normal particle filtering algorithm. Bray et al. [82] used stochastic meta-descent algorithm to improve the particle filtering algorithms. Pupilli and Calway [83] used particle filter to recursively estimate the pose of handheld camera based on matching edge map of the scene with predefined edges of the scene. Zheng and Bhandarkar [84] combined iterative particle filtering and adaboost algorithm for face detection and tracking.

2.4.4.3 *Auxiliary Particle Filtering Algorithm (AuxPF)*

An optimal filtering scheme must define the proposed distribution (PDF) to be equivalent to the original distribution. For APF, an auxiliary weight $\lambda_k^{(i)} \propto w_{k-1}^{(i)}\, p(z_k \,|x_{k-1}^{(i)})$ is computed before drawing the samples $x_k^{(i)}$. Then, the weighted particles set $\{x_{k-1}^{(i)},\ \lambda_k^{(i)}\}|_{i=1}^{N}$ is resampled which represent a smoother version of probability distribution $p(x_{k-1} \mid z_{1:k})$. After that, auxiliary particles $x_{k-1}^{(s^{(i)})}$ are generated from the weighted samples by identical independent sampling. The

approximation $\{x_{k-1}^{(i)},\ \lambda_k^{(i)}\}|_{i=1}^{N}$ is resampled into $\left\{x_{k-1}^{(s^{(i)})}, \frac{1}{N}\right\} \sim p\left(x_k^{(i)} \middle| x_{k-1}^{(s^{(i)})}, z_k\right)$. Finally particles are independently sampled from the auxiliary ones $x_k^{(i)} = p\left(x_k^{(i)} \middle| x_{k-1}^{(s^{(i)})}, z_k\right)$. The weights must be corrected to account for dissimilarity between $\lambda_k^{(i)}$ and $w_k^{(i)}$ [90, 100]. Table 2.5 gives a detailed description of the auxiliary particle filter algorithm.

Table 2.5: Pseudo-code for auxiliary particle filter [90, 100].

1) IF k = 0 THEN
   Draw $x_0^{(i)} | \ i = [1, \cdots, N]$ from $p(x_0)$ using identical independent distribution and weights $w_k^{(i)} = \frac{1}{N}$
2) ELSE THEN
   i. inputs: $\langle x_{k-1}^{(i)}, w_{k-1}^{(i)} \rangle | \ i = [1, \cdots, N]$, observation z_k, number of samples N
   ii. FOR $i = 1, \cdots, N$DO
      a) Generate auxiliary weights $\acute{\lambda}_k^{(i)} \propto w_{k-1}^{(i)} \hat{p}\left(z_k \middle| x_{k-1}^{(i)}\right)$
   iii. END FOR
      a) Weights normalization $\lambda_k^{(i)} = \frac{\acute{\lambda}_k^{(i)}}{\sum_i \acute{\lambda}_k^{(i)}}$, $\sum_i \lambda_k^{(i)} = 1$
   iv. FOR $i = 1, \cdots, N$DO
      a) Generate auxiliary particles $x_{k-1}^{(s^{(i)})}$by i.d.d sampling from $\langle x_{k-1}^{(i)},\ \lambda_k^{(i)} \rangle$
      b) Assign equal weights to each particle $\langle x_{k-1}^{(s^{(i)})}, \frac{1}{N} \rangle$ where $\frac{1}{N} \sum_i \delta\left(x_{k-1} - x_{k-1}^{(s^{(i)})}\right) \approx p(x_{k-1} \mid z_{1:k})$
   v. END FOR
   vi. FOR $i = 1, \cdots, N$DO
      a) Generate the particles by independently sampling $x_k^{(i)} \sim p\left(x_k \mid x_{k-1}^{(s^{(i)})}\right)$
      b) Update weights $\acute{w}_k^{(i)} \propto p\left(z_k | x_{k-1}^{(i)}\right) / \hat{p}\left(z_k \middle| x_{k-1}^{(s^{(i)})}\right)$
   vii. END FOR
   viii. Weights normalization $w_k^{(i)} = \frac{\acute{w}_k^{(i)}}{\sum_i \acute{w}_k^{(i)}}$, $\sum_i w_k^{(i)} = 1$
   ix. Compute the expectation value of x_k from the approximation $\sum_{i=1}^{N} w_k^{(i)} \delta(x_k - x_k^{(i)})$
3) END ELSE
4) At any time or depending on the efficiency criterion, resample $\langle x_k^{(i)}, w_k^{(i)} \rangle | \ i = [1, \cdots, N]$, by evenly weighted particles$\langle x_k^{(i)}, \frac{1}{N} \rangle \ | \ i = [1, \cdots, N]$,

McKenna and Nait-Charif [85] used auxiliary with Iteratively Likelihood Weighting for head tracking using wide angle camera. ILW is a process of resampling the particles in order to concentrate it around the higher likelihood region. Brethes et al. [74] used several particle filters for mobile robot navigation using a camera mounted on the robot's head. In this work the authors implemented auxiliary particle filter, SIR particle filter and hierarchical particle filter. These filters were evaluated using different number of particle and with various visual cues such as the tracked object color, shape and motion. This work presents a comprehensive evaluation of particle filters. In a different work, Peursum et al. [86] used different particle filters for smoothing the results of articulated body tracking. The particle filters used were generic particle filter, annealed particle filter and factored sampling hierarchical particle filter. According to [86] smoothing tracking results using particle filters did not yield any improvement in tracking accuracy but rather added more computational complexity.

2.4.4.4 *Sequential Importance Resampling Particle Filtering Algorithm (SIRPF)*

SIR is known as the Condensation algorithm which is an abbreviation for Conditional Density Propagation algorithm. In SIR algorithm, firstly new auxiliary particles are generated from the prior as in equation (2.51), this is known as the deterministic drift step. Then the new particles are generated from the auxiliary ones by a stochastic sampling as shown in equation (2.52). $A\ and\ B$ matrices are the coefficient matrices for the deterministic drift and stochastic diffusion respectively. This is known as the stochastic diffusion step. Finally, the weights are updated and normalized. The complete description of the condensation algorithm is illustrated in Table 2.6 [87].

$$x_{k-1}^{(s^{(i)})} \sim p(x)|_{i=1}^{N} \aleph \tag{2.51}$$

$$x_k^{(i)} = A x_{k-1}^{(s^{(i)})} + B \aleph \tag{2.52}$$

Table 2.6: Pseudo-code for the Condensation algorithm [87].

1) IF k = 0 THEN
   Draw $x_0^{(i)} | \ i = [1, \cdots, N]$ from $p(x_0)$ using identical independent distribution and weights $w_k^{(i)} = \frac{1}{N}$
2) ELSE THEN inputs: $\langle x_{k-1}^{(i)}, w_{k-1}^{(i)}, c_{k-1}^{(i)} \rangle \mid \ i = [1, \cdots, N]$, observation z_k and number of samples N
   i. FOR $i = 1, \cdots, N$ DO
      a) Select sample $x_k^{\prime(i)}$
         - Generate a uniformly distributed random number $r \in [0,1]$
         - Find the smallest j for which $c_{k-1}^{(j)} \geq r$
         - $x_k^{\prime(i)} = x_{k-1}^{(j)}$
      b) Predict new sample $x_k^{(i)} = Ax_{k-1}^{(s^{(i)})} + B\aleph$ where $\aleph \rightarrow$ noise; A,B $\rightarrow$ coefficients
      c) Compute weight $w_k^{(i)} = p\left(z_k | x_{k-1} = x_k^{(i)}\right)$
   ii. END FOR
   x. Normalize weights $w_k^{(i)} = \frac{w_k^{(i)}}{\sum_i w_k^{(i)}}$, $\sum_i w_k^{(i)} = 1$
   iii. Compute the accumulative weight $c_k^{(0)}, c_k^{(j)} = c_k^{(j-1)} + w_k^{(j)} \ where \ j = [1, \ldots, N]$
   iv. Compute the expectation value of x_k from the approximation $\sum_{i=1}^{N} w_k^{(i)} \delta(x_k - x_k^{(i)})$
3) END ELSE

SIR particle filtering algorithm was proposed by Israd and Blake [87] for tracking curves in dense scene. In SIR algorithms, prior particles are resampled using previous weights in order to get enhanced prior. Meuter et al. [12] employed SIR particle filter for tracking traffic sign board from a moving vehicle. Bando et al. [88] combined SIR particle filter with annealed particle filter for tracking a free moving ball. The SIR filter enhances particles prior while the annealed particle filter enhances the likelihood computations.

2.4.5 Object Tracking in Computer Vision

When implementing object tracking in computer vision, measurements are extracted from images. These measurements include extracting information about the location, size and color of the tracked object. There are important parameters that should be

defined prior to tracking such as the selection of visual cue, defining the likelihood function and defining a suitable model for the object motion using state space representation [74].

2.4.5.1 *Defining State Space Model*

State space variables are the variables to be estimated throughout the filtering process. For example, for tracking an object in a 2D scene, the state variables can be the x, y coordinates of the objects as well as their first derivative (velocity). In addition, the dominant object color could also be added, the state matrix is written as $x = [x, y, \dot{x}, \dot{y}, c]^T$. If the object is being tracked in a 3D scene, then its state variables will include the third dimension (z), thus $x = [x, y, z, \dot{x}, \dot{y}, \dot{z}, c]^T$. After selecting the state variables, state space for the system is formed using physical relationships between state variables such as relationships between distance, velocity and acceleration.

Let's consider an object having a constant velocity motion similar to the one shown in Figure 2.11. In this figure, the triangle represents the original position of the moving object while the circle represents its new position. Constant velocity model means that the acceleration is zero and thus it is omitted from the state space.

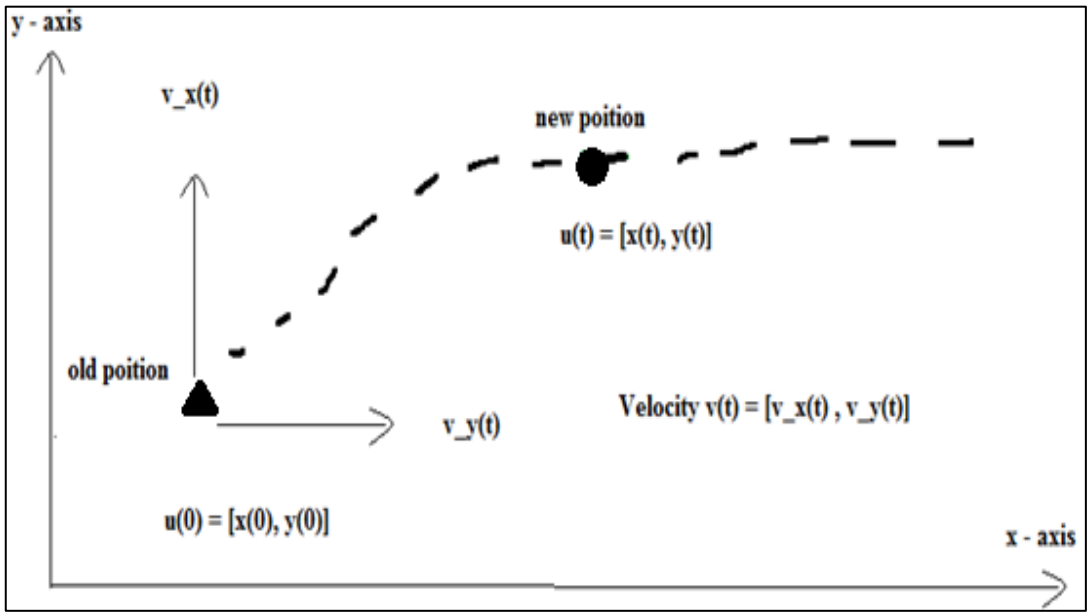


Figure 2.12: Illustration for a constant velocity drifting point motion.

Using a drifting motion model, the new position (u_t) and new velocity (v_t) can be derived as in equation (2.53) and (2.54). Δt is the time step, ϵ_t and ε_t are noise terms for the position and velocity respectively. The final state space model can be expressed as in equation (2.56) where W_i is the noise matrix and A_{i-1} is the state transition matrix.

$$u_t = u_{t-1} + \Delta t v_{t-1} + \epsilon_t \tag{2.53}$$

$$v_t = v_{t-1} + \varepsilon_t \tag{2.54}$$

$$\underbrace{\begin{bmatrix} u_t \\ v_t \end{bmatrix}}_{X_t} = \underbrace{\begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}}_{A_{t-1}} \underbrace{\begin{bmatrix} u_{t-1} \\ v_{t-1} \end{bmatrix}}_{X_{t-1}} + \underbrace{\begin{bmatrix} \epsilon_t \\ \varepsilon_t \end{bmatrix}}_{W_t} \tag{2.55}$$

$$X_t = A_{t-1}X_{t-1} + W_t \tag{2.57}$$

If the moving target is exhibiting a varying velocity but with a constant acceleration, the new position (u_t), new velocity (v_t) and new acceleration (a_t) can be written as in equations (2.58) – (2.60). Δt is the time step, ϵ_t, ε_t and ς_t are noise terms for the position, velocity and acceleration respectively. The final state space model can be expressed as in equation (2.62) where W_i is the noise matrix and A_{i-1} is the state transition matrix.

$$u_t = u_{t-1} + \Delta t v_{t-1} + \epsilon_t \tag{2.58}$$

$$v_t = v_{t-1} + \Delta t a_{t-1} + \epsilon_t \tag{2.59}$$

$$a_t = a_{t-1} + \varsigma_t \tag{2.60}$$

$$\underbrace{\begin{bmatrix} u_t \\ v_t \\ a_t \end{bmatrix}}_{X_t} = \underbrace{\begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}}_{A_{t-1}} \underbrace{\begin{bmatrix} u_{t-1} \\ v_{t-1} \\ a_{t-1} \end{bmatrix}}_{X_{t-1}} + \underbrace{\begin{bmatrix} \epsilon_t \\ \varepsilon_t \\ \varsigma_t \end{bmatrix}}_{W_t} \tag{2.61}$$

$$X_t = A_{t-1}X_{t-1} + W_t \tag{2.62}$$

2.4.5.2 *Measurement Function*

The measurement function is used to obtain the current observations from the image. Measurements are used to correct the Kalman filter estimation and in particle filter it

is used to update the particles' weights. The measurements are based on physical cues obtained from the image such as shape cue, motion cue and color cue [74].

The use of shape feature for likelihood computation requires learning the shape of the object beforehand. Object shape can be characterized using silhouette, template, and skeleton or object contour [1]. When using shape features, at the initial stage of the tracking, a shape template $x(j)$ is computed. Then when measurements $z(j)$ are presented, the similarity between the shape template and measured shape is evaluated using a distance $\{D\}$ evaluation as shown in equation (2.63). Finally the likelihood is computed using equation (2.64). N_p is the number of points in the shape template and σ_s is the standard deviation for the likelihood function [74].

$$D = \sum_{j=1}^{N_p} |x(j) - z(j)| \tag{2.63}$$

$$p(z^S|x) \propto \exp\left(-\frac{D^2}{2\sigma_s^2}\right) \tag{2.64}$$

Object color can be represented using normalized histograms of N_p points. Let h_{ref}^c $c \in \{R, G, B\}$ to be color histogram of a color image. The histogram model for each color channel for the object x is $h_x^c$where $c \in \{R, G, B\}$. In a similar manner to shape features, the similarity between a reference color histogram and the current color histogram of the object is evaluated with Bhattacharyya distance as shown in equation (2.65). Finally the likelihood is computed for each channel in the image using equation (2.66). The reference histogram is learnt before the tracking starts. σ_s is the standard deviation for the color likelihood function [74].

$$D\left(h_x^c, h_{ref}^c\right) = \sqrt{\left(1 - \sum_{j=1}^{N_p} \sqrt{h_{j,x}^c . h_{j,ref}^c}\right)} \tag{2.65}$$

$$p(z^C|x) \propto \exp\left(-\sum_{c=R,G,B} \sum_{j=1}^{N_p} \frac{D^2\left(h_{j,x}^c, h_{j,ref}^c\right)}{2\sigma_c^2}\right) \tag{2.66}$$

Motion feature employs the frame difference information computed for successive frames. Optical flow is the most common technique for object motion detection.

Firstly the background of the scene is identified; this is achieved via median averaging or by assigning a specific frame to be the background. After that, the background is subtracted from every frame and the regions of motion will be highlighted in the foreground image. The histogram of the foreground image for motionless object falls in the lower bins. For regions with motion, the histogram will have higher values. In order to compute the likelihood, the reference histogram is set to lower bin value as in equation (2.67). Equation (2.68) explains the likelihood computations for motion cue. $h_{j,x}^{M}$ is the current frame histogram and σ_M is the standard deviation for the likelihood function [74].

$$h_{j,ref}^{M} = \frac{1}{N_p} \quad j = 1, \ldots., N_p \tag{2.67}$$

$$p(z^M|x) \propto \exp\left(-\frac{D^2\left(h_{j,x}^{M}, h_{j,ref}^{M}\right)}{2\sigma_M^2}\right) \tag{2.68}$$

Multiple visual cues can be combined in order to get robust measurement function. If there are N visual cues to be combined, the final likelihood is a multiplication of these individual likelihood functions as it is shown in equation (2.69) [74].

$$p(z^1, \cdots, z^N|x) = \prod_{i=1}^{N} p\left(z^i|x\right) \tag{2.69}$$

2.5 **Chapter Summary**

This chapter summarizes the published work on the main components of a 3D tracking system. The aim of this summary is to establish a background about existing 3D tracking systems and select the best methods to be used in the proposed 3D tracking scheme. Firstly, this section discussed about object tracking techniques in general specially background modeling techniques because it is the most suitable one to be used in cluttered environments. It also discussed about shadow removal and object representation techniques. Moreover, the section concluded that multi-point pair background model is the best one among the existing background modeling techniques to be used for visual surveillance.

Secondly, this chapter discussed about the existing depth estimation techniques and highlighted their advantages and disadvantages. Active depth estimation methods involve higher cost. Stereovision, structure from motion and shape from focus use multiple cameras or multiple images and they require high computational complexity. Other monocular depth estimation methods work in specific situations. Multiple depth cues give better accuracy and can work for various scenarios. However, it involves a higher computational burden which is not desirable for visual tracking. Existing depth estimation techniques are not suitable for real time applications. Thus, a new depth estimation technique is developed for the proposed 3D tracking system.

Finally, this chapter discussed existing object tracking methods and listed their implementations in the published work. Kalman filters and particle filters are the most widely used object tracking methods. The section summarized previous works about three types of Kalman filters and three types of particle filters. In the literature, there is no objective comparison between these methods. Thus, it is hard to judge which method has superior accuracy. Therefore, an objective comparison between these methods has been conducted in chapter 2 in order to choose the most accurate estimation techniques to be used in the proposed 3D tracking system.

CHAPTER 3
OBJECT DETECTION AND TRACKING

3.1 **Introduction**

Any tracking system requires a detection method in order to identify the location of the object of interest in image coordinates. This section implement the method developed in the previous chapter for object detection using background subtraction. Also it implements shadow removal techniques for foreground image filtering. In addition, this chapter conducts a comparative study of six Kalman filters and particle filters objectively based on computational time and estimation accuracy. The selected methods from this chapter will be used for implementing 3D tracking system using single camera in chapter 5.

3.2 **Moving Object Detection**

Accurate detection of moving objects in the scene is a key element for accurate depth estimation. As it was stated in chapter 2, background subtraction is the best method for detecting an unknown object because it does not require any prior knowledge about the moving object compared to template matching and segmentation techniques. Figure 3.1 shows a flow diagram for an object detection scheme using background subtraction. Firstly, the background is modeled for the first M frames. The background modeling is performed only one time assuming that the background is stationary. If the background is dynamic then this process is repeated frequently and the moving object is detected by comparing the new frame with the learnt background. After that, the detected object is filtered using shadow removal component and if there is any occlusion, occlusion compensation is performed. Finally, the location and the size of the moving object are extracted. The bottom location of the moving object will be used by depth estimation method for computing the depth and calculating the world coordinates of the moving object.

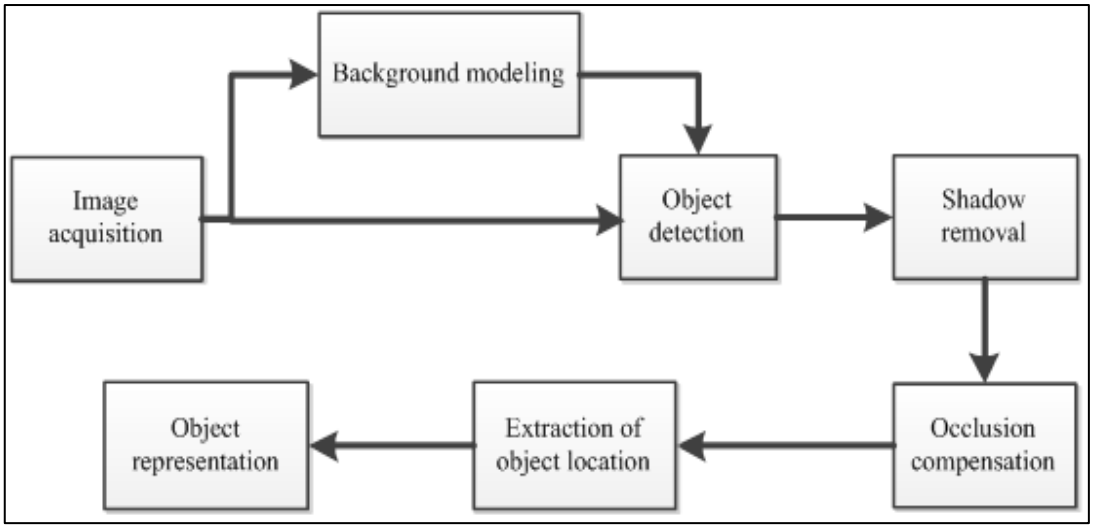


Figure 3.1: Flow diagram of moving object detection.

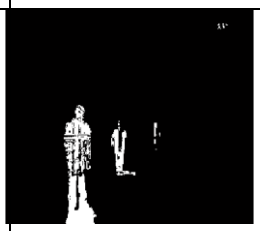
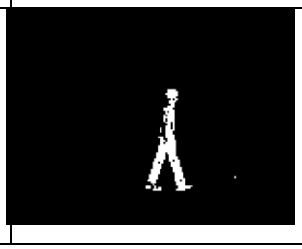
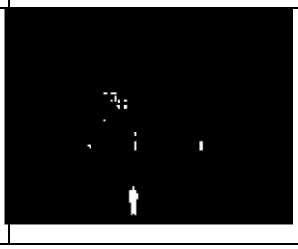
This section implement background modeling and background subtraction using the Grayscale Arranging Pairs (GAP) introduced in the previous chapter and compares with other existing background modeling techniques. After that, this section implement shadow removal methods using luminance and chromaticity gain proposed by [25] on several images. Finally, this section shows object representation using its centroid and object representation using its bottom most point in the image.

3.2.1 Object Detection Using GAP Algorithm

Table 3.1 compares the detection results obtained by these method using three video sequences. The first row represents the current frame while the second row represents the background of the scene obtained using median filtering. The third row shows the foreground image obtained by subtracting background from current frame where the shadow effect is very prominent especially in sequence 1 and sequence 2. In the fourth row, the moving object is been extracted using Otsu's threshoding method. In the first sequence, the shadow has been considered as a foreground element. Similar effect can be seen in the other two sequences. The fifth row shows the detected moving object using MoG method. MoG managed to eliminate some of the shadow regions but there are considerable numbers of false alarms from the background particularly in the second sequence. The last row shows the detected moving object obtained by the GAP algorithm. GAP eliminated all shadow regions in the image, this

is clear in first sequence where the shadow blob is been removed. These results show that GAP method is better than median filter and MoG background subtraction algorithm. In [21] the GAP method was compared with state the art background subtraction algorithm and it showed better results than all of them.

Table 3.1: Object detection using GAP method and median filtering.

	Sequence 1	Sequence 2	Sequence 3
Image			
Background using median filtering			
Foreground			
Detected objects using Otsu's thresholding			
Detected objects MoG method			
Detected objects using GAP method			

3.2.2 Shadow Removal Using Luma/Chroma Gain

Figure 3.2 shows shadow removal examples. The shadow presents in the foreground image is detected as an object when threshoding is applied. In the last column, the shadow is been eliminated using the proposed method. Using traditional background subtraction techniques shadow regions are identified as blobs during the thresholding process. This will results in false alarms and increase the number of tracked object. In the last column, the shadow is removed considering the luminance and the chromaticity gain as shown in equations (2.1-2.2).

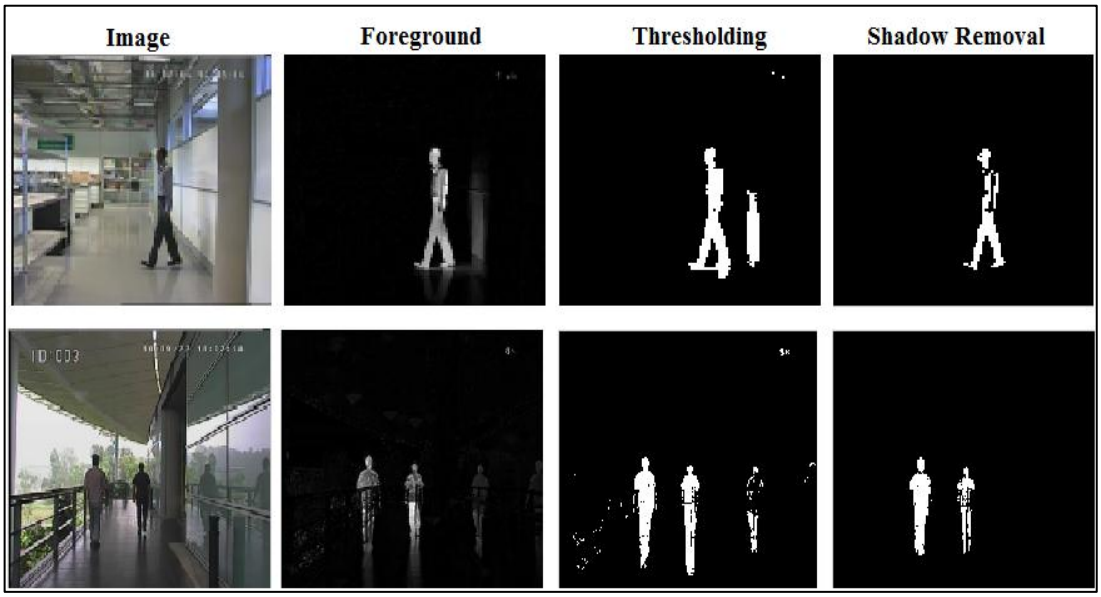


Figure 3.2: Shadow removal using the luminance and chromaticity gain.

3.2.3 Occlusion Compensation

Locating an occluded object from single view can be a nightmare because not much information can be extracted from the image. The only way forward is to use the motion history of the object for estimating the new location. If there is an object in image J_{k-1} at location $I(x, y)$ and with motion velocity $v(\dot{x}, \dot{y})$, assuming that this object undergoes a fixes velocity motion. In the new frame J_k if this object is fully or partially occluded in a way that its location is not measureable from the image. The new location can be estimated from the previous frame data using equation (3.3) where Δt is time step and ϵ is a random noise.

$$J_k(x,y) = J_{k-1}(x,y) + v_{k-1}(\dot{x},\dot{y}) \times \Delta t + \epsilon \quad (3.1)$$

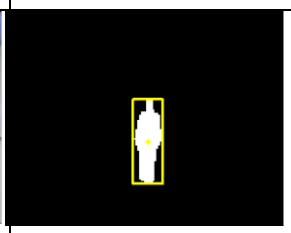
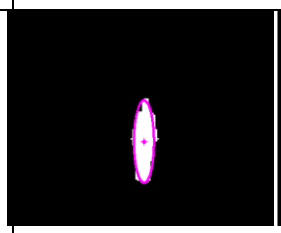
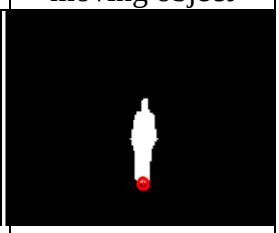
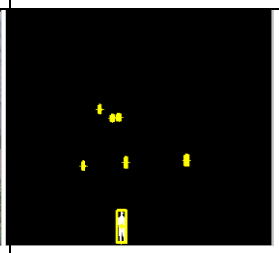
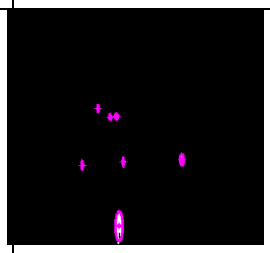
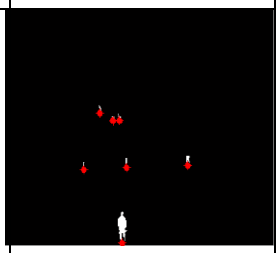
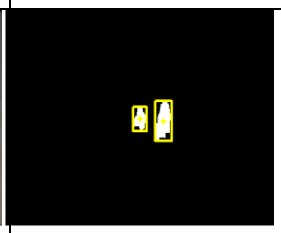
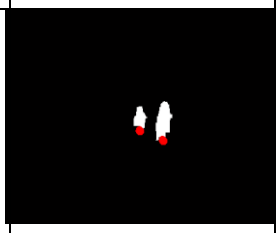
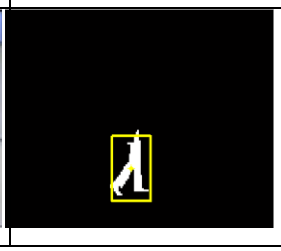
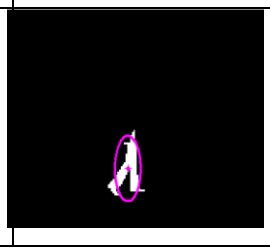
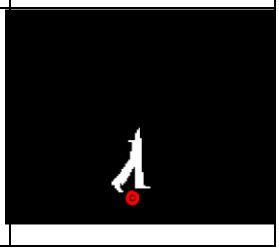
3.2.4 Moving Object Representation

Primitive representation has been adopted for representing the moving object where the moving object is represented by a rectangular patch or elliptical patch. The size information is extracted from the image using connected component analysis. For representing a rectangular patch, the center point of the connected component is computed and the width and the height as well. Similarly for representing the object with elliptical patch, the center location is computed from the image in addition to the major axis and minor axis of the object. Moreover, elliptical representation requires measuring the orientation angle as well. In Table 3.2 shows rectangular and elliptical patches representation of different video sequences. The last column shows feet location of the moving object which is required by the depth estimation algorithm of Chapter 4 in order to compute the distance between the moving object and the camera. The feet location is directly computed from the rectangular patch representation using equations (3.2) and (3.3) where (W) is the width of the blob.

$$x_{feet} = x_{centroid} \quad (3.2)$$

$$y_{feet} = y_{centroid} + \frac{W}{2} \quad (3.3)$$

Table 3.2: Object representation schemes.

Original image	Rectangular patch	Elliptical patch	Feet location of moving object
			
			
			
			

3.3 **Qualitative and Quantitative Comparison of Stochastic Filters**

All tracking algorithms described previously estimate the next state of the moving object using previous estimation and measured observations. Kalman filters approximate system dynamics using a Gaussian probability distribution. While particle filters uses the Monte Carlo approximation for estimating the next state of the system. Although these filters are widely used for object tracking for a long time, no attempt was made for an objective comparison between them. But rather these methods were discussed separately for different applications. In order to evaluate these 3D tracking algorithms quantitatively and qualitatively, we have implemented particle filter and Kalman filter tracking methods and tested them on several video

sequences for indoor and outdoor scenes. The implemented methods are listed in Table 3.3 along with visual cues used for each method, the estimated state variables and the related work published in the literature. This comparison aims to identify the method that has the best estimation accuracy among the discussed ones.

The performance measures used for evaluating these methods are root mean square error (E), computational time per frame and correlation. This error measures the mean error in term of pixels as the case for estimating the object position or the size of the bounding ellipse or in term of degrees for estimating the orientation angle. The time per frame is the time elapsed for performing the algorithm on single frame from image acquisition until displaying the tracking results. Correlation is a dimensionless measure that relates the estimated value to the ground truth data. Equations (3.4) and (3.5) are for the root mean square error and correlation respectively [31], [89] and [90].

Table 3.3: The implemented methods and visual cues used in the comparison.

Algorithm	**Visual cue**	**Estimated state variables**	**Related work in the literature**
Kalman filters (linear, extended and unscented)	Shape and motion features	Object location (x,y,z), object velocity, bounding ellipse size and the ellipse orientation angle	[62], [61], [64] and [71]
Particle filters (generic, auxiliary and the condensation particle filter)	Shape and motion features	Object location (x, y, z), object velocity, bounding ellipse size and the ellipse orientation angle	[77], [91], [86] and [8]
Particle filters (generic, auxiliary and the condensation particle filter)	Color features	Object location (x, y, z), object velocity, bounding ellipse size and the ellipse orientation angle	[88], [80], [91], [92], [75] and [79]

$$E = \sqrt{\frac{1}{NumFrames} \sum_{i}^{NumFrames} (X_i - Z_i)^2} \tag{3.4}$$

$$r_{xy} = \frac{\sum_{i}^{NumFrames}\left((X_i - \bar{X})(Z_i - \bar{Z})\right)}{\sqrt{\sum_{i}^{NumFrames}(X_i - \bar{X})^2}\sqrt{\sum_{i}^{NumFrames}(Z_i - \bar{Z})^2}} \quad (3.5)$$

3.3.1 Data Collection

We have collected several video sequences for evaluating the selected tracking algorithms. The images have been captured using Samsung high resolution 37X zoom color camera, model number SDZ-375. These images are scaled down from (704 x 576) to smaller size as explained in table 9 which helps in reducing the computational time [93]. Figure 3.3, shows the data collection system, which contains the zoom camera, video recorder and a computer.

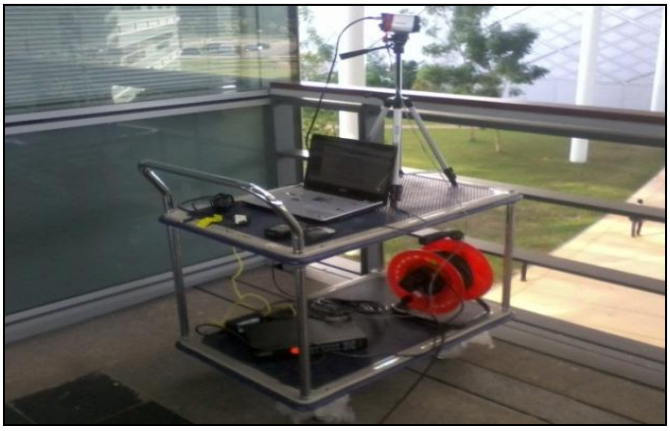


Figure 3.3: Data collection unit.

Table 4.4 illustrates the video sequences used for testing in this manuscript. The word custom in Table 3.4 indicates that these data have been generated by us while standard indicates that these video sequences have been used in the already published literature. First three sequences are collected in indoor scenes. These sequences contain people walking in front of the camera. The second three sequences have been captured in outdoor environment. It contains people as well as vehicles moving in different directions. In addition these sequences include static occlusion and crowded scenes. Additional data have been used from standard computer vision datasets. Sequences 1, 2, 3 and 4 have been collected from the CAVIAR project at the university of Edinburgh [94], PETS datasets at the university of Reading [95] and sequence 5 has been taken from Matlab demos.

Table 3.4: Summary of dataset collected.

Video sequence	No. Frames	Size	Type	Description
Sequence 1	300	288 x 384	standard	Vehicle motion in highway taken CVAIR [94]
Sequence 2	300	288 x 384	Standard	Human motion in a corridor taken from CVAIR [94]
Sequence 3	612	288 x 384	Standard	Human motion a room taken from PETS [95]
Sequence 4	190	120 x 160	Standard	Vehicle motion in highway taken from Matlab
Sequence 5	500	120 x 160	Custom	Indoor environment, the depth changes
Sequence 6	310	120 x 160	Custom	Indoor environment, the depth is fixed
Sequence 7	300	240 x 320	Custom	Indoor, multiple object, occlusion
Sequence 8	500	480 x 640	Custom	Vehicle tracking in outdoor
Sequence 9	200	576 x 704	Custom	Tracking in crowded environment (outdoor)

3.3.2 Experiment Setup

We have adopted a common test bed for testing six tracking algorithms. We only used these methods because it clear from the literature that these are the best existing algorithms for object tracking. The evaluation includes computing the estimation accuracy as well as the computational time taken by each one of these method. In order to compute the computational time we have implemented these methods on two computers with different specifications:

- Portable computer (laptop) with Intel Processor 2.1 GHz and 3GB RAM.
- Workstation with Intel i7 CPU 2.80 GHz and 6.0 GB RAM.

The moving object is been detected in videos using the object detection algorithm described in chapter 2. Kalman filter and particle filter has been tuned for the specific tracking scenario by specifying the motion model and initializing the covariance matrices.

3.3.2.1 *Tuning Kalman filters*

A constant velocity (zero acceleration) motion model is assumed throughout the tracking process. This model assumes a drifting point tracking scenario in which the new point is the old one plus noise as it has been shown in the previous chapter. Kalman filter requires initializing the state space matrices of equation (2.18) and (2.19). In general for a Kalman filter, matrices $\{F, Q, H$ and $R\}$ have to be specified prior to tracking. Matrix F which is the state transition matrix has been calculated from equation (2.62). For noise covariance, we assumed an identical noisy effect on all state space parameters; thus, Q and R have been set to identity. H matrix is the measurement transition matrix which is set to identity too since we measure the actual value of the estimated state.

3.3.2.2 *Tuning particle filters*

State space matrices of particle filter has the same value as that for Kalman filter since the same motion model is used. The number of particles is selected in order to reduce the estimation error and the computational time. Table 3.5 shows the computational time and estimation accuracy of three types of particle filters at a varying number of particles. Similarly is shown in Figure 3.4 with more number of particles. As the number of particle increase, the computational time increase while the error is not changing significantly. As a result, 500 particles are found to be the most appropriate number to be used throughout the coming experiments. This number of particles has small error and also short computational time.

Table 3.5: computational time and estimation error computed for particle filters at a varying number of particles

# particle	Computational time (sec)			Estimation error (pixel)		
Filter	GPF	SIR-PF	Aux-PF	GPF	SIR-PF	Aux-PF
50	0.09228	0.0844034	0.08439	2.252467	1.989355	2.27333
250	0.10921	0.1108619	0.11594	1.770901	1.424242	1.66339
500	0.10983	0.1180892	0.12508	1.944094	1.881090	1.42589
750	0.11030	0.1283104	0.14209	1.903934	2.423164	2.11270
1000	0.11080	0.1407283	0.16317	1.423323	1.493355	2.27556
1500	0.11027	0.174931	0.22449	1.707053	1.724448	2.06860
2000	0.11435	0.209213	0.30866	2.020600	1.645364	2.53650
2500	0.11513	0.269421	0.39922	1.624925	2.256329	2.45133
3000	0.11478	0.339741	0.51491	2.130686	1.629254	1.60344

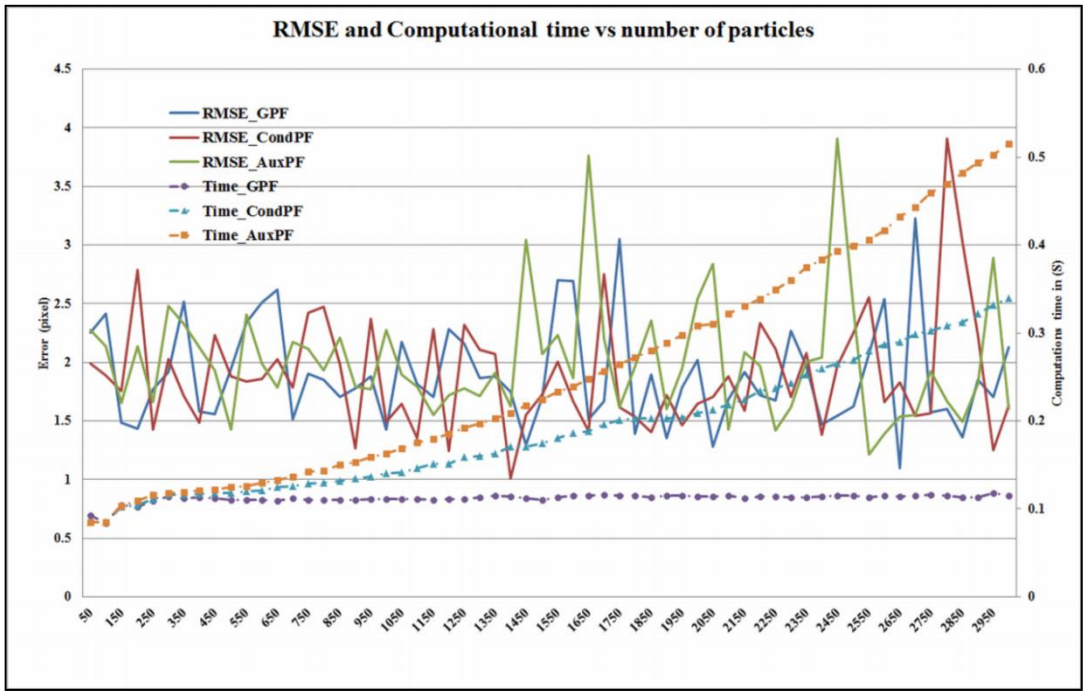


Figure 3.4: Variations of mean square error value and computational time of particle filters with the number of particles.

3.3.3 Subjective Comparison of Stochastic Filters

Table 3.6 displays tracking results using the tracking algorithms discussed in Section 2.3. These results are for selected frames from video sequence 5. The frames have been selected randomly. The first row shows the original images. The remaining rows show tracking results for the implemented algorithms. The true object location is represented by a blue box while the estimated target location is represented by a red box.

Table 3.6 : Results for implementing six tracking algorithms using the single moving object motion sequence.

Frame No.	70	151	270	367	490
Original Image					
LKF					
EKF					
UKF					
GPF					
SIR-PF					
AuxPF					

The second row of Table 3.6 until the fourth row show the tracking results for linear, extended and unscented Kalman filters respectively. From the table; these filters appear to have similar performance with great accuracy since the red box overlay the blue one. This mean the estimation of the center point and the box size is similar to the measured center point and box size. The last three rows display the tracking results for generic, SIR and auxiliary particle filters. It is clear that all of them have less accurate results than the one obtained by Kalman filters. The generic particle filter has clear deviation in frame number 270 and frame number 490 while the SIR particle filter has clear deviation in frame 70 and frame 270. The auxiliary particle filter has shown good results for all selected frames, however, it is still less accurate than the one achieved by the Kalman filters. In general, Table 3.6 shows that the accurate performance with small errors.

The tracking algorithms have been tested on additional video sequences in order to validate the earlier results. Table 3.7 shows tracking results for two indoor video sequences and two outdoor video sequences. The first row shows the original images while the second row shows the detected moving object. The images have been pseudo-colored to indicate different object in the case of occlusions. The third row shows the tracking results for linear Kalman filter. The true object location is represented by a blue ellipse while the measured object location is represented by red ellipse; we have chosen an elliptical shape in order to display the orientation angle clearly. Tracking results for LKF, EKF and UKF are shown in the third, fourth and fifth rows respectively. The three filters have shown similar estimation accuracy which is very high for the object position and the size of the bounding ellipse, while the orientation angle has a substantial error especially in sequence 3.The remaining three rows show the tracking performance for generic particle filter, SIR particle filter and auxiliary particle filter. Similar to table 4.7 they all have less estimation accuracy than the Kalman filters. In general, all filters fail to have good estimation accuracy for estimating the orientation angle of the tracked object. This is due to erroneous measurement of the orientation angle. This is seen from the blue ellipse which represented the measured location; e.g. in sequence 3 the orientation angle for both objects in scene is very much deviated from its true value.

Table 3.7: object tracking results for different video sequences.

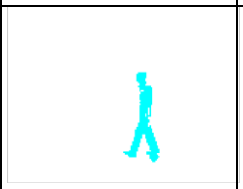
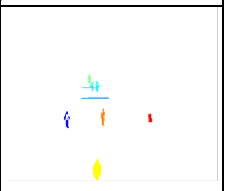
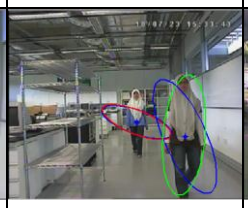
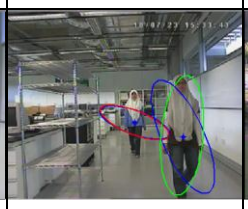
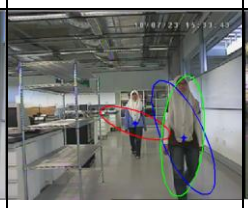
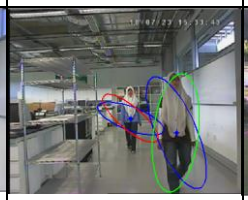
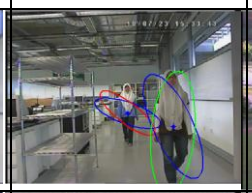
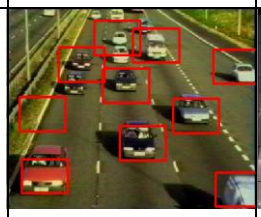
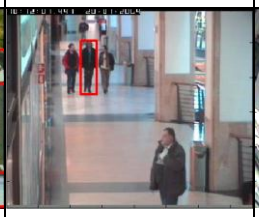
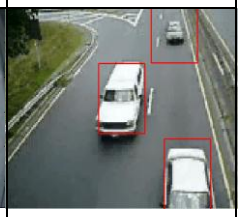
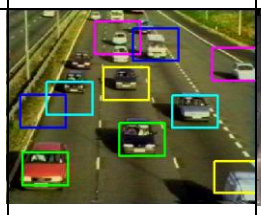
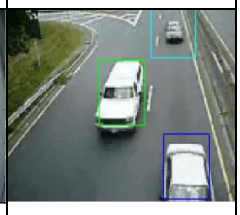
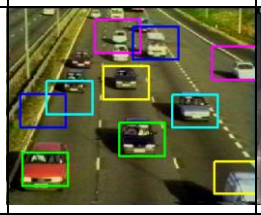
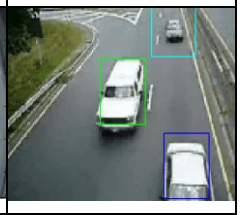
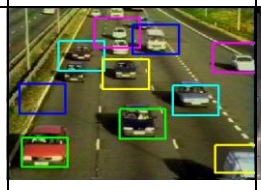
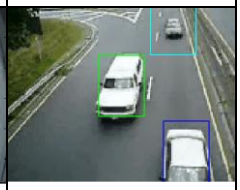
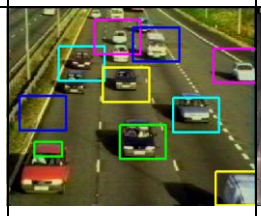
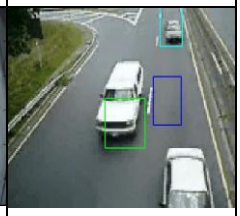
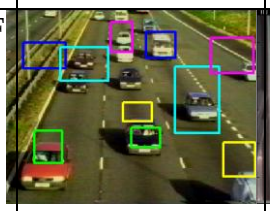
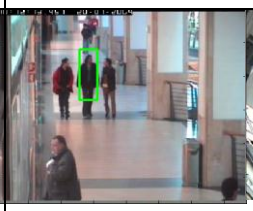
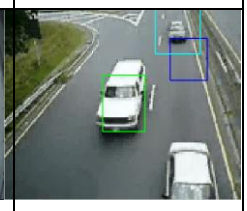
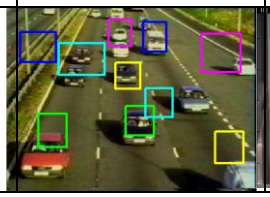
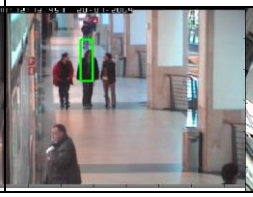
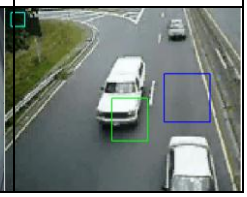
Method	Sequence 6	Sequence 7	Sequence 8	Sequence 9
Original Image				
Blobs image				
LKF				
EKF				
UKF				
GPF				
CondPF				
AuxPF				

Table 3.8: Result for object tracking using the standard datasets.

Method	Sequence 6	Sequence 7	Sequence 8	Sequence 9
Object selection				
LKF				
EKF				
UKF				
GPF				
CondPF				
AuxPF				

In a similar manner to Table 3.7, these algorithms have been tested on standard datasets and similar results have been achieved. Sequence 1 and sequence 4 contain vehicle images in highway, and sequence 2 and sequence 3 are for indoor scenes. For the outdoor sequences (sequence 1 and sequence 4), the tracking is performed based

on object motion and similar to the one shown in Table 4.7. The first row contains the measured object location while the remaining rows display the tracking results for the implemented filters. Similar to Table 4.7, Kalman filters have better tracking performance than particle filters. For the case on indoor sequences, the tracking is conducted based on the color of a selected object in the scene. Firstly, the object of interest is selected interactively as shown in the first row of Table 3.8 (sequence 2 and sequence 3). Then, the selected ROI is highlighted by a red box. After that the object of interest is tracked in every new frame by matching the reference color with the color of the object. The rest of rows in sequence 2 and sequence 3 display the implemented tracking algorithms. This method allows us to track an object in a cluttered scene with great accuracy. The selected object is successfully tracked using the implemented methods.

3.3.4 Objective Subjective Comparison of Stochastic Filters

In this section, we evaluate the tracking method objectively based on their computational time, estimation error and correlation. For simplicity, the objective analysis has been conducted on the first video sequence only.

3.3.4.1 *Computational time*

The selected tracking methods have been tested on two computers with different specifications as explained earlier. Figure 4.9 shows the computational time for the selected methods. When comparing the computational time on both computers, PC2 has shorter computational time because it has better specs than PC1. However, when comparing the computational time of the different tracking algorithms on the same computer, the results are different from what has been mentioned in the literature. We found that there is no significant difference in the computational time between the six tracking algorithms. The EKF has the shortest computational time (0.2121 sec for PC1 and 0.1096 sec for PC2) while the AuxPF has the maximum computational time (0.2334 sec for PC1 and 0.1226 sec for PC2) as shown in Table 3.9. We can see that the maximum change in computational time is only 10%; thus the difference is not significant. This result is different from the published work mentioned in section 2.3.

For example, the EKF estimation is propagated using equations which only contains matrix multiplications and summations of size NxN (N is the number of estimated variables). On the other hand, the AuxPF has a larger matrix of size (N x number of particles) and moreover, it has a sequential resampling steps. Therefore, it is obvious that the computational requirement of AuxPF is much larger than EKF. Additionally, in [66], [67], [23] and [14], it has been stated that the Kalman filter has much lower computational complexity than particle filters. However, we did not find significant difference in computational time during our implementation of these methods. Our results are different from the published results due to reasons discussed as follows. During the implementation of these algorithms, we have avoided creating nested loops; instead the nested loops are replaced with matrix operations. Matlab software performs matrix operations very efficiently despite of its large dimensions or complexity. This is done by using only the non-zero valued elements of the matrices (sparse matrix operations [68]); thus the computational time is reduced significantly. Another important reason is the availability of high performance PCs. As can be seen from the description of the two PCs used for experiments, one has Intel Core 2 Duo processor while the other has Intel i7 Quad core processor with 3GB and 6GB RAM respectively. Hence, the difference in the computational time for the implemented algorithms is not significant.

Table 3.9: Computational time of six stochastic filtering methods on two computers.

Selected tracking algorithm	Computational time using computer PC1	Computational time using computer PC2
Linear Kalman filter	0.2153	0.1099
Extended Kalman filter	0.2121	0.1096
Unscented Kalman filter	0.2128	0.1114
Generic particle filter	0.2125	0.1113
SIR particle filter	0.2195	0.1177
Auxiliary particle filter	0.2334	0.1226

3.3.4.2 *Estimation Accuracy*

For six tracking algorithms we have computed the accuracy for estimating the centroid location and the size of the moving object for 100 frames of the first video sequence. In general, Kalman filters performed better than particle filters. This is because of the poor prior used for particle filters. From figure 4.10, the unscented Kalman filter has the highest accuracy because it uses the unscented transformation for approximating nonlinear system. Linear Kalman filter has less accuracy the UKF because the object motion is not always linear, thus it fails to correctly estimate the nonlinear motions. The extended Kalman has similar performance with linear Kalman filter with slightly better accuracy because of it uses first order approximation to linearize nonlinear state functions. On the other hand, particle filters have higher error because the Monte Carlo approximation of random variables is not perfect and it only has better results for systems that do not follow a known distribution. In addition, the proposal distribution poorly resembles the actual one. Among particle filters, the auxiliary particle filter has higher accuracy but larger computational time because it uses current observations in order to correct the estimation prior which induce additional resampling step. SIR particle filter has higher RMSE value than the auxiliary particle filter. However it is still better than the generic particle filtering algorithm because it resamples the prior particles before drawing the new estimation compared to the generic particle filter which directly draws new estimation from the prior. In the case of RMSE error is higher than the one obtained for centroid location because the variations in the bounding ellipse size are bounded by the algorithms ability to segment the full size of the tracked object; this can vary due to occlusion or light variations (nonlinear variations).

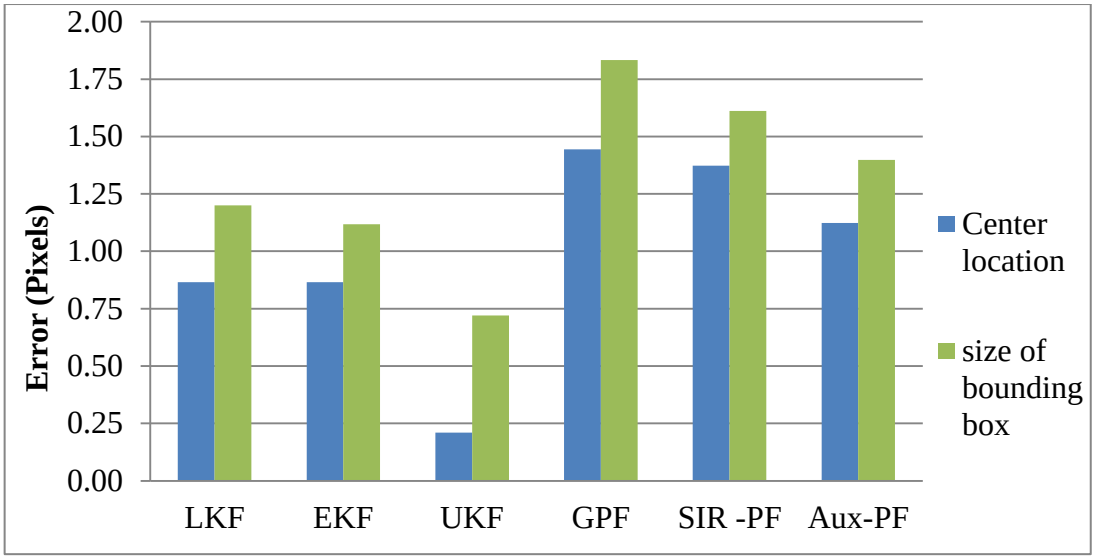


Figure 3.5: Root mean square error of estimating the object location and object size.

Figure 4.11, shows the correlation relationship between the measured and estimated variables for the six algorithms. The correlation gives dimensionless quantification for errors. Clearly, Kalman filters have outperformed the particle filters because of the poor prior used in the case of particle filters. For estimating object position the correlation for all filters is greater than 95% which is very high. The UKF has 100% correlation. For the size of bounding box size, Kalman filter can still achieve better correlation index than particle filters although overall it is less than the one obtained for estimating the object centroid location.

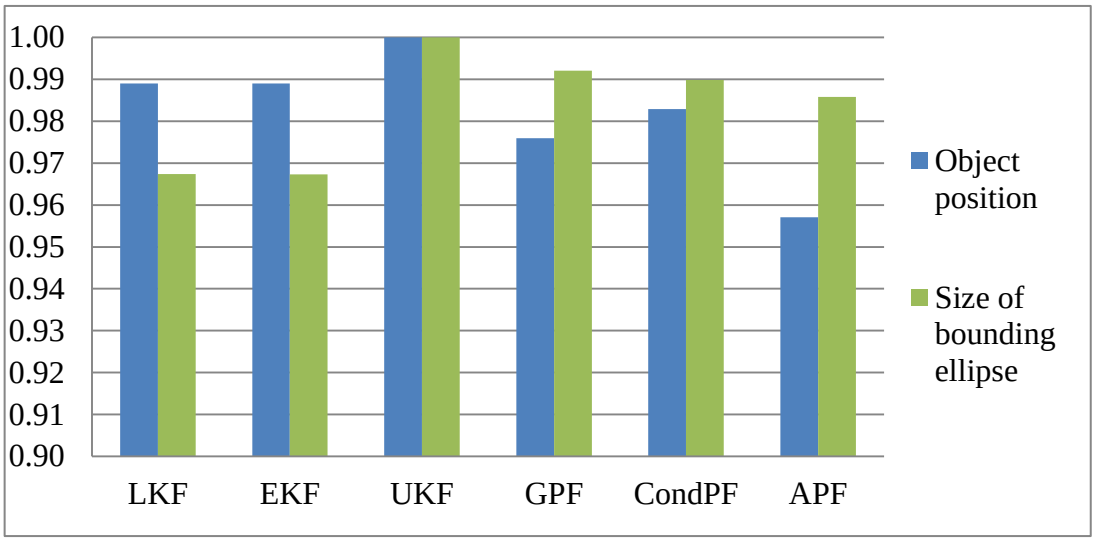


Figure 3.6: Correlation relationship between measured and estimated variable for the object location and size of bounding box.

As a conclusion, the unscented Kalman has better estimation accuracy among the discussed ones because it can deal with both linear and nonlinear systems. In addition it has shorter computational time compared particle filters. Moreover, in visual surveillance the tracked object (mostly humans and vehicles) exhibit nonlinear motion that can be approximated with a normal distribution. Therefore, the unscented Kalman filter is the best one to be used in these scenarios. The other type of stochastic filters and particle filters have an acceptable accuracy but it is less than the one achieved by the unscented Kalman filter.

3.3.4.3 *Effect of Initial Conditions on Settling Time*

Usually in conditions of the stochastic filters are set to a reasonable random choice. However, the choice of the initial condition should have an effect on how fast the filter converge to its normal value which is known as settling time of the filter. In this experiment the nominal value of the six filters is 100. However, the initial condition has been set to 10^6 which is very far from the nominal value. The settling time has been recorded which is around ±5% of the nominal value which is presented in Table 3.10. The motion involve in this experiment is nonlinear therefore linear Kalman filter and extended Kalman filter has larger settling time compared to the unscented Kalman filter and the and the particle filters. This proves that the initial condition are generally not significantly important in the especially for the unscented Kalman filter and the particle filters.

Table 3.10: Settling time of six filtering algorithm for visual tracking.

Filter	Settling time
Linear Kalman filter (LKF)	17
Extended Kalman filter (EKF)	17
Unscented Kalman filter (UKF)	2
Generic particle filter (GPF)	2
Sequential importance sampling particle filter (SIR-PF)	2
Auxiliary particle filter (AuxPF)	2

3.4 **Chapter Summary**

This chapter discusses the implementation of object detection using background subtraction. The GAP method has been implemented for background modeling and compared with median filtering and mixture of Gaussian. The experiment proves that the GAP method has better accuracy for detecting moving objects. In addition this method is robust for eliminating brightness variation and noise effects. This section also implemented shadow removing method using luminance and chromaticity gains and it also shows how to represent the moving object using it bottom most point instead on using the centroid which is most common moving object representation.

The second section of this chapter presents an objective comparison of stochastic filtering algorithms presented in the previous Section 2.4. Three Kalman filters and three particle filters has been implemented. The objective of our comparison is the select the best method to be combined with the depth estimation method presented in Chapter 4. In this comparison we concluded that the unscented Kalman filter has the highest accuracy among the discussed methods and it has less computational requirement than particle filter. Therefore, the unscented Kalman filter is the most suitable one to be used for 3D tracking using the depth estimation method proposed in the previous chapter.

CHAPTER 4
DEPTH ESTIMATION USING TRIANGULATION

4.1 **Introduction**

Once the moving object is detected in the scene, the next step is to find its coordinates in 3D space. As discussed in details in chapter 2, there is no method available for 3D localization of the moving object for real-time 3D tracking using the existing 2D camera installations. The existing solution includes replacing these cameras with multiple cameras, stereovision, 3D cameras or to use methods with large computational complexity and do tracking offline instead of online. Hence, a new depth estimation algorithm is proposed that uses the existing 2D camera installations for developing 3D tracking system without the need for any new hardware installations. This chapter introduces a novel depth computational method that is efficient and can be used for real time applications such as object tracking. Experimental validations and analysis is presented for this depth estimation method.

4.2 **Camera Geometrical Model**

Most of the visual surveillance cameras are installed in such a way that they look downward with a known pitch angle. This step enables the camera to cover a larger zone and avoid occlusion from background objects. Figure 3.4 shows a camera setup where the green trapezium area shows the area covered by the camera view. The camera is installed at a height (h) from the ground with a pitch angle (θ) with the vertical axis. The field of view of the camera is FOV_H for the horizontal direction and FOV_V for the vertical direction. Some cameras have similar vertical and horizontal field of view. For zoom cameras, the field of view is given in a range of maximum field of view angle when the camera is zoomed out and minimum field of view angle when it is zoomed in.

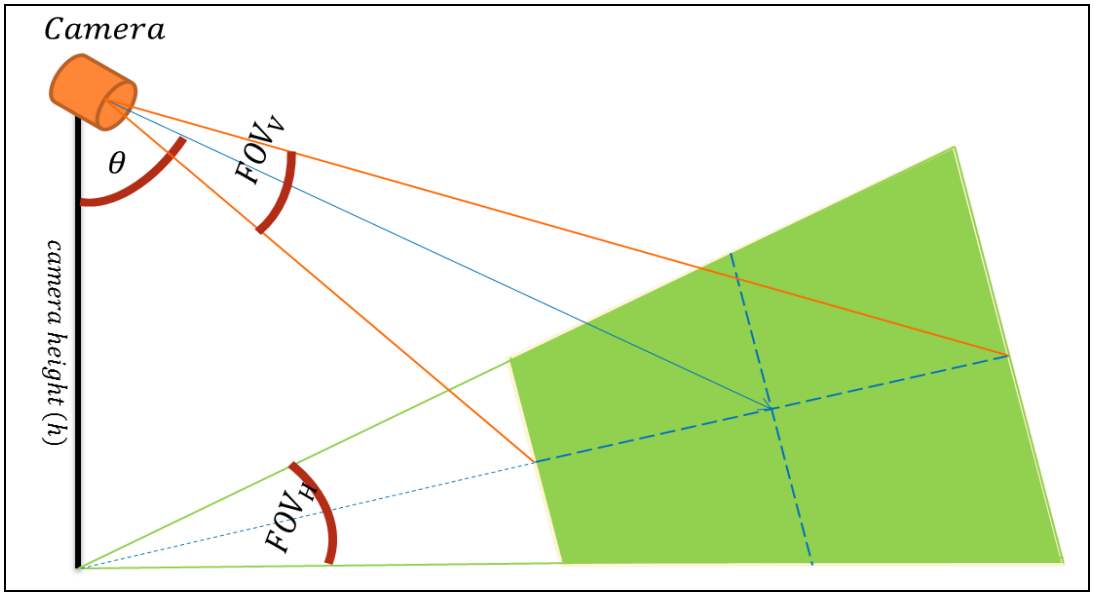


Figure 4.1: Model of a typical surveillance camera installation.

The size of the area covered by the camera depends on these three parameters. For example, the greater the field of view angle means it covers larger area. For zoom cameras, only the field of view changes. Larger camera height covers large viewing area. The pitch angle (θ) is constrained by the trigonometry constraints of equation (4.1). If the pitch angle exceeds this constraint, the trigonometry relationship cannot be maintained, and thus we will obtain erroneous results. The parameters mentioned earlier are generally known for all cameras and it is tuned during the camera installation process.

$$\theta < 90 - \frac{FOV_V}{2} \tag{4.1}$$

4.2.1 Depth from Triangulation Algorithm

In figure 3.5, there is an object at location $I(i,j)$. The image size is with Width (W) and height (H). The resolution of the scene depends on the image resolution as well as the size of the area covered by the camera. The larger the image resolution, the finer the image element is and it can have more accurate localization of the object location.

However, if the size of the covered is large (the camera height is large or the field of view is large), the resolution is reduced and the pixel element will be larger in size. Thus, there is a larger quantization error. In Figure 4.1, the rotation angle (ϕ) and the pitch angle (ψ) are computed for the object located at point $I(i,j)$. Then the distances Y and L are computed using these two angles.

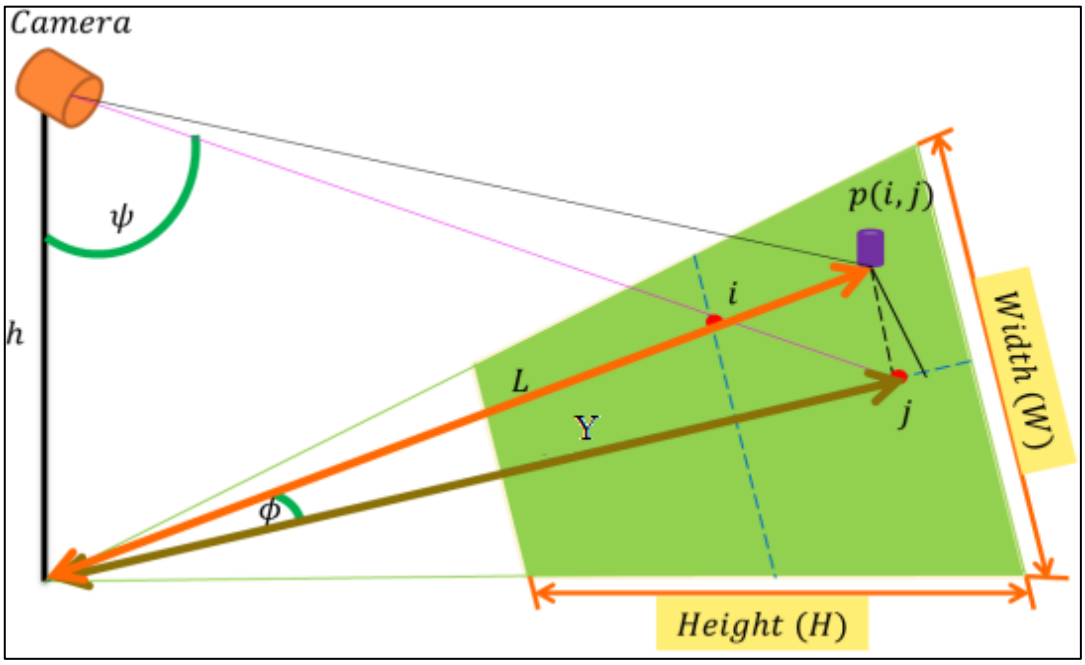


Figure 4.2: Trigonometry model of an object in the scene at location $\boldsymbol{I(i,j)}$.

The angular step in the horizontal direction is given by $\left(\frac{FOV_H}{W}\right)$. This is the field of view angle in the vertical direction divided by the image width.

$$\phi = \left(i - \frac{W}{2}\right) \times \left(\frac{FOV_H}{W}\right) \tag{4.2}$$

The angular step in the vertical direction is given by $\left(\frac{FOV_V}{H}\right)$. This is the vertical field of view divided by the image height.

$$\psi = \theta + \left(\frac{H}{2} - j\right) \times \left(\frac{FOV_V}{H}\right) \tag{4.3}$$

Given the vertical angle (ψ) and the camera height (h) the distance (Y) in Figure 4.2 is computed using equation (4.4).

$$Y = h \times \tan(\psi) \tag{4.4}$$

Then, given the distance (Y) and the rotation angle (ϕ), the distance (L) in Figure 4.3 is computed using equation (4.5).

$$L = \frac{Y}{\tan(\phi)} \tag{4.5}$$

Now by using the distances (L) and (Y) and the angles (ψ) and (ϕ), the 3D coordinates of the object at point $I(i,j)$ are computed assuming that the center of coordinates is beneath the camera exactly as shown in Figure 4.3.

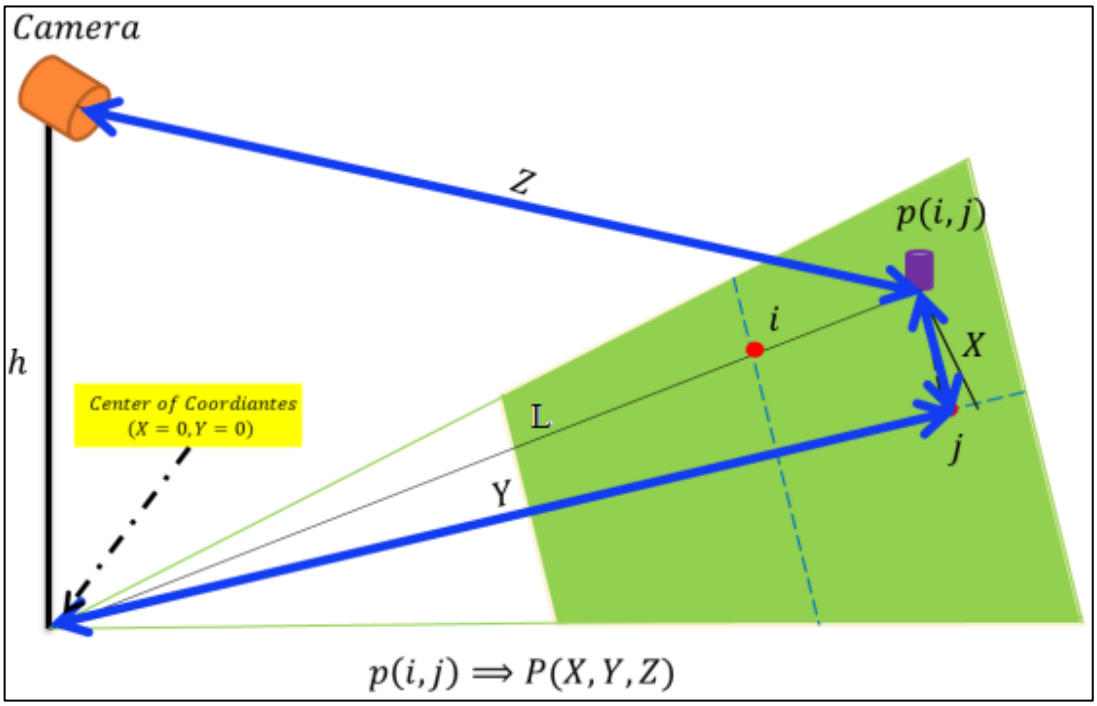


Figure 4.3: Computing the 3D world coordinate of an object at location $\boldsymbol{I(i,j)}$.

$$X = Y \times \tan(\phi) = h \times \tan(\psi) \times \tan(\phi) \tag{4.6}$$

$$Y = h \times \tan(\psi) \tag{4.7}$$

$$Z = \sqrt{h^2 + L^2} = \sqrt{h^2 + \left(h \times \frac{\tan(\psi)}{\tan(\phi)}\right)^2} \tag{4.8}$$

X and Y in equation (4.6) and (4.7) respectively denote the moving object

coordinates with respect to ground of the scene. Z in equation (4.8) is the depth of field or the actual distance between the moving object and the camera. The algorithm for depth estimation of moving object can be summarized in details in the steps below and also in the flow diagram of Figure 4.4.

Inputs:

1. Camera extrinsic parameters:
    a. camera height (h)
    b. camera pitch angle (θ)
2. Camera intrinsic parameters:
    a. vertical field of view (FOV_V)
    b. horizontal field of view (FOV_H)
3. Image parameters:
    a) Image size (Width, Height) → (W, H)
    b) Moving object location $p(i, j)$

Algorithm:

1. Compute the angles

$$\psi = \theta + \left(\frac{H}{2} - j\right) \times \frac{FOV_V}{H} \tag{4.9}$$

$$\phi = \left(i - \frac{W}{2}\right) \times \frac{FOV_H}{W} \tag{4.10}$$

2. Compute the world coordinates

$$X = h \times \tan(\psi) \times \tan(\phi) \tag{4.11}$$

$$Y = h \times \tan(\psi) \tag{4.12}$$

3. Compute the depth of field

$$Z = \sqrt{h^2 + h \times \left(\frac{\tan(\psi)}{\tan(\phi)}\right)^2} \tag{4.13}$$

Outputs:

$$p(i, j) \Longrightarrow P(X, Y, Z) \tag{4.14}$$

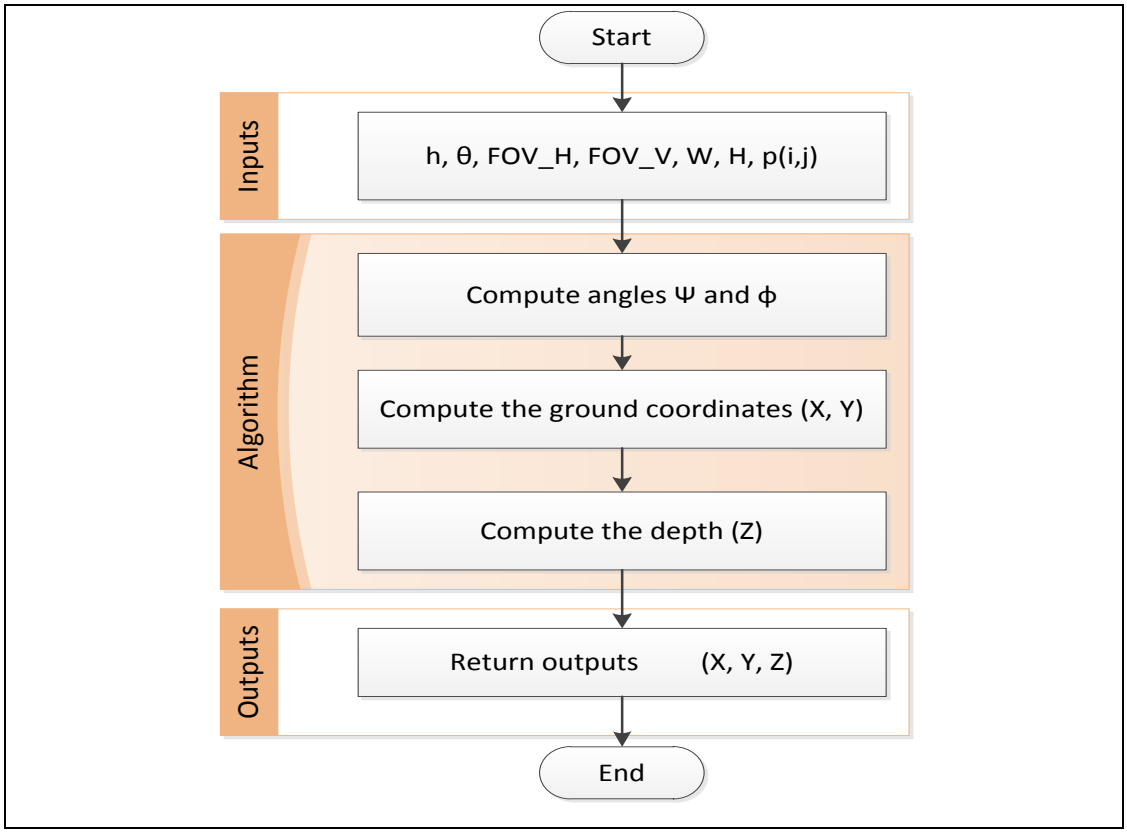


Figure 4.4: Flow diagram for geometry from triangulation method.

4.2.2 Practical Implementation of the Proposed Method

The proposed method employs geometrical technique in the scene for computing the depth of any point in the field of view. This requires the location of the moving object to be in a known geometrical structure with the camera center. Thus the object has to be represented in a way that serves this purpose. In addition, non-flat surfaces, occlusion and shadow have to be taken care off in order to get correct and accurate depth computation.

4.2.2.1 *Moving Object Representation*

The proposed algorithm computes the depth for an object which is located on the ground. This means the object has to be moving on the ground as the algorithm cannot compute the depth for a floating object in air unless its ground location is known. Figure 4.5 shows a sectional view of the camera model. If the moving object is represented by the centroid location, then it will be seen in the image at point (C)

which is not the true location of the moving object on the ground. Thus the moving object is represented by the bottom most point (BP) which is the ground point (feet location) of the moving object. Hence, the ground is the reference for calculating the depth.

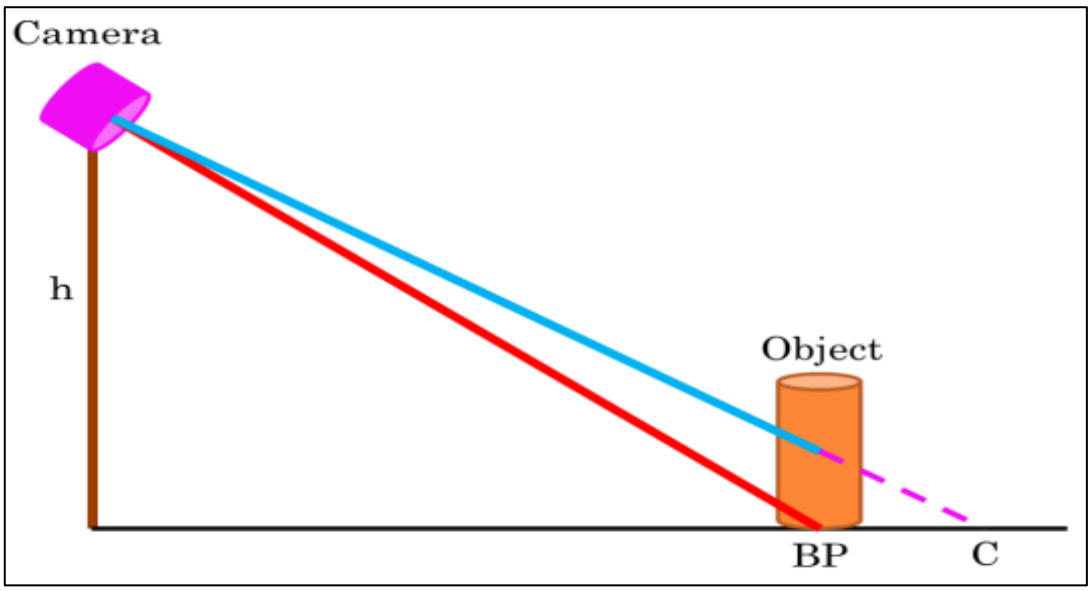


Figure 4.5: Object representation by the bottom point (BP) and by the centroid (C).

4.2.2.2 *Non-flat Surface*

If the surface is not-flat the as in Figure 4.6, a correct trigonometry relationship cannot be established. Thus a wrong position is recorded for the moving object. In Figure 4.6, the true location of the moving object is at point (BP) while it is seen in the image at point (C) of the ground which is not the ground location of the object. Therefore, the estimated depth will be larger than the actual depth of the object. If the height of the uneven surface is known the true depth could be computed by taking the camera height from the tip of the uneven surface. Thus, this method can be implemented on flat surfaces and non-flat surfaces with a known height.

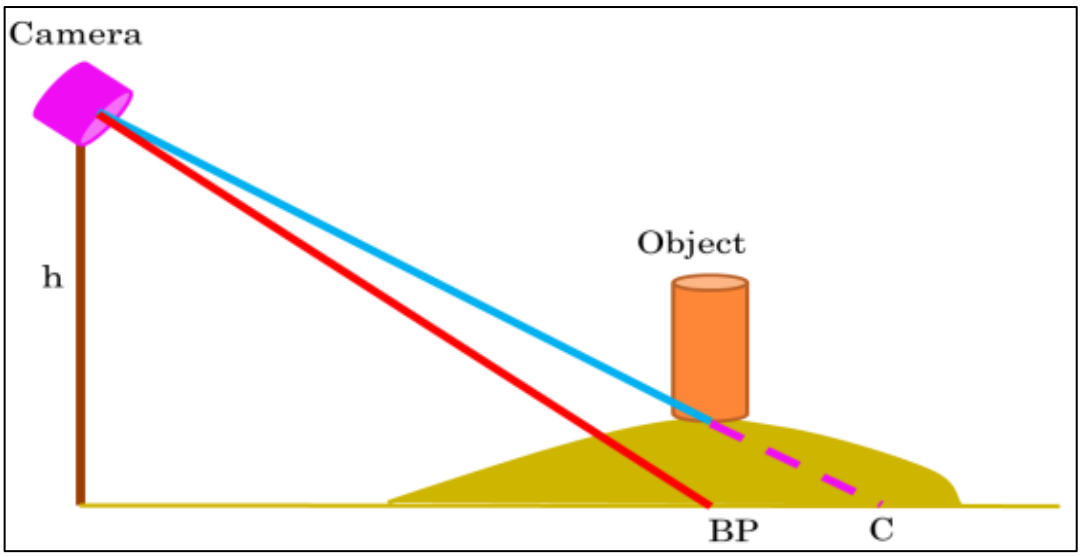


Figure 4.6: A model for a moving object in non-flat surface.

4.2.2.3 *Cast Shadow/Highlights*

Cast shadow gives false location for the moving object which affects the depth computation algorithm. Therefore, shadow/highlights must be eliminated before computing the depth. In Figure 4.7, the moving object is at location (C) in the image but the shadow is at (BP) and normally detection methods confuse shadow with moving object. If the shadow is in front the object, then it may be detected as the feet location of the object. Therefore, a shadow removal algorithm has to be used before the depth estimation in order to refine the detection results.

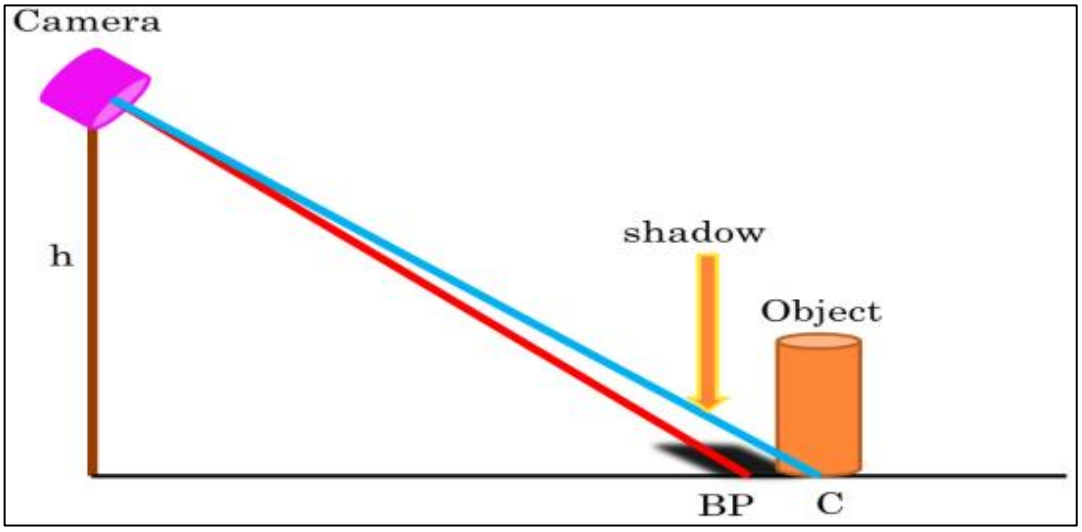


Figure 4.7: Moving object with cast shadow.

4.2.2.4 *Occlusion*

If a moving object is occluded by the background or another moving object partially or fully, it cannot be seen by the camera at its correct place. Therefore, motion history should be utilized in order to estimate the location of the occluded object. Motion history can give an estimate for the motion direction and velocity. Thus the new location can be estimated using the previously known location and the motion vectors.

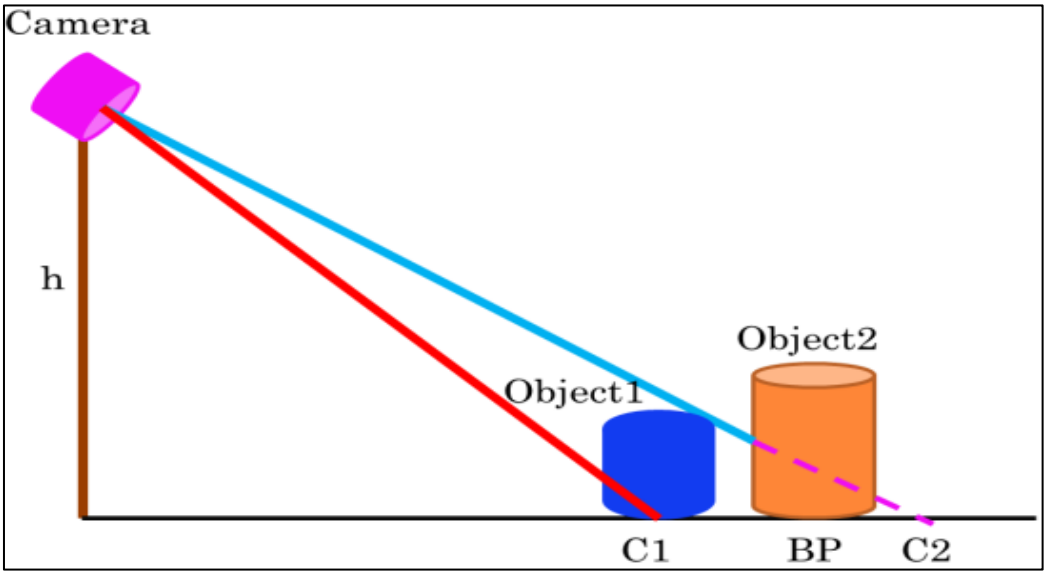


Figure 4.8: Camera model with Object1 occluding Object2.

In Figure 4.8, Object1 with apparent location (C1) is occluding Object2 with apparent location (C2). The true location for Object 2 is point (BP) but since it has been occluded by Object 1, the camera will see its bottom-most point at point (C2) rather than point (BP). Therefore, the location of Object 2 should be computed from its motion history rather than directly computing it from the image.

4.2.3 Height of Moving Object

The proposed algorithm computes the depth and the 3D location of moving objects that are located on the ground (the bottom point of the moving object). Thus the height of the moving object cannot be computed directly. But it can be computed by constructing a geometrical model using the bottom point and the top point of the object. Then similar triangles in the scene are used to compute the height.

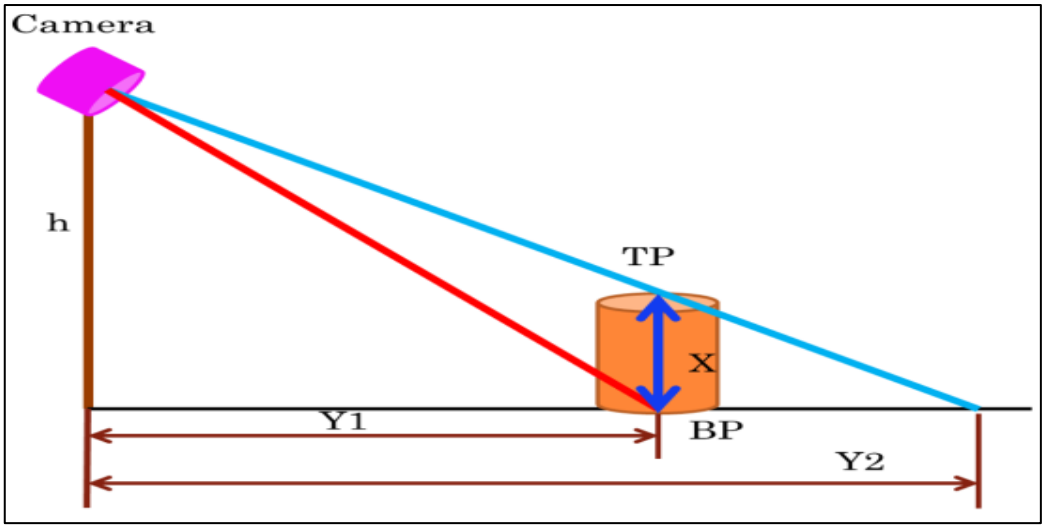


Figure 4.9: Computing the height of the moving object.

In figure 4.9, the ground distance ($Y1$) is computed using point (BP) and the ground distance ($Y2$) is computed using the triangulation algorithm from point (TP). Then by using similar triangles, the object height (X) is computed using equation (4.15).

$$X = h \times \frac{Y2 - Y1}{Y2} \tag{4.15}$$

4.2.4 Ground Distance Measurement Using Triangulation

Triangulation algorithm can be used to compute the actual distance between two points in the image. This can be very useful in aerial images where the pilot can choose two points in the image and the DfT algorithm measures the exact distance between them in meters or inches. In addition, DfT method could also be used to compute the distance between two detected object in the scene. This is used in automated surveillance for detecting unattended object in the scene by measuring the distance between the object and the person who left it. In Figure 4.10 there are two objects in the scene one at pixel $I_1(i,j)$ and the second object at pixel $I_2(i,j)$ in the image. The ground location and depth of field for the first object is computed by the DfT as $p_1(i,j) \leftrightarrow P(X_1, Y_1, Z_1)$ and for the second object is computed as $p_2(i,j) \leftrightarrow P(X_2, Y_2, Z_2)$. Then the distance between these two points is computed using equation (4.16).

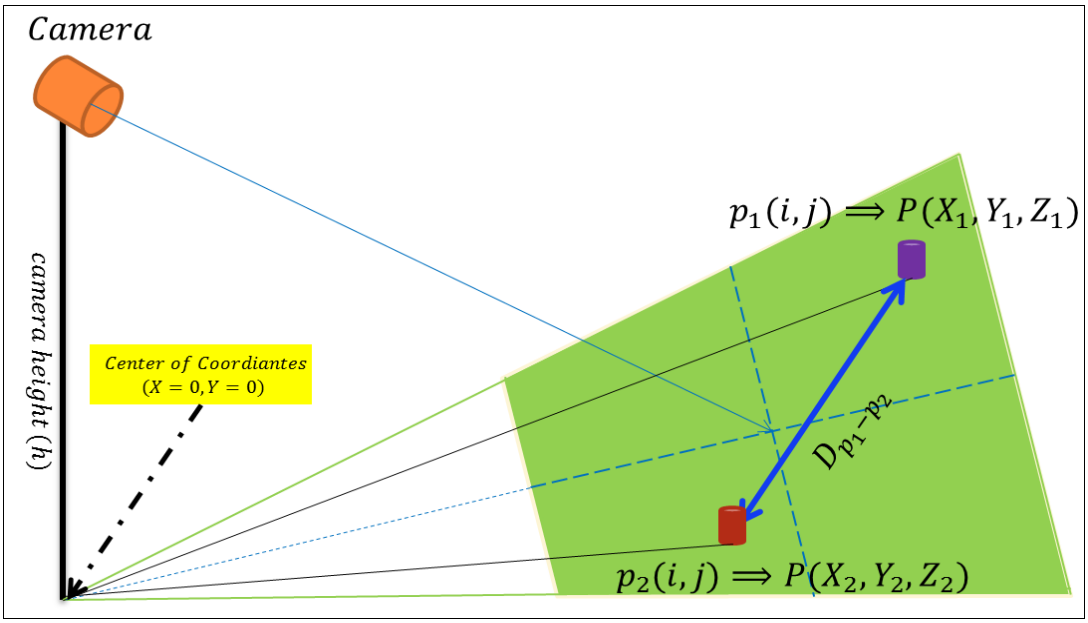


Figure 4.10: Measuring the ground distance between two objects in the scene using DfT algorithm.

$$D_{p_1-p_2} = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} \tag{4.16}$$

4.2.5 Testing and validating the proposed method

In order to validate the proposed method, different images we collected using a Canon DSLR 1000D camera. The image resolution is (2592x3888) which is considered to be very high resolution. The camera field of view is 64.3^0 horizontally and 45.3^0 vertically. Different images are acquired by varying the pitch angle and the camera height. Figure 4.11 shows ground measurement using the proposed method. For the first box, the actual width of the box is 0.31m but the measurement indicated as 0.273 m which is very close to the actual measurement. For the second box the measurement is even more accurate where the actual width is 0.570 and the measurement by the proposed method is 0.547m. For the last distance, the actual measurement is 1.72m while the proposed method measured it as 1.75m.

Figure 4.11: Distance measurement in images using the proposed method.

Figure 4.12 shows further measurement using the proposed method. The distance measurement using the proposed algorithm between the parallel grid lines in the floor is the very similar to the actual measurements and the error is in the scale of centimeters. Similarly, Figure 4.13 shows height measurement using the proposed method.

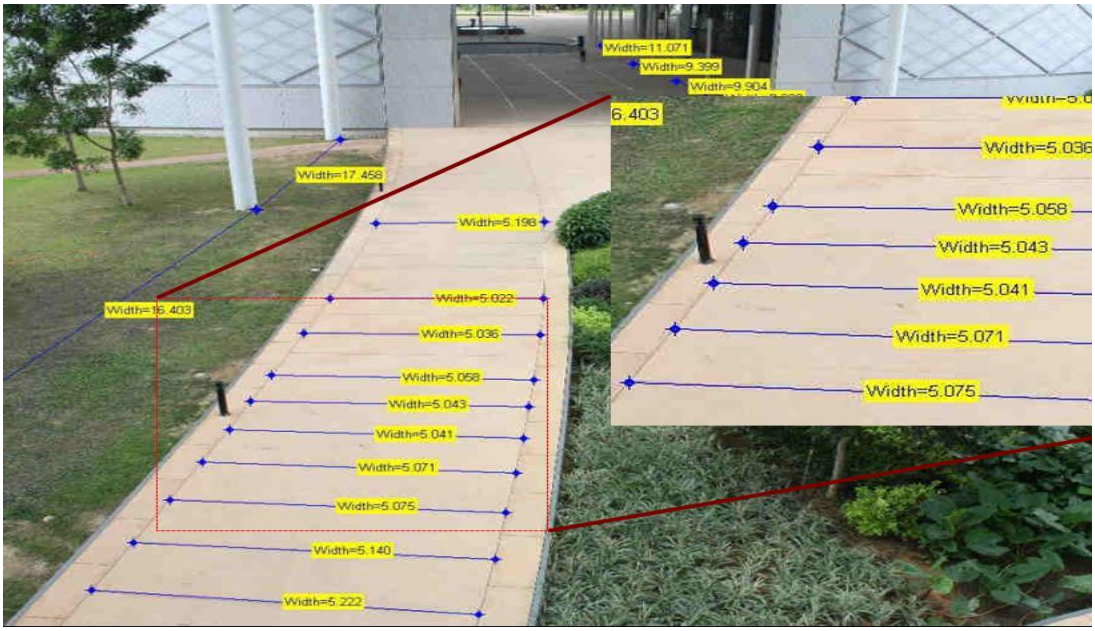


Figure 4.12: Measuring distance using the proposed method.

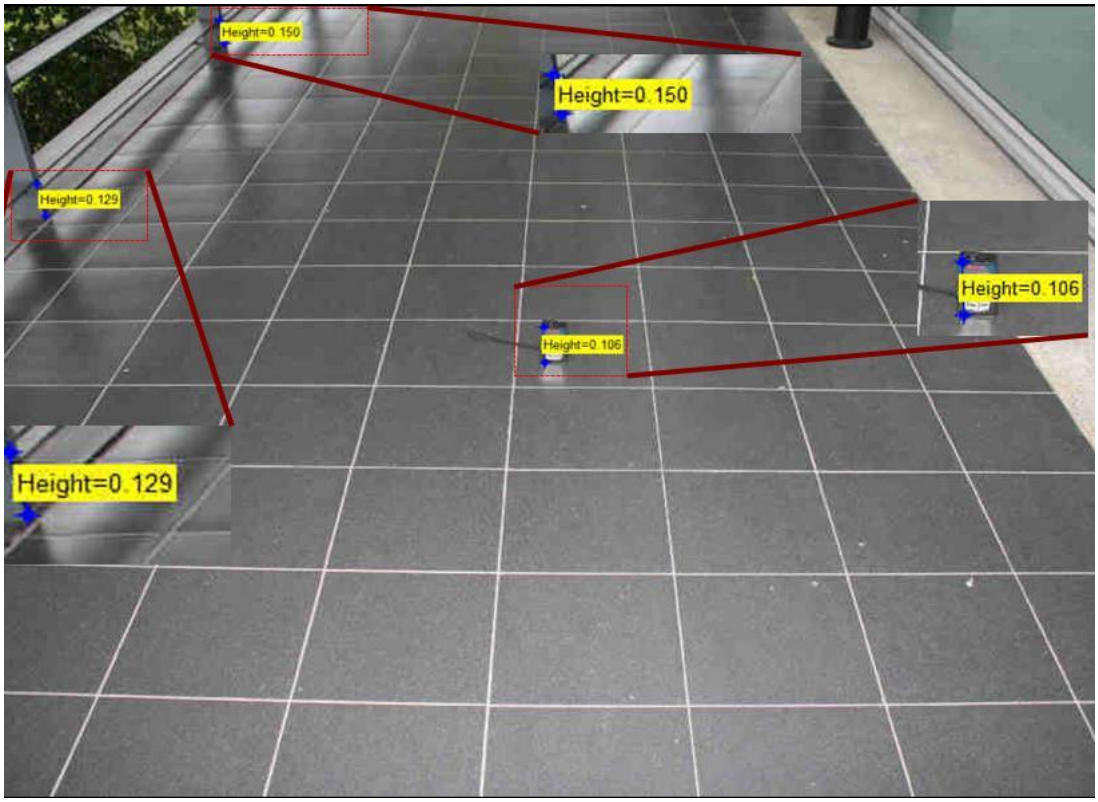


Figure 4.13: Measuring height of objects using the proposed method.

In Table 4.1, 10 measurements are recorded that represent the distance between an object in the image and the camera. The ground truth measurements are acquired using laser rangefinder. In Table 4.1, the ground truth measurements are compared with the recorded measurements. The absolute error was recorded and the error ratio to the ground truth data. Most of these measurements show good accuracy compared to ground truth measurements.

Table 4.1: Comparsion of measured depth using the proposed method and ground turth depth acquired by rangefinder.

Case	Ground truth (m)	Estimated depth (m)	Error (m)	Error (%)
1	3.723	3.5923	0.1307	3.50
2	5.343	5.1703	0.1727	3.20
3	7.721	7.2307	0.4903	6.40
4	11.37	10.705	0.2950	2.60
5	4.128	3.8841	0.2439	5.90
6	4.900	4.6563	0.2437	5.00
7	2.370	2.4000	0.0300	1.30
8	2.780	2.8400	0.0600	2.16
9	4.590	4.8800	0.2900	6.32
10	3.390	3.5800	0.1900	5.61

4.2.6 Effect of Distance on Depth Estimation

Image resolution is reduces over the distance and the quantization error increase when we go far from the camera. As a result, distance from the image center has a significant impact on the depth of field computed the triangulation method. In this section an experiment has been conducted to study the effect of distance on the estimated depth. The experiment measures the depth of field for an object at a distance range from 17m to 150m. The camera was place at a height of 10.48m and with viewing angle of 67.00^0. Table 4.2 list ground truth depth of field and computed depth of field in this experiment. Table 4.2 also tabulates the error obtain in this experiment. The table and Figure 4.14 show error variations with the distance from the camera. In general, the error increases with distance because the quantization error in increasing. However, for the last three points the error is decreasing which might be due to errors in selecting the object in the image.

Table 4.2: Comparsion of measured depth using the develped method and ground turth depth acquired by rangefinder.

Case	Ground truth (m)	Estimated depth (m)	Error (m)	Error (%)
1	17.374	17.369	0.005	0.027
2	21.029	21.175	0.146	0.696
3	24.939	24.937	0.002	0.008
4	30.391	30.091	0.301	0.989
5	37.423	37.120	0.303	0.808
6	43.325	42.631	0.693	1.601
7	49.498	48.472	1.027	2.074
8	58.827	57.725	1.103	1.874
9	68.724	66.022	2.702	3.931
10	81.286	78.553	2.733	3.362
11	97.478	93.870	3.608	3.701
12	112.522	108.102	4.421	3.929
13	129.804	127.179	2.625	2.023
14	134.871	131.089	3.782	2.804
15	143.430	144.833	1.403	0.978

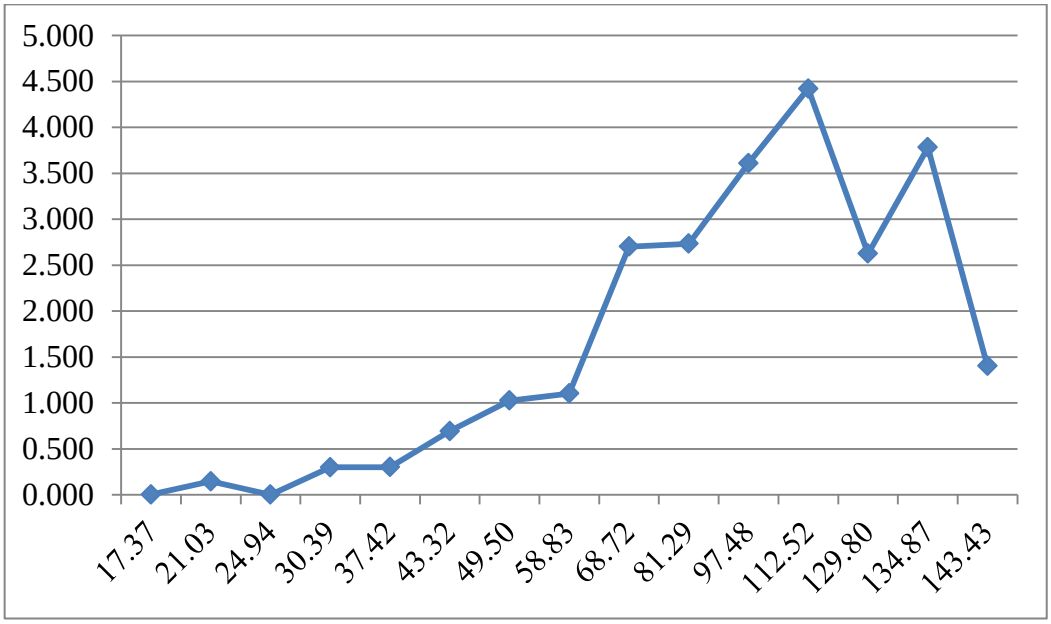


Figure 4.14: Analysis of depth estimation error with distance from the camera.

4.2.7 Error Analysis

From the Tables 4.1 – 4.2 and the Figures 4.11 - 4.13, the proposed method generally has a good depth computational accuracy compared to existing methods. However, there are some errors that need to be analyzed. This section tries to highlight the sources of possible errors in the proposed method. This method takes three measured inputs; which are the camera height, camera pitch angle and the location of moving object in the image. Other inputs such as image size and field of view are standard and can be obtained from the camera specifications.

This section discusses how much error is generated if there is error in measuring one or more of these parameters. Table 4.3 shows the error obtained due to 1% error in measuring camera height, camera angle or object location. For the object location 1% error is relative to the image size. For example, if the image size is (640×480), 1% error means 6 pixels error in width and 5 pixels error in height. Table 3.6 shows error analysis for three type of error. If there is 1 degree error in measuring the pitch angle, it will yield to 1.75% error in computing the Y-axis, 1.75% error in computing the X-axis and 2.48% error in computing the depth of field. Similarly, 1% error in measuring the camera height will have effect on all the three coordinates by 1% for X and Y and 1.73% in the depth value. Errors in measuring the moving object location in the image is very prominent because existing object detection tools are not very accurate and 1% error is very common. 1% error in the width coordinate only affects the X-axis coordinate and the depth of field (Z) by 3.49% however 1% error in the height-coordinate will have effect on X, Y and Z coordinates by 3.49%, 3.49% and 4.94% of camera height respectively. This algorithm is very sensitive to error in the vertical direction compared to the horizontal one. This is because the magnification increases drastically in the vertical direction as compared to the horizontal one.

Table 4.3: Error analysis for the proposed method showing maximum possible error.

Input	Error	$\Delta\psi$	$\Delta\phi$	ΔX	ΔY	ΔZ
Pitch angle	1.00^0	1.00^0	0.00^0	1.75%	1.75%	2.48%
Camera height	1% of actual camera height	0.00^0	0.00^0	1.00%	1.00%	1.73%
x-coordinate	1% of image width	0.00^0	2.00^0	3.49%	0.00%	3.49%
y-coordinate	1% of image height	2.00^0	0.00^0	3.49%	3.49%	4.94%

4.3 **Chapter Summary**

In the chapter, a novel depth estimation method was introduced using the triangulation concept. The basic principles and the implementation steps of the method have been discussed. Moreover, all the issues related to the method have been explained such as how to represent the object in the image, effect of non-flat surface, occlusion and shadow in the computation of the depth. Besides computing the depth this method compute the real world coordinates of the moving object with respect to a known reference such as the camera pole. In addition, this method has been extended for computing the height of the moving object using the top-most and bottom-most points of the object in the image. The depth from triangulation method is been validated by measuring the dimension of known objects in the scene and comparing it with ground truth measurements. The results obtained showed good accuracy with small errors. In addition, analysis of error with the distance from the camera has been presented in this chapter. Finally, an uncertainty analysis is presented for this method by identifying the possible sources of errors and how much they can affect the computed depth and geometry of an object in the scene.

CHAPTER 5
EVALUATING THE DEVELOPED 3D TRACKING SYSTEM

5.1 **Introduction**

Previous chapters of this thesis discussed the three major components of the proposed 3D tracking system. These three components are the object detection, depth computation and object tracking component respectively. Moreover, suitable techniques have been identified for each one of these components. This section will combine these components in order to achieve a 3D tracking system using a single camera. Figure 5.1 provides a flow diagram for the complete system. In the first component the GAP method is been used for detecting the moving object. Then a newly proposed method is used for computing the depth of the moving object using triangulation. Finally, the unscented Kalman filter is used for tracking the moving object across frames. In addition, this section will discuss the data collection process and the evaluation criteria for comparing the performance of the system using four video sequences.

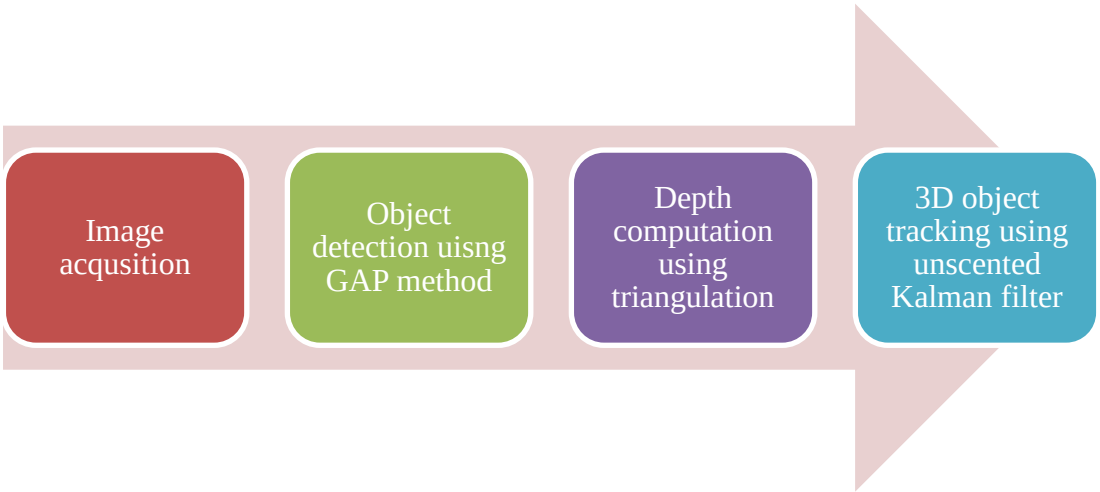


Figure 5.1: Flow diagram of the implemented 3D tracking system using single camera.

5.2 Evaluation Criteria

It is important to define a quality measures by which the performance of the system is measured. The first component of this system takes an image and it detects the location of the moving object in the image. An objective measure for measuring the accuracy of the detected object is not available because there is no ground truth image for the moving object. However, this accuracy can be measured visually by human observer. Therefore, the accuracy of the object detection system is measured subjectively by visual inspection. The second component computes the depth of field given the object location in the image. In this component it is possible to perform an objective comparison between the depth of field computed by the triangulation algorithm and a ground truth depth computed using a laser rangefinder. The last part is the object tracking using the unscented Kalman filter. In this case, an objective comparison is possible because the ground truth data is available. Therefore the evaluation criteria for this method include:

- Subjective evaluation of the object detection in each frame.
- Objective evaluation of the estimated depth for all the moving objects.
- Objective evaluation of the estimation results using RMSE and correlation similar to what is shown in section 3.3.2.

5.3 Implementation of 3D Tracking System

In this section, a detailed description is given about each component of the 3D tracking system. The section focuses more on how the parameters of each component are tuned.

5.3.1 Image Acquisition

Images have been captured from three types of camera shown in Figure 5.2 and explained in detail in the data collection section. These camera include two IP cameras; d'link DCS-2120 (wireless) and Samsung SNB-3000 and one analog camera Samsung SDZ-37X. For the IP cameras, images are streamed directly from the camera while for the analog one, the images are saved in video format then the

tracking is performed offline using the video file.

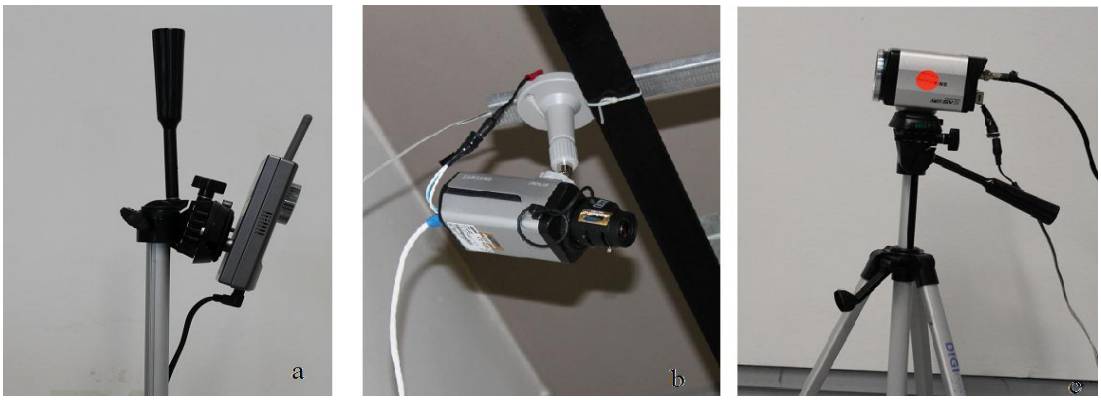


Figure 5.2: Video acquisition devices: a) D'link DCS-2120, b) Samsung SNB-3000 and c) Samsung SDZ-375.

5.3.2 Tuning the GAP Modeling Algorithm for Object Detection

GAP method associates each image point with a set of reference points. In this implementation, 20 frames ($M = 20$) were used for modeling the background of the scene assuming that the background is static; thus it is modeled only one time prior the tracking process. For each image point, 16 reference points ($N = 16$) have been allocated where half of these points (8 points) maintain an intensity difference larger than some positive threshold ($W_G = 20$) with the target point for at least 90% ($W_P = 0.90$) of the training frames. The other half maintains an intensity difference smaller than some negative threshold $W_G = -20$ for at least 90% of the training frames ($W_P = 0.90$). For reducing the search range, two neighboring pixels are considered similar if they maintain intensity difference less than 5 ($W_S = 5$) for at least 90% of the training frames ($W_B = 0.90$).

In the detection part, an image point is considered as background if it is larger than 70% of its positive reference points and smaller than 70% of its negative reference points. Shadow removal component was not implemented because the GAP method managed to eliminate most of the shadow effect present in the scene. Since some of the used video sequences include occlusion case, the occlusion compensation algorithm proposed in the Section 2.1 is used where the occluded object is replaced by its previous estimation. The results of the object detection are in the form of the bottom most point and the top most point of the moving object as it is shown in

Figure 5.3 where these two points will be used for depth estimation and computing the height of the moving object in the image. The bottom point is used for computing the depth of field and the ground location of the object in the scene. The top point will be used to compute the height of the object.

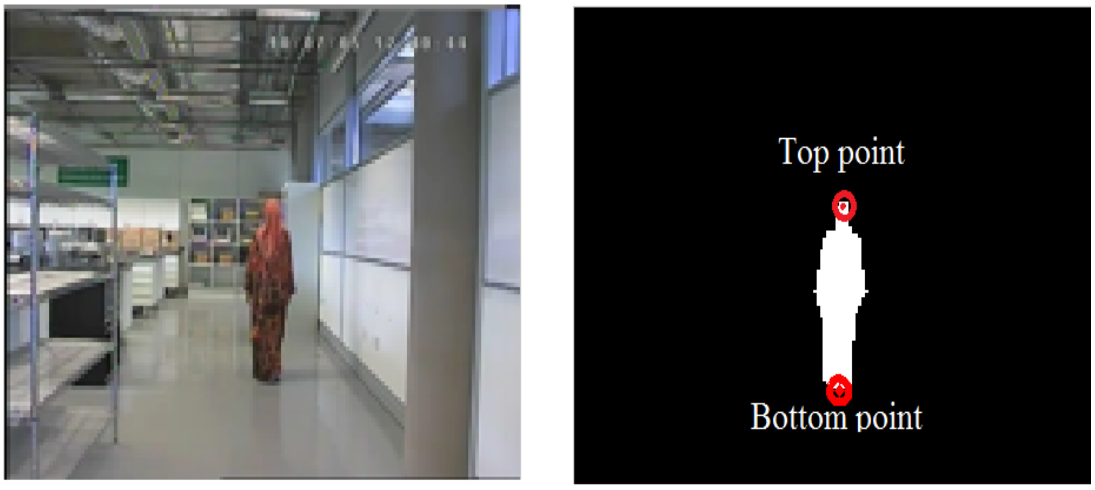


Figure 5.3: Detection of top most and bottom most points of a moving object.

5.3.3 Computing Depth from Triangulation (DfT)

The input to the DfT method is in a form of two 2D points which represents the bottom and the top point of the detected object blob. The DfT algorithm computes the ground location of the moving object (in meter) and the distance between the moving object and the camera (depth of field). In addition, the algorithm also returns the height of the moving object.

5.3.4 Tuning the Unscented Kalman Filter

Similar to Section 3.3 a constant velocity motion model is assumed throughout the tracking process where the object motion is assumed as a drifting point with a fixed velocity as in Equation (2.58). At each time step, the unscented Kalman filter predicts six variables which are the image ground location in X axis and Y axis, the depth of field and the velocities for these three parameters. Matrix A is computed from Equation (2.58) while the measurement matrix is considered to be the identity matrix

since measured variable exactly resembles the predicted states. The covariance matrices Q and R for state and measurement respectively are assumed to have identical effect on all estimated parts. Thus both are set to the identity matrix.

The unscented Kalman filter firstly computes $(2n+1)$ sigma points for n predicted variables. In this case there are 13 reference points because the state space contains 6 variables. Then the unscented transform is used to propagate these sigma points through the state and measurement functions.

5.4 **Data Collection**

The proposed method requires normal camera setup where the camera height, pitch angle and field of view angles have to be known. Four video sequences have been collected where these sequences resemble different tracking scenarios which include single/multiple object tracking in indoor/outdoor environment. Table 5.1 summarizes the collected video sequences and specifies their geometrical setup. The first sequence includes one moving object in indoor environment. The second and third sequences include two moving objects in outdoor without occlusion. The last sequence includes two moving objects in the opposite direction with occlusion in some frames.

Table 5.1: Data collection for evaluating the proposed 3D tracking system.

Sequence	Camera model	Image size	Field of view	height (m)	Pitch angle
Seq 1 (Indoor)	Dlink DCS-2120	(320x240)	(62.0^0x62.0^0)	2.6540	62.0^0
Seq 2 (Outdoor)	Samsung SDZ-3750	(704x576)	(55.5^0x42.5^0)	10.406	60.0^0
Seq 3 (walkway)	SNB-3000	(160x140)	(44.7^0x44.7^0)	3.5790	60.0^0
Seq 4 (Indoor)	Dlink DCS-2120	(320x240)	(62.0^0x62.0^0)	2.6540	62.0^0

5.5 **Results and Analysis**

In this section tracking results will be displayed for four video sequences and they will be evaluated using evaluation criteria defined in section 5.2.

5.5.1 First Experiment (Seq 1)

In this sequence, there is a single moving object in indoor environment. The camera is installed at a height of 2.65m and the maximum visible depth of field is around 10m. The subject moves from far to near then it turns right. Figure 5.4 shows the object trajectory in XY plane (ground plane). Five images have been sampled at a fixed step of 10 frames and marked with yellow color (Im1 – Im5). The 3D system performance is analyzed in Table 5.2 using these five samples.

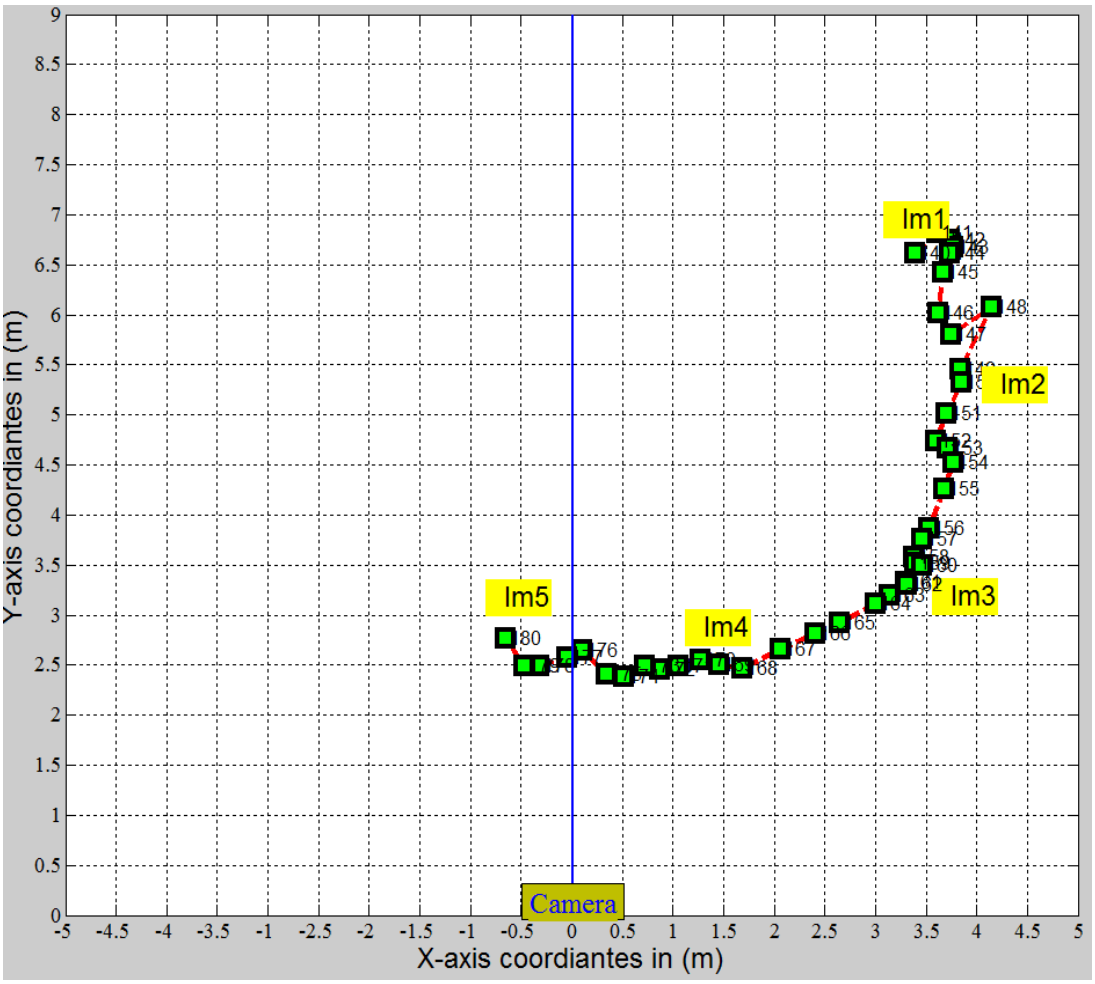
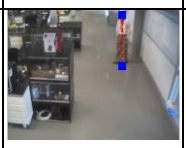
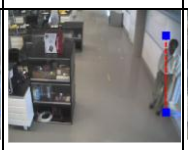
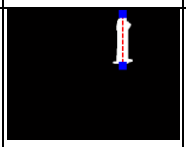
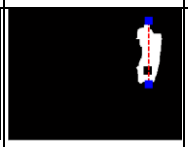
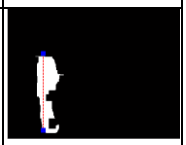


Figure 5.4: Trajectory of motion in XY domain for the moving object.

In Table 5.2, the first row shows the original image where the top and bottom points of the moving object have been marked with blue dots. The second row shows the detected object using GAP algorithm; in all images, the moving object has been correctly detected using the GAP hence the top and bottom points have been correctly identified in the image. The third row shows the location of the moving object on the ground. This location corresponds to the yellow marks in Figure 5.4 which shows where the object is located for the specific frame number. Ground location of moving object is not available in the image because an image return the object location with respect to image coordinates (not ground coordinates) while the proposed method computes the real world geometrical location of the moving object (ground location and depth of field).

Table 5.2: Results analysis for select frames in first experiment.

# Frame	140 (Im1)	150 (Im2)	160 (Im3)	170 (Im4)	180 (Im5)
Image					
Blobs image					
Location in XY plane	(3.388, 6.615)	(3.852, 5.320)	(3.463, 3.500)	(1.266, 2.556)	(-0.6488, 2.772)
Groundtruth depth using rangefinder	7.775	7.010	4.976	3.593	3.839
Computed depth using DfT	7.892	7.084	5.594	3.896	3.892
Estimated depth using UKF	7.781	7.107	4.905	3.500	3.862

The fourth ground truth depth of field computed using laser rangefinder while the fifth row shows the depth of field as computed by the triangulation algorithm. The last row shows the estimated depth of field using the unscented Kalman filter. From the last three rows there is a good match between the computed and ground truth depth of field which validate the accuracy of the proposed depth computation method. In addition, the correlation between the estimated depth of field in the last row and the computed one in the fifth row is very high which proves that the unscented Kalman filter has very high estimation accuracy.

The depth estimation results provide the measurements for the tracking algorithm using the unscented Kalman filter. The filter uses the measurement and the previous estimation to estimate the new location of the moving object in the ground coordinates as well as the depth of field. The next three figures show the measured vs. the estimated location of the moving object using the unscented Kalman filter. Figure 5.5 shows the changes in the X-coordinate for the moving object at each frame. The blue line is the ground X-coordinate computed by the proposed triangulation method while the red circles are the estimated values using the unscented Kalman filter. The figure shows a good match between the measured and the estimated value. The mean square error is 0.0437 m and the correlation greater than 98%.

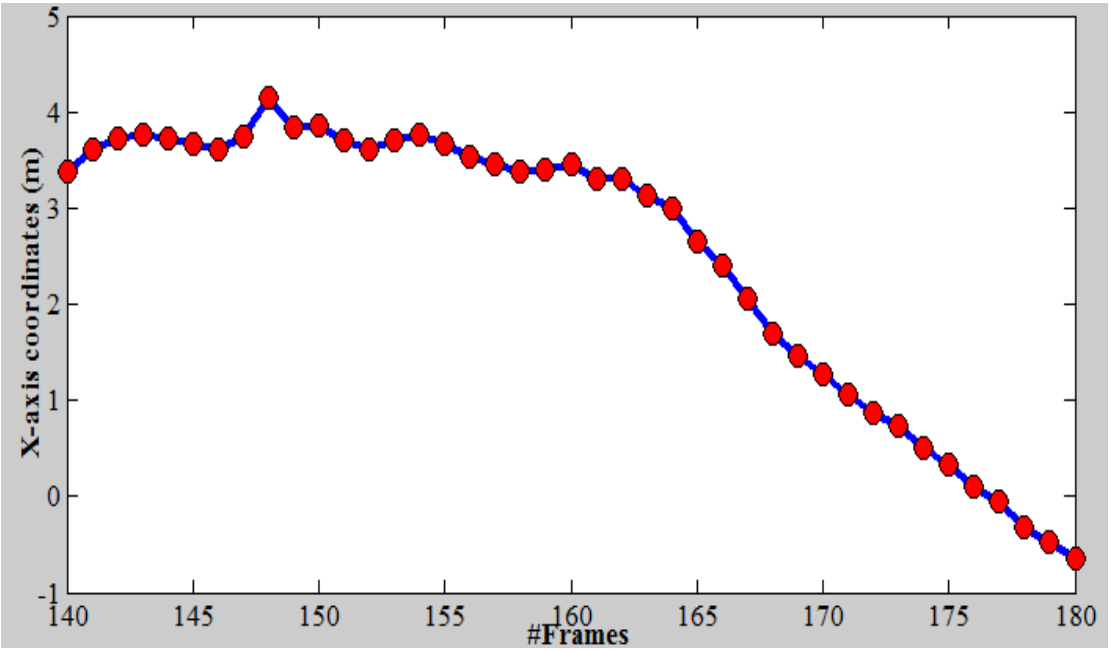


Figure 5.5: Trajectory of the changes in X-axis coordinates.

Figure 5.6 shows the changes in Y-axis coordinate at each frame where the blue line represented the values computed by the proposed triangulation algorithm while the red triangle represents the estimated values using the unscented Kalman filter. Similar to Figure 5.5, the estimation accuracy is very high in this case too and the mean square error is only 0.0531 m and the correlation is greater than 98%.

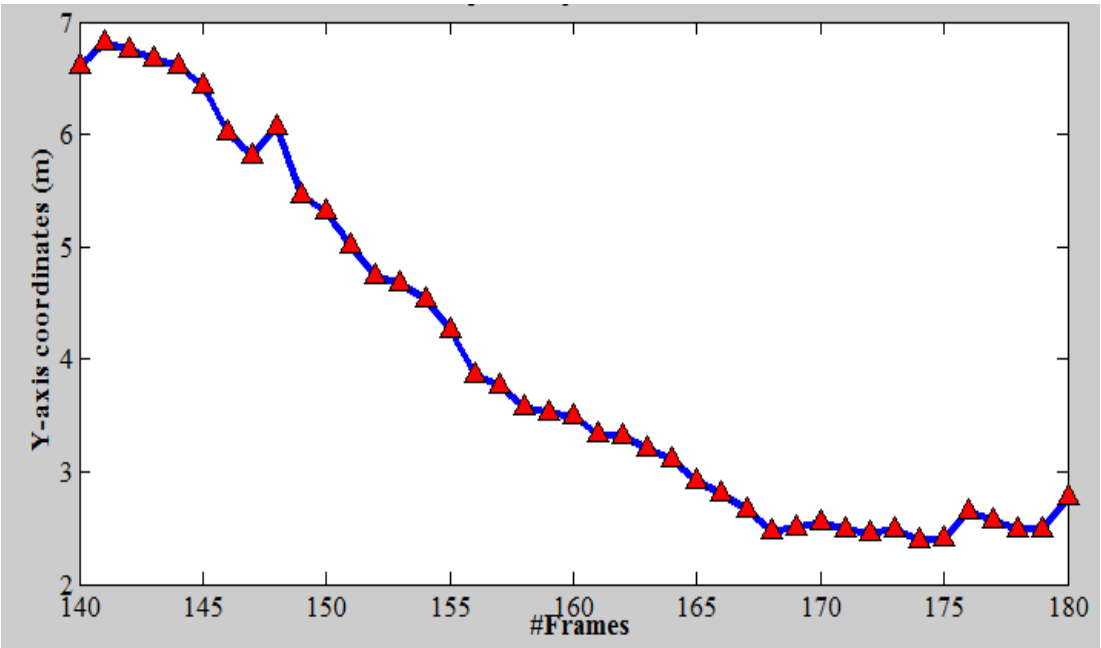


Figure 5.6: Trajectory of the changes in Y-axis coordinates.

Figure 5.7 shows changes in the depth of field at each frame in the video sequence. Since the object is coming closer to the camera the depth of field decreases from maximum value to around 4m then when the object turn right the depth of field start increasing slowly again. In Figure 5.7, the blue line represents the depth of field computed by the triangulation algorithm while the red square is the estimated depth of field computed by the unscented Kalman filter. Similar to Figure 5.5 and Figure 5.6, the estimation accuracy is high with mean square error = 0.0668 m and the correlation is larger than 97%.

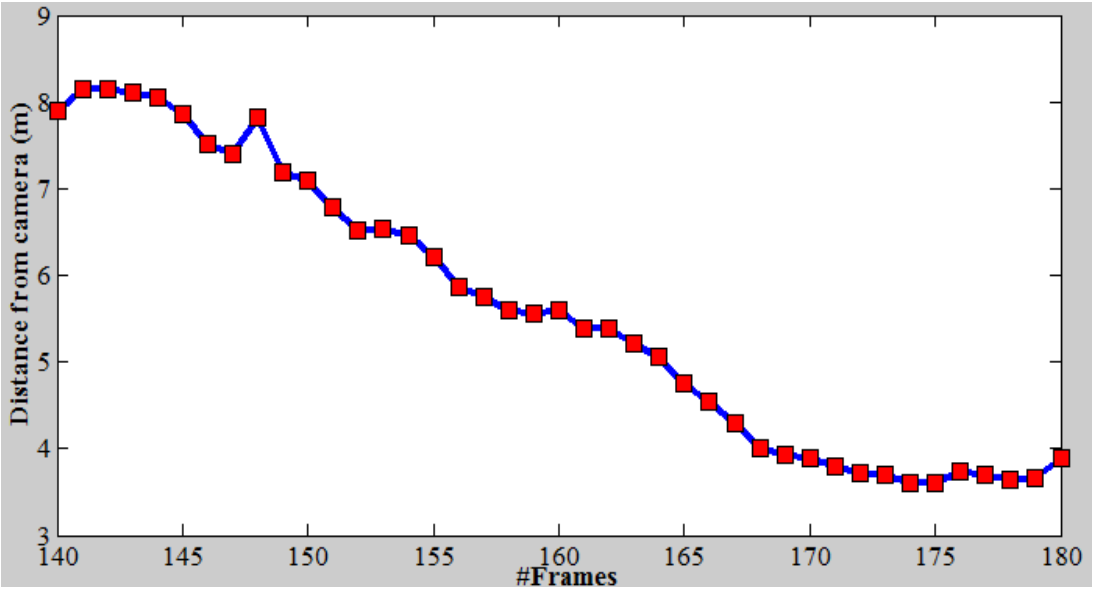


Figure 5.7: Trajectory of the depth of field for the moving object.

As an overall evaluation, the proposed 3D tracking system performed very accurately in all tracking aspects starting from the moving object detection where the extracted blobs resemble the actual objects location. The geometry computational algorithm generated accurate estimation for the depth of field similar to the data collected by the laser rangefinder. Finally, the unscented Kalman filter has very high estimation results and all errors are less than 0.1 m and the correlation is 97%.

5.5.2 Second Experiment (Seq 2)

This sequence includes two moving object walking away from the camera. The images are captured using an analog camera in outdoor environment where the camera was installed at height of 10.406 m and pitch angle of 60 degrees. The camera has a horizontal field of view of 55.5 degrees and vertical field of view of 42.5 degrees. Since the two subjects move in parallel away from the camera, both subjects will have similar depth of field. In addition, the parallel motion should be maintain in the X-axis of the motion. Figure 5.8 shows the subjects movements in the ground plane. Since the field of view is large, the image is been zoomed-in to enhance the visibility. Similar to the previous experiment, five images have selected for further analysis. The ground plane motion shows that the two subjects move parallel to each other. Then subject 2 (the right subject) starts to diverge to the right.

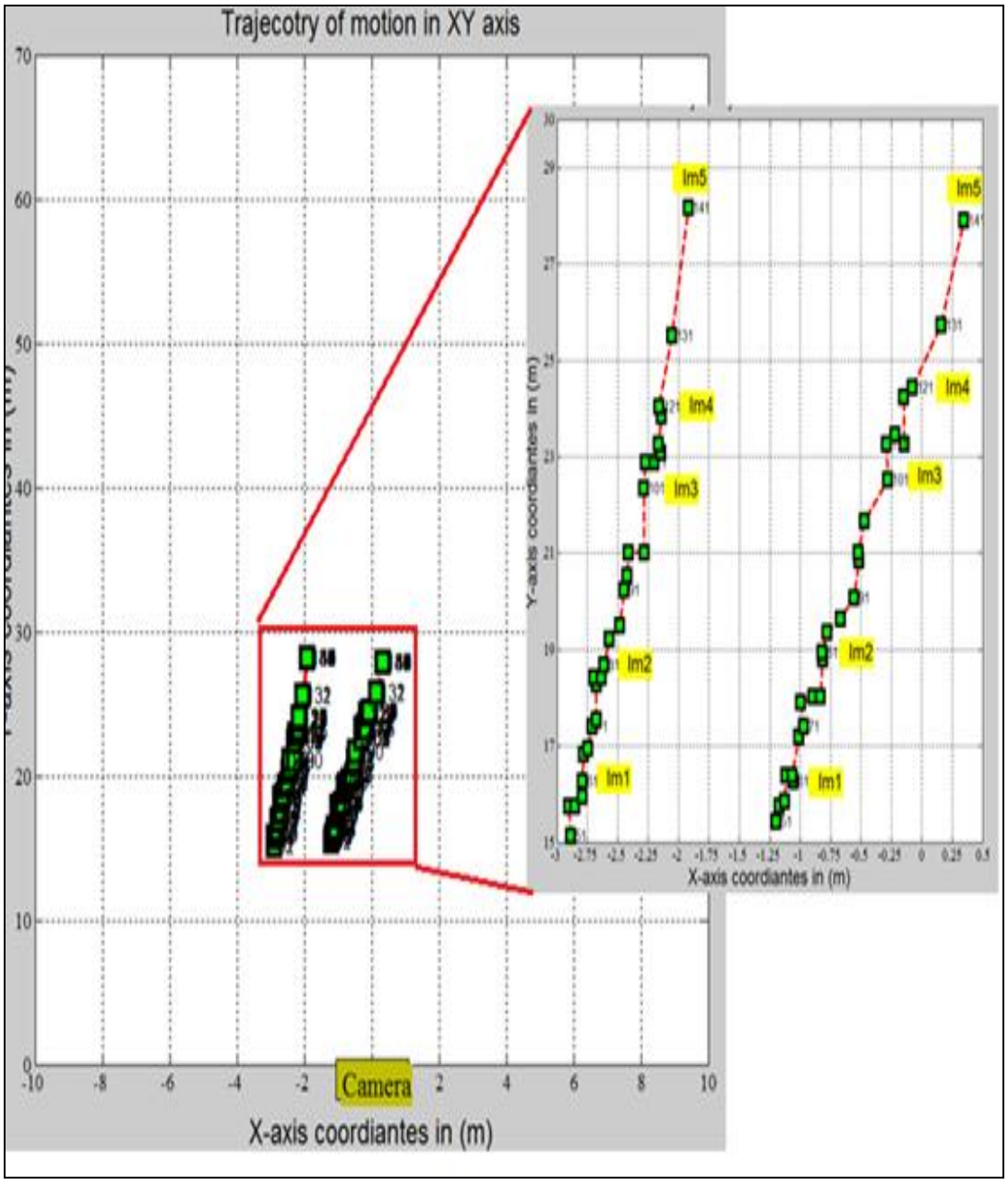
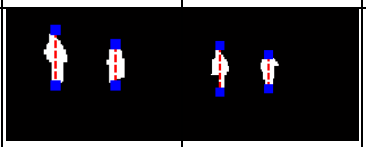
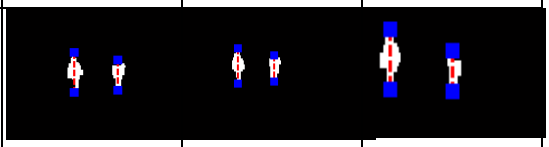


Figure 5.8: Trajectory of motion in XY domain for the moving object.

Table 5.3 shows selected images from the second video sequence. The first row shows the original image with two moving objects. The second row shows binary images of the extracted moving objects. These two rows show that the moving target is correctly detected by GAP method. The images shown in Table 5.3 correspond to the yellow marks in Figure 5.8 (ground plane). The third row shows the location of the moving subject in the ground plane, the first coordinate corresponds to the right

subject and the second coordinate corresponds to the left subject. The fourth row shows the ground truth depth of field (distance between moving object and camera) computed using a laser range finder. The fifth row shows the depth of filed computed using the developed triangulation method. The computed depth of field is similar to the ground truth one for most of the selected sequences which proves the accuracy of the depth computation algorithm. The last row shows the estimated object depth using the unscented Kalman filter for both moving objects. The estimation accuracy is very high and the error and is in the scale of centimeters which is very small compared to the image scale. The mean square error for estimating the depth is 0.0678m and correlation is 99% which is very high.

Table 5.3: Results analysis for selected frames in second experiment.

# Frame	61 (Im1)	81 (Im2)	101 (Im3)	121 (Im4)	141 (Im5)
Image					
Blobs image					
Location in XY plane	(-2.8, 16.3) (-1.1, 16.3)	(-2.6, 18.7) (-0.8, 18.9)	(-2.3, 22.3) (-0.3, 22.5)	(-2.2, 24.0) (-0.1, 24.4)	(-1.9, 28.1) 0.34, 27.9)
Ground truth depth	18.78 18.53	22.88 23.14	26.03 25.18	24.17 24.96	29.05 28.74
Computed depth using DfT	19.5144 19.3424	21.5340 21.6255	24.7560 24.8130	26.2808 26.5640	30.0729 29.7701
Estimated depth of field using UKF	19.5455 19.3867	21.5785 21.6558	24.6814 24.7441	26.3450 26.6364	29.9989 29.6961

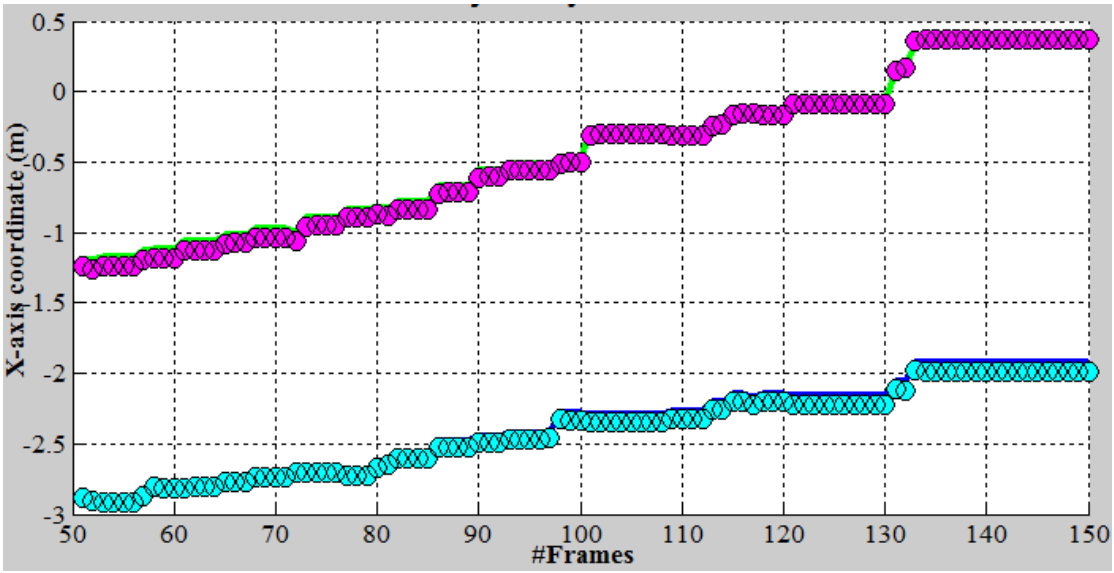


Figure 5.9: Trajectory of the changes in X-axis coordinates.

Figure 5.9 above shows the changes in the X-coordinate for the moving object at each frame. The blue solid line is for the first object while the green solid line is for the second object. These solid lines show the computed X-coordinate using the triangulation method. The cyan and magenta circles show the estimated X-coordinate using the unscented Kalman filter. The circles almost overlay the solid lines which mean the estimation is very close to the depth data estimated by the triangulation method. The estimation has a small mean square error of 0.0474 m and the correlation is 99%.

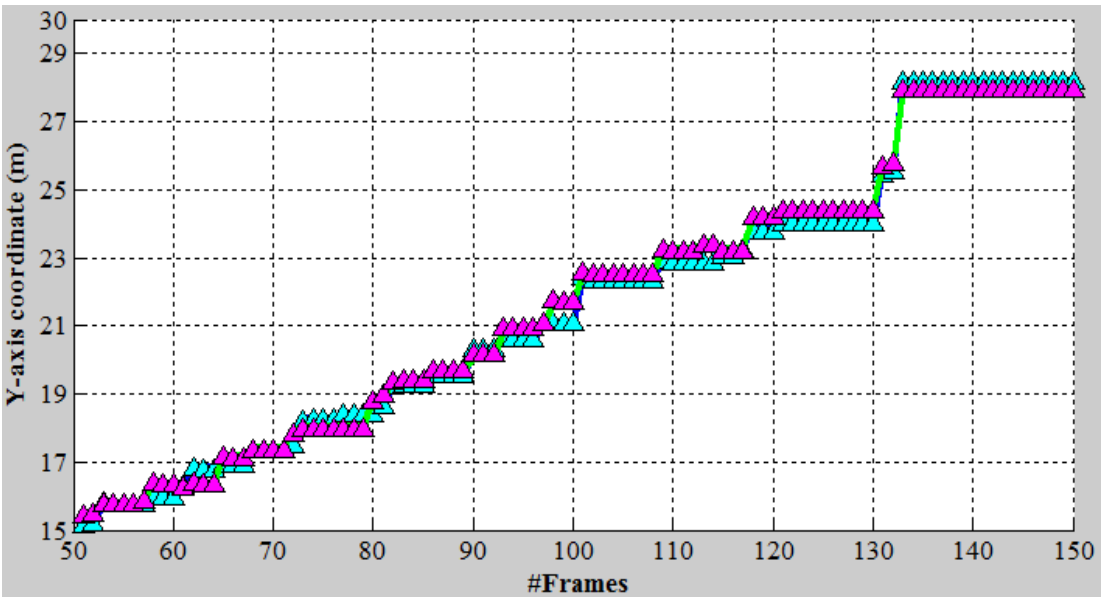


Figure 5.10: Trajectory of the changes in Y-axis coordinates.

Figure 5.10 above shows the changes in the Y-coordinate for the moving object at each frame. The two objects have similar distance from the camera which is clear in Figure 5.10 where the solid lines overlay each other. Similar to the X-coordinate estimation, the cyan and magenta colored triangles overlay the solid lines which indicate the unscented Kalman filter estimation is very close to the ground truth data computed by the triangulation algorithm. The estimation has small mean square error 0.0497 m and the correlation is 99%.

Figure 5.11 shows changes in the depth of field at each frame in the video sequence. The depth of field is increasing with frame number because both objects are moving away from the camera. Both objects have similar depth of field because there are moving in parallel away from the camera. The cyan and magenta colored square represents the unscented Kalman filter estimation for the depth of field of both objects. The estimated depth is very close to the computed one with mean square error 0.0678m and correlation of 99%.

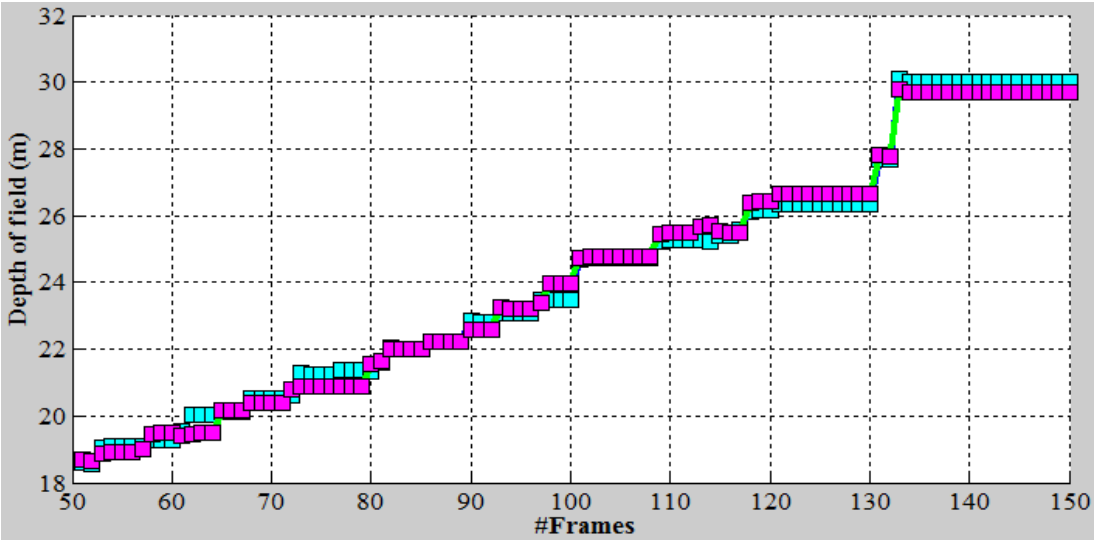


Figure 5.11: Trajectory of the depth of field for the moving object.

In this video sequence, the proposed 3D tracking system correctly detected the moving object in the image then it computes the ground coordinates and the depth of field for each one of the detected object. Finally, the unscented Kalman filter produced accurate estimation for the moving object location and the depth of field. The error of this estimation is in the scale of few centimeters and with very high correlation index.

5.5.3 Third Experiment (Seq 3)

This sequence was shot in a walkway where two moving objects located at different depth move away from the camera. The images were captured using a camera at a height of 3.579m and with a viewing angle of 60 degrees. This camera has an equal vertical and horizontal field of view of 44.7 degrees.

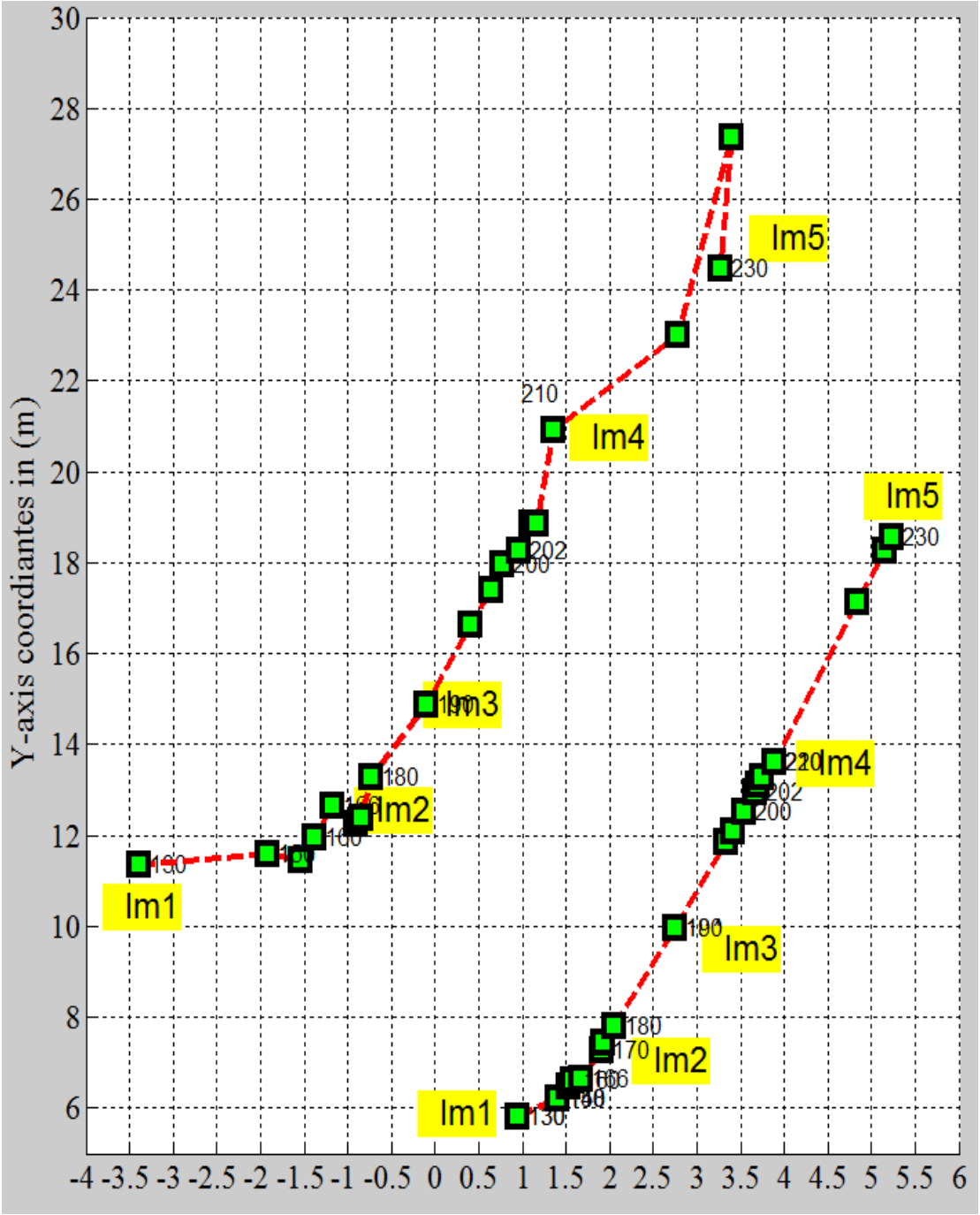
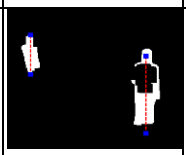
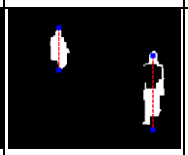
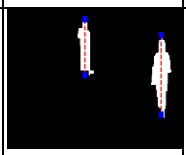
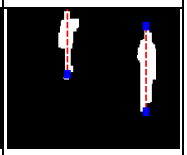


Figure 5.12: Trajectory of motion in XY domain for the moving object.

Figure 5.12 shows the subjects movements in the ground plane. Similar to other experiments, five frames have been extracted from the video sequence and displayed in Table 5.4 for detailed analysis. The first row of Table 5.4 shows the original image where the bottom most and top most points of the moving objects have been marked by a blue dot. The second row shows the detected objects blobs using the GAP method. These blobs show that the moving object has been correctly detected in the image and the full size of the object has been correctly detected. The third row shows the location of the two moving subjects in the ground plane (XY plane of Figure 5.12). The location information cannot be extracted directly from the image but rather computed by the triangulation algorithm.

Table 5.4: Results analysis for selected frames in third experiment.

# Frame	125 (Im1)	145 (Im2)	165 (Im3)	185 (Im4)	105 (Im5)
Image					
Blobs image					
Location in XY plane	(-3.2, 7.17) (0.69, 5.71)	(-1.9, 11.5) 1.47, 6.19)	-1.18, 14.9) 1.65, 6.66)	(-0.1, 12.7) (2.74, 9.96)	(1.17, 18.8) (3.75, 13.3)
Ground truth depth using laser rangefinder	8.69 4.87	12.37 7.35	13.195 7.92	14.569 9.698	19.531 13.52
Computed depth of field	8.3329 6.7740	12.2539 7.3600	13.2149 7.7422	15.3054 10.9326	19.2261 14.2541
Estimated depth of field using UKF	8.3329 6.8146	12.272 7.3532	13.2628 7.8170	15.3297 10.8577	19.2552 14.3289

The fourth row shows the depth of field computed using laser range finder which the fifth row show the depth of field computed by the triangulation algorithm. For Im1 the depth of the first subject is larger than the ground truth depth (computed depth = 6.77m and ground truth depth = 4.78m). This is because the subject is standing on the staircase above the ground. Therefore, the triangulation method cannot compute its correct depth. For the other images, the moving subjects are on the ground level thus accurate depth is computed. The last row shows the estimated depth of field of both moving subjects using the unscented Kalman filter.

Figures 5.13 shows the changes in X-axis coordinate with the number of frames; the green solid line is the computed value for the first subject while the blue solid line is for the second subject. The magenta circles is the unscented Kalman filter estimation for the first subject while the cyan colored circles represent the estimation for the second subject. The mean square error for this estimation is very small (5.3cm).

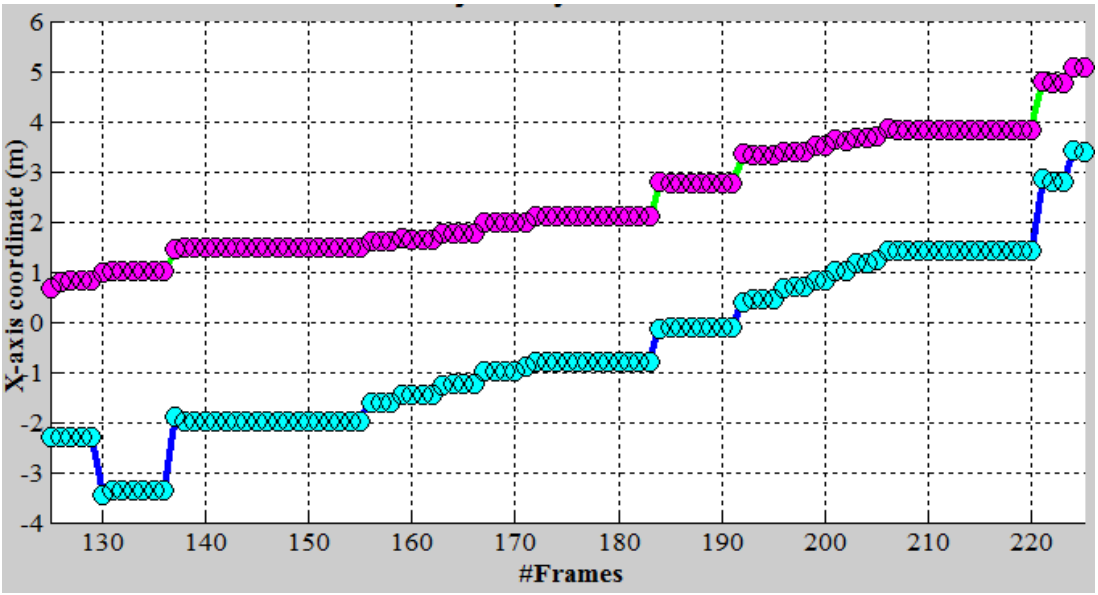


Figure 5.13: Trajectory of the changes in X-axis coordinates.

Similarly, figure 5.14 shows the changes in the Y-axis coordinate with the number of frames. The figure shows that the two subjects kept a distance of 6m between them through the motion. In addition, the figure shows that subject 2 (red line) is much closer to the camera than subject 1 (blue line). For both subjects, the unscented Kalman filter estimation is very similar to the ground truth data computed by the

triangulation method because the triangles (magenta for first subject and cyan for second subject) overlay the solid lines. The RMSE error for this estimation is 5.51cm and the correlation is more than 98%.

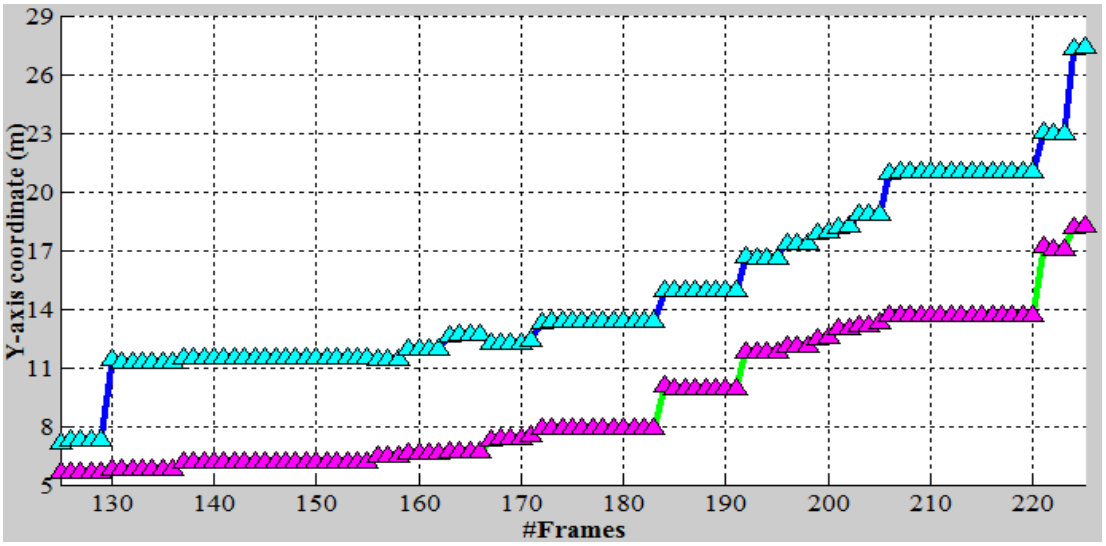


Figure 5.14: Trajectory of the changes in Y-axis coordinates.

Figure 5.15 shows the estimation for the depth of field (distance between the camera and the moving subject). The depth of field has similar behavior like the Y-axis coordinate where subject 1 has larger depth of field than subject 2 for all frames. Similarly, the unscented Kalman filter estimation have very high accuracy with RMSE = 6.1cm and correlation larger than 98%.

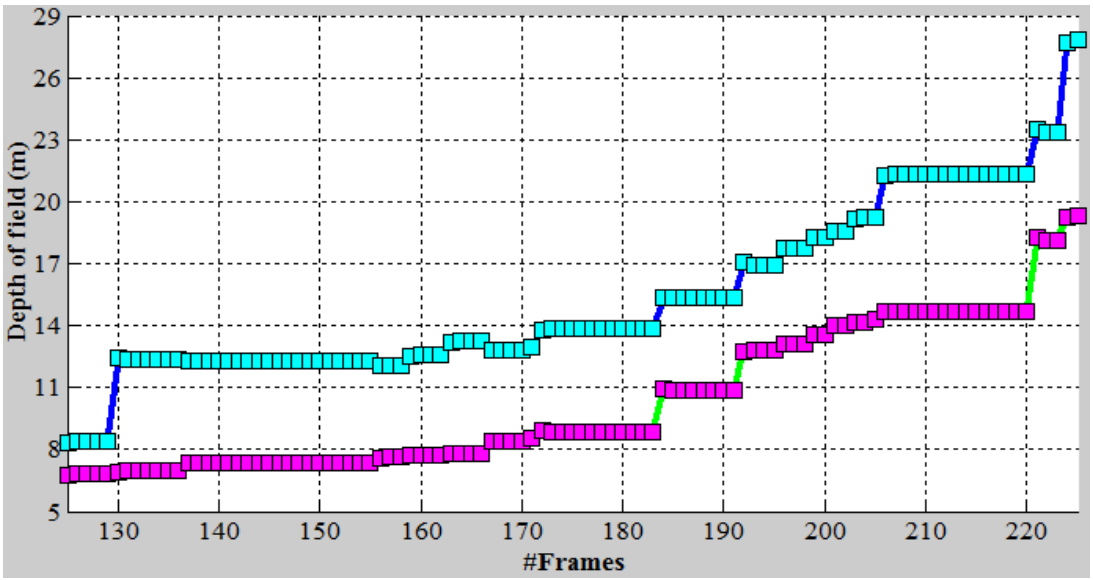


Figure 5.15: Trajectory of the depth of field for the moving object.

5.5.4 Fourth Experiment (Seq 4)

This experiment uses similar imaging setup like the first one where the camera has an identical field of view of 62 degree, 60 degrees pitch angle and it is installed at a height of 2.654m. This video sequence includes two moving subjects moving in the opposite direction. The two objects occlude each other for some time in the sequence. In the case of occlusion, the ground location of the occluded object cannot be detected from the image. Therefore, the occluded object is compensated using its motion history as it has been explained in the Section 3.2.3. However, motion compensation cannot reveal the true location of the moving object and this has its effect on the accuracy of the computed depth of field by the triangulation method.

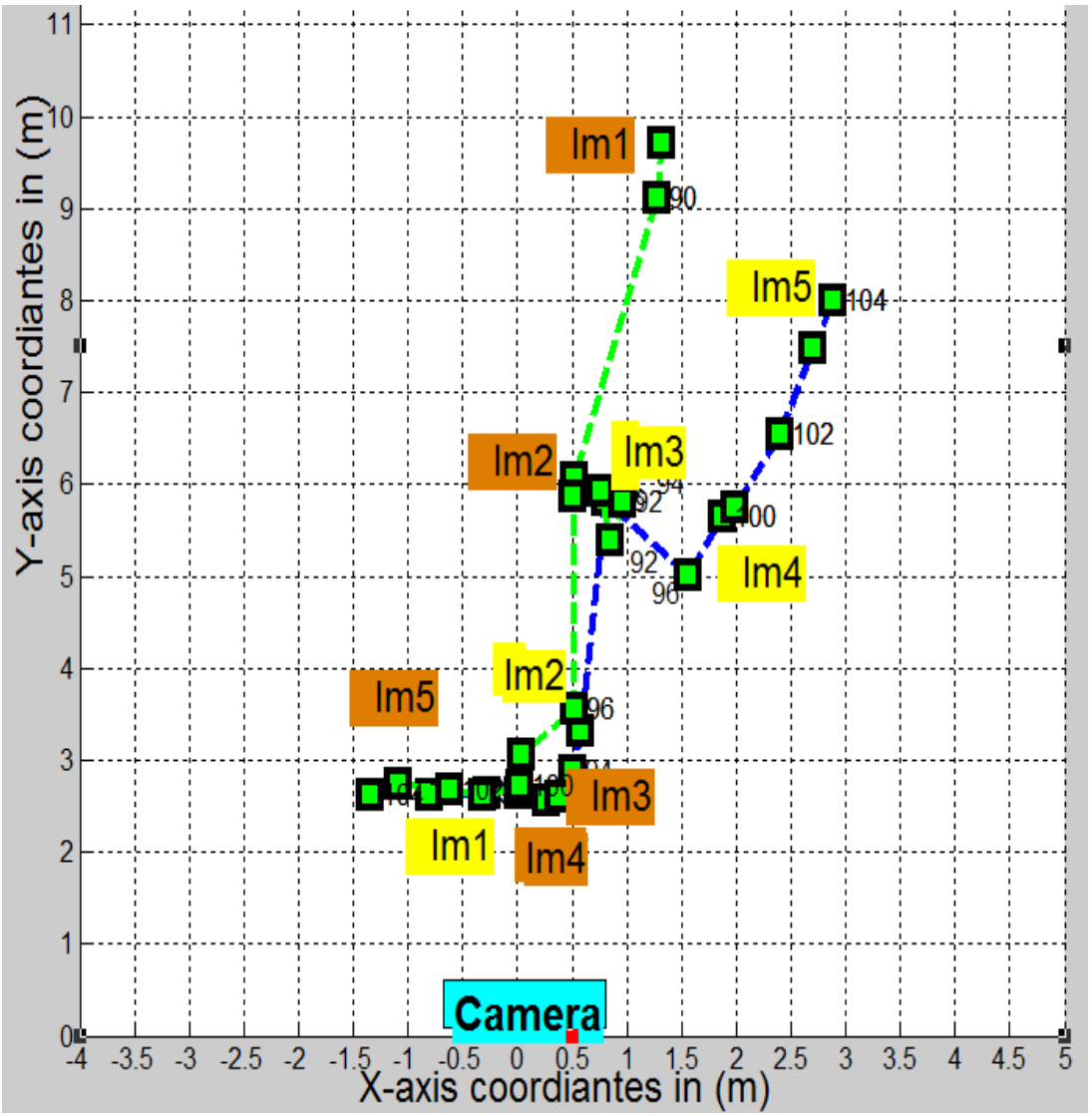
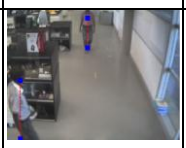
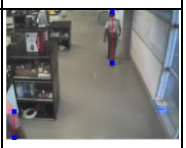
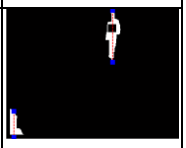


Figure 5.16: Trajectory of motion in XY domain for the moving object.

Similar to the previous sequences, firstly ground plane of the motion is laid out in Figure 5.16. The blue trajectory is for the first subject while the green trajectory is for the second subject. The first subject is going from near the camera to far from the camera in the scene while the second object is taking the opposite direction.

Table 5.5 shows selected images from the fourth video sequence. These images correspond to the yellow colored marks for the trajectory of the first subject (blue line) and the orange colored marks for the trajectory of the second subject (green line). In the first image, the two objects were not in occlusion thus the geometrical computation follows the standard flow. In the second image, subject 1 occlude subject 2. In this case the blobs image shows only one long object. The object detection algorithm returns only the top and bottom point of the moving subject without any recognition work.

Table 5.5: Results analysis for selected frames in fourth experiment.

# Frame	89 (Im1)	93 (Im2)	96 (Im3)	101 (Im4)	104 (Im5)
Image					
Blobs image					
Location in XY plane	(-0.8, 2.63) (1.32, 9.72)	0.38, 2.59) (0.92, 6.85)	0.52, 3.55) 1.39, 6.02)	1.99, 5.75) (-0.3, 2.63)	(2.89, 7.99) (-1.3, 2.63)
Ground truth depth	3.6580 7.0690	3.6240 6.1940	4.2040 4.6210	5.8700 3.4790	6.8990 4.0790
Computed depth	3.8193 7.2604	3.7277 7.1315	4.1624 5.3250	6.3411 3.7472	7.9016 3.9683
Estimated depth of field using UKF	3.8757 7.3324	3.7210 6.6396	4.2848 5.5453	6.3768 3.7171	7.9674 3.9179

Normally in multiple object detection, a data association algorithm is used for corresponding new measurement to previous one. This is very important in order to establish the accurate trajectory especially in the cases of occlusion and swapping direction. In this implementation, data association is done based on maximum similarity between the current and previous measurements. In the case of occlusion, the number of current measurement is less than the previous one. Thus the new measurement is match to the previous one in order to identify the occluded object. Then the occluded object measurement is computed by propagation of the previous measurement though the motion model. For example, if the motion model follows a fixed acceleration behavior, the new location will be computed using Equation 3.1. In Im2, the occluded object is the second object and the estimation error for the occluded object is much larger than the one obtained for the occluding object (error is 0.0067m for the first subject and 0.3919m for the second subject). In Im3, the two subjects are still in occlusion and thus they have large estimation error compare to Im1. In the last two images Im4 and Im5, there is no occlusion and thus the estimation accuracy is similar to Im1.

Figure 5.17 shows the changes in the X-coordinate for the moving object at each frame. The blue solid line is for the first object while the green solid line is for the second object. During the occlusion, the geometry of the occluded object will not be available and it is replaced by an estimated location. In figure 5.17, the occlusion starts at frame 93 until frame 98. The computed geometry (location and depth of filed for these frames) has large error compared the other frames. This is because the occlusion compensation does not give the true location of the object. Similarly, the unscented Kalman filtering for these frames has larger error compared to the non-occlusion frames. For estimated the X-coordinate the overall mean square error is 0.445m and the correlation is less than 92%

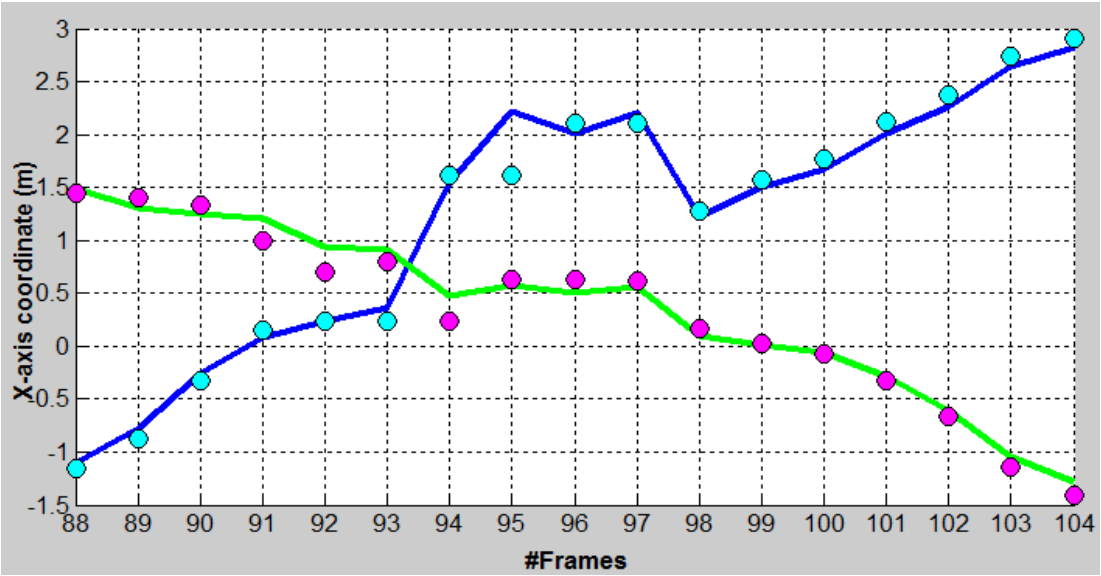


Figure 5.17: Trajectory of the changes in X-axis coordinates.

Figure 5.18 shows the changes in the Y-coordinate for the moving object at each frame. In frame 91, the Y-coordinate of the second subject (green color) changes from 9m to 3.5m which is very big jump. This is because at this frame the object was occluded and there was no measurement to correct the estimation error. Similar behavior can be said about Figure 5.19 where the second object exhibits the same big change. The mean square for estimating the Y-axis coordinate is 0.655m and the correlation is 90% and the mean square for the depth of field is around 0.674m and the correlation is around 90%.

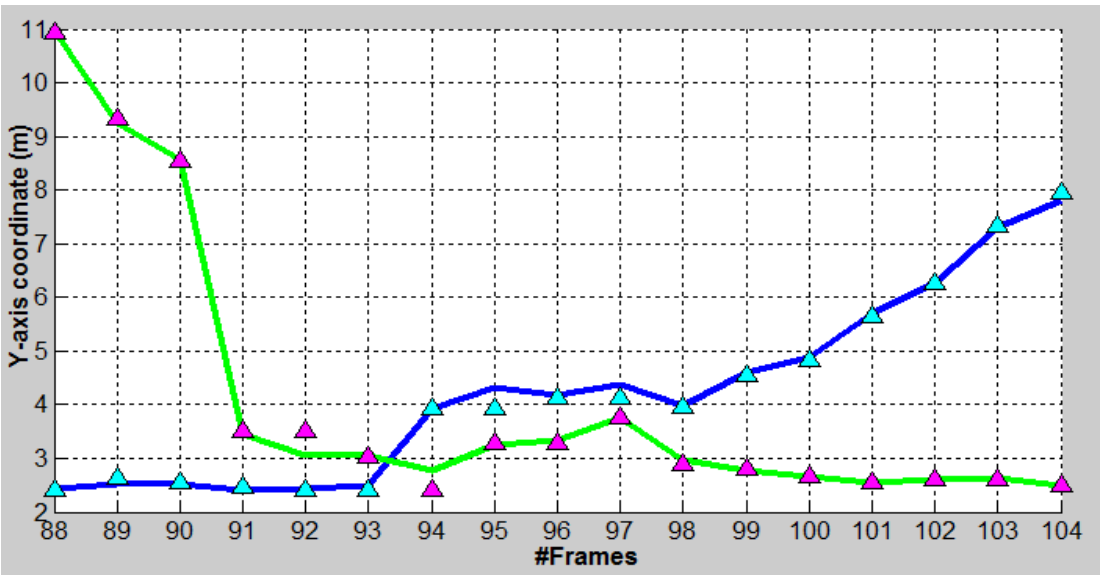


Figure 5.18: Trajectory of the changes in Y-axis coordinates.

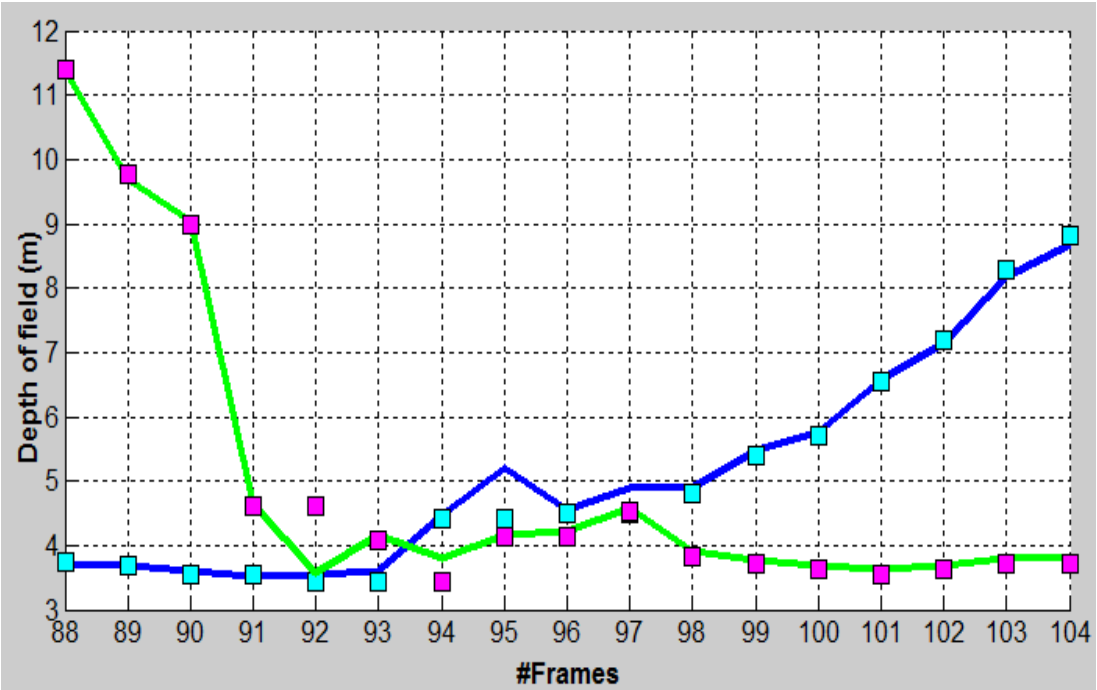


Figure 5.19: Trajectory of the depth of field for the moving object.

In this sequence, the moving object occludes each other. Thus, the accuracy of the depth computation as well as the unscented Kalman filter is reduced compared to non-occlusion frames. This is because occlusion compensation module does not give the true location of the moving object but rather an estimated one.

5.5.5 Fifth Experiment (Seq 5)

This sequence includes two moving object walking away from the camera on a grassy land. The images are captured using an analog camera in outdoor environment where the camera was installed at height of 10.48 m and pitch angle of 59.5 degrees. The camera has a horizontal field of view of 55.5 degrees and vertical field of view of 42.5 degrees. The two objects are moving far from the camera therefore the depth of field is increasing with the time. Figure 5.20 shows the subjects movements in the ground plane. Similar to the previous experiment, five images have selected for further analysis. The two objects moves away from the camera in different direction. The first object moves in straight line away from the camera while the second object moves away to the right direction of the scene.

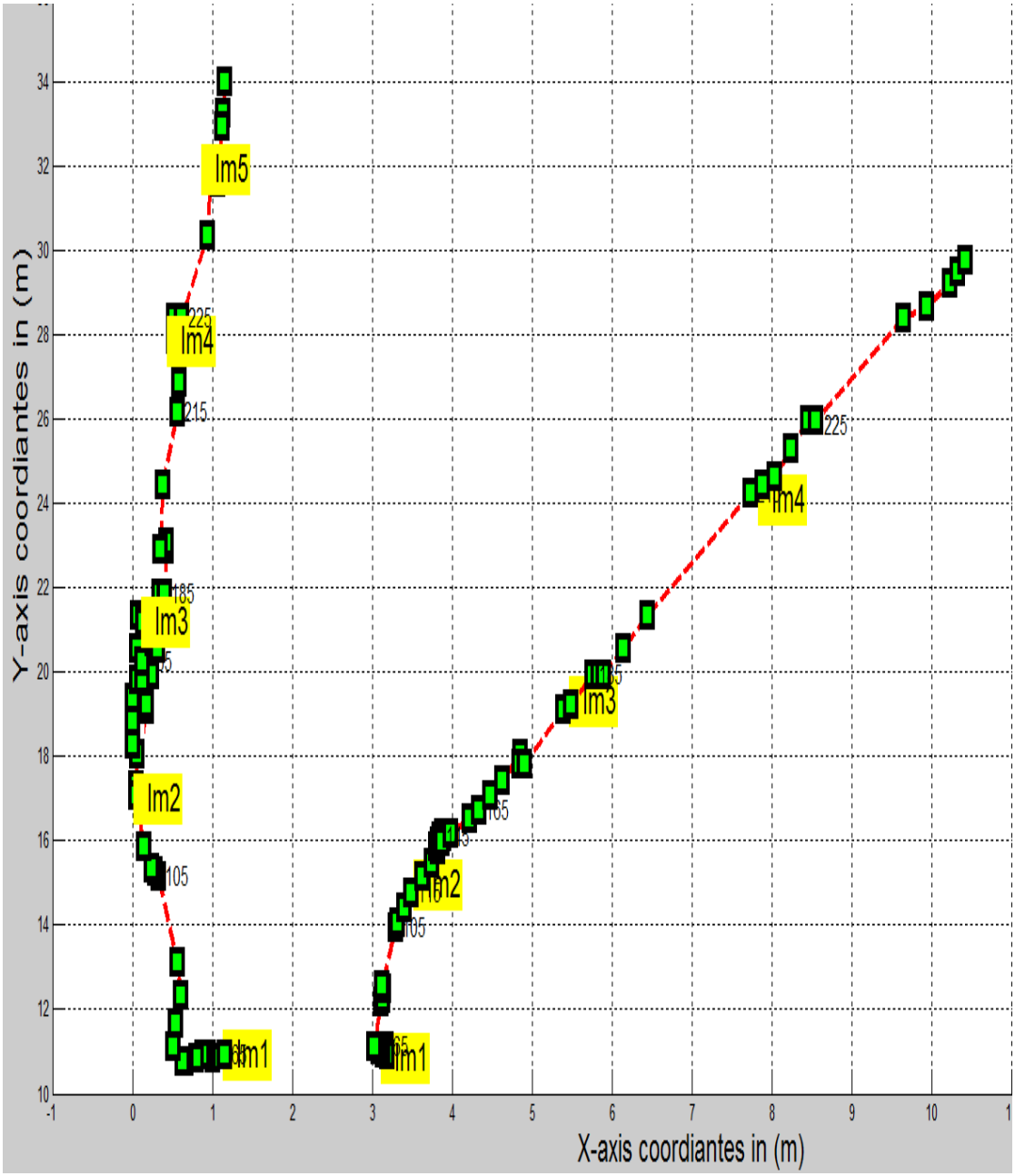
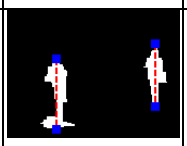
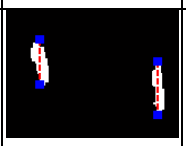
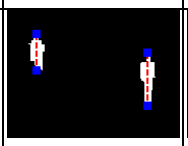
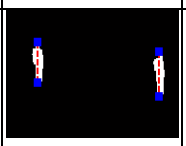
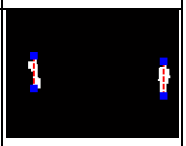


Figure 5.20: Trajectory of motion in XY domain for the moving object.

Table 5.5 shows selected images from the second video sequence. The first row shows the original image with two moving objects. The second row shows binary images of the extracted moving objects where these blobs have been zoomed in to enhance the visibility. These two rows show that the moving target is correctly detected by GAP method. The images shown in Table 5.5 correspond to the yellow marks in Figure 5.20 (ground plane). The third row shows the location of the moving subject in the ground plane, the first coordinate corresponds to the left subject and the

second coordinate corresponds to the right subject. The fourth row shows the ground truth depth of field (distance between moving object and camera) computed using a laser range finder. The fifth row shows the depth of filed computed using the developed triangulation method. The computed depth of field is similar to the ground truth one for most of the selected sequences which proves the accuracy of the depth computation algorithm. The last row shows the estimated object depth using the unscented Kalman filter for both moving objects. The estimation accuracy is very high and the error and is in the scale of centimeters which is very small compared to the image scale. The mean square error for estimating the depth is 0.0571m and correlation is 99% which is very high.

Table 5.5: Results analysis for selected frames in second experiment.

# Frame	90 (Im1)	130 (Im2)	170 (Im3)	210 (Im4)	250 (Im5)
Image					
Blobs image					
Location in XY plane	0.51, 11.1) 3.1, 12.2)	0.0, 18.3) 3.8, 15.8)	0.3, 20.6) 4.6, 17.4)	0.6, 26.2) 7.7, 24.2)	1.1, 31.6) 10.4, 29.8)
Ground truth depth	14.88 16.73	22.85 20.14	24.03 22.18	27.27 26.96	32.05 31.78
Computed depth using DfT	15.2829 16.3424	21.0876 19.3129	23.3427 20.8602	28.1965 27.5154	33.2960 33.2343
Estimated depth of field using UKF	15.3118 16.3194	21.1450 19.3487	23.2709 20.9254	28.2019 27.5642	33.2955 33.3091

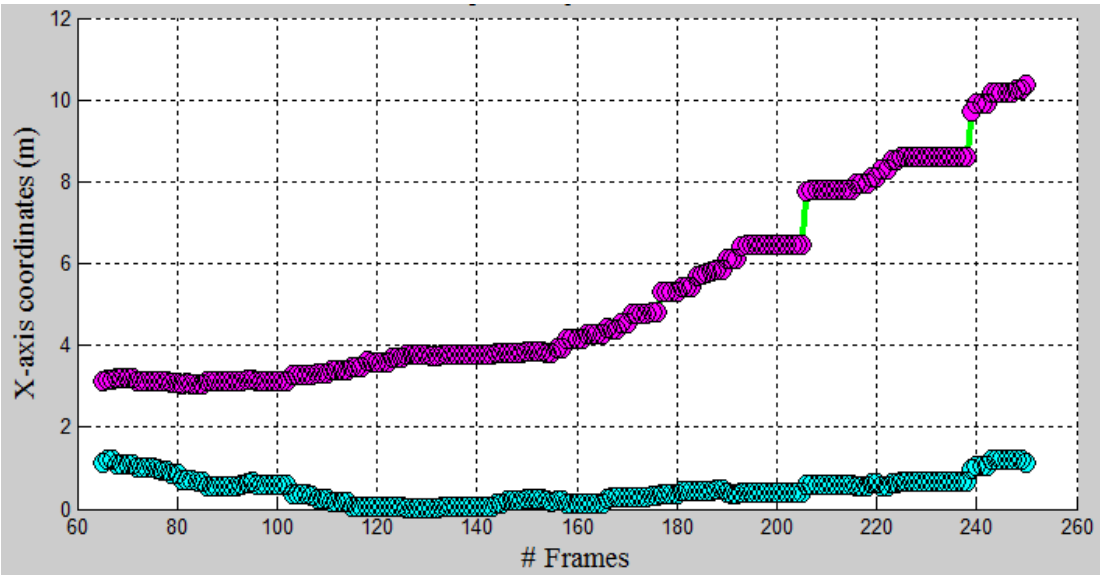


Figure 5.21: Trajectory of the changes in X-axis coordinates.

Figure 5.21 above shows the changes in the X-coordinate for the moving object at each frame. The blue solid line is for the right object while the green solid line is for the left object. These solid lines show the computed X-coordinate using the triangulation method. The cyan and magenta circles show the estimated X-coordinate using the unscented Kalman filter. The circles almost overlay the solid lines which mean the estimation is very close to the depth data estimated by the triangulation method. The estimation has a small mean square error of 0.0413 m and the correlation is 99%.

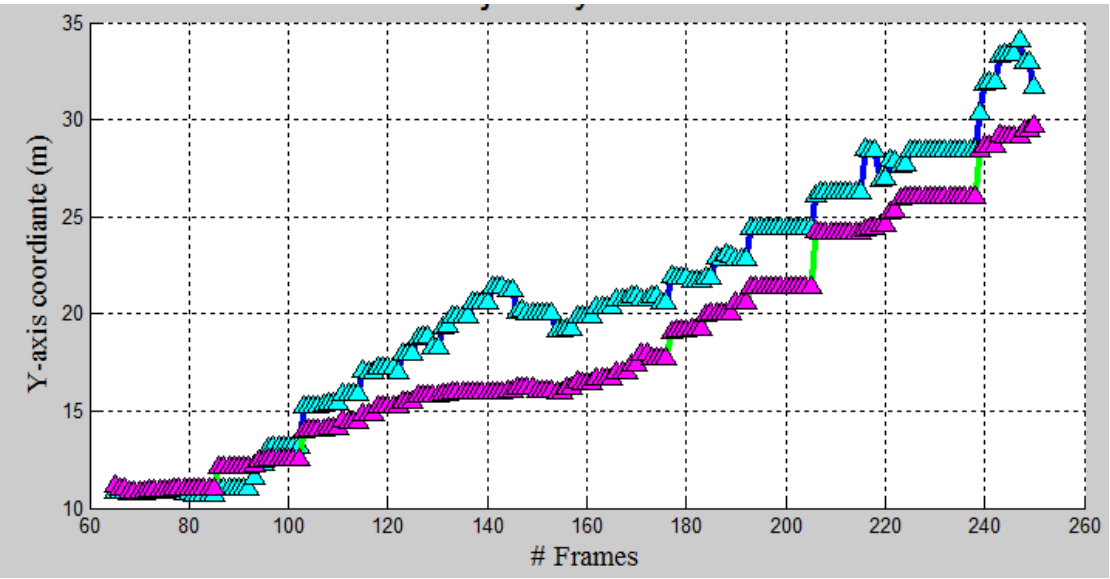


Figure 5.22: Trajectory of the changes in Y-axis coordinates.

Figure 5.22 above shows the changes in the Y-coordinate for the moving object at each frame. The two objects have similar distance from the camera which is clear in Figure 5.22 where the solid lines overlay each other. Similar to the X-coordinate estimation, the cyan and magenta colored triangles overlay the solid lines which indicate the unscented Kalman filter estimation is very close to the ground truth data computed by the triangulation algorithm. The estimation has small mean square error 0.0539 m and the correlation is 99%.

Figure 5.23 shows changes in the depth of field at each frame in the video sequence. The depth of field is increasing with frame number because both objects are moving away from the camera. Both objects have similar depth of field because there are moving in parallel away from the camera. The cyan and magenta colored square represents the unscented Kalman filter estimation for the depth of field of both objects. The estimated depth is very close to the computed one with mean square error 0.0571m and correlation of 99%.

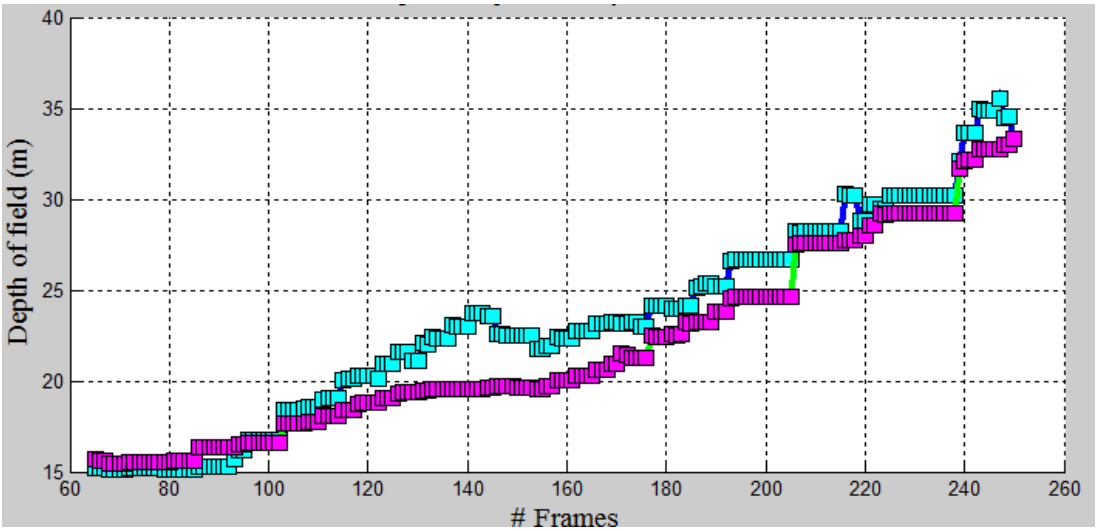


Figure 5.23: Trajectory of the depth of field for the moving object.

In this video sequence, the proposed 3D tracking system correctly detected the moving object in the image then it computes the ground coordinates and the depth of field for each one of the detected object. Finally, the unscented Kalman filter produced accurate estimation for the moving object location and the depth of field. The error of this estimation is in the scale of few centimeters and with very high correlation index.

5.6 **Chapter summary**

In this chapter, the proposed 3D tracking system is been tested and validated in five different experiments. Initially, this chapter discussed the experimental setup and the data collection process. After that an evaluation criteria is defined for evaluating the performance of the system. The proposed system has shown good accuracy in tracking moving object using only one camera while existing 3D tracking system uses multiple cameras. Moreover, this system has real time performance both at the geometrical computation as well as the depth computation. These experiments presented a comparison between ground truth depth of field and depth of field computed using the developed triangulation techniques. In addition, the experiment also compared the depth of field estimated using the unscented Kalman filter with the ground truth depth of field as well as the computed depth of field. Although the Kalman filter estimation uses the computed depth of field as measurement. It has high estimation accuracy than the computed depth of field specially in the case of occlusion.

CHAPTER 6
CONCLUSION AND FUTURE WORKS

6.1 **Introduction**

This chapter concludes the work presented in this thesis and summarizes findings achieved in this work. Moreover it also discusses the possible future works and enhancement that can be added to the ideas proposed in this thesis.

6.2 **Conclusions**

In this thesis, a novel 3D tracking system is been proposed using single camera. The system uses single camera to compute 3D location of the moving object. Then it performs the tracking process using 3D measurement computed using triangulation technique. This system has many advantages over existing 3D tracking systems. Firstly, this system uses only one image from single 2D camera for computing the depth of field and the geometry of the scene while the existing 3D systems use multiple cameras (stereovision) or multiple images from the same camera (shape from focus and structure from motion). Secondly, this system has very small computational requirements compared to other 3D tracking systems. Thus, it is suitable to be used in real time application such as traffic management and computer games. Thirdly, this system does not require any special hardware installations rather it can be used with the existing 2D camera installation.

The proposed 3D tracking system consists of three main components. The thesis has described all these components in details in the previous chapters. The first component is about detecting the moving object in the image. Background subtraction is the best algorithm to be used for detecting an unknown object in the scene. The grayscale arranging pairs (GAP) algorithm has been used for learning the background of the scene because it uses intensity difference rather than intensity value for modeling the background which is more stable against illumination variations. After

that, for every new frame, the modeled background is used for detecting any new object in the scene as well as for identifying its coordinates in the image. The detection algorithm returns the bottom most and top most points of the moving object blob. These points are used for computing the geometry of the moving object by the next component of the tracking system.

The second component of the tracking system is a novel depth and geometry computation method from single 2D image. This method uses basic information about the scene such as the height and the pitch angle of the camera as well as the field of view. Then it forms a geometrical model using triangulation relationships in scene in order to compute the depth and the ground location of any object in the image given its feet/bottom-most location in image coordinates. In addition, the method can compute the height of the object given its bottom most and top most points in image coordinates.

The third component of this system is the object tracking algorithm. This component predicts the new location of the moving object given its previous location and measured observation. Many algorithms have been proposed for 3D tracking especially stochastic filters which represent more than 50% of the published work on object tracking for the last five years. In this thesis, a thorough comparison has been conducted in order to study the existing stochastic fileting algorithms and selecting the best one to be used in the proposed system. The study has concluded that the unscented Kalman filter is the best algorithm to be used for 3D tracking because it has higher accuracy than other Kalman filters and particle filters. In addition, it has shorter computational time compared to the particle filters and it can work well with both linear and nonlinear systems. 3D tracking using unscented Kalman filter has shown good accuracy and it has small error in the scale of centimeters. Moreover, the correlation is higher than 97% except in the cases of occlusion where some information about the occluded object is missing.

As a conclusion, 3D tracking system using single 2D camera has been presented in this thesis. The system uses the GAP method for detecting the moving object in the image. Then it uses the proposed triangulation method for computing the depth field and the geometry of the detected object. Finally, the unscented Kalman filter is used

for tracking from frame to frame. This system has been tested using several video sequences and it shows good performance in term of accuracy and computational time.

6.3 **Contribution**

This thesis presented a 3D tracking system using single camera. The salient features of this work include a novel depth and geometry computation algorithm from single image using triangulation. This algorithm uses only basic camera setup information such as camera height, field of views and pitch angle and it can compute the depth of field and the ground location of any object in the scene given its location in image coordinates. This algorithm has very high accuracy and short computational time and moreover, it can be embedded to the existing camera installation.

Secondly this thesis presented a thorough comparison of the existing object tracking algorithms used in the past five years, specially the stochastic filters. Three types of Kalman filters and three types of particle filters have been thoroughly analyzed using several video sequences and evaluated based on their estimation accuracy and computational complexity. The results of these studies have been published in [3], [22], [23] and [33].

6.4 **Limitations of the Proposed System**

The proposed 3D tracking system relies heavily on the object detection algorithm and the developed triangulation algorithm in order to provide accurate measurement for the unscented Kalman filter. However, both methods have some limitations.

The implemented object detection system has the following limitations. Firstly, the background modeling component has a very high computational time and it work well with static backgrounds. Secondly, the shadow removal component does not remove all types of shadow. Thirdly, the occlusion compensation uses a simple motion model and it does not produce accurate estimation especially for long time occlusion.

This technique requires knowledge of the basic camera installation information such as the camera height and pitch angle. Therefore, this technique cannot be used with video sequences from an unknown source. In addition, measuring these variables might be a difficult task specially measuring the pitch angle.

The proposed triangulation algorithm computes the depth and geometry for any object in the scene provided that the object is located on the ground level. This means if the object is floating above the ground level the proposed algorithm cannot compute its depth of field. In addition, if the surface is not flat the algorithm cannot compute the depth of field for this object because correct triangulation algorithm cannot be formed between this object and the camera center.

6.5 **Publications**

- **Yasir Salih**, Aamir S. Malik, "Comparison of Stochastic Filtering Methods for 3D Tracking", 2011 *Pattern Recognition* 44 (10-11), pp. 2711-2737, March 2011.
- **Yasir Salih**, Aamir S. Malik, Zazilah May, "Depth Estimation Using Monocular Cues from Single Image", *2011 National Postgraduate Conference (NPC 2011)*, UTP, Perak, Malaysia, 19-20 September 2011.
- **Yasir Salih**, Aamir S. Malik, "3D Object Tracking Using Three Kalman Filters" , *2011 IEEE Symposium of Computer & Informatics (ISCI 2011)*, Kuala Lumpur, Malaysia, 20-22 March 2011.
- **Yasir Salih**, Aamir S. Malik, "3D Tracking Using Particle Filters" , *2011 International Instrumentation and Measurement Technology Conference (I2MTC 2011)*, Binjiang, Hangzhau, China, 10-12 May 2011.
- **Yasir Salih**, Aamir S. Malik, "Stochastic Filters for Object Tracking", *2011 the 15th IEEE Symposium on Consumer Electronics (ISCE 2011)*, Singapore, 14-17 June 2011.
- Muzaffar Djalalov, Humaira Nisar, **Yasir Salih**, and Aamir S. Malik, “An Algorithm for Vehicle Detection and Tracking”, *2010 International*

Conference on Intelligent and Advanced Systems (ICIAS 2010), Kuala Lumpur, Malaysia, 15-17 June 2010.

6.6 **Recommendations**

Further improvements can be added to the proposed system in order to enhance its reliability and expand its usage to new applications.

- Improving the object detection component by using background model algorithm that can work with both static and dynamic backgrounds. This can be done by using parallel processing where the background is modeled frequently.
- Using more sophisticated shadow elimination algorithm that can handle all types of shadows and highlights.
- Using more accurate occlusion compensation algorithm that uses the whole history of the object motion rather than only using previous frame data.
- Expanding the triangulation algorithm so that it can compute the depth of objects that float above the ground level and non-flat surfaces.

REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking," *ACM Computing Surveys*, vol. 38, no. 4, pp. 13-58, Dec. 2006.

[2] The Dialy Mail new paper, Available at: <http://www.dailymail.co.uk/news/article-444819/UK-1-worlds-population-20-CCTV-cameras.html>, 2007 (Last visit 11/06/ 2011).

[3] Y. Salih and A. S. Malik, "Comparison of Stochastic Filtering Methods for 3D Tracking," *Pattern Recognition*, vol. 44, no. 10-11, pp. 2711-2737, Apr. 2011.

[4] W. Hu, T. Tan, L. Wang, and S. Maybank, "A Survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems Man and Cybernetics, Part C: Applications and Reviews*, vol. 34, no. 3, pp. 334-352, 2004.

[5] H. M. Dee and S. a Velastin, "How close are we to solving the problem of automated visual surveillance?," *Machine Vision and Applications*, vol. 19, no. 5-6, pp. 329-343, May. 2007.

[6] S. Mitra and T. Acharya, "Gesture recognition : A survey," *IEEE Transactions on Systems Man and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 3, pp. 311-324, 2007.

[7] J. Madapura and B. Li, "3D articulated human body tracking using KLD-annealed Roa-blackwellized filter," in *IEEE International Conference on Multimedia and Expo*, 2007, pp. 1950-1953.

[8] X. Xu and B. Li, "Exploiting motion correlations in 3-D articulated human motion tracking," *IEEE Transactions on Image Processing*, vol. 18, no. 6, pp. 1292-1303, 2009.

[9] J. Ziegler, K. Nickel, and R. Stiefelhagen, “Tracking of the articulated upper body on multi-view stereo image sequences,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006, pp. 774-781.

[10] W. Hu, X. Xiao, D. Xie, and T. Tan, “Traffic accident prediction using vehicle tracking and trajectory analysis,” in *International Conference on Intelligent Transportation Systems*, 2003, pp. 220-225.

[11] K. Kiratiratanapruk and S. Siddhichai, “Vehicle detection and tracking for traffic monitoring system,” in *TENCON*, 2006, pp. 1-4.

[12] M. Meuter, A. Kummert, and S. Muller-Schneiders, “3D traffic sign tracking using a particle filter,” in *11th International IEEE Conference on Intelligent Transportation Systems*, 2008, no. 1, pp. 168-173.

[13] J. Batista, P. Peixoto, C. Fernandes, and M. Ribeiro, “A dual-stage robust vehicle detection and tracking for real-time traffic monitoring,” in *IEEE Intelligent Transportation Systems Conference*, 2006, pp. 528-535.

[14] J. C. Lee, S. E. Hudson, J. W. Summet, and P. H. Dietz, “Moveable interactive projected displays using projector based tracking,” in *18th annual ACM Symposium on User Interface Software and Technology - UIST ’05*, 2005, pp. 63-72.

[15] X. Gao, T. E. Boult, F. Coetzee, and V. Ramesh, “Error analysis of background adaption,” in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2000, pp. 503-510.

[16] Y. Salih and A. S. Malik, “3D Object Tracking Using Three Kalman Filters,” in *IEEE Symposium of Computer and Informatics*, 2011, pp. 501-505.

[17] Y. Salih and A. S. Malik, “3D Tracking Using Particle Filters,” in *IEEE International Instrumentation and Measurement Technology Conference*, 2011, pp. 1-5.

[18] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proceedings. International Conference on Computer Vision and Pattern Recognition*, 1999, pp. 246-252.

[19] Y. Satoh, H. Tanahashi, S. Kaneko, Y. Niwa, and K. Yamamoto, “Robust event detection by radial reach filter (RRF),” in *the 16th International Conference on Pattern Recognition*, 2002, vol. 0, no. 1, pp. 623-626.

[20] Y. Satoh and K. Sakaue, “Robust Background Subtraction based on Bi-polar Radial Reach Correlation,” *TENCON 2005 - 2005 IEEE Region 10 Conference*, pp. 1-6, Nov. 2005.

[21] X. Zhao, Y. Satoh, H. Takauji, S. Kaneko, K. Iwata, and R. Ozaki, “Object detection based on a robust and accurate statistical multi-point-pair model,” *Pattern Recognition*, vol. 44, no. 6, pp. 1296-1311, Jun. 2011.

[22] Y. Sheikh and M. Shah, “Bayesian modeling of dynamic scenes for object detection.,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1778-92, Nov. 2005.

[23] M. Seki, T. Wada, H. Fujiwara, and K. Sumi, “Background subtraction based on cooccurrence of image variations,” *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, p. II-65-II-72, 2003.

[24] L.-Q. Xu, J. L. Kandabaso, and M. Pardas, “Shadow removal with blob-based morphological reconstruction for error correction,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2005.

[25] A. Branca, G. Attolico, and A. Distante, “Cast shadow removal in foreground segmentation,” in *International Conference on Pattern Recognition 16 (1)*, 2002, pp. 214-217.

[26] R. Cucchiara, C. Grana, M. Piccardi, and a Prati, “Detecting moving objects, ghosts, and shadows in video streams,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1337-1342, Oct. 2003.

[27] Y. Salih, A. S. Malik, and S. M. Ieee, “Stochasitc filters for object tracking,” in *The 15th IEEE International Symposium on Consumer Electronics*, 2011, pp. 1-5.

[28] C. Paramanand and A. N. Rajagopalan, “Unscented Transformation for Depth from Motion-Blur in Videos,” *Image Processing*, pp. 38-44, 2010.

[29] D. Scharstein, R. Szeliski, and R. Zabih, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” in *IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV)*, 2001, pp. 131-140.

[30] D.-jye L. Paul, M. Zhaoyi, and B. E. Nelson, “Two-frame structure from motion using optical flow probability distributions for unmanned air vehicle obstacle avoidance,” *Machine Vision and Applications*, vol. 21, no. 3, pp. 229-240, 2010.

[31] A. S. Malik and T. S. Choi, “A novel algorithm for estimation of depth map using image focus for 3D shape recovery in the presence of noise,” *Pattern Recognition*, vol. 41, no. 7, pp. 2200-2225, 2008.

[32] Y. N. Shree K. Nayar, “Shape from Focus,” *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 16, no. 8, pp. 824-831, 1994.

[33] W. L. Hwang, C. S. Lu, and P. C. Chung, “Shape from texture: estimation of planar surface orientation through the ridge surfaces of continuous wavelet transform.,” *IEEE Transactions on Image Processing*, vol. 7, no. 5, pp. 773-80, Jan. 1998.

[34] R. White and D. a Forsyth, “Combining Cues: Shape from Shading and Texture,” in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR’06)*, 2006, no. 4, pp. 1809-1816.

[35] A. Criminisi, I. Reid, A. Zisserman, and P. Francesca, “Single View Metrology,” *International Journal of Computer Vision*, vol. 40, no. 2, pp. 123-148, 2000.

[36] D. Rother, K. a Patwardhan, and G. Sapiro, “What Can Casual Walkers Tell Us About A 3D Scene?,” in *2007 IEEE 11th International Conference on Computer Vision*, 2007, pp. 1-8.

[37] O. Barinova, V. Konushin, A. Yakubenko, K. Lee, H. Lim, and A. Konushin, “Fast automatic single-view 3-d reconstruction of urban scenes,” in *European Conference on Computer Vision*, 2008, pp. 1-14.

[38] D. Hoiem and A. A. Efros, “Geometric Context from a Single Image,” in *IEEE International Conference on Computer Vision*, 2005, pp. 654-661.

[39] Y. Lila, C. Lursinsap, and R. Lipikorn, “3D shape recovery from single image by using texture information,” in *International Conference on Control, Automation and Systems*, 2008, no. x, pp. 2801-2806.

[40] A. Torralba and A. Oliva, “Depth estimation from image structure,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 1226-1238, Sep. 2002.

[41] J.-il Jung and Y.-sung Ho, “Depth map estimation from single-view image using object classification based on Bayesian learning,” in *3D TV Conference: The Ture Vision - Capture, Transmission and Display on 3D Video*, 2010, pp. 1-4.

[42] D. Hoiem, A. a Efros, and M. Hebert, “Automatic photo pop-up,” *ACM Transactions on Graphics*, vol. 24, no. 3, p. 577, Jul. 2005.

[43] A. Saxena, S. H. Chung, and A. Ng, “3-D Depth Reconstruction from a Single Still Image,” *International Journal on Computer Vision*, no. 76, pp. 53-69, 2008.

[44] A. Saxena, M. Sun, and A. Y. Ng, "Make3D: learning 3D scene structure from a single still image.," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 824-40, May. 2009.

[45] A. Das, V. Ramnani, J. Bhavsar, and S. K. Mitra, "Improvised Filter Design for Depth Estimation from Single Monocular Images," in *3d International Conference on Pattern Recognition and Machine Intelligence*, 2009, pp. 333-338.

[46] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1253-1260.

[47] Z. Chen, "Bayesian filtering: From Kalman filters to particle filters, and beyond," *Statistics*, pp. 1-69, 2003.

[48] G. Welch and G. Bishop, *An introduction to the Kalman filter*, 1st ed. University of North Carolina at Chapel Hill, 1995, pp. 1-80.

[49] J.-S. Hu, T.-M. Su, C.-W. Juan, and G. Wang, "3D object tracking using mean-shift and similarity-based aspect-graph modeling," in *33rd Annual Conference of the IEEE Industrial Electronics Society*, 2007, pp. 2383-2388.

[50] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000, no. 7, pp. 142-149.

[51] T. Gang, H. U. Rui-min, W. Zhong-yuan, and Z. H. U. Li, "Object tracking algorithm based on meanshift algorithm combining with motion vector analysis," in *1st International Workshop on Education Technology and Computer Science*, 2009, pp. 987 - 990.

[52] L. Zhang and H. Zhao, "Real time mean shift tracking using the Gabor wavelet," in *IEEE International Conference on Mechatronics and Automation*, 2007, pp. 1617-1621.

[53] M. Boonsin, W. Wettayaprasit, and L. Preechaveerakul, “Improving of mean shift tracking algorithm using adaptive candidate model,” in *I International Conference on Electrical Engineering, Electronics Computer Telecommunications and Information Technology (ECTICON)*, 2010, pp. 894 - 898.

[54] Z. H. E. Chen, *Bayesian Filtering : From Kalman Filters to Particle Filters, and Beyond*. 2003, pp. 9-13, 25-46.

[55] H. Zhou, Y. Yuan, and C. Shi, “Object tracking using SIFT features and mean shift,” *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345-352, Mar. 2009.

[56] S. Weng, C. Kuo, and S. Tu, “Video object tracking using adaptive Kalman filter,” *Journal of Visual Communication and Image Representation*, vol. 17, no. 6, pp. 1190-1208, Dec. 2006.

[57] P. R. Devarakota, M. Castillo-Franco, R. Ginhoux, B. Mirbach, S. Kater, and B. Ottersten, “3-D-skeleton-based head detection and tracking using range images,” *IEEE Transactions on Vehicular Technology*, vol. 58, no. 8, pp. 4064-4077, Oct. 2009.

[58] Y. Du and F. Yuan, “Real-time vehicle tracking by Kalman filtering and Gabor decomposition,” in *1st International Conference on Information Science and Engineering*, 2009, no. c, pp. 1386-1390.

[59] I. Fernández, M. Mazo, J. L. Lázaro, D. Pizarro, E. Santiso, and P. Martín, “Guidance of a mobile robot using an array of static cameras located in the environment,” *Autonomous Robots*, vol. 23, no. 4, pp. 305-324, 2007.

[60] A. Suppes, F. Suhling, M. Hötter, F. Hannover, and R. Stadtweg, “Robust obstacle detection from stereoscopic image sequences using Kalman filtering,” in *23rd DAGM-Symposium on Pattern Recognition*, 2001, pp. 385-391.

[61] Z. Jia, A. Balasuriya, and S. Challa, "Sensor fusion-based visual target tracking for autonomous vehicles," *Artificial Life and Robotics*, vol. 12, no. 2, pp. 317-328, 2008.

[62] Q. Zhou and J. K. Aggarwal, "Object tracking in an outdoor environment using fusion of features and cameras," *Image and Vision Computing*, vol. 24, no. 11, pp. 1244-1255, 2006.

[63] F. Ababsa, "Robust extended Kalman filtering for camera pose tracking using 2D to 3D lines correspondences," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2009, pp. 1834-1838.

[64] J. Gao, A. Kosaka, and A. C. Kak, "A multi-Kalman filtering approach for video tracking of human-delineated objects in cluttered environments," *Computer Vision and Image Understanding*, vol. 99, no. 1, pp. 1-57.

[65] V. Lippiello, B. Siciliano, and L. V. Ã, "Adaptive extended Kalman filtering for visual motion estimation of 3D objects," in *Control Engineering Practice*, 2007, vol. 15, pp. 123-134.

[66] M. C. Vandyke, J. L. Schwartz, and C. D. Hall, "Unscented Kalman filtering for spacecraft attitude state and parameter estimation," *Virginia State Polytechnic Institut*e. pp. 1-13, 2004.

[67] G. A. Terejanu, "Unscented Kalman filter tutorial," *Workshop on Large-Scale quantification of Uncertainty, Sandia National Laboratories*, no. 1, Sandia National Laboratories, pp. 1-6, 2009.

[68] P. Janis, "Unscented Kalman filter," *Postgraduate Seminar on Signal Processing, Aalto University*, pp. 1-20, 2006.

[69] A. Causo, E. Ueda, Y. Kurita, Y. Matsumoto, and T. Ogasawara, "Model-based hand pose estimation using multiple viewpoint silhouette images and unscented Kalman filter," in *17th IEEE International Symposium on Robot and Human Interactive Communication*, 2008, pp. 291-296.

[70] C. Tsai, K. Song, X. Dutoit, H. V. Brussel, and M. Nuttin, “Robust mobile robot visual tracking control system using self-tuning Kalman filter,” in *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 2007, pp. 161-166.

[71] D. Ponsa, A. Lopez, J. Serrat, F. Lumbreras, and T. Graf, “Multiple vehicle 3D tracking using an unscented Kalman filter,” in *IEEE International Conference on Intelligent Transportation Systems*, 2005, pp. 1108-1113.

[72] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking,” *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174-188, 2002.

[73] A. Doucet, S. Godsill, and C. Andrieu, “On sequential Monte Carlo sampling methods for Bayesian filtering,” *Statistics and Computing*, vol. 10, no. 3, pp. 197-208, 2000.

[74] L. Brèthes, F. Lerasle, P. Danès, and M. Fontmarty, “Particle filtering strategies for data fusion dedicated to visual tracking from a mobile robot,” *Machine Vision and Applications*, vol. 21, no. 4, pp. 427-448, Oct. 2008.

[75] M. Taiana, J. Santos, J. Gaspar, J. Nascimento, A. Bernardino, and P. Lima, “Tracking objects with generic calibrated sensors: An algorithm based on color and 3D shape features,” *Robotics and Autonomous Systems*, vol. 58, no. 6, pp. 784-795, Jun. 2010.

[76] S. Kim, C. Park, and S. Lee, “Tracking 3D human body using particle filter in moving monocular camera,” in *18th International Conference on Pattern Recognition*, 2006, pp. 805-808.

[77] G. Catalin and S. Nedevschi, “Object tracking from stereo sequences using particle filter,” in *4th International Conference on Intelligent Computer Communication and Processing*, 2008, pp. 279-282.

[78] Y. Lao, J. Zhu, and Y. F. Zheng, “Sequential particle generation for visual tracking,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 9, pp. 1365-1378, Sep. 2009.

[79] S. Ongkittikul, S. Worrall, and A. Kondoz, “Enhanced hand tracking using the k-means embedded particle filter with mean-shift vector re-sampling,” in *5th International Conference on Visual Information Engineering VIE*, 2008, pp. 23-28.

[80] C. Shan, T. Tan, and Y. Wei, “Real-time hand tracking using a mean shift embedded particle filter,” *Pattern Recognition*, vol. 40, no. 7, pp. 1958-1970, Jul. 2007.

[81] Z. Feng, B. Yang, Y. Zheng, Z. Wang, and Y. Li, “Research on 3D hand tracking using particle filtering,” in *4th International Conference on Natural Computation*, 2008, pp. 367-371.

[82] M. Bray, E. Koller-Meier, and L. V. Gool, “Smart particle filtering for high-dimensional tracking,” *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 116-129, 2007.

[83] M. Pupilli and A. Calway, “Real-time camera tracking using known 3D models and a particle filter,” in *International Conference on Pattern Recognition*, 2006, pp. 199-203.

[84] W. Zheng and S. M. Bhandarkar, “Face detection and tracking using a Boosted Adaptive Particle Filter,” *Journal of Visual Communication and Image Representation*, vol. 20, no. 1, pp. 9-27, 2009.

[85] S. J. Mckenna and H. Nait-Charif, “Tracking human motion using auxiliary particle filters and iterated likelihood weighting,” *Image and Vision Computing*, vol. 25, no. 6, pp. 852-862, Jun. 2007.

[86] P. Peursum, S. Venkatesh, and G. West, “A study on smoothing for particle-filtered 3D human body tracking,” *International Journal on Computer Vision*, vol. 87, no. 1-2, pp. 53-74, 2010.

[87] M. Isard and A. Blake, “CONDENSATION — Conditional Density Propagation for Visual Tracking,” *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5-28, 1998.

[88] T. Bando, T. Shibata, K. Doya, and S. Ishii, “Switching particle filters for efficient visual tracking,” *Robotics and Autonomous Systems*, vol. 54, no. 10, pp. 873-884, 2006.

[89] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed. Pearson Education Ltd, 2010, pp. 376 - 377.

[90] R. C. Gonzalez, R. E. Woods, and Steven L. Eddins, *Digital Image Processing using MATLAB*, 2nd ed. Gatesmark Publishing, 2010, pp. 211 - 250.

[91] L. Brèthes, F. Lerasle, P. Danès, and M. Fontmarty, “Particle filtering strategies for data fusion dedicated to visual tracking from a mobile robot,” *Machine Vision and Applications*, vol. 21, no. 4, pp. 427-448, 2010.

[92] C. R. Del-Blanco, R. Mohedano, N. Garcia, L. Salagado, and F. Jaureguizar, “Color based 3D particle filtering for robust tracking in heterogeneous environment,” in *2nd ACM/IEEE International Conference on Distributed Smart Cameras*, 2008, pp. 1-10.

[93] Samsung Techwin, “SDZ-375 high resolution 37X zoom color camera user guide,” *Samsung Inc*, 2008. [Online]. Available: http://www.samsungtechwin.com/product/file_data/manual/20091026_0_091016_SDZ-375_Eng.pdf. [Accessed: 28-Dec-2010].

[94] R. Fisher, J. Santos-Victor, and J. Crowley, “CAVIAR datasets,” *EC Funded CAVIAR project*, 2001. [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/. [Accessed: 28-Dec-2010].

[95] PETS datasets," *Computational Vision Group, University of Reading*, available at: http://www.cvg.cs.rdg.ac.uk/, Last visit (05/01/2011).