OBJECT TRACKING SYSTEM WITH SEAMLESS OBJECT HANDOVER BETWEEN STATIONARY AND MOVING CAMERA MODES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

DENİZ EMEKSİZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

NOVEMBER 2012

OBJECT TRACKING SYSTEM WITH SEAMLESS OBJECT HANDOVER BETWEEN STATIONARY AND MOVING CAMERA MODES

Submitted by **DENİZ EMEKSİZ** in partial fulfillment of the requirements for the degree of **Master of Science in Information Systems, Middle East Technical University** by,

Prof. Dr. Nazife Baykal ____________________
Director, **Informatics Institute**

Prof. Dr. Yasemin Yardımcı Çetin ____________________
Head of Department, **Information Systems**

Assist. Prof. Dr. Alptekin Temizel ____________________
Supervisor, **Work Based Learning, METU**

Examining Committee Members:

Prof. Dr. Yasemin Yardımcı Çetin ____________________
Information Systems, METU

Assist. Prof. Dr. Alptekin Temizel ____________________
Work Based Learning, METU

Prof. Dr. Gözde Bozdağı Akar ____________________
Electrics and Electronics Engineering, METU

Assist. Prof. Dr. Banu Günel ____________________
Information Systems, METU

Assoc. Prof. Dr. Erkan Mumcuoğlu ____________________
Information Systems, METU

Date: 20.11.2012

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Deniz Emeksiz

Signature : ____________

ABSTRACT

OBJECT TRACKING SYSTEM WITH SEAMLESS OBJECT HANDOVER BETWEEN STATIONARY AND MOVING CAMERA MODES

Emeksiz, Deniz

M. Sc. , Department of Information Systems

Supervisor: Assist. Prof. Dr. Alptekin Temizel

November 2012, 56 Pages

As the number of surveillance cameras and mobile platforms with cameras increases, automated detection and tracking of objects on these systems gain importance. There are various tracking methods designed for stationary or moving cameras. For stationary cameras, correspondence based tracking methods along with background subtraction have various advantages such as enabling detection of object entry and exit in a scene. They also provide robust tracking when the camera is static. However, they fail when the camera is moving. Conversely, histogram based methods such as mean shift enables object tracking on moving camera cases.

Though, with mean shift object’s entry and exit cannot be detected automatically which means a new object’s manual initialization is required.

In this thesis, we propose a dual-mode object tracking system which combines the benefits of correspondence based tracking and mean shift tracking. For each frame, a reliability measure based on background update rate is calculated. Interquartile Range is used for finding outliers on this measure and camera movement is detected. If the camera is stationary, correspondence based tracking is used and when camera is moving, the system switches to the mean shift tracking mode until the reliability of correspondence based tracking is sufficient according to the reliability measure.

The results demonstrate that, in stationary camera mode, new objects can be detected automatically by correspondence based tracking along with background subtraction. When the camera starts to move, generation of false objects by correspondence based tracking is prevented by switching to mean shift tracking mode and handing over the correct bounding boxes with a seamless operation which enables continuous tracking.

Keywords: Object Tracking, Moving Camera, Mean Shift, Correspondence Based Tracking, Background Subtraction

ÖZ

DURAĞAN VE HAREKETLİ KAMERA KİPLERİNDE KESİNTİSİZ NESNE DEVİRLİ NESNE TAKİP SİSTEMİ

Emeksiz, Deniz

Yüksek Lisans, Bilişim Sistemleri Bölümü

Tez Yöneticisi: Yrd. Doç. Dr. Alptekin Temizel

Kasım 2012, 56 Sayfa

Güvenlik kameraları ve kamera içeren hareketli platformların sayıları arttıkça, bu sistemlerde otomatik nesne tespit ve takibinin önemi de artmaktadır. Sabit ve hareketli kameralar için birçok takip sistemi tasarlanmıştır. Sabit kameralar için, arka plan çıkarımıyla beraber eşleştirmeye dayalı takip yöntemlerinin bir nesnenin bir sahneye giriş ve çıkışını tespit etmek gibi avantajları vardır. Bu sistemler kamera sabitken nesneleri gürbüz biçimde takip etmeye de olanak sağlarken kamera hareket etmeye başladığında nesne takibinde başarısızdırlar. Diğer taraftan, ortalama kayma gibi histograma dayalı yöntemler hareketli kamerada nesne takibine olanak sağlar. Fakat, ortalama kayma ile nesnenin giriş ve çıkışları otomatik olarak tespit edilemez ve bu nesnenin elle ilklendirilmesi gerektiği anlamına gelir.

Bu tezde, eşleştirmeye dayalı takip yöntemi ile ortalama kaymayı birleştiren çift kipli nesne takip sistemi önerilmektedir. Her bir çerçeve için arka plan güncelleme oranına dayalı bir güvenilirlik ölçüsü hesaplanır. Orta çeyrekler aralığı bu ölçü üzerindeki aykırı elemanları bulmak için ve kamera hareketini tespit etmek için kullanılır. Eğer kamera sabitse, eşleştirmeye dayalı takip yöntemi kullanılır, kamera hareketliyken sistem ortalama kayma moduna geçip eşleştirmeye dayalı takip yönteminin güvenilirliği güvenilirlik ölçüsüne gore yeterli olana kadar devam eder.

Sonuçlara göre, durağan kamera kipinde yeni nesneler arka plan çıkarımıyla beraber eşleştirmeye dayalı takip yöntemiyle tespit edilebilmektedir. Kamera hareket ettiğinde eşleştirmeye dayalı takip yönteminden yanlış karakter kutularının oluşturulması ortalama kayma moduna geçip doğru karakter kutuları kesintisiz bir işlemle devredilir ve sürekli takip sağlanır.

Anahtar Kelimeler: Nesne Takibi, Hareketli Kamera, Ortalama Kayma, Eşleştirme Tabanlı Takip, Arka Plan Çıkarımı

To my friends and family…

ACKNOWLEDGMENTS

I express my sincere thanks to my supervisor Assist. Prof. Dr. Alptekin Temizel for his guidance, support and assistance throughout my thesis process. I am also heartily thankful to my friend and co-worker Püren Güler for her encouragement, motivation and help in capturing videos.

I am also thankful to Çağlar Çetintaş and Mustafa Teke for their support and help in capturing the videos.

I would like to thank my friends Ömer Özgün Işıkman, Özkan Kapçak and Mina Nabii for their support and understanding throughout my thesis.

Finally, I would like to thank my family Ayşe Nilgün Emeksiz and Vedat Emeksiz for their love, encouragement and faithfulness.

This research was supported by Turkish Ministry of Science, Industry and Technology SAN-TEZ project number 00542.STZ.2010-1.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Object tracking is a field that is widely studied and many algorithms have been proposed in the literature. Tracking systems are generally either designed with stationary camera assumptions (such as systems using background subtraction) or designed to work in moving camera scenarios as well (such as mean-shift or particle filter tracking based systems).

Correspondence Based Tracking (CBT) is a method that is based on background subtraction. It uses foreground information that is extracted from the background model and matches the objects in consecutive frames using object features such as size, color and distance. When the camera is stationary, object correspondence based tracking using background subtraction has a number of benefits such as enabling automatic detection of new objects in the scene and high tracking accuracy. However these algorithms fail when the camera is moving or the background is affected by sudden light changes.

Mean Shift tracking (MS) is a robust method that matches the objects using their histogram information. Once the object initiation is done, it uses the histogram information of the kernel and hence it is able to track objects while the camera is moving. But, there are some drawbacks of mean shift. Firstly, it does not provide any mechanism for object birth and death and hence require manual selection of object to be tracked. Second drawback is tracking drift where the tracking kernel shifts to the background objects, as the tracking kernel includes some of background information as well as the tracked object.

In this thesis, our motivation is to implement a system that starts tracking when the camera is stationary and maintains tracking in the situations when the camera starts moving. Camera angle in security cameras in airports, underground stations etc. are limited and when the tracked object goes out of scene, tracking is lost. However, with the help of PTZ cameras, it is possible to continue tracking the moving object. Also, mobile robots or platforms can be used to track people for security purposes such as military applications. By combining the proposed system with these types of platforms, we can obtain continuous tracking of moving objects.

Our aim is to follow a moving object that comes into the view of a camera in a stopped car or autonomous system and continue tracking with a seamless manner when car or system is moving. Our purpose is to propose a framework that combines the advantages of two tracking methods and we do not aim to improve the existing tracking methods. In this thesis it is assumed that the camera does not provide any information whether it is stationary or moving.

- Before tracking, camera should be stationary during background learning and outlier detection. After finishing these operations, for at least one frame, camera should not move to obtain moving objects for the first time.
- New objects can be detected only when the camera is not moving.
- The object can move freely in front of the camera for tracking. However, the tracking might fail if;
    - the moving object gets too close to camera to obstruct the camera view,
    - the moving object gets too far away the camera and becomes too small to track.
- Sudden light changes or new objects entering the scene might adversely affect the tracking performance.
- In crowded areas, multiple people can be tracked as one person.

The camera is assumed to be on a moving platform, such as a land vehicle controlled by a driver, and we aim to track the object continuously whether the platform is stationary or moving. Test videos include two different camera movements. First movement type is panning the camera to follow the object. Second type is following the moving object from behind as it is going forward.

For this purpose, we propose a hybrid system using different algorithms in stationary and moving camera modes to benefit from their advantages as the algorithm providing better performance differs in these modes. Under normal operating conditions, when the background estimation is working reliably, correspondence based tracking is used. In this mode, object detection is done automatically and the objects are tracked with high accuracy using the foreground information. When the reliability of background estimation decreases, due to factors such as camera starts moving, the system automatically switches to mean shift tracking until the reliability of background information increases again. The system hands over the objects in a seamless manner to provide continuous tracking during mode change.

In Chapter 2, a literature survey is presented, in Chapter 3 the methods and algorithms used in this thesis are explained, in Chapter 4 the results and benchmarks are presented, in the concluding remarks are given in the final chapter.

CHAPTER 2

LITERATURE REVIEW

In this chapter, studies related with this thesis are reviewed. First, a literature review on object tracking techniques is provided. Then, tracking systems which work on moving cameras are analyzed. Finally, popular background subtraction techniques ın the literature are presented.

2.1 Object Tracking

According to [1], the purpose of object tracking is to produce a path of a tracked object from the location of it in each video frame. In object tracking systems, objects that come into view of the camera need to be tracked continuously. Object tracking is performed on the frames obtained from different type of cameras such as video and thermal cameras or other modalities such as radars. Various tracking methods have been developed to perform tracking on those systems. Our aim is to perform tracking on visible band video cameras that gives RGB images as output. There are various complications that cause object tracking to fail. In [1], some issues that cause object tracking to fail are listed including noise, complex nature of objects, occlusions, lighting changes, information loss and constraints that the tracking needs to work real-time. Some object tracking systems are designed to work on only stationary cameras. So, moving camera can be considered as another complication. In [1], different object tracking techniques that can handle these conditions are analyzed.

In [1], object tracking is divided into three main categories: point tracking, kernel tracking and silhouette tracking.

In point tracking, moving objects are represented with points. Using these points, object matching with other frames can be performed taking object location and motion parameters into account.

Kalman filters [2] can be used as a point tracker. In Kalman filters, object state, which is represented with a Gaussian, is estimated. This method consists of two parts; prediction phase followed by the correction phase. In the prediction phase, the new states are calculated with the information of the state model. In the correction phase, object's state is updated with the results of prediction phase. As the variables in the state model have a normal distribution, for the variables that do not have a normal distribution, Kalman filtering may not give accurate results. In particle filter tracking, a state density function is defined. This function is denoted with particles. Particles are a sample set of the density function. First, a random group of samples are selected, then a corresponding new sample set is generated. By using new samples, positions of the objects are estimated [3].

In kernel tracking, tracking is performed using a kernel with a predefined shape containing each tracked object. According to [1], kernel tracking methods are simple and have low computation cost. They are based on generating a template and searching the position of a similar template in other frames. One of the kernel tracking techniques is template matching. An object tracking system is proposed in [4] which calculates the mean of the pixels' colors in tracked object's kernel and searches the similar kernel in 8-neighborhood. A similar algorithm by using an elliptical kernel and utilized a weighted histogram instead of mean color is proposed in [5]. For searching the template in other frames, they use a method named mean shift which finds the similarity between template and target objects based on Bhattacharya coefficient.

In [6], a correspondence based object tracking algorithm by feature matching is proposed. In this study, first background is estimated with background subtraction, and then moving objects that enter the scene are detected. Objects in consecutive frames are matched using the size, centroid and histogram values of their bounding boxes. This algorithm is very effective of detecting objects and matching between consecutive frames but fails when the camera moves as it is dependent on background subtraction information. In [7], a segmentation based object matching algorithm is presented by calculating Mahalanobis distance between frames. In [8], a correspondence based algorithm by using Gaussian Mixture Model to extract objects and match them with their different features such as size, shape, direction and change of location is proposed. Because of using a template which has a fixed shape, changes in object size might lead the algorithm to fail. In [9], a correspondence based algorithm is fused with motion estimation for object tracking. In this study, object parts can be tracked along with the main body. In [10], a similar method to [6] is combined with mean shift by setting the main tracking method as mean shift because of its robustness and low computational complexity. In [10], correspondence based tracking is used for detecting new objects and handling occlusion. Also, to overcome the problem of tracking kernel shifting to the background in mean shift, correspondence based tracking is used to redetect the objects and match with former objects every 25 frames.

Silhouette tracking methods track the objects using the information of the shape of the object area. As it is said in [1], silhouette tracking is usually performed by matching object shapes or contour growth. In [11], a tracking method using support vector machines (SVM) is presented. According to this method, two training sets are used. In one set, images of the objects that are going to be tracked are kept. In other set, the components that are not wanted to be tracked are kept. Then a classification method is used to give scores to the image regions to find moving objects. A limitation of this method is that it uses a training set which prevent the system to work real time and automated object initialization.

There are various studies which propose to provide object tracking or motion estimation on moving camera.

In [12], a segmentation based shape estimation method for moving cameras is proposed. According to [12], first camera motion is detected, then change detection is applied. In the end, background pixels which do not have foreground pixels are estimated. In [13], a system that performs motion tracking with moving cameras with combining segmentation of color and region merging is proposed. A system that detects and performs tracking in a camera which has a slidable angle is proposed in

[14]. In [15] a system which can match actions between frames in the case of camera moving based on calculating epipolar geometry of objects' two different views is proposed. In this study, the transition should be short and smooth. In [16], an object detection method in a moving camera by using an approach based on Markov random field (MRF) models is proposed. In [17], authors used an image mapping and motion estimation technique to perform object tracking with pan/tilt camera. An object tracking algorithm with a moving camera by using a probability matrix to find the change of object location is proposed in [18]. In [19], authors propose an object tracking system with a moving camera that films from air. In this method, they detect objects by segmentation and track them with a histogram method.

2.2 Background Subtraction

According to [1], background subtraction is important, because when the background is learned, a change in the background is a moving object candidate. So, choosing a good background subtraction method is important to obtain these moving object candidates.

Different background subtraction techniques are analyzed in [20]. A background subtraction method in which each pixel has its own background model is proposed in [21]. Background models are represented with Gaussian probability density function (pdf). They use a running average technique shown in Equation (2.1) to calculate the pdf.

$$\mu_t = \alpha I_t + (1 - \alpha)\mu_{t-1} \tag{2.1}$$

where I_t is pixel's current value, μ_t is the previously calculated running average, α is a constant which can be set to give weight to either stability or update rate. According to [20], a pixel's value is determined if it belongs to foreground or background with the Equation (2.2). If this inequality is true, the pixel is said to belong to foreground.

$$|I_t - \mu_t| > k\sigma_t \tag{2.2}$$

An improved method which uses median filter is proposed in [22]. In [22], instead of getting background model of each pixel, they propose to get the median of set of values. This set includes last n subsampled values and the last computed median value. However, in these methods that are based on median, median values need to be kept in memory and also median do not provide a correct background model for background estimation [20].

In [23], a simple background subtraction method is used based on frame differencing. Where $I_n(x)$ is the pixel in current frame, $I_{n-1}(x)$ is the pixel in previous frame and $I_{n-2}(x)$ is the pixel in the frame that is 2 frames prior, equation given in Equation (2.3) is used

$$\begin{aligned} |I_n(x,y) - I_{n-1}(x,y)| \; &> \; T_n(x,y) \\ |I_n(x,y) - I_{n-2}(x,y)| \; &> \; T_n(x,y) \end{aligned} \tag{2.3}$$

If these two subtractions are higher than threshold T_n, a pixel said to belong to foreground.

Codebook [24] is a background subtraction method in which a set of vectors named codebooks is set for each pixel. Change in background is observed with these vectors in the codebook.

In [25], the probability of a pixel value x at time t is described with the Equation (2.4).

$$P(x_t) = \sum_{i=1}^{K} \omega_{i,t} \eta(x_t - \mu_{i,t}, \Sigma_{i,t}) \tag{2.4}$$

where $\omega_{i,t}$ represents the fraction of data is used for i^{th} Gaussian, η denotes Gaussian probability density function, $\mu_{i,t}$ is mean of i^{th} Gaussian, $\Sigma_{i,t}$ is the covariance matrix. Here, all K distributions define one foreground or background object. To determine where the pixel belongs, the pdfs are ordered along with their ω_i / σ_i. The higher result means that it has a high probability to belong to background. After that, Equation (2.5) is used.

$$x_t - \frac{\mu_{i,t}}{\sigma_{i,t}} > 2.5 \tag{2.5}$$

In this order, the first one gets to be x_t's match.

In [26] authors used a model based on Kernel Density Estimation by looking at a buffer of *n* recent background pixels. By smoothing histograms of each frame, a continuous histogram is obtained. Background probability density function is created as a sum of Gaussian kernels. At a given time *t*, probability of detecting a pixel is given with Equation (2.6).

$$P(x_t) = \frac{1}{n}\sum_{i=1}^{n} \eta(x_t - x_i, \Sigma_\mathrm{t}) \tag{2.6}$$

where, $\sum_t$ is covariance matrix, η is probability density function, If probability density function of *x* in time *t* is greater than a threshold, the pixel is said to belong to background.

Even though this method has a high accuracy rate memory cost is high because of using two models at the same time.

The method proposed in [27] is also a fast background subtraction method. In [27] instead of building a probability density function, for pixel model, sample values are obtained. Then, a sphere is defined whose center is the current pixel and these values are compared with the values that are present in this sphere. Then, a cardinality function is defined to find the pixel belonging to background. If result of this function is greater than a threshold, a pixel is said to belong to background. Just like GMM [25] , Vibe [27] is a fast and computationally not complex method. But, because of not replacing new samples with the old ones, Vibe is strongly attached to the first initialized frame. If there is a moving object in the first frame, a “ghost” is generated and stays very long in the foreground image.

The background subtraction method proposed in [23] is fast and easy to implement, but it is not efficient as [25] or [26] which and more prone to generate inaccurate results and detect background objects inaccurately as foreground.

In [28], an improved Gaussian Mixture Model (IGMM) is proposed. This method is similar to GMM method but there are improvements. In [28], unlike [25], component numbers and parameters are selected automatically.

In this thesis, as a background subtraction method improved GMM method IGMM proposed in [28] is chosen to be implemented because of its low computational complexity, success in background estimation and adapting to changes quickly. The implementation details of IGMM are explained in detail in Section 2.2.

CHAPTER 3

METHODOLOGY

3.1 Overview

Correspondence Based Tracking (CBT) is an efficient and fast tracking method when stationary camera assumption can be made. It mainly uses features of detected foreground objects and aims to match these objects with detected foreground objects in consecutive frames. So, for obtaining foreground objects, first the background should be estimated. Background subtraction methods learn the background information from a video sequence and using the background extracts the foreground (moving) objects. Most object tracking techniques require manual initiation of objects by selecting an area prior to tracking. Background subtraction eliminates this requirement as objects can be detected by segmentation of the foreground. These benefits make using background subtraction with correspondence based tracking an effective method for detecting and tracking moving objects. However, these techniques assume that the camera is stationary and if the camera starts to move, background information becomes unreliable, resulting in erroneous object detection or misdetection of objects.

Mean Shift (MS) is a fast tracking method which does not require any background information. It uses the color histogram information of chosen objects as tracking feature and doesn't have stationary camera assumption, i.e. camera motion does not prevent tracking. However, it does not provide any mechanism for object birth and death and hence require manual selection of object to be tracked. Also, its performance might be adversely affected by kernel drift, i.e. the kernel may shift from tracked object to the objects in the background after a while. Usually object tracking systems are built for stationary cameras and there are limited number of studies about how to maintain tracking if the camera starts moving. In this thesis, a continuous object tracking system with stationary and moving camera modes based on correspondence based tracking and mean shift is proposed. This system is able to detect when the camera is starts moving by observing the background update rate and automatically switches between correspondence based tracking and mean shift.

A general overview of the system is given in Figure 1. The system starts with background estimation. In our experiments, foreground estimation is completed in 125 frames. After this stage, foreground image can be segmented reliably and from 125th frame to 225th frame, masked background update rate, a metric proposed for calculating the background reliability, is observed. Then quartiles are calculated and a threshold for detecting outliers in this background reliability metric is obtained. The frame is said to be moving if the masked background update rate is higher than this threshold. However, as there is a lag in detection, some frames might have been processed since the camera first started to move. So, a buffer is held for each frame that holds the tracked object information for the previous *n* frames. When the camera is detected as moving, mean shift tracking is applied to the past objects data that are kept in buffer and mean shift continues with the buffer result. This mechanism prevents generation of the false objects when camera starts to move.

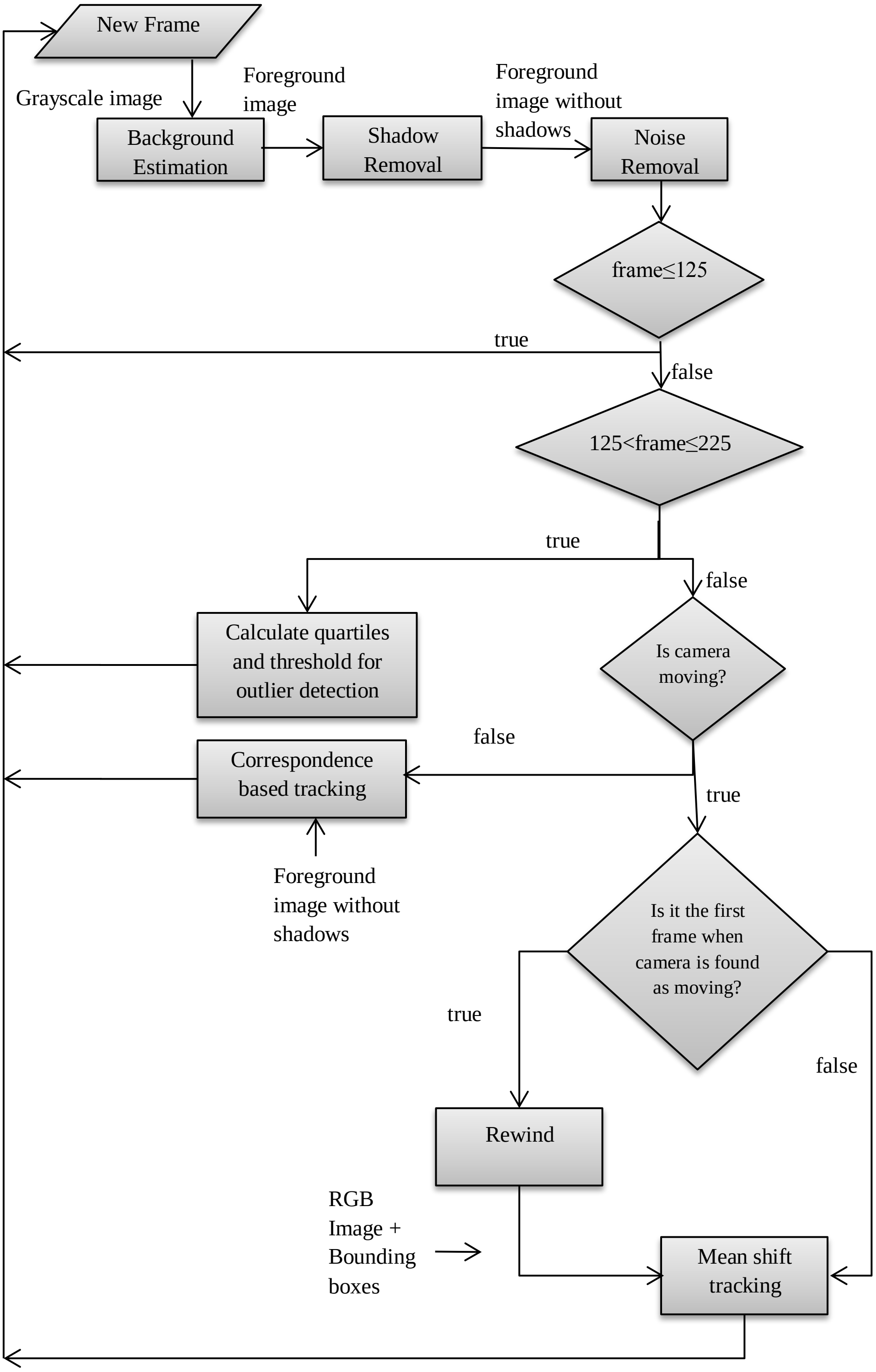

New Frame
Grayscale image
Background Estimation
Foreground image
Shadow Removal
Foreground image without shadows
Noise Removal
frame≤125
true
false
125<frame≤225
true
false
Calculate quartiles and threshold for outlier detection
Is camera moving?
false
Correspondence based tracking
true
Foreground image without shadows
Is it the first frame when camera is found as moving?
true
false
Rewind
RGB Image + Bounding boxes
Mean shift tracking

Figure 1. General flow chart of the system

3.2 Background Subtraction

As correspondence based tracking uses foreground information, background should be estimated for each frame. For background estimation, because of its low complexity and ability to estimate foreground correctly, Improved Adaptive Gaussian Mixture Model (GMM) [27] is used.

According to the GMM method, main aim is to determine whether the pixel belongs to the foreground or the background. This can be calculated with the Equation (3.1).

$$R = \frac{p(BG|\ \vec{x}(t))}{p(FG|\ \vec{x}(t))} = \frac{p(\vec{x}(t)|BG)\ p(BG)}{p(\vec{x}(t)|FG)p(FG)} \tag{3.1}$$

Where R is the Bayesian decision, BG is background, FG is foreground, $\vec{x}(t)$ is the pixel value at time t and p represents probability. As there is no information about the probability of the pixel belonging to a foreground or background at first, probabilities of being foreground and background are set equal as $p(BG) = p(FG)$. According to the equation (3.2), a pixel is a part of the background if its probability is greater than a threshold c_{th}.

$$p(\vec{x}(t)|\ BG)\ > c_{th} \tag{3.2}$$

In this formula, $p(\vec{x}(t)|\ BG)$ is background model which can be represented as c_{FG}. To adapt to the alterations in the background image the background model is updated by replacing the old samples with the new ones. In a selected time t in a time period T, the sample set $X_T = \{x(t), \dots, x(t-T)\}$ is updated and $p(\vec{x}(t)|\ BG)$ is calculated again.

In some frames prior to current frame, to find the probability of pixels that belong to the foreground, Equation (3.3) is used.

$$p(\vec{x}|X_T, BG + FG) = \sum_{m=1}^{M} \hat{\pi}_m N(\vec{x}; \widehat{\vec{\mu}_m}, \widehat{\sigma_m^2} I) \tag{3.3}$$

According to Equation (3.3), $p(\vec{x}|X_T, BG + FG)$ is the estimate of pixels in recent frames that might belong to foreground, *M* is the number of components, $\hat{\pi}_1$, $\hat{\pi}_2$, ..., $\hat{\pi}_M$ are means, $\widehat{\sigma_1}$, $\widehat{\sigma_2}$,..., $\widehat{\sigma_m}$ are variances, $\hat{\pi}_1$, $\hat{\pi}_2$,..., $\hat{\pi}_m$ are positive mixing weights. Sum of $\hat{\pi}_1$, $\hat{\pi}_2$,..., $\hat{\pi}_m$ equals to 1.

Updating formulas for a data sample *x(t)* at time *t* are shown in Equations (3.4), (3.5) and (3.6).

$$\hat{\pi}_m \leftarrow \hat{\pi}_m + \alpha(o_m^{(t)} - \hat{\pi}_m) \tag{3.4}$$

$$\widehat{\vec{\mu}_m} \leftarrow \widehat{\vec{\mu}_m} + o_m^{(t)}(\frac{\alpha}{\hat{\pi}_m})\,\vec{\delta}_m \tag{3.5}$$

$$\widehat{\sigma_m^2} \leftarrow \widehat{\sigma_m^2} + o_m^{(t)}(\frac{\alpha}{\hat{\pi}_m})(\,\overrightarrow{\delta^T}_m\vec{\delta}_m - \widehat{\sigma_m^2}) \tag{3.6}$$

Where $\delta_m = x(t) - \mu_m$, and is a α constant equals to *1/T* that reduces the effect of the old data on the new data. All pixels are set to *0* while the close component of largest π_m is set to *1*.

Closeness is described by Mahalanobis distance. A component is said to be a close component if Mahalanobis distance from the component is less than a threshold. In this study this threshold is defined as four standard deviations. From *m*th component, Mahalanobis distance formula is given in Equation (3.7).

$$D_m^2(\vec{x}(t)) = \overrightarrow{\delta^T}_m\,\vec{\delta}_m/\widehat{\sigma_m^2} \tag{3.7}$$

According to Equation (3.7), $D_m^2(\vec{x}(t))$ is Mahalanobis distance from the component *m*. If any close component is not found, new component is generated with equations (3.8), (3.9) and (3.10).

$$\hat{\pi}_{M+1} = \alpha \tag{3.8}$$

$$\mu_{M+1} = x(t) \tag{3.9}$$

$$\sigma_{M+1} = \sigma_0 \tag{3.10}$$

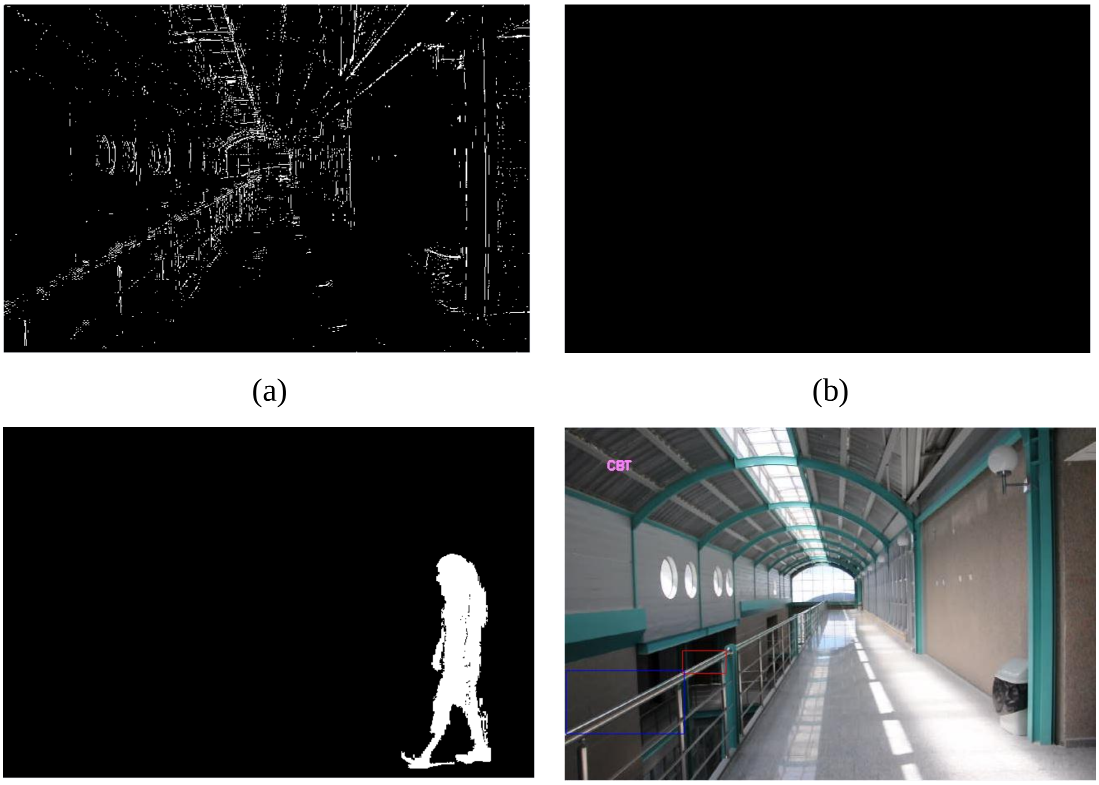
In Equation (3.10), σ_0 is initial variance. As number of components gets higher, old components need to be deleted. Thus, if the component number is equal to the maximum number, the component with the smallest $\hat{\pi}_m$ is deleted. The components are listed from the component having the least $\hat{\pi}_m$ to the most, largest distribution B is calculated with Equation (3.11).

$$B = \arg\min_b \left(\sum_{m=1}^{b} \hat{\pi}_m > (1 - c_f) \right) \tag{3.11}$$

where c_f is used to calculate the maximum part of the data which a foreground object can have. When a moving object appears in the frame and waits for a while, its $\hat{\pi}_m$ will be larger than c_f. Thus, the waiting object will be a part of the background and erased from the foreground.

First 125 frames are used for background learning. While the system is learning the background, to avoid detection of false objects, tracking is not initiated. Once an initial background is estimated, object detection is done followed by correspondence based tracking.

Figure 1 (a) and 1 (b) both represent a frame while the background is still being learned while Figure 1 (a) represents the foreground image before the noise removal process and Figure 1 (b) represents the foreground image after the noise removal process. From Figure 2 (a) we can see that background is fully learned and only a foreground object can be seen. Figure 2 (b) shows an image having false object detection (shown with blue and red bounding boxes). False object detection might occur if tracking is allowed in the initial background learning phase before a stable background is estimated.



(a) (b)

(c) (d)

Figure 2. Noise removal process (a) Foreground image during background learning before noise removal, (b) Foreground image during background learning after noise removal, (c) Foreground image after background is learned, (d) RGB Image result with false tracking if the noise is not cleaned.

3.3 Shadow and Noise Removal

Because of lighting, in videos, usually shadows of objects appear. This shadow causes the trackers to consider it as a part of the object resulting in larger bounding boxes than the object. If these kernels with shadow are handed in to mean shift tracker, it is likely that the tracking kernel drifts to the background and lose tracking. To remove the shadows, the HSV color space based shadow removal method is applied [22].

To determine if the pixel in a frame belongs to foreground or it is a shadow. The equations (3.12), (3.13) and (3.14) are applied.

$$|I_t(x,y).H - B_t(x,y).H| \leq T_H \tag{3.12}$$

$$I_t(x,y).S - B_t(x,y).S \leq T_S \tag{3.13}$$

$$\propto \leq \frac{I_t(x,y).V}{B_t(x,y).V} \leq \beta \qquad (3.14)$$

where, H is the hue component, S is the saturation component and V is the value component in HSV color space. $\propto$ and β are two threshold values which can be between 0 and 1. Higher $\propto$ means that, the lesser the shadows make objects darker. With higher β the noise robustness is increased. In a given time t, $I_t(x,y)$ is the foreground pixel, $B_t(x,y)$ is the background pixel. T_S is the threshold that is used to control the saturation value difference, T_H is the threshold to calculate the change in hue value. If all the equations stated in (3.12), (3.13) and (3.14) are found as true, pixel is said to belong to foreground and if one of them is found as false, the pixel is said to be a shadow. To make the system adaptive to changes in the background, when there is no object in the background, the HSV image is renewed.

For noise removal, connected component analysis in [29] is used. According to connected component analysis, each different object in a binary image is labeled with different numbers. To find connected components, 8-connectivity is used. A pixel's 8-connected pixels are all pixels that are next to that pixel.

All 8-connected pixels generate a connected component. In an image, connected components with various sizes are generated. A generated connected component is regarded as noise if the density of an object D_{cc} is smaller than a threshold Th_{cc}. A connected component's density is calculated with Equation (3.15).

$$D_{cc} = N_{cc}/A_{cc} \qquad (3.15)$$

where, N_{cc} is the number of pixels in a connected component, A_{cc} is the area of the connected component.

3.4 Calculating the masked background update rate

At the end of the background estimation phase, we obtain binary images generally with noise. The noise in a frame that comes from a stationary camera can be handled with simple noise removal techniques. The noise is removed with connected component technique in [10]. A foreground image is a binary image where foreground objects are shown as white. When camera starts to move, background starts to get updated with new scene information and gets noisy. So, when the camera moves there should be an increase in the background update rate and foreground image is expected to get noisy. The increase of noise in the foreground image when camera starts to move can be observed from Figure 3. As it can be seen from Figure 3, besides the person many pixels from the background are also detected as foreground incorrectly.

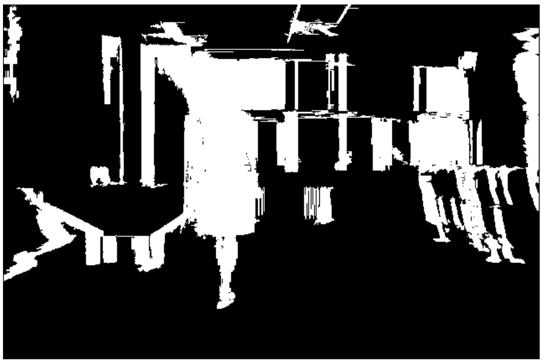


Figure 3. Foreground image when camera moves

To detect the camera movement, we propose to measure the increase in the background update rate and use the information that the background gets updated at a higher rate when the camera starts moving. To increase the accuracy, we exclude the areas belonging to the moving objects. This procedure is explained in more detail in the remaining of this section.

First, a binary mask image *Mask* is created with the equations (3.16) where $FG_{t\text{-}1}$and FG_t are the detected binary foreground images at time *t-1* and *t* respectively.

$$Mask_t(i) = \begin{cases} 1 & if\ FG_t(i) = 1\ \vee\ FG_{t-1}(i) = 1 \\ 0 & otherwise \end{cases} \tag{3.16}$$

As a result of this operation, *Mask* carries the information about the areas belonging to moving objects at the last consecutive two frames. An example of a mask image is given in Figure 4.

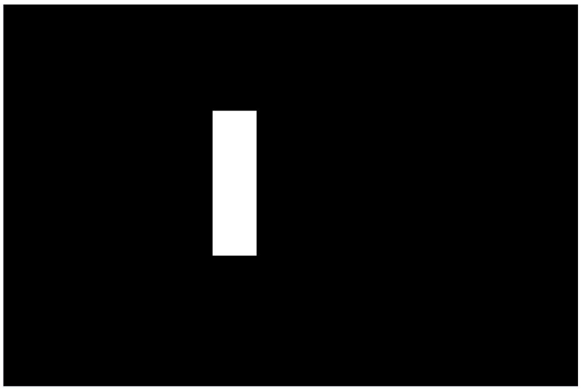


Figure 4. Mask image

Then for all pixels i at time t, we calculate the difference between the previous grayscale background image $BG_{t-1}$and the current grayscale background image BG_t excluding the areas belonging to moving objects (Equation 3.17).

$$X_t = \begin{cases} X_t + |BG_t(i) - BG_{t-1}(i)| & if\ Mask_t(i) = 0 \\ X_t & otherwise \end{cases} \tag{3.17}$$

As a result of this operation we have X_t holding the masked background update rate where the initial value of X_t is zero.

There might be a sudden increase in masked background update rate in foreground image for just a few frames causing the system switch to mean shift incorrectly. To prevent this, instead of taking only the current image's calculated pixel count, the mean of the last 10 frames is used. Thus, we obtain smoother variations in total pixel graph.

Figure 5 shows an example of masked background update rate in a video for each frame. The dotted lines show the masked background update rate without any postprocessing, the solid lines show the masked background update rate if the mean of the last 10 frames is calculated. As it can be inferred from the figure, by getting the mean of the masked background update rate, we smooth the graph and prevent sudden increases that cause camera mode switching which may result in incorrect detection.

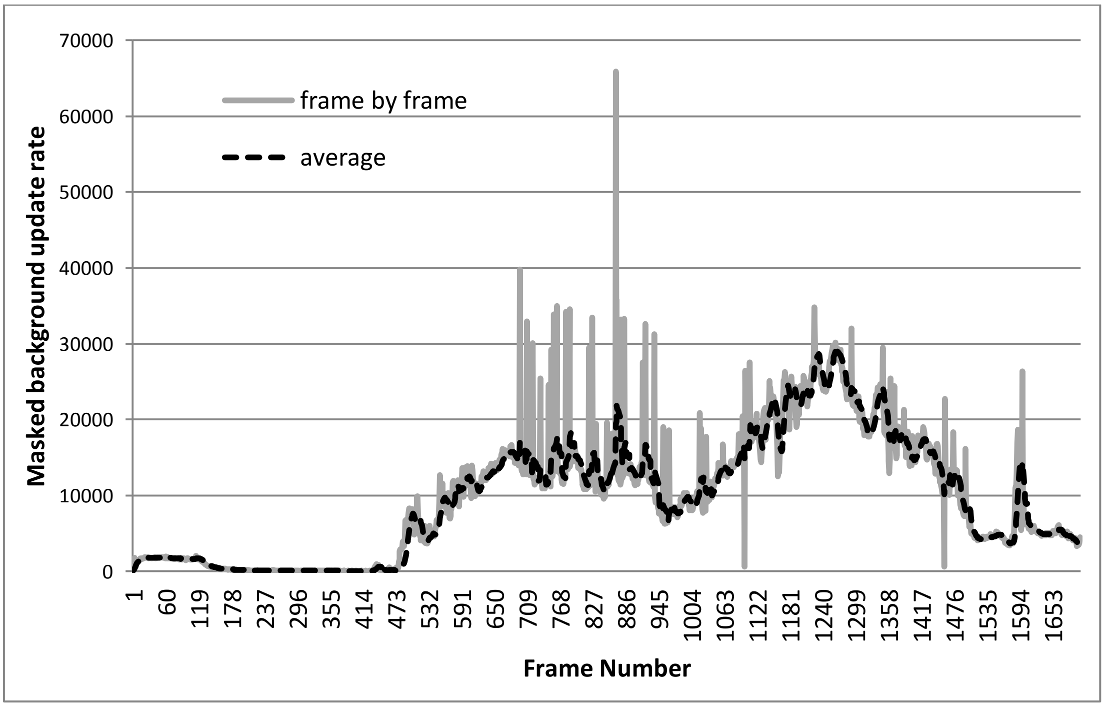


Figure 5. Masked background update rate graph

Interquartile Range is an easy but effective method for finding outliers in a data. To apply Interquartile Range, a sample set should be defined. Let X_s be a set of all masked background update rate of detected foreground pixels X_t where 125<t≤225, $X_s = \{X_{126}, X_{127}, \dots, X_{225}\}$. After obtaining the sample set, the values are put in order in ascending order. So, the new set can be represented as $X_s{}' = \{X_1', X_2' \dots, X_{100}'\}$. Then, quartiles are found. These quartiles are Q_1, Q_2 (median) and Q_3. Quartiles are calculated with the Equations (3.18), (3.19) and (3.20).

$$Q_1 = X_{25}^{'} \tag{3.18}$$

$$Q_2 = X_{50}^{'} \tag{3.19}$$

$$Q_3 = X_{75}^{'} \tag{3.20}$$

The outliers o_1 and o_2 are calculated with Equation (3.21) and (3.22) respectively.

$$o_1 = Q_1 - k * (Q_3 - Q_1) \tag{3.21}$$

$$o_2 = Q_3 + k * (Q_3 - Q_1) \tag{3.22}$$

where, k is a constant that we set as 3, o_1 is the minimum and o_2 is the maximum thresholds. In this thesis, o_2 will be used as a threshold for switching between moving and stationary camera modes.

The data between the first and the 125th frames are used for background learning. Then data from 126th frame to 225th frame are used for calculating the quartiles and the threshold o_2 is in order to detect outliers for the remaining of the data. During this period no tracking is performed.

If camera is on correspondence based tracking (CBT) mode, Equation (3.23) is used. If the camera is on mean shift, another switching formula is defined. The reason for using such a mechanism and the algorithm for switching method is given in the Section 3.7.

If at time t Equation (3.23) is satisfied, system switches from CBT to mean shift mode.

$$X_t \geq o_2 \tag{3.23}$$

For the test data shown in Figure 5, correspondence based tracking is applied until 1225th frame due to the calculated masked background update rate. From around 1191th frame, camera starts to move and system switches to mean shift.

The flow of background estimation and outlier detection is demonstrated in Figure 6.

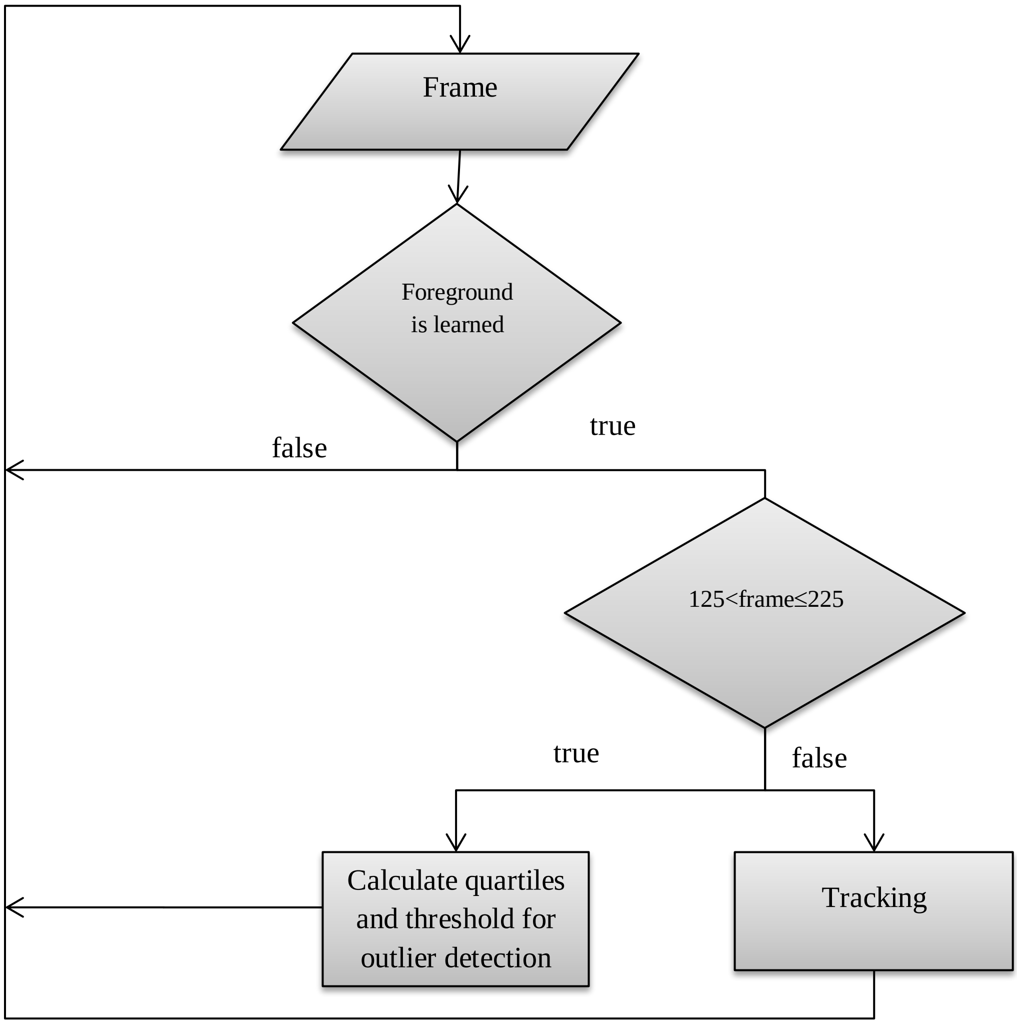


Figure 6. Flow chart of outlier detector training

3.5 Correspondence based tracking

In this thesis we use the correspondence based tracking method proposed in [10]. Correspondence based tracking is an efficient tracking method which both can track and detect objects. It detects the objects at each frame, and matches these objects between consecutive frames with size and closeness metrics. For matching, foreground images are needed. Therefore, foreground is extracted by using the estimated background information. To find the similarity between two objects in current and previous frames, we first look at the closeness factor. If there is a moving object in previous frame (O_p), in the current frame, the object (O_c) should not be far from its initial state. To find the distance between O_p and O_c, Euclidean formula is used.

$$d(O_p, O_c) = \sqrt{\sum_{i=1}^{2} (x_{ip} - x_{io})^2 + (y_{ip} - y_{io})^2 \ )} \ \leq \ t_d \tag{3.24}$$

where, $d(O_p, O_c)$ is the Euclidean distance between O_p and O_c, x is the x coordinate of object's center of mass, y is the y coordinate of the object's center of mass, t_d is the threshold.

The other matching formula is matching the objects' sizes. Between consecutive frames, objects can change size either if the moving object is going distant from the camera, or tracked person bends. Even under these circumstances, an object's size does not change significantly between 2 consecutive frames. Calculation of size ratio S_D is given in Equation (3.25).

$$S_D = \begin{cases} \dfrac{S_p}{S_c} & if \ S_p > S_c \\ \dfrac{S_c}{S_p} & if \ S_c > S_p \end{cases} \tag{3.25}$$

Where, S_p is the bounding box size of the previous frame and S_c is the bounding box size of the current frame.

Deciding if O_c is the same object as O_p or it is a new object is defined in Equation (3.26) where t_{size} is a size threshold for matching criteria.

$$O_c = \begin{cases} O_p & if \ d(O_p, O_c) \ \leq \ t_d \ and \ \ S_D \leq t_{size} \\ new\ object & otherwise \end{cases} \tag{3.26}$$

3.6 Mean shift tracking

According to mean shift [5], first of all, a target model of tracked object is created with the probability density function in Equation (3.27).

$$q_u = C \sum_{i=1}^{n} k \ (||x_i||^2) \delta[b(x_i) - u] \tag{3.27}$$

where x_i represent pixel locations of target model, δ is the Kronecker delta function, $b(x_i)$ represents the color bin at the pixel location x_i, k is a number that denotes more weight to the pixels at centroid, u is the color bin, C is the normalization constant, q_u is the probability of color u.

Mean shift seeks the most similar object in the neighborhood of the selected object. To perform this, a target candidate model is described:

$$p_u(y) = C_h \sum_{i=1}^{n_h} k \left(\left\| \frac{y - x_i}{h} \right\|^2 \delta[b(x_i) - u] \right) \tag{3.28}$$

According to Equation (3.28), y represents pixel locations of target candidate, h is the kernel size. To find the similarity between target model and the target candidate, Bhattacharya coefficient is used.

$$\rho[q_u, p_u(y)] = \sum_{n=1}^{m} \sqrt{p_u(y) q_u} \tag{3.29}$$

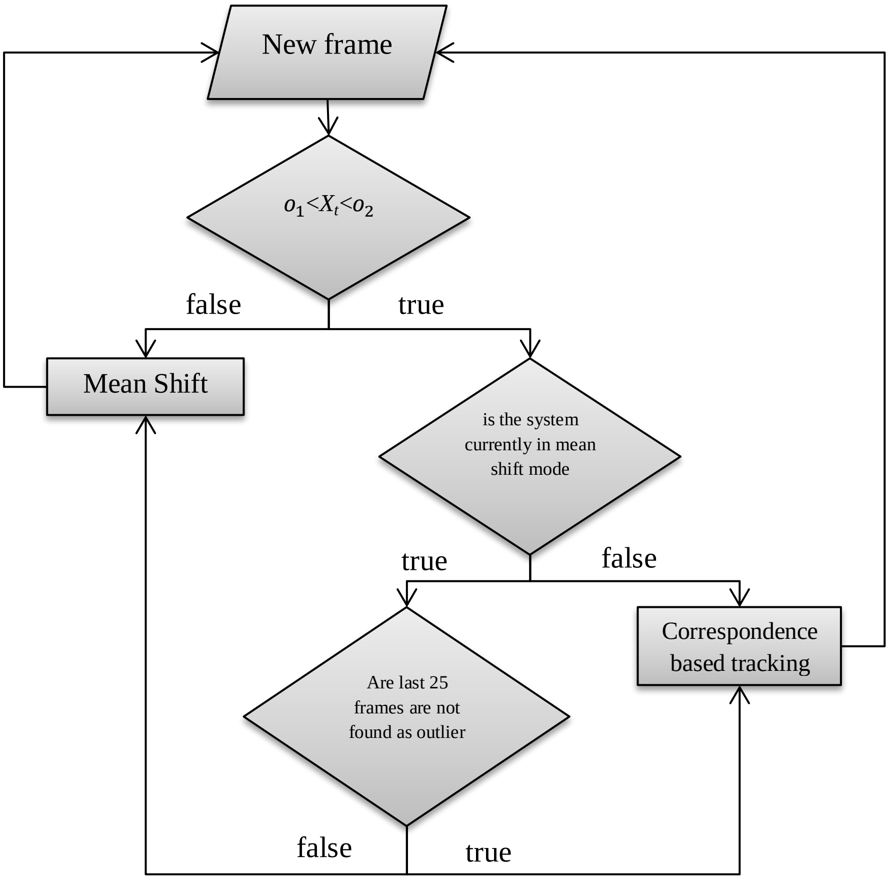
where ρ is the cosine of vectors. According to this metric, a bigger ρ number represents a good similarity between two objects is obtained.

Searching for the best match starts from the pixels closer to the object and if the similarity is not sufficient, mean shift shifts to the pixels that are far from the object.

3.7 Switching between tracking methods

If the camera is once detected as moving, in some consecutive frames, due to slowing down the camera, the found masked background update rate might be found in the normal distribution. So, switching back to correspondence based tracking in these conditions may result in creating false bounding boxes and false tracking. To prevent this, if a pixel count is found to be in the normal distribution, consecutive 25 frames are checked if all the pixels are in the normal distribution. If any of them are found as outlier, mean shift continues. The switching formula is given in Equation (3.30). If this equation is satisfied system switches to CBT mode, otherwise it carries on with mean shift mode. A flow chart of the switching mechanism is shown in Figure 7.

$$\text{If } \forall X_n,\ n \in \{t-24, ..t\}\ X_n \leq o_2 \qquad (3.30)$$



New frame
o_1<X_t<o_2
false
true
Mean Shift
is the system currently in mean shift mode
true
false
Correspondence based tracking
Are last 25 frames are not found as outlier
false
true

Figure 7. Flow chart of switching between tracking methods

3.8 Buffer Mechanism

Although camera is detected as moving as a result of the outlier detection, there might have been some frames since the camera first started to move. Figure 8 illustrates the change in masked background update rate in video 7 when camera starts moving. As it can be seen from this figure, due to the lag in detecting camera movement, background reliability might get low and result in detection and tracking of false objects. In this example, camera starts moving at frame 473 and false objects are detected at frame 477, the camera movement is detected by the system at frame 488 and false objects are also passed to mean shift tracking.

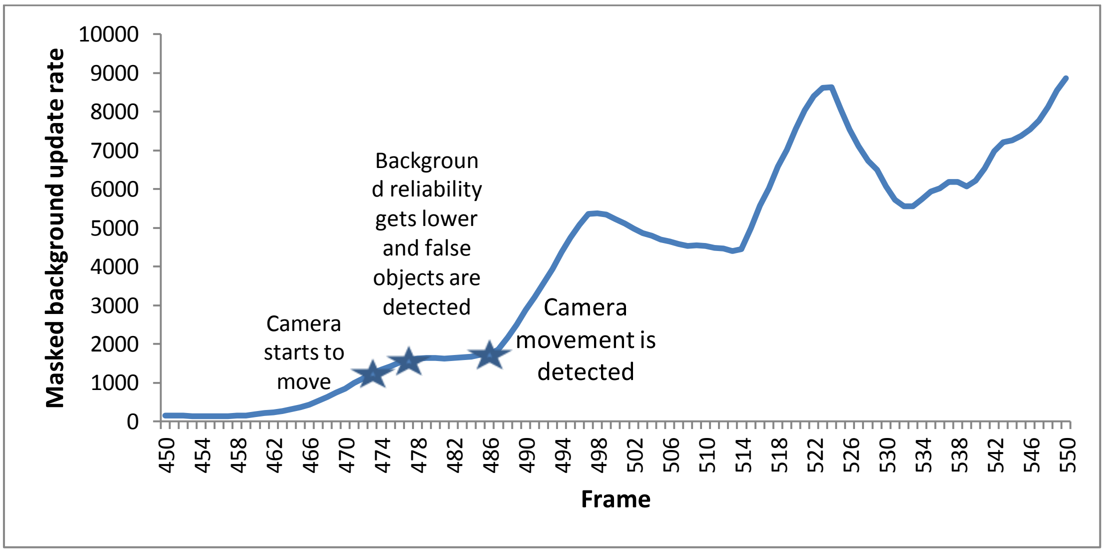


Figure 8. Increase in masked background update rate when camera starts to move.

To prevent this problem, a circular buffer mechanism is used. Using this circular buffer, the system can be rewind, system state before camera movement can be restored and the objects are passed to mean shift tracker a predetermined number of frames before the current frame preventing tracking of false objects.

We hold an array which has a size of $S_{B,}$ in this array we hold both the last $S_{B,}$ frames and information about the detected objects in these frames by the CBT algorithm. B_L is the index that keeps the last added index. Starting from the index right after the B_L, for each time, we need to look at the next index. The index which will be looked (B_c) is found by applying the Equation (3.31).

$$B_c = (B_{L+1})\%S_B \tag{3.31}$$

With this way, mean shift can be applied to last S_B frames before the detection of camera movement.

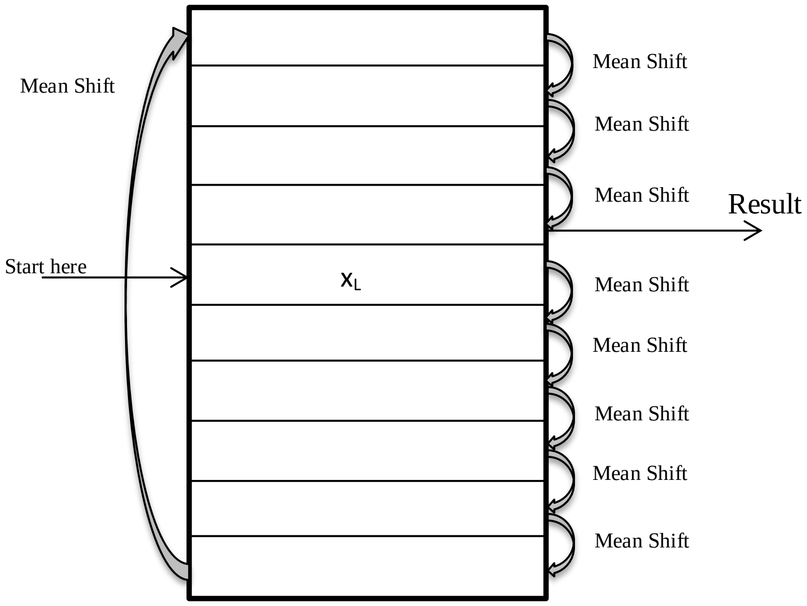
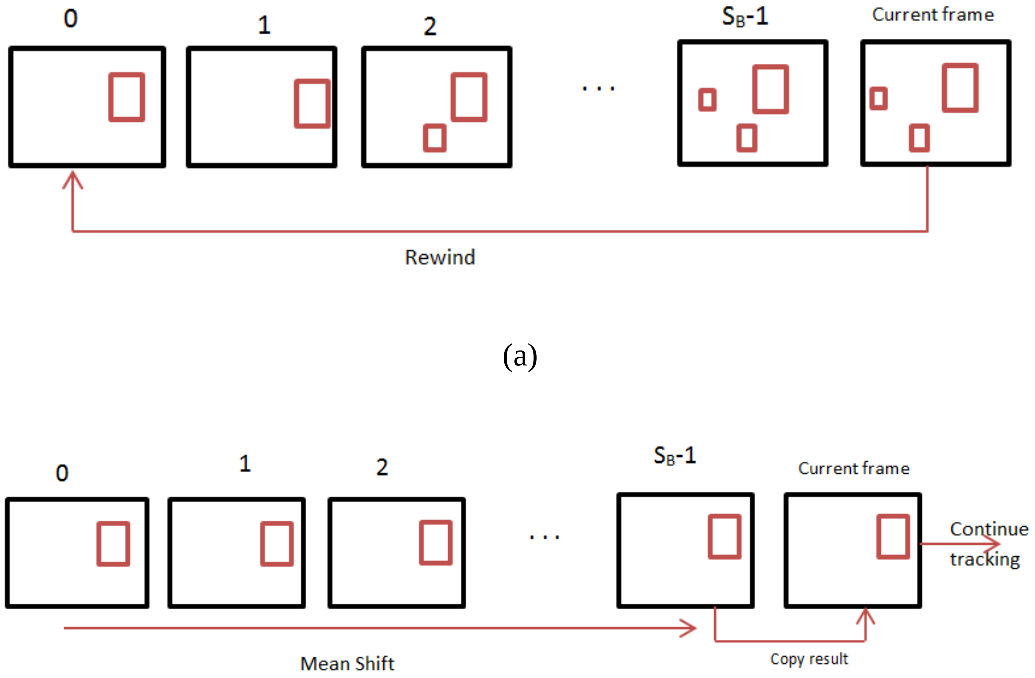


Figure 9. Applying mean shift on buffer

After reaching at the end of the buffer, the found objects are transferred to the current frame and mean shift continues on newly coming frames until the camera stops. Figure 10 (a) shows an example of false object detection in frames if the buffer mechanism is not used, Figure 10 (b) that these false objects can be eliminated by restoring the system state using a buffer holding information before the camera movement.



(a)

(b)

Figure 10. Illustration of the bounding boxes for the detected objects (a) False bounding boxes are created, (b) Eliminating false bounding boxes by rewinding and restoring the system state S_B frames prior to the current frame and applying mean shift.

3.9 Reducing the Bounding Box Size

The tracked object information is extracted from its bounding box. In that bounding box, besides the tracked object information, there might be some information from the background. In correspondence based tracking mode, size of the bounding box is updated in each frame. But in mean shift mode, the size of the bounding box stays the same during the entire tracking. As tracking continues with mean shift, because of the inclusion of background information, the tracker might drift to the background and lose the object after a while. Also, object might move away from the camera causing a larger bounding box than the object size. To overcome this drawback, when we switch to mean shift tracking mode, bounding box size is scaled down. Bounding box is cropped from top and bottom with ¼ of its size and right and left with ⅛ of its size. With this way, bounding box size gets smaller and background's effect on color histogram of the bounding box is reduced. Figure 11 shows the reduction rates in the bounding box dimensions relative to the original size. In this figure, width of the old size is represented by w_{bb} and height of the old size is represented by h_{bb}. Width of the reduced bounding box (new size) is represented by w'_{bb} and height of the reduced bounding box is represented by h'_{bb}.

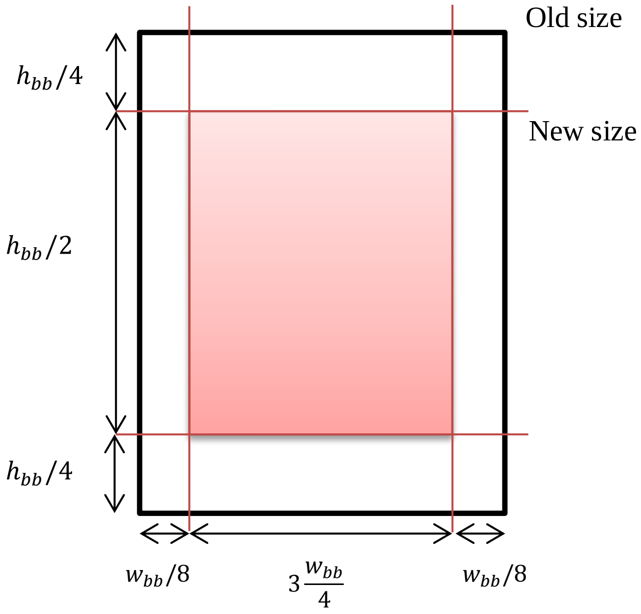
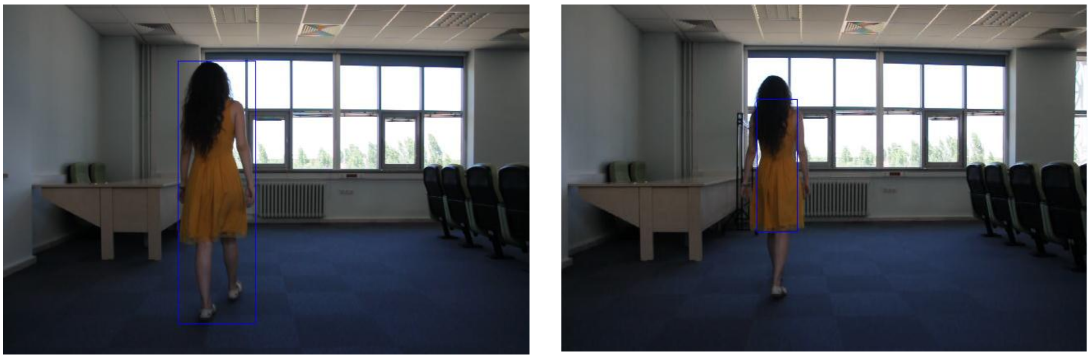


Figure 11. Reducing the bounding box size

$$w'_{bb} = 3\frac{w_{bb}}{4} \tag{3.32}$$

$$h'_{bb} = h_{bb}/2 \tag{3.33}$$

Figure 12 (a) shows a bounding box generated by correspondence based tracking, Figure 12 (b) shows when the camera switches to moving and bounding box is resized.



(a) (b)

Figure 12. Bounding boxes while tracking (a) Bounding box while tracking with correspondence based tracking, (b) Reduced bounding box while tracking with mean shift

CHAPTER 4

EXPERIMENTAL RESULTS

4.1 Dataset

There is no publicly available dataset that includes both cases when the camera is stationary and when the camera moves. Therefore, to test our system we created our own database by recording video sequences with different scenarios and in different places. 20 videos in 7 different places have been captured with a Nikon D90 DSLR camera fitted with AF-S DX NIKKOR 18-105mm f/3.5-5.6G ED VR lens. Videos were captured at various places in the Graduate School of Informatics of Middle East Technical University. The resolution of the captured videos is 640x424 and frame rate is 24 fps. The places the videos are captured are:

- Corridor at the 2nd floor:

  In this place, 5 videos are captured in two different views. The lighting and background are different in two views. Also, these videos include different people. In first and second video, tracked person is the same but the amount of camera motion is different. In 5th video, two people are in the scene. Camera motion is forwards.

- Front courtyard of the Institute:

  In this place, 3 videos are captured in three different places. There are different weather conditions such as snow and sunshine. In 11th and 12th videos, weather is snowy and in 13th video, weather is sunny. Camera motion is performed by panning the camera from left to right or right to left.

- Basement floor captured from top:

  One video is captured in this place. Apart from other videos, moving objects are captured with an angle from the top, not from on their level. Camera motion is performed by zooming forwards.

- A road near the Institute:

  Two videos are captured in this place. In first video, one person is tracked, in second video, two people are tracked. In both videos, after people come into scene and walk for a while, they go forwards, getting distant from camera. While people are going forwards, a car with a camera follows them from back.

- Seminar room

  6 videos are captured in this place. These videos include different lighting conditions, different number of people and different places in seminar room. Video 6 and 7 are captured in the same place but the walking people are different. In these videos, curtains are closed and room is illuminated with fluorescent lighting. Videos 8, 9 and 10 are captured when curtains are open and room is illuminated with sunshine. These videos are taken in three different places of seminar room. Last video is prepared by cutting video 7 when the camera is moving and reversing it backwards. With this way, we obtained a video that includes the case returning back to stationary mode from camera moving mode. Camera motion is performed by both following the tracked person forwards and panning the camera from right to left.

- Virtual Reality and Computer Vision Lab

  One video is captured with one person in this place. Camera motion is performed by panning the camera from right to left.

- Basement floor

Three videos are taken with different people and in different places. Video 16 and 17 are in the same place but the closeness to camera is different. Camera motion is performed by panning the camera from right to left.

The dataset is publicly available and can be accessed from http://ii.metu.edu.tr/content/object-tracking-stationary-and-moving-camera-modes.

The information about the videos in the dataset is given in Table 1. This table includes the video number, location where the video were captured, tracking result, frame number when the system switches to mean shift and frame number when the camera actually starts to move.

Table 1. Tracking results of videos in dataset

Video #	Location	Tracking result	# of objects	# of detected objects	Frame# switching to mean shift	Frame# camera starts moving
1	Corridor	OK	1	1	878	902
2	Corridor	Tracks but finds another blob	1	1	536 and 645	533 and 639
3	Corridor - back view	Tracks but finds another blob	1	1	687	680
4	Corridor - back view	OK	1	1	630	622
5	Corridor - back view	Tracks one of two people	2	1	669	669
6	Seminar room	Tracks at first, then MS shifts	1	1	450	450
7	Seminar room	OK	1	1	486	473
8	Seminar room	OK	1	1	491	490
9	Seminar room	OK	1	1	572	572
10	Seminar room	OK	2	2	571	571
11	Front courtyard of the Institute	OK	2	1	1217	1216
12	Front courtyard of the Institute	OK	1	1	357	353

Table 1 (cont.)

13	Front courtyard of the Institute	Tracks but then MS shifts	1	1	540	535
14	Outside	OK	1	1	615	615
15	Outside	OK	2	2	411	411
16	Basement floor	Tracks one of two people	2	1	786	812
17	Basement floor	Tracks then MS shifts	1	1	686	686
18	Basement floor	Tracks then MS shifts	1	1	481	478
19	Basement captured from upstairs	OK	1	1	359	356
20	Lab	Tracks then MS shifts	1	1	371 and 751	355 and 750
21	Seminar room	OK	1	1	486	473

In Table 1, location column indicates where the video is captured, tracking result demonstrates if the system works properly or not along with the problem description, number of objects specify the actual number of moving objects in the scene, number of found objects show how many moving objects are detected with the proposed system, frame number of switching to mean shift shows at which frame the proposed system switches to mean shift and frame number of camera starts moving is the actual frame number when camera starts to move. For 2 videos (2 and 20), the camera starts moving twice, so there are two numbers for "Frame# camera starts moving" field.

In all videos, the proposed system was able to detect the camera movement shortly after it occurs, switch between tracking modes without any problem and track the main object that we want to be tracked. In videos 2 and 3 while the object was tracked successfully, there were some other false bounding boxes created due to environment conditions such as illumination changes. In videos 5 and 16 the first object is tracked without problem, but the second object cannot be tracked. As the detection algorithm runs only in correspondence based tracking mode; when the system is running in mean shift mode, new object cannot be detected. On the other hand, if multiple objects enter the scene before movement of the camera, such as in video 15, the system was able to track all the objects. In videos 6, 13, 17 and 18 the system was able to detect moving camera and transfer bounding box to mean shift, but because mean shift failed to track the object until the end of the sequence due to kernel drift. In video 1, the camera starts to move at frame 902, but there is camera vibration starting from frame 874, which can be observed in the foreground image. Camera vibration causes the system to switch to mean shift tracking and when the camera moves the system is already in mean shift tracking mode. Similarly, in video 16, starting from frame 750, foreground image gets noisy because of camera vibration; however this does not cause correspondence based tracking to fail. In frame 786, because of the noise, the system switches to mean shift before the real camera motion at frame 812. The system does not switch back to correspondence based tracking mode, mean shift mode persists through the camera motion and the object is tracked without any problem.

Video 21 is prepared by merging video 7 until frame 579 with its reverse from frame 579 until the start of the video. The reason for making this video is to prove that the system goes back to non-moving mode when the camera stops. In 672th frame, camera motion stops. In our experiments, system goes back to CBT mode on frame 716. The reason for the delay is using the 25 frame buffer. To switch back to CBT mode, all consecutive 25 frames should be found as non-moving.

4.2 Test and Application Environment

The system is implemented with C++ on Visual Studio 2008. OpenCV (Open Source Computer Vision) library is used for image processing algorithms such as reading and processing video frames. OpenCV is widely used in computer vision area and offers a variety of efficient computer vision algorithms. The version of OpenCV library used in this thesis is 2.0. Testing of the algorithms is performed with C++ and MATLAB.

4.3 Experimental Results

4.3.1 Constants Used in Proposed System

Background Subtraction

According to Gaussian Mixture Model, some constants need to be given as parameters. The most appropriate parameters are determined for object tracking and used. Background learning rate α is taken as 0.0002, a greater learning rate causes system to adapt to changes quickly, but leaving a lot of noise because of the fast learning. Using a smaller learning rate causes system to adapt to changes slower. So, after using different learning rates, we decided 0.0002 is the best thinking in in both speed and accuracy cases. Mahalanobis distance $D_m^2(\vec{x}(t))$ is taken as 16 which means number of Gaussians are taken as 4.

Shadow and Noise Removal

According to shadow removal formulas in (3.12), (3.13) and (3.14) , α is taken as 0.6, β is taken as 0.6, T_s is used as 0.6 and T_H is taken as 0.9. Pixel density threshold Th_{cc} of noise removal is taken 100. The details of these parameters are explained in detail in Section 3.3.

Correspondence Based Tracking

The same parameters used in [10] are used for correspondence based tracking. The distance threshold between two objects t_d is taken as 25 and the size threshold t_{size} is taken as 1.3.

Buffer

In our experiments, camera moving is detected in 10 frames, so buffer size that holds the frames and bounding boxes (S_B) is taken as 10. For deciding maximum number of frames that needs to be detected as non-moving to switch from mean shift to correspondence based tracking, different numbers are tested such as 10, 25 and 50. Among these numbers the most efficient one is 25 because 10 is not enough frames for deciding mode change and 25 is enough for deciding mode change and more than 25 is long for mode change which can cause to lose tracking. So this parameter is taken as 25 frames. According to our experiments, the frame number reserved for background subtraction is 125 and the frame number for outlier detection to perform is 100 frames.

4.3.2 Results

Table 2 represents a continuous video tracking process in video 8. Different frames are captured in different time and RGB images with their corresponding foreground images are represented. The tracking status is shown at the top left of the images. Tracking status can be “No Tracking”, “Correspondence Based Tracking (CBT)” or “Mean Shift”. Frame 1 represents the initial RGB and foreground images, Frame 32 represents a frame while foreground is estimated, in Frame 124 foreground estimation is finished and foreground does not include any moving objects. In Frame 444, a new object enters and it is started to be tracked by correspondence based tracking. In Frame 487, there is no camera motion, so object is still tracked by correspondence based tracking. In Frame 490, as it can be seen from the foreground image, camera starts to move. In Frame 491, camera motion is detected, and system switches tracking method to mean shift. If the system state was not restored back from older data using the buffer, system would find the line shaped noise in front of the object as a moving object and we would obtain a false object. From Frame 491 to the end of the video, the object and the camera move continuously. Random frames are chosen and demonstrated from starting of the camera motion at frame 490 until the end of the video.

Table 2. RGB and Foreground Image Frames at different times

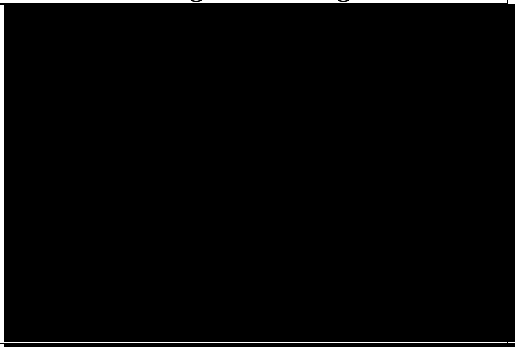
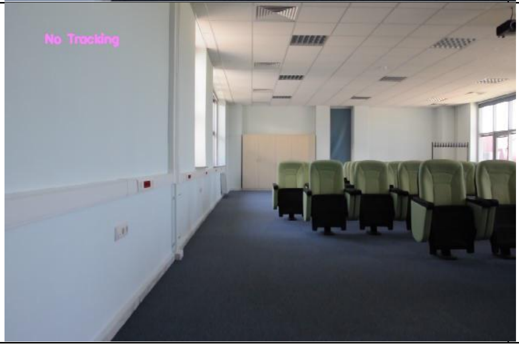
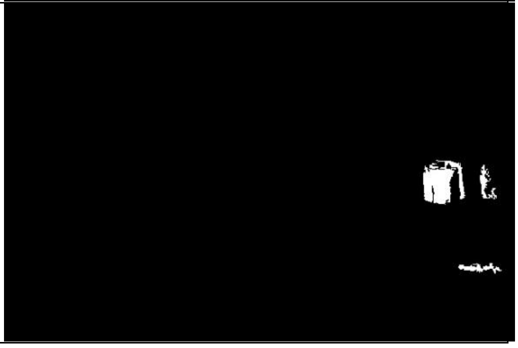
#	RGB Image	Foreground Image
1		
32		

Table 2 (cont.)

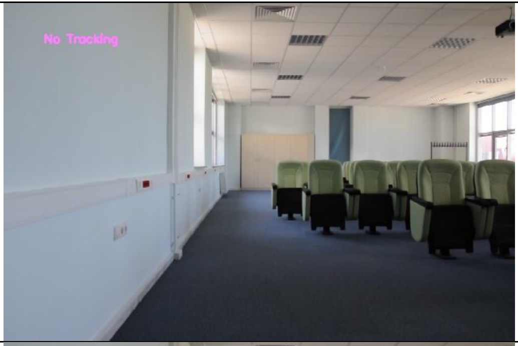
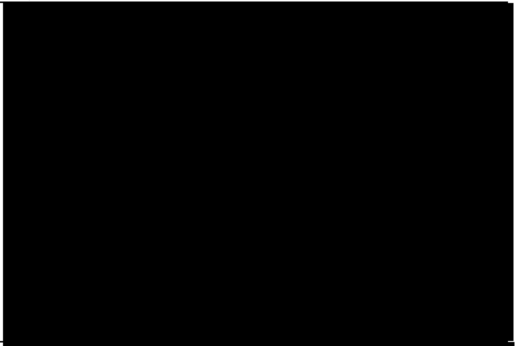
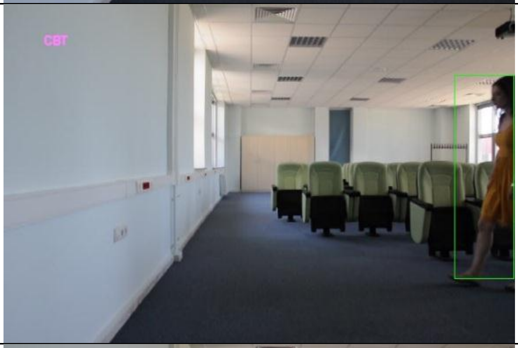
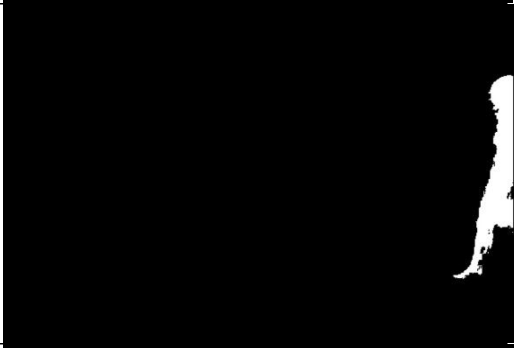
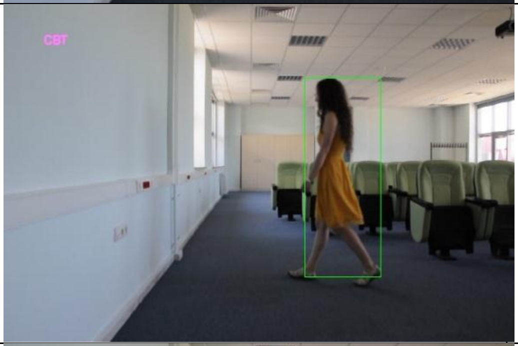
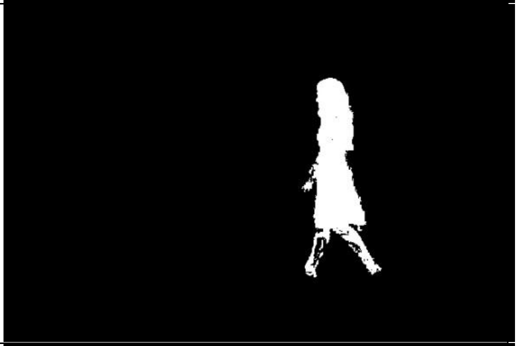
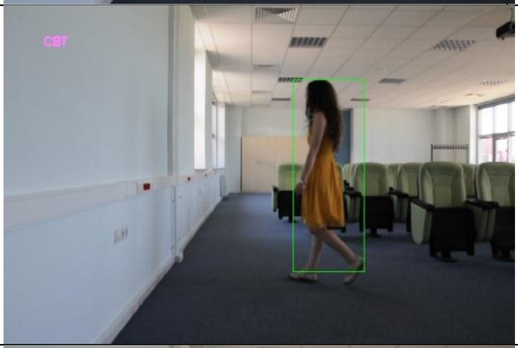
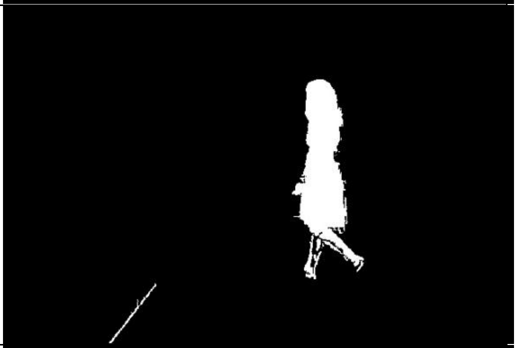
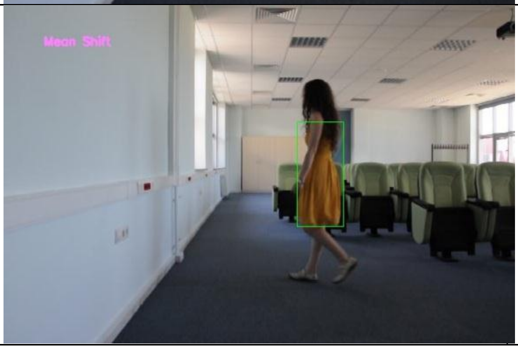
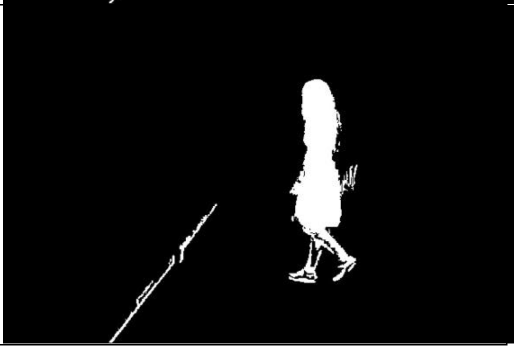
124		
444		
487		
490		
491		

Table 2 (cont.)

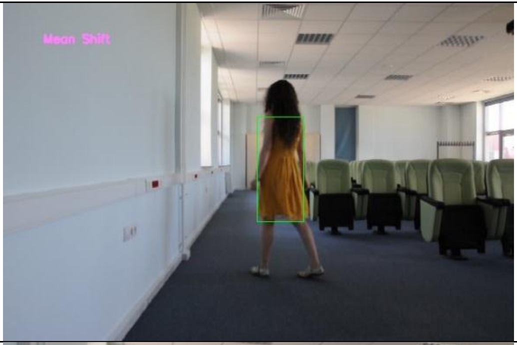
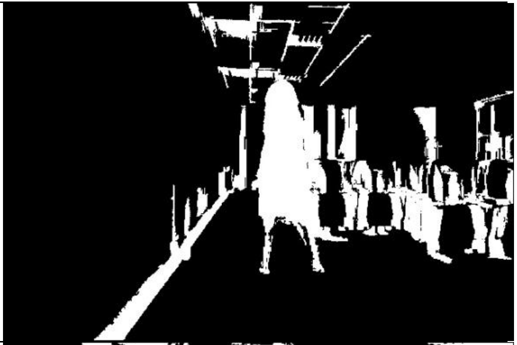
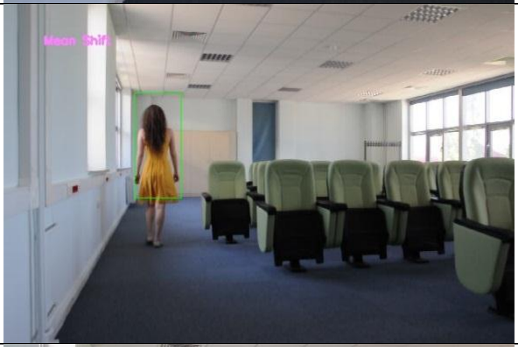
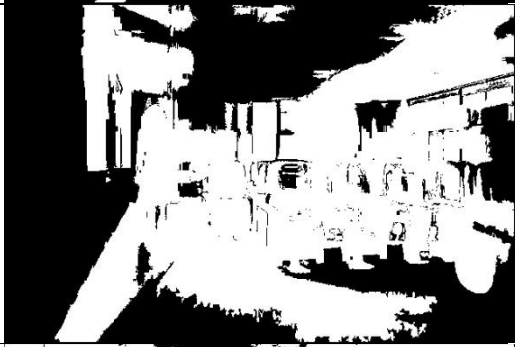
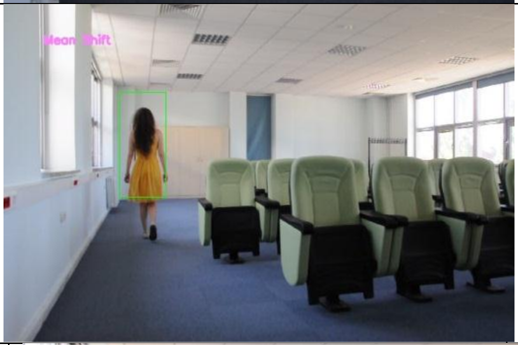
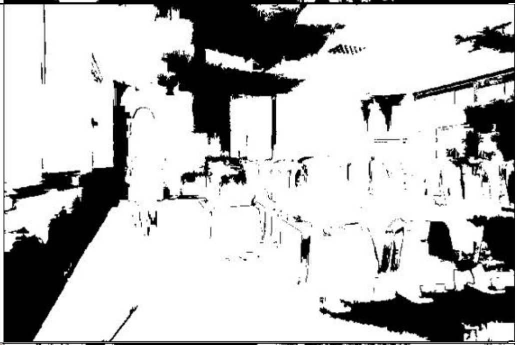
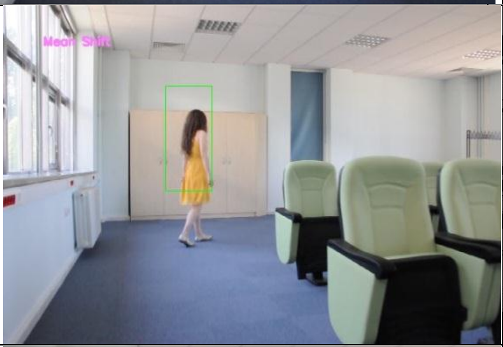
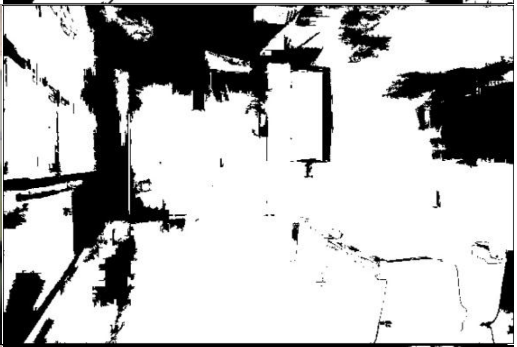
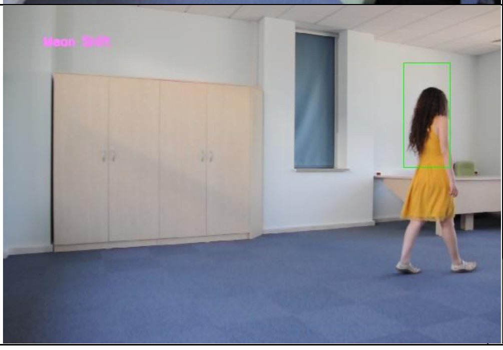
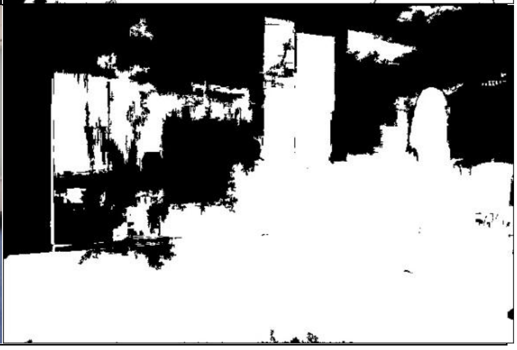
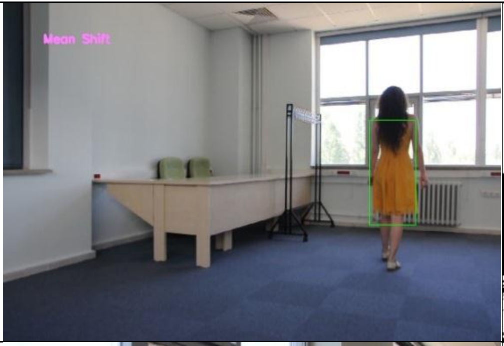
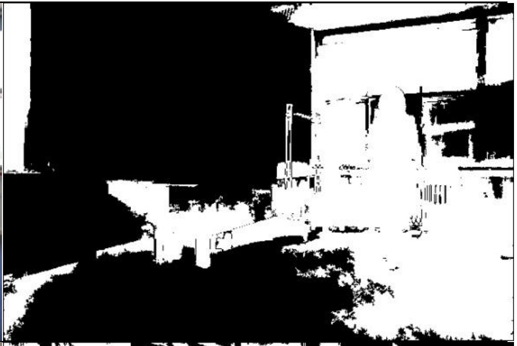
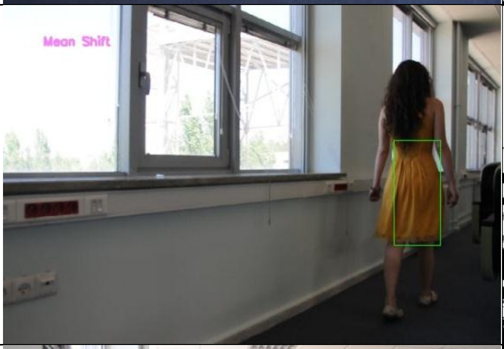
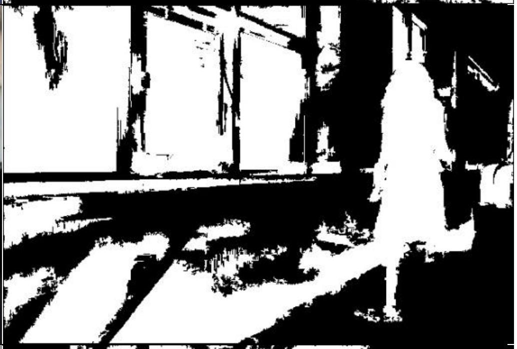
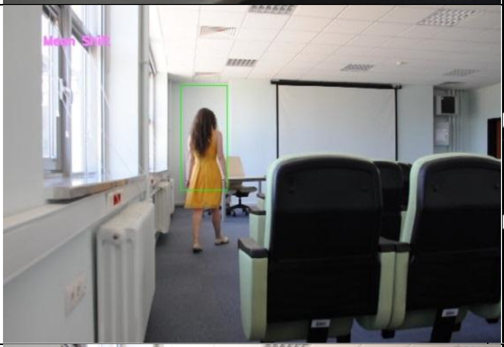
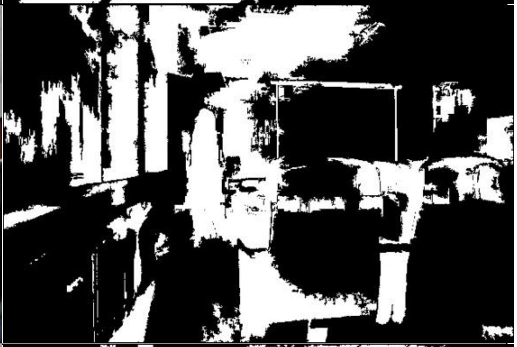
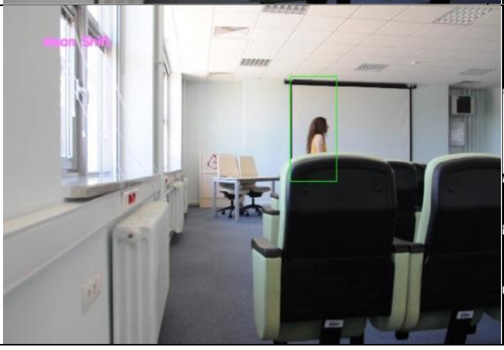
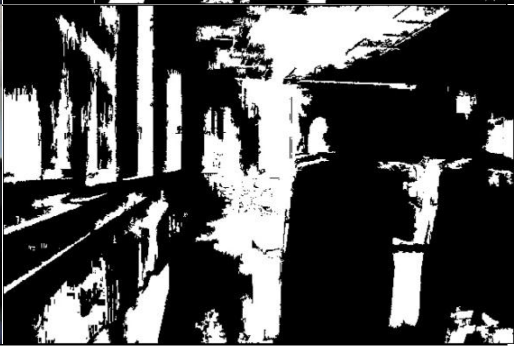
507		
660		
710		
876		
975		

Table 2 (cont.)

1089		
1317		
1536		
1610		

To show the comparison of the proposed system with standard correspondence based tracking and standard mean shift, precision, recall and f-score metrics are used.

To calculate precision and recall rates of the proposed system, first, ground truth is created by deleting noise by hand in foreground images. For video 8, from 449th to 550th frame, noise is cleaned. The reason to choose this frame range is it both includes the cases when the camera is not moving and the camera is moving. After cleaning the noise, because of using kernel based tracking, a kernel which includes the moving object is generated. Figure 13 (a) is a foreground image taken randomly while the camera is moving. Figure 13 (b) is the obtained ground truth.

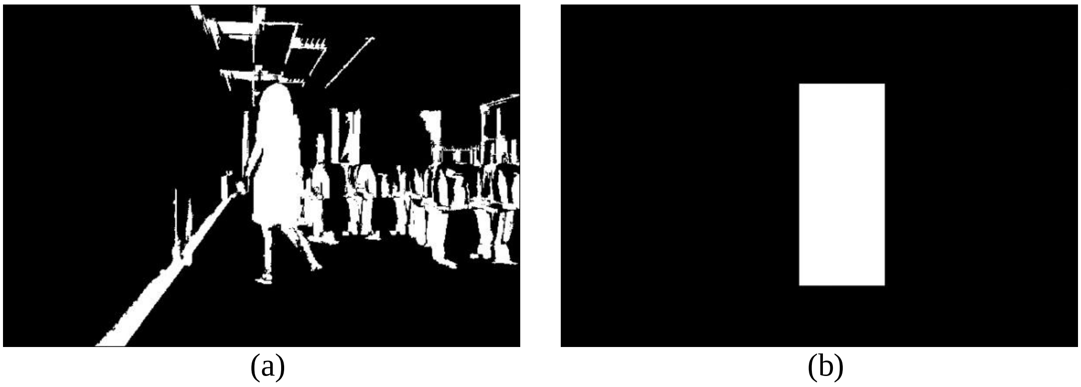


Figure 13. Foreground image while camera is moving (a) Foreground image with noise, (b) Ground truth image generated by cleaning the noise by hand

3 different tracking methods are compared to find the proposed system's accuracy. Like the ground truth image, 3 different frame sets are created for 3 methods. In these images, bounding boxes of moving objects are shown as white; other pixels are set as black. Because of reducing the bounding box size in standard mean shift and proposed system, we obtain smaller bounding boxes than the tracked object. Using the binary images created from these bounding boxes would cause obtaining wrong performance results. For pixel by pixel comparing, bounding boxes are returned to its original size. The important point here is, mean shift is performed by small bounding boxes in order to prevent shifting, but comparing with the ground truth is done by using the original size of the bounding boxes after obtaining the mean shift result. The created binary images for 3 different methods are given in Figure 14 (a), Figure 14 (b) and Figure 14 (c).

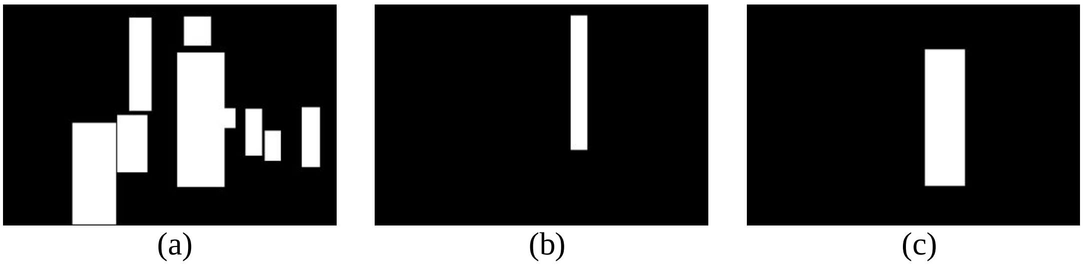


Figure 14. Test images (a) Generated bounding boxes from different methods at the same time (a) Bounding boxes from correspondence based tracking, (b) Bounding boxes from standard mean shift, (c) Bounding boxes from the proposed system

After obtaining ground truth and the bounding boxes, each image is compared with the ground truth image by using a voting system. O_r represents the original moving object and O_f represents found moving object by a tracking method. Equation (4.1) is applied to find true positive (TP), false positive (FP) and false negatives (FN). Precision is calculated with the Equation (4.1), recall is calculated with the Equation (4.2) and f-score is calculated with Equation (4.3).

For each frame we look at the pixels from the first pixel to the last. If a pixel on O_r is white and the corresponding pixel on $O_f$is white, TP is incremented by 1. If a pixel on O_r is white but the corresponding pixel on O_f is black, FP is incremented by 1. If a pixel on O_r is white, but the corresponding pixel on O_f is white, FN is incremented by 1. Thus, we calculate TP, FP and TN values for each frame.

$$precision = \frac{TP}{TP + FP} \tag{4.1}$$

$$recall = \frac{TP}{TP + FN} \tag{4.2}$$

$$fscore = 2.\frac{precision.recall}{precision + recall} \tag{4.3}$$

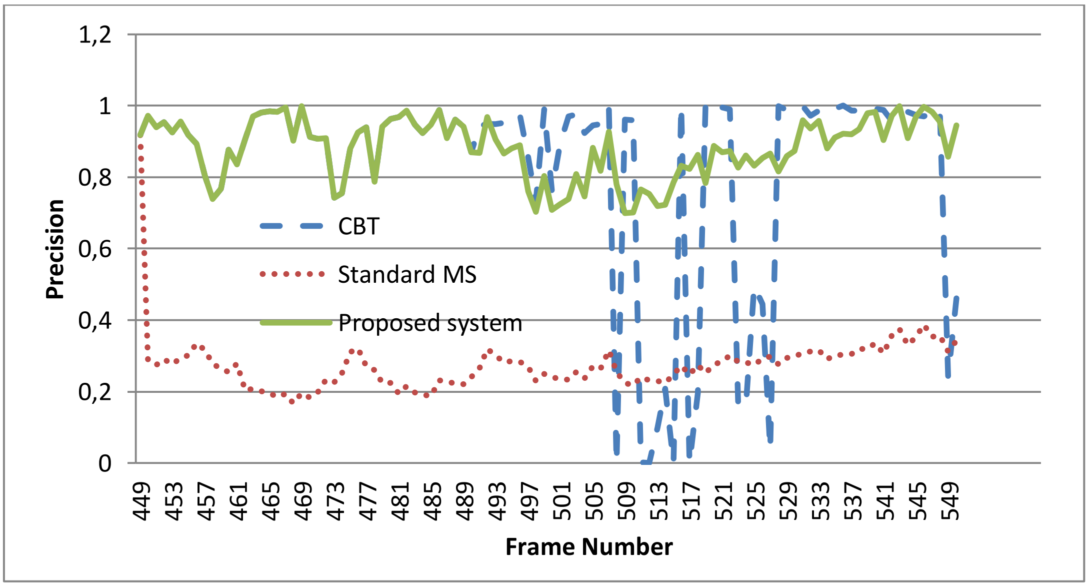


Figure 15. Precision graph of video 8

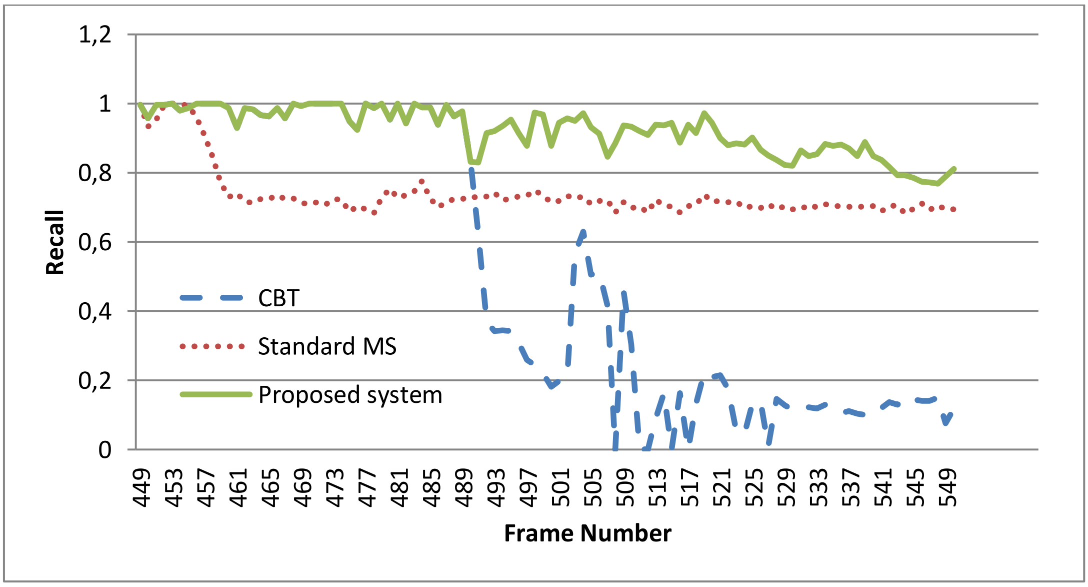


Figure 16. Recall graph of video 8

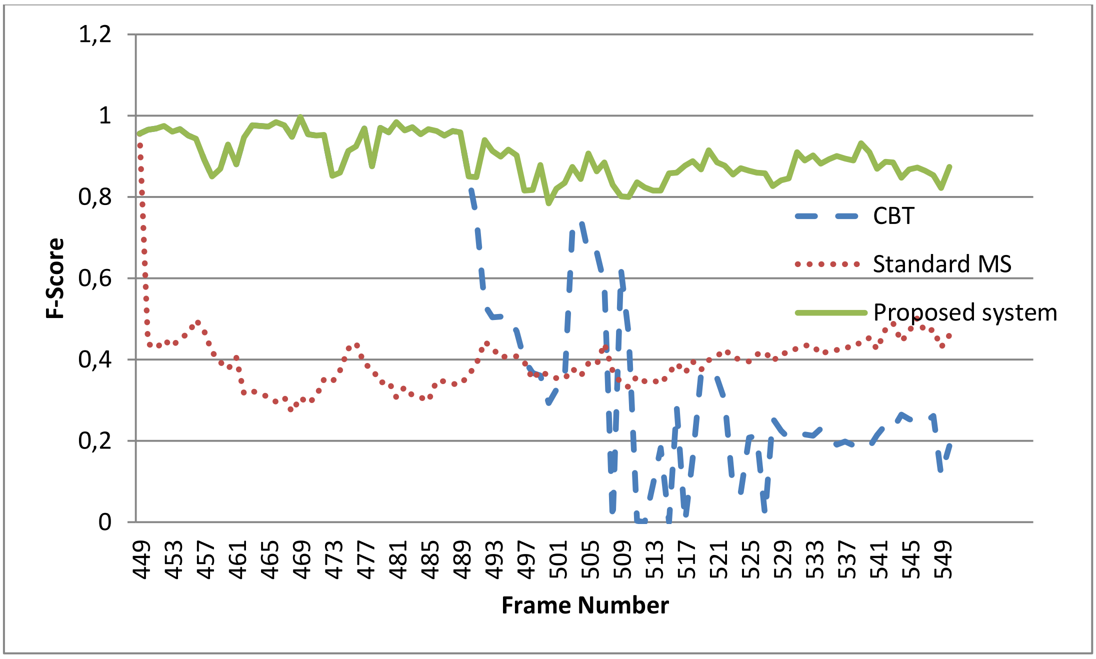
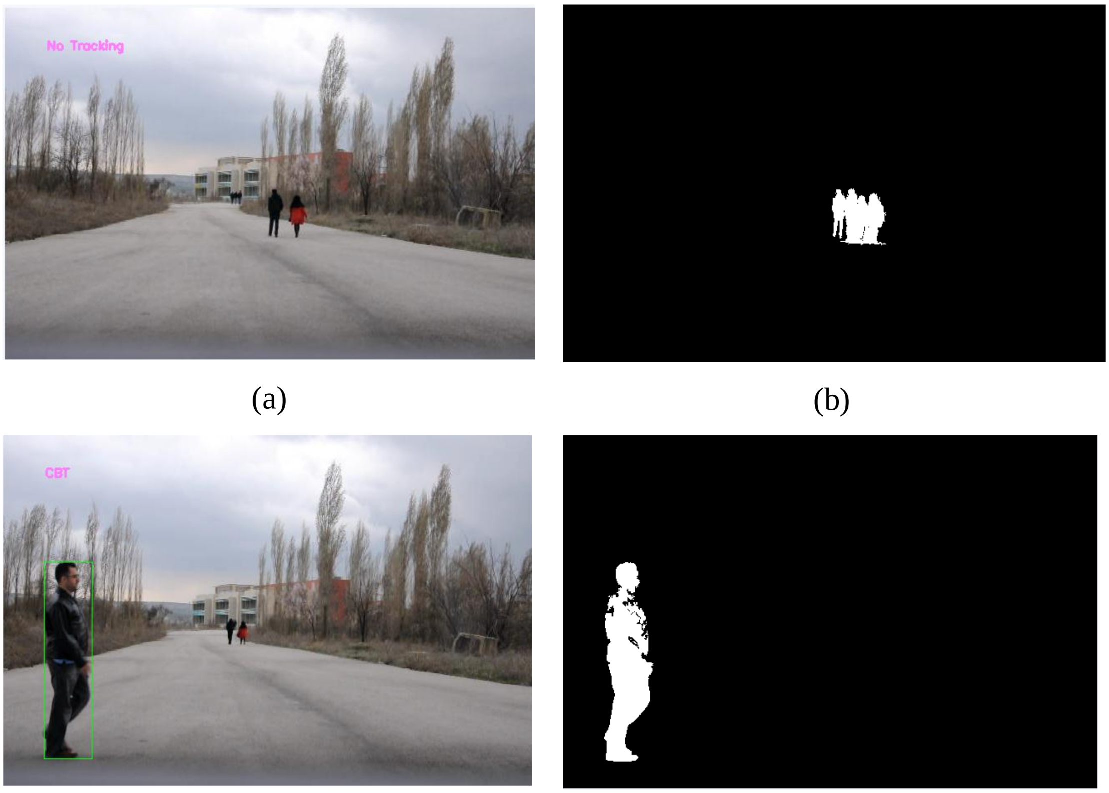


Figure 17. F-Score graph of video 8

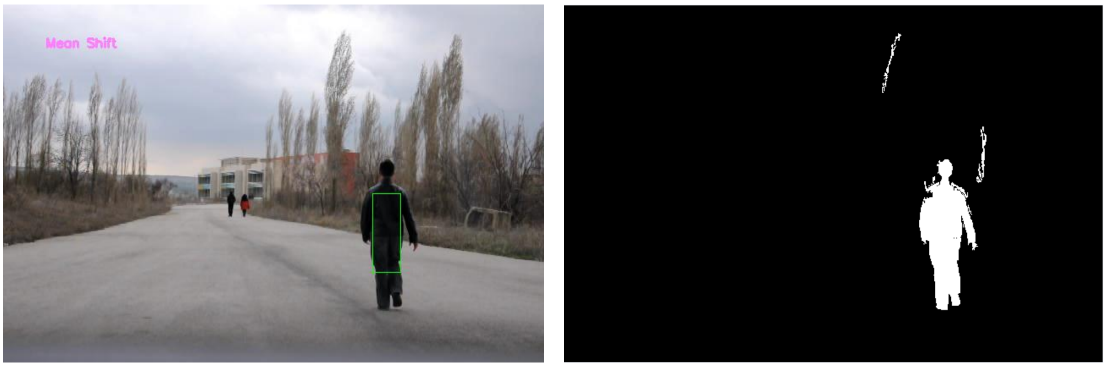
By looking at Figure 15, Figure 16 and Figure 17, it can be clearly said that the proposed method works the best among the tested algorithms. In the first frame, for standard mean shift tracking, objects are detected with background subtraction. From the precision graph, it can be seen that standard mean shift is not as good as correspondence based tracking when the camera is not moving. In recall graph we can see that correspondence based tracking performance drops significantly when the camera starts to move. F-Score graph is used to compute the accuracy of a system. It is the harmonic mean of precision and recall graphs.

According to the f-score graph in Figure 17, it can be seen that the proposed system has the same f-score values until the camera moves with the CBT as the proposed system uses the same algorithm with the CBT. Both CBT and proposed system have better f-score than mean shift until the camera moves. When the camera starts moving, CBT's f-score drops significantly and the proposed system has better f-score than both CBT and mean shift. It has to be noted that the bounding boxes generated by correspondence based tracking method change their size in each frame due to actions of the tracked object while standard mean shift uses a fixed kernel which may shift to background in time, reducing its performance in time. In long video sequences, the tracking kernel of mean shift's moving to background can be seen more clearly.



(a) (b)

(c) (d)



(e) (f)

Figure 18. Tracking result and foreground image for different frames of video 14. (a) RGB image when background subtraction is processing, (b) Foreground image when background subtraction is processing, (c) RGB image when camera is stationary, (d) Foreground image when camera is stationary, (e) RGB image when camera starts to move, (f) Foreground image when camera starts to move.

Precision, recall and f-score graphs for video 14 are given in Figure 19, Figure 20 and Figure 21. In precision graph, it can be seen that correspondence based tracking has good results but from the recall and f-score graph, we can see that when camera is moving, correspondence based tracking fails. The reason for this is, the generated bounding boxes by correspondence based tracking include the correct bounding boxes, but they also include additional false bounding boxes which lower the recall result. From the f-score graph we can see that correspondence based tracking fails again after camera starts to move and standard mean shift's f-score is lower than that of the proposed system.

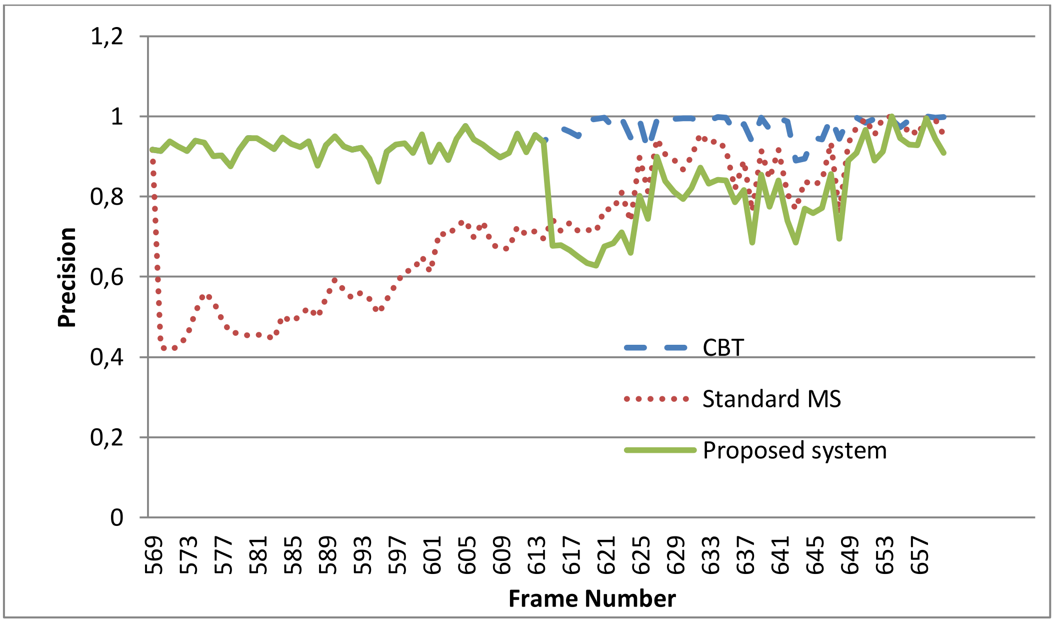


Figure 19. Precision graph of video 14

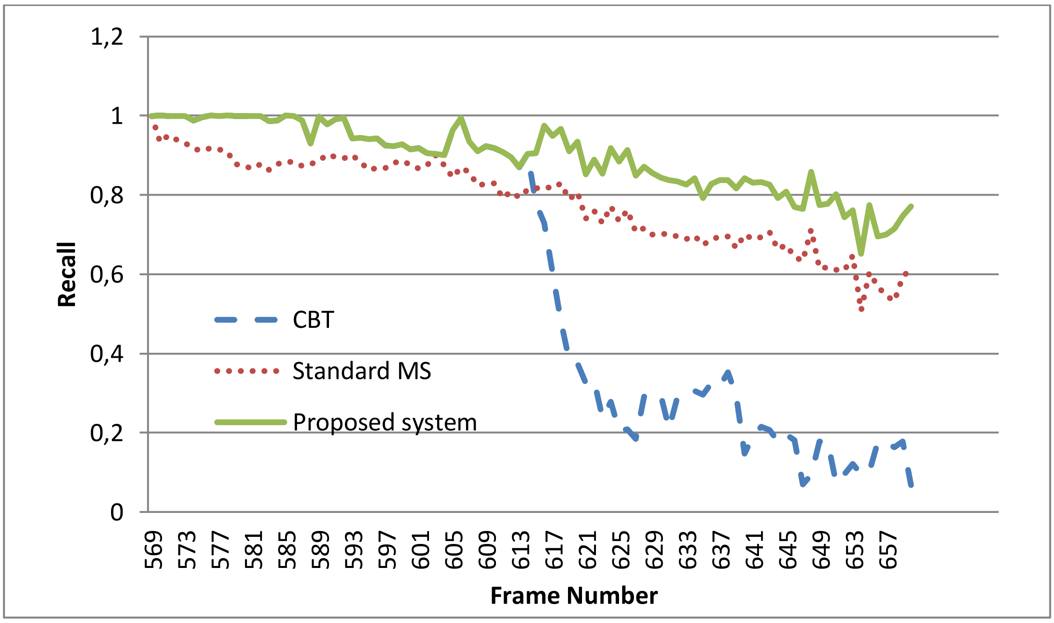


Figure 20. Recall graph of video 14

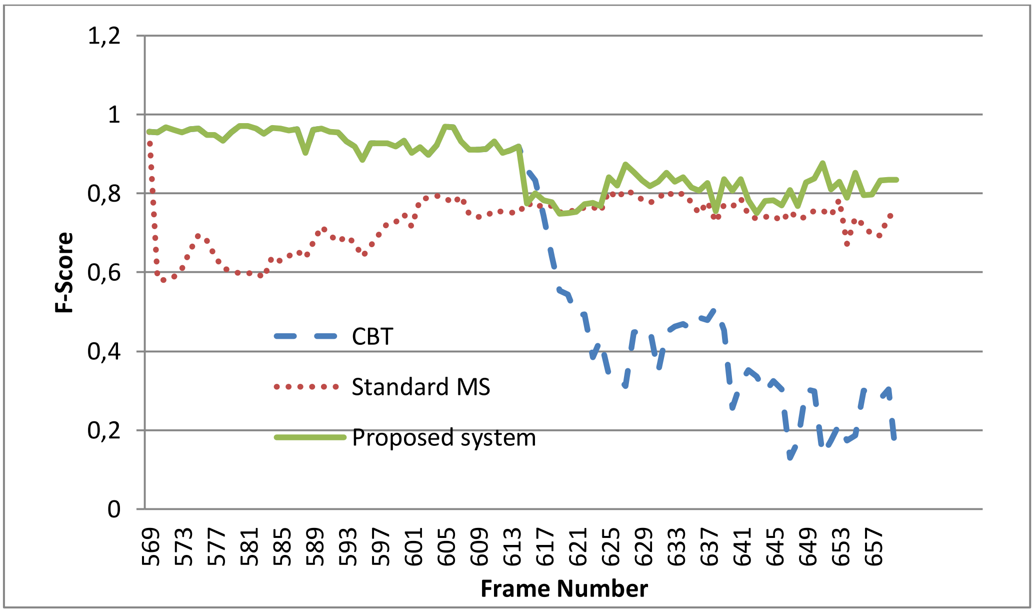


Figure 21. F-Score graph of video 14

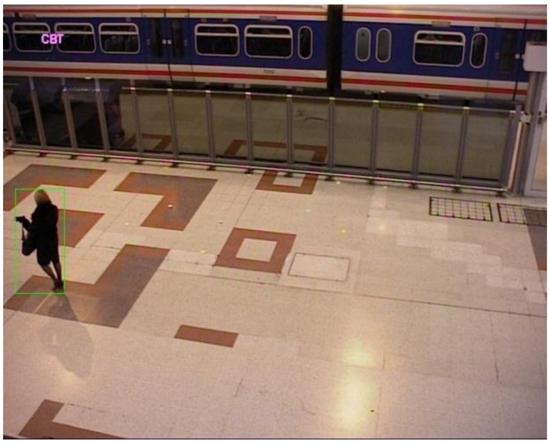
In order to observe if it switches to mean shift erroneously when the camera is not moving, three videos of PETS 2006 dataset [30] are used. These videos were prepared for tracking people with a stationary camera and left luggage detection. These videos are captured with a stationary camera and include numerous people passing by in an indoor area. All videos are captured in a train station.

Video #1 (Dataset S1 (T1 – C - 3)) : This video is captured from above. It includes moving people with luggage. In this video in some frames, system switches to mean shift mode incorrectly, because in the background there are many people walking near the train, which are small to detect, causing noise in the background. Due to this noise, system switches to mean shift. Sample images of incorrect mode change is given in Figure 22.

Video #2 (Dataset S5 (T1 – G - 1)) : This video is captured on people’s height. It includes people walking and waiting. Generally, people are going forwards or backwards. In this video, the test is completed without any incorrect switch to mean shift mode.

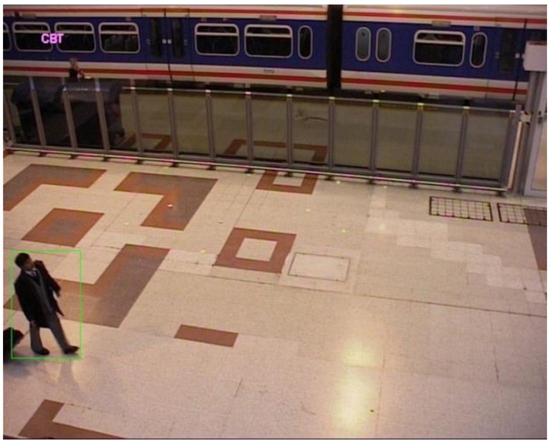
Video #3 (Dataset S3 (T7 – A - 3)) : This video is similar to the first one but does not include people in the background which are very far from the camera. In this video, no incorrect mode change is observed. Sample frames are given in Figure 23.

For video#3, the masked background update rates are shown in Figure 24. In the outlier detection phase, threshold for switching to mean shift is detected as 1648. As it can be seen from the Figure, there is no masked background update rate of a frame that is higher than this threshold.

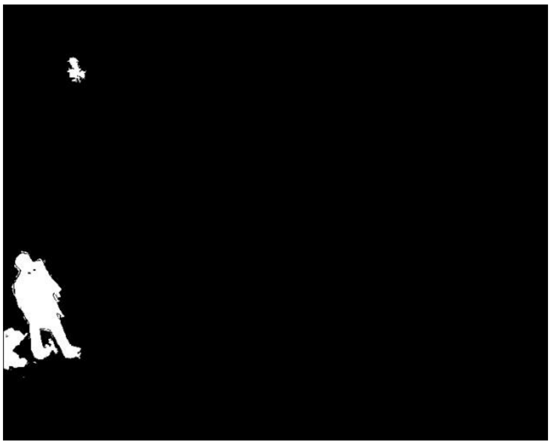


(a)

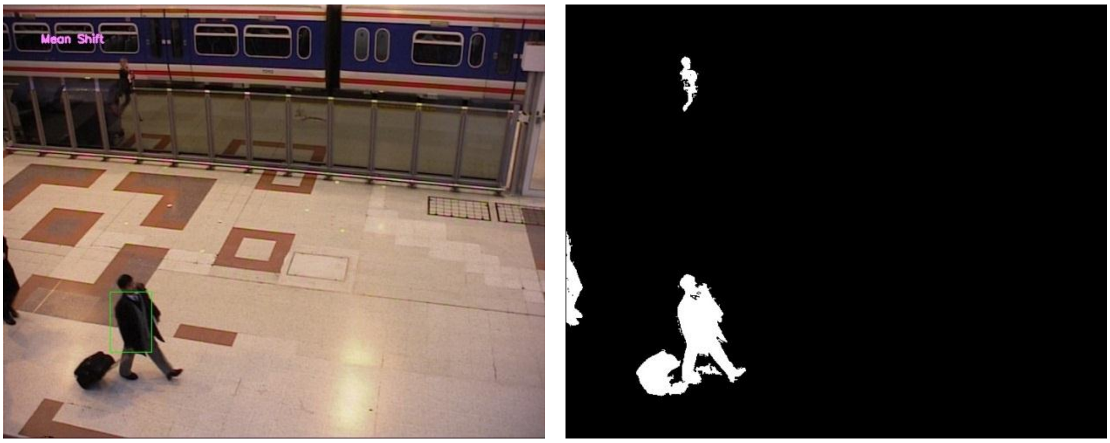
(b)



(c)

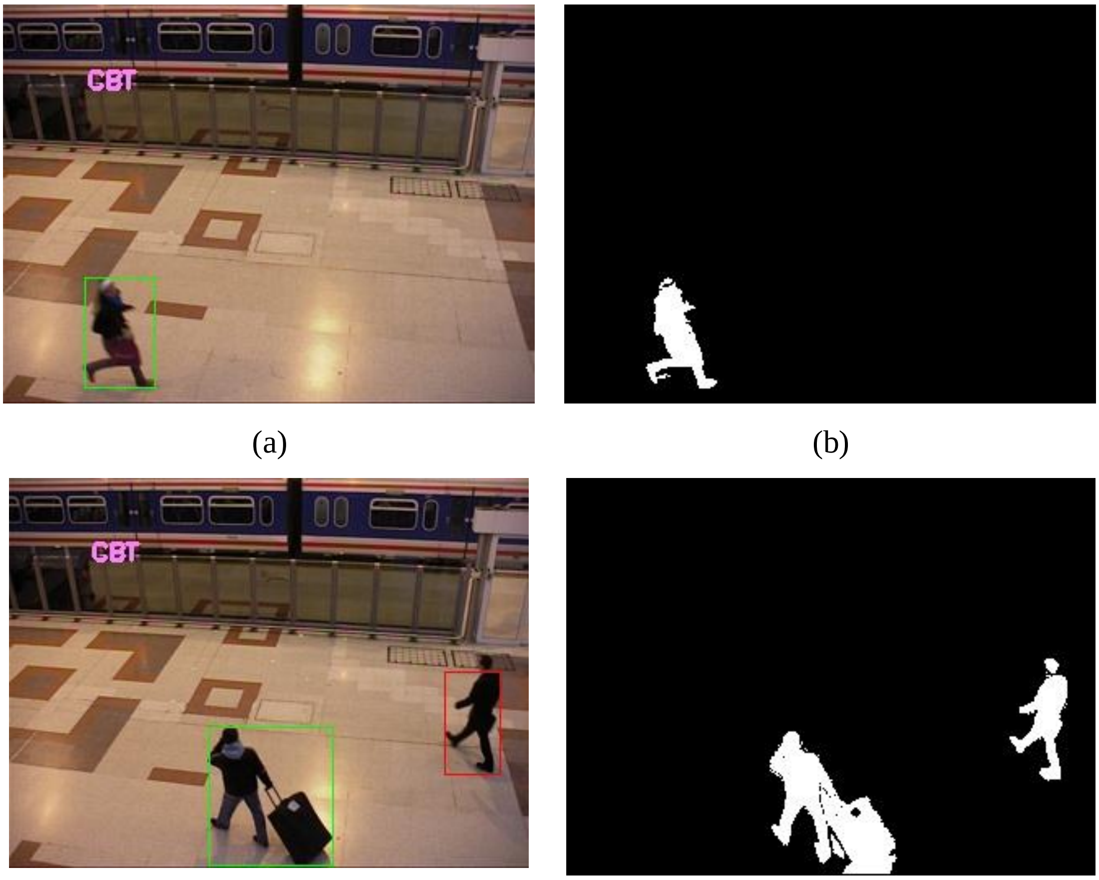


(d)



(e) (f)

Figure 22. Captures from video#1. (a) RBG image when camera is on CBT mode,(b) Foreground image when camera is on CBT mode, (c) RGB image when the person which cannot be detected because of its size appears (d) Foreground image when the person which cannot be detected because of its size appears, (e) RGB image when the system is on mean shift mode, (f) Foreground image when the system is on mean shift mode.



(a) (b)

(c) (d)

Figure 23. Some captures from video #3. The system does not switch to mean shift incorrectly and keeps tracking on CBT mode.

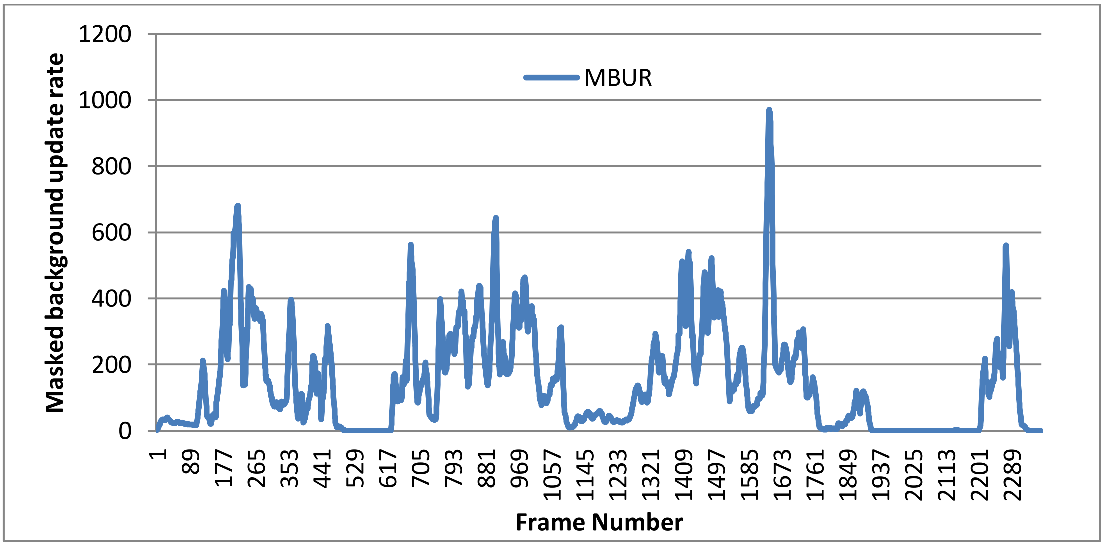


Figure 24. Masked background update rate of video 3

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this thesis, we proposed a hybrid tracking system that can perform continuous tracking on both stationary and moving camera modes. To detect new objects and perform tracking on stationary camera mode, correspondence based tracking (CBT) along with background subtraction is used. When the camera starts moving, background subtraction fails to provide satisfactory results. We propose a background update rate metric to measure the background information reliability and detect the camera movement automatically. In the moving camera mode, mean shift tracking is used. Mean shift is not affected from moving camera, because it deals with histogram information of the tracked object. But because of using a kernel which both includes background information and the tracked object information while tracking, after a while, mean shift might shift to the background. So whenever the camera is found to be stationary, we switch to CBT mode for more robust tracking.

According to the test results, the proposed system is able to combine the advantages of these different methods and provide more robust tracking compared to CBT and mean shift. While mean shift tracking require manual instantiation of the objects, the proposed system can detect the new objects when the camera is stationary similar to the CBT. On the other hand, the system cannot detect newly emerging objects in the scene in the moving camera mode.

As it can be seen from video 21, proposed system is able to switch back to stationary mode after camera stopped moving. This proves that our system is adaptive to the mode changes and handles these changes without problem.

In 5 videos out of a total of 21, while the system is able to detect the camera movement and switch to mean shift satisfactorily, mean shift kernel shifted to background and tracking failed after a while. As a future work, improved versions of mean shift or other tracking methods such as particle filter can be integrated into the system to increase system performance in the moving camera mode.

Tests performed with videos of PETS 2006 demonstrate that, in conditions that are explained in introduction section, our system prevents its stability in conditions that the camera is not moving and does not switch to mean shift incorrectly. In other conditions, such as moving objects being too far from the camera, system is not able to detect the moving objects and might confuse them as noise. As a future work, detecting objects that are too close or far from the camera can be studied and can be adapted to the system.

Because of sudden light changes may lead the system incorrect mode-change, detecting and handling the sudden light changes can be performed as future work.

REFERENCES

[1] Yilmaz, A., Javed, O., & Shah, M. (2006). Object tracking: A survey. *ACM Computing Surveys*, *38*(4). doi:10.1145/1177352.1177355

[2] Broida, T. J., & Chellappa, R. (1986). Estimation of object motion parameters from noisy images. *IEEE transactions on pattern analysis and machine intelligence*, *8*(1), 90–9.

[3] Ristic, B., Arulampalam, S., & Gordon, N. (2004). *Beyond the Kalman filter: particle filters for tracking applications*. Artech House.

[4] Fieguth, P., & Terzopoulos, D. (1997). Color-based tracking of heads and other mobile objects at video frame rates. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 21–27. doi:10.1109/CVPR.1997.609292

[5] Comaniciu, D. (2003). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(5), 564–577.

[6] Dedeoglu, Y. (2004). *Moving Object Detection, Tracking and Classification for Smart Video Surveillance*. Bilkent University, Department of Computer Engineering, Turkey.

[7] Colombari, A., Fusiello, A., & Murino, V. (2007). Segmentation and tracking of multiple video objects. *Pattern Recognition*, *40*(4), 1307–1317. doi:10.1016/j.patcog.2006.07.008

[8] Liu, Z., Shen, H., Feng, G., & Hu, D. (2012). Tracking objects using shape context matching. *Neurocomputing*, *83*, 47–55. doi:10.1016/j.neucom.2011.11.012

[9] Haritaoglu, I., Harwood, D., & Davis, L. S. (1998). W4: A Real Time System for Detecting and Tracking People. *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231)*, 962–962. doi:10.1109/CVPR.1998.698720

[10] Beyan, C., & Temizel, A. (2011). Adaptive Mean-Shift for Automated Multi Object Tracking. *IET Computer Vision*, *6(1)*, 1-12.

[11] Avidan, S. (2004). Support vector tracking. *IEEE transactions on pattern analysis and machine intelligence*, *26*(8), 1064–72. doi:10.1109/TPAMI.2004.53

[12] Mech, R., & Wollborn, M. (1998). A noise robust method for 2D shape estimation of moving objects in video sequences considering a moving camera. *Signal Processing*, *66*(2), 203–217. doi:10.1016/S0165-1684(98)00006-1

[13] Cucchiara, R., Prati, A., & Vezzani, R. (2004). Real-time motion segmentation from moving cameras. *Real-Time Imaging*, *10*(3), 127–143. doi:10.1016/j.rti.2004.03.002

[14] Daniilidis, K., Krauss, C., Hansen, M., & Sommer, G. (1998). Real-Time Tracking of Moving Objects with an Active Camera. *Real-Time Imaging*, *4*(1), 3–20. doi:10.1006/rtim.1996.0060

[15] Yilmaz, A., & Shah, M. (2006). Matching actions in presence of camera motion. *Computer Vision and Image Understanding*, *104*(2-3), 221–231. doi:10.1016/j.cviu.2006.07.012

[16] Odobez, J. M., & Bouthemy, P. (1994). Detection of multiple moving objects using multiscale MRF with camera motion compensation. *Image Processing*, *2*, 257-261.

[17] Murray, D., & Basu, A. (1994). Motion tracking with an active camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *16*(5), 449–459. doi:10.1109/34.291452

[18] Rosenberg, Y., & Werman, M. (1998). Object Tracking from a Moving Video Camera: A Software Approach. *Applications of Computer Vision, 1998. WACV '98. Proceedings., Fourth IEEE Workshop*, 238–239.

[19] Kalarot, R., & Morris, J. (2009). Real-time correlogram tracking for airborne traffic surveillance. *2009 24th International Conference Image and Vision Computing New Zealand*, 18–22. doi:10.1109/IVCNZ.2009.5378414

[20] Piccardi, M. (2004). Background subtraction techniques: a review. *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, 3099–3104. doi:10.1109/ICSMC.2004.1400815

[21] Wren, C., Azarbayejani, A., Darrell, T., & Pentland, A. (1996). Pfinder: real-time tracking of the human body. *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, 51–56. doi:10.1109/AFGR.1996.557243

[22] Cucchiara, R., Grana, C., Piccardi, M., & Prati, A. (2003). Detecting objects, shadows and ghosts in video streams by exploiting color and motion information. *Proceedings 11th International Conference on Image Analysis and Processing*, *25*(10), 1337–1342. doi:10.1109/ICIAP.2001.957036

[23] Collins, R., Lipton, A., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, N., Hasegawa, O., Burt, P., Wixson, L. (2000). *A System for Video Surveillance and Monitoring: VSAM Final Report*.

[24] Kyungnam K., Chalidabhongse, T.H., Harwood, D., Davis, L. (2004). Background modeling and subtraction by codebook construction, *Image Processing, 2004. ICIP '04. 2004 International Conference* , 5, 3061-3064.

[25] Stauffer, C., & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, 246-252.

[26] Elgammal, A., Duraiswami, R., Harwood, D., & Davis, L. S. (2002). Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE*, *90*(7), 1151–1163. doi:10.1109/JPROC.2002.801448

[27] Barnich, O. (2009). ViBe: a powerful random technique to estimate the background in video sequences. *IEEE International Conference on Acoustics, Speech and Signal Processing* , 945–948.

[28] Zivkovic, Z. (2004). Improved adaptive Gaussian mixture model for background subtraction. *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, 28–31.

[29] Glassner, A. (2001). Fill 'Er Up! *IEEE Computer Graphics and Applications*, *21*(1), 78–85.

[30] Performance Evaluation of Tracking and Surveillance (PETS) 2006 Benchmark Data, http://www.cvg.rdg.ac.uk/PETS2006/data.html.