SALIENCY-BASED VISUAL TRACKING USING CORRELATION FILTERS FOR SURVEILLANCE APPLICATIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

EMRE TUNALI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2014

Approval of the thesis:

SALIENCY-BASED VISUAL TRACKING USING CORRELATION FILTERS FOR SURVEILLANCE APPLICATIONS

submitted by **EMRE TUNALI** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Gönül Turhan Sayan
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. A. Aydın Alatan
Supervisor, **Electrical and Electronics Engineering Dept., METU** _____

Examining Committee Members:

Prof. Dr. Gözde Bozdağı Akar
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. A. Aydın Alatan
Electrical and Electronics Engineering Dept., METU _____

Assoc. Prof. Dr. Umut Orguner
Electrical and Electronics Engineering Dept., METU _____

Assist. Prof. Dr. Fatih Kamışlı
Electrical and Electronics Engineering Dept., METU _____

Assist. Prof. Dr. Sinan Kalkan
Computer Engineering Dept., METU _____

Date: _____

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: EMRE TUNALI

Signature :

ABSTRACT

SALIENCY-BASED VISUAL TRACKING USING CORRELATION FILTERS FOR SURVEILLANCE APPLICATIONS

TUNALI, Emre
M.S., Department of Electrical and Electronics Engineering
Supervisor : Prof. Dr. A. Aydın Alatan

August 2014, 119 pages

In recent years intelligent transportation systems (ITS) have been an active research area in computer vision. One of the main goals of ITS is producing systems to guide surveillance operators and reduce human resources for observing hundreds of cameras in urban traffic surveillance. Thus, this thesis is devoted to realization of low level tasks, target detection and tracking, for an autonomous video surveillance system. The initial step of the proposed system is moving object detection which is utilized based on a recently proposed Self Adaptive Gaussian Mixture Model technique. The resulting moving object mask is further enhanced via post processing steps containing morphological operations and shadow/highlight removal. Benefiting from this enhanced binary mask, track initialization is achieved for each detected moving blob entering to the scene and a track is intended to be maintained until the target leaves the scene. For target tracking, multiple model visual tracking methodology is proposed based on correlation filters. Moreover, in order to adjust tracking parameters online and provide high level tasks with extra information together, a target bounding

box generation methodology which is capable of target silhouette extraction is proposed based on temporal consistency of the saliency map of tracking window. The proposed algorithm is tested on synthetic as well as real data and based on these experimental results, it can be concluded that it yields competitive tracking results in real life scenarios.

Keywords: Moving object detection, object tracking, correlation filters, saliency

ÖZ

GÖZETLEME UYGULAMALARİ İÇİN BENZEŞİM FİLTRELERİ KULLANARAK BELİRGİNLİK TABANLI GÖRSEL TAKİP

TUNALI, Emre
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. A. Aydın Alatan

Ağustos 2014 , 119 sayfa

Son yıllarda akıllı ulaşım sistemleri görüntü işleme alanında aktif bir araştırma alanı olmuştur. Bu çalışmaların temel amaçlarından biri operatörleri yönlendirecek sistemler oluşturarak kent trafiğini gözetlemede kullanılan yüzlerce kameranın incelenmesinde gereken insan iş gücünün azaltılmasıdır. Buradan hareketle, bu tez otonom video gözetleme sistemi oluşturmak için gerekli bazal işlemler olan hedef tespiti ve takibinin gerçekleştirilmesi amacını gütmektedir. Önerilen sistemin ilk adımını yakın geçmişte ortaya atılan Özuyarlamalı Gauss Karışım Modeli temel alınarak gerçeklenen hareketli objelerin tespiti oluşturmaktadır. Elde edilen hareketli obje maskesi morfolojik işlemler ile gölge/yansıma arındırma işlemlerini içeren ardişlemler uygulanarak daha kaliteli hale getirilmiştir. Bu işlemler sonucu elde edilmiş olan ikili imgeden yararlanılarak görüntüde tespit edilmiş her hareketli hedef için takip başlatılmakta ve hedef görüntüden çıkana kadar takibin devamlılığı sağlanmaya çalışılmaktadır. Hedefin takibi için önerilen metodoloji, benzeşim filtrelerinden temellendirilmiş çoklu görsel modelidir.

Buna ek olarak takip parametrelerini işlem sırasında ayarlamak ve daha karmaşık işlemlerde ihtiyaç duyulan ekstra bilgileri sağlamak amacıyla; hedefin siluetini belirginlik haritalarının zamansal tutarlılığının kontrolü ile çıkartan bir hedef sınırlama kutusu üretim metodu da önerilmektedir. Önerilen sistem sentetik ve gerçek data ile test edilerek gerçek hayat problemlerinin çözümündeki başarısı da tez kapsamında gösterilmiştir.

Anahtar Kelimeler: Hareketli obje tespiti, hedef takibi, benzeşim filtreleri, belirginlik

To my family...

ACKNOWLEDGMENTS

I would like to express my deep appreciation to my supervisor Prof. Dr. Aydın Alatan for his guidance, encouragement and for his efforts on providing a stimulating research environment.

I feel also indebted to my co-workers in ASELSAN, Sinan Öz for his advices, criticism and encouragement; and Berk Ülker for helping out with CUDA.

I also express my gratitude to TÜBİTAK BİDEB "National Scholarship Program for MSc Students".

I would like to deeply thank to my family; my father *(Yüksel TUNALI)*, my mother *(Hülya TUNALI)*, my sister *(Elıf Tunalı ÇALIŞKAN)* for their emotional support and belief in me. I also want to thank to my future wife *(Türkü ÇOBANOĞLU)* for her love, support and patience. Finally, I want to express my gratitude to my grandparents *(M. Ihsan KURUCU and Şevkiye KURUCU)*, who have put us in the center of their lives for more than 15 years.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ACE	Average Correlation Energy
ASEF	Average of Synthetic Exact Filter
ASM	Average Similarity Measure
BD	Brightness Distortion
CD	Chromacity Distortion
CF	Correlation Filters
CHF	Circular Harmonic Functions
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
ECPSDF	Equal Correlation Peak Synthetic Discriminant Function
FFT	Fast Fourier Transform
FPR	False Positive Rate
GMACE	Gaussian Minimum Average Correlation Energy
GMM	Gaussian Mixture Model
GPU	Graphics Processing Unit
IR	Infrared
ITS	Intelligent Transportation Systems
JPDAF	Joint Probability Density Association Filters
LR	Large Region
MACE	Minimum Average Correlation Energy
MACE-MRH	Minimum Average Correlation Energy Mellin Radial Harmonic Function
MACH	Maximum Average Correlation Height
MHI	Motion History Images
MHT	Multiple Hypothesis Tracking
MINACE	Minimum Noise and Correlation Energy
MOSSE	Minimum Output Sum of Squared Error
MRHF	Mellin Radial Harmonic Functions
MSE	Mean Square Error
MSESDF	Minimum Squared Error Synthetic Discriminant Function
MVSDF	Minimum Variance Synthetic Discriminant Function
NCC	Normalized Cross Correlation

OCOF	Optimized Correlation Output Filters
ONV	Output Noise Variance
OTCHF	Optimal Trade-off Circular Harmonic Function
OTSDF	Optimal Tradeoff Synthetic Discriminant Function
PCF	Polynomial Correlation Filters
QCF	Quadratic Correlation Filter
PCA	Principal Component Analysis
PCE	Peak-to-Correlation Energy
PSR	Peak-to-Sidelobe Ratio
PTZ	Pan-Tilt Zoom
SAGMM	Self-Adaptive GMM
SR	Small Region
TPR	True Positive Rate
UMACE	Unconstrained Minimum Average Correlation Energy
UMSESDF	Unconstrained Minimum Squared Error Synthetic Discriminant Function
ZHGMM	Zivkovic and Heijden's GMM

CHAPTER 1

INTRODUCTION

"Science and technology multiply around us. To an increasing extent they dictate the languages in which we speak and think. Either we use those languages, or we remain mute."

-J. G. Ballard

1.1 Intelligent Video Analytics for Traffic Management

The significant increase in the number of vehicles in the traffic emerged the necessity of detailed analysis on traffic flow to manage and improve the traffic system. To estimate some critical traffic parameters, intelligent transportation systems (ITS) have been designed and become a centralized approach in traffic systems. In general, ITS cover any technology designed to provide innovative services to transportation systems keeping the traffic flow in control in order to make transport more sustainable, which means efficient, clean, safe; and acquire relevant information for future planning. Moreover, these systems can also be used for homeland security applications by observing every action of a suspicious vehicle throughout the ITS network.

The older generations of ITS were using sensors that were expensive and requiring destructive installation such as inductive loop detectors which needed to be placed under the roadbeds to detect vehicles passing through. These detectors can only identify the existence of vehicles with low speeds. Moreover, the

information obtained was also limited to the number of vehicles in unit time, weights and lengths of the vehicles. The developments in sensor technology enabled usage of vision based monitoring systems for traffic applications and they become very popular in ITS applications. The main advantage can be listed as: low setup costs, nondestructive installation, and rich information flow for human understanding. However, until the last decade, the workload was mainly weigh upon the human operators. The human operators had to observe hundreds of cameras 7 days 24 hours to achieve traffic analysis which was extremely burdensome. At this point, the increased computing power together with the advancement of analytical techniques to process the video formed a basis and many algorithms have been proposed in order to aid human operators. The proposed automatic pre-processing techniques not only allow efficient guidance for operators on selecting the relevant camera to observe but also enable them to intervene the traffic flow more efficiently in real time.

Figure 1.1: ITS Operator should be Offload from Observing Hundreds of Cameras Simultaneously

Traffic monitoring systems for urban areas have several tasks which can be solved by computer vision techniques including congestion detection, vehicle count, lane change detection, average traffic flow estimation, detection of traffic violations such as illegal turn and parking to a non-parking zone, vehicle classification,

accident or emergency detection. The information obtained from these kinds of applications is not only used for simultaneous interventions on traffic flow but also analyzed for planning new routes for transportation network and making important decisions for planning of the urbanization in future. One should note that all these important tasks require the same low-level tasks as a preliminary which is automatic target detection and tracking. Therefore, in this thesis work, a solution is proposed for the fulfillment of these low level tasks, which provides higher level tasks with valuable information.

For the solution, the imaging system plays an important factor. Although usages of different types of visual sensors are possible, i.e. CCTV, fish-eye, IR etc., nearly all of the ITS applications retrieves the scene from the cameras mounted on poles or high grounds above the roads as illustrated in Fig. 1.2. Therefore, in the proposed solution the cameras are assumed to be stationary (at the home position) until the operator changes their position. Although the proposed solution can respond to different image resolutions and object sizes effectively, installation of the pole can still drastically change the performance; since the location of the pole or its height may result in increased number of occlusions. In this sense, pole implementation should be accomplished on a per case basis to achieve better performance.

1.2 Scope of the Thesis

This thesis focuses on developing main building blocks of an automated ITS system to address the main bottleneck of these systems which is the limited number of operators. Although the ultimate goal is the tracking of the vehicles passing by the scene, to achieve full autonomy target detection stage is also a must and provided in the solution. By the nature of the problem, the targets are considered as moving vehicles; hence the tracks are initialized from the output mask of moving object detection mechanism. For moving object detection various algorithms are utilized from the literature and their outputs are enhanced with post-processing steps consisting of morphological operations and shadow/highlight removal strategies proposed in the literature. All these

Figure 1.2: An Exemplary Installation of Cameras for an ITS Application

moving object detection methods are compared both in theory and practice to reveal the best possible option for the overall system.

Target tracking in urban scenes contains different challenges than highway monitoring including various maneuvering angles for the vehicles moving in different lanes and different sections of junction points, high probability of partial occlusions and distractive patterns such as similar targets in close neighborhood, random motion patterns due to frequent stop-and-go activities. In this manner, the proposed tracker should be capable of tracking without any motion model assumption, handle partial occlusions effectively and not be effected by similar targets in the vicinity. Considering the attractive properties of the correlation filters, which are explained in the background section; for target tracking, a multiple model visual tracking methodology is proposed based on a recent example of correlation filters [1]. Moreover, the proposed filter is also capable of target size estimation by silhouette extraction of the target which compensates a general weakness of the correlation filters and provides higher levels of the ITS system with valuable information.

The performance of the proposed system is also validated by the experiments which are conducted both in subsystem and system level. To enhence our observations on the proposed solution experiments included usage of synthetic data as well as real life data.

1.3 Outline of the Thesis

In Chapter 2, both moving object detection and object tracking taxonomies are presented and explained in brief for each proposed system. Some methodologies that are suitable for the usage in the proposed solution are specified and further analyzed to determine their possible advantages and disadvantages with respect to each other.

The Chapter 3 is dedicated for the detailed explanations of the proposed solution both in subsystem level and including their collaborative working strategy.

After the explanation of the algorithm, the experiments are conducted to analyze performance of the both subsystems and the integrated solution. The integrated solution is executed on a real world problem and the results are presented in Chapter 4.

Finally, the thesis is concluded in Chapter 5, by summarizing the work accomplished and presenting the obtained observations. Chapter 5 also contains suggestions on some future extensions.

CHAPTER 2

BACKGROUND REVIEW

2.1 Moving Object Detection

Moving object detection is defined as identifying pixels belonging to moving objects in a frame and considered to be the first stage of many visual surveillance systems; since detecting moving blobs provides focus of attention for high level tasks such as tracking, recognition, classification and activity analysis in which only "moving" pixels need to be considered. Although human being is well adapted for moving object detection, it is still a challenging task as a computer vision application. A robust moving object detection algorithm should overcome the common problems that are encountered in the literature such as gradual/sudden illumination changes (including automatic gain control of camera), moving backgrounds (swaying trees), slow moving objects, clutter and other types of noise.

In order to achieve this task, diverse set of algorithms exist in literature [2–4] can be categorized into three groups as: temporal differencing [5–10]; background subtraction [11–27]; and optical flow [28–30]. In this section, different solutions from each category of the conventional moving object detection methodologies are discussed. This discussion includes a rough comparison for the strengths, weaknesses and computational complexities of the conventional methods. In the light of this discussion, possible algorithms are selected by their suitability of the overall system for further analysis. Their detailed comparison considering to both their performance and execution time is achieved in Chapter 4.

Figure 2.1: A Moving Object Detection Example with the Resultant Binary Mask (Red)

2.1.1 Conventional Methods for Moving Object Detection

Examining the literature, it is seen that moving object detection can be mainly divided to three subcategories as temporal differencing, background subtraction, and optical flow. Each subcategory has its own weaknesses and strengths and explained in the following section.

2.1.1.1 Temporal Differencing

Temporal differencing aims to detect moving objects by using pixel-wise difference of frames in video imagery. Although, temporal frame difference methods can easily detect the motion, they generally show poor performance of localizing the objects. An exemplary situation is illustrated in Fig. 2.2. If object is moving fast or temporal distance of two differencing frames is large, two objects are detected; one where the object is and one where it used to be,Fig. 2.2a. On the contrary, if object is moving slowly or the temporal distance between frames is low, only a part of the object could be detected,Fig. 2.2b. Moreover, when the moving object ceases moving, most of temporal differencing method fails detection just in a few frames.

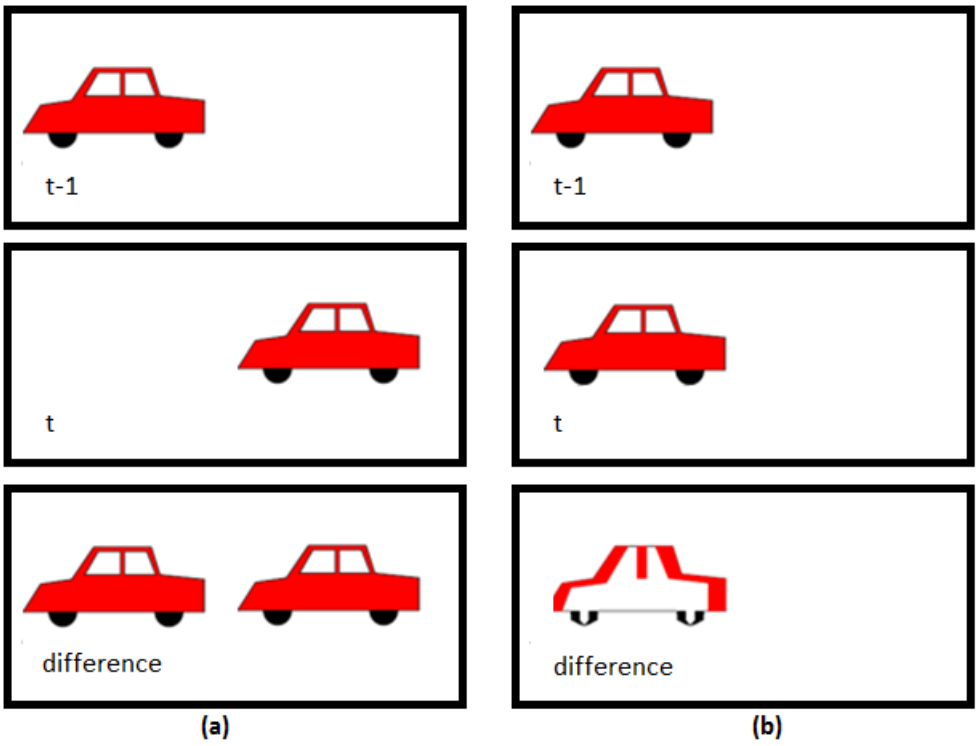

Figure 2.2: (a) Double Detection due to Fast Motion or High Temporal Distance (b) Incomplete Detection of Moving Object due to Slow Motion or Low Temporal Distance

In this part, 4 different temporal differencing methodologies are explained which try to overcome the problems mentioned above in addition to the naive temporal differencing approach.

2.1.1.1.1 Naive Approach

The simplest and the fastest method of temporal differencing is the usage of only two consecutive frames which simply means that background model is considered to be the previous frame as presented in the work of Lipton et al. [5]. Eqn. 2.1 shows the formal definition of the naive approach, where $I_t(x,y)$ corresponds to intensity value of the current frame at position (x,y) and $I_{t-1}(x,y)$ represents the intensity value for the same pixel for the previous frame.

$$Mask_t(x,y) = \begin{cases} 1, & \text{if } |I_t(x,y) - I_{t-1}(x,y)| > \tau \\ 0, & \text{else} \end{cases} \tag{2.1}$$

Since this method generally produces incomplete information, many moving

pixels can be missing; about the moving object; it should be supported with other algorithms to achieve complete information.

2.1.1.1.2 Inter-frame Difference

If the slow moving objects are the interest, the naive approach may fail to detect existence of moving object by comparing two consecutive frames. However, in such scenarios it is still possible to detect moving object by using frames having temporal distance more than one frame. Therefore, the simplest inter-frame difference algorithm is nothing but the modified version of naive algorithm to use frames with N frame distant to each other and given in the Eqn. 2.2

$$Mask_{\mathrm{t}}(x,y) = \begin{cases} 1, & \text{if } |I_{\mathrm{t}}(x,y) - I_{\text{t-N}}(x,y)| > \tau \\ 0, & \text{else} \end{cases} \tag{2.2}$$

The choice of temporal distance between frames should be selected according to expected size and speed of the moving object. Since most of the time scene includes objects with different speeds and sizes, this methodology is generally used to detect motion not for locating the moving object which means it should be supported with another algorithm for object localization as in [5].

2.1.1.1.3 Three Frame Difference

The importance of the three frame difference is the notion behind the algorithm. This algorithm simply states that the moving object should preserve its motion in consecutive frames. Therefore, three-frame differencing rule states that a pixel is legitimately moving if a significant intensity change is observed not only in this frame but also in the previous frame and formulated in [7] as in Eqn. 2.3:

$$Mask_{\mathrm{t}}(x,y) = \begin{cases} 1, & \text{if } |I_{\mathrm{t}}(x,y) - I_{\text{t-1}}(x,y)| > \tau \ AND \ |I_{\text{t-1}}(x,y) - I_{\text{t-2}}(x,y)| > \tau \\ 0, & \text{else} \end{cases} \tag{2.3}$$

Obviously, the frames that are used in three frame difference methodology can be changed to be used as inter-frame difference. Since this algorithm contains information from more than one frame, it shows better performance than the naive approach as expected.

2.1.1.1.4 Motion History Images

Motion history images (MHI) also based on the same notion used in the three-frame difference algorithm and uses the combinations of the object movements from different frames and shows cumulative object motion [8–10]. MHI has two main differences than three frame algorithm: First, it does not limit itself to three frames from past but makes use of L frames from both future and past; and the second while using more than one frames it gives less weight to the motion detected from frames with high temporal distance to the frame in process by using a decay term. Although, usage of future frames introduces delay to system, this algorithm shows the best object localization performance among all mentioned temporal differencing methodologies. Considering the overall system, better target localization is something more desirable than a system without any delays. Although more computational load is required than previous methodologies in this category, its workload introduces still less computational burden than methodologies from other two categories.

Since MHI is selected to be a possible option for the overall system, further explanation is given below. The MHI proposed in [10] consists of 3 sub-steps namely: preprocessing, MHI generation and object localization.

2.1.1.1.4.1 Preprocessing

This module is implemented to solve two important problems: image stabilization, and rapid change in pixel intensities. Since the proposed system is designed for stationary cameras, the stabilization part is not in the interest. However, the solution of the second problem actually corresponds to making MHI robust to abrupt changes which may occur due to sudden illumination changes, automatic

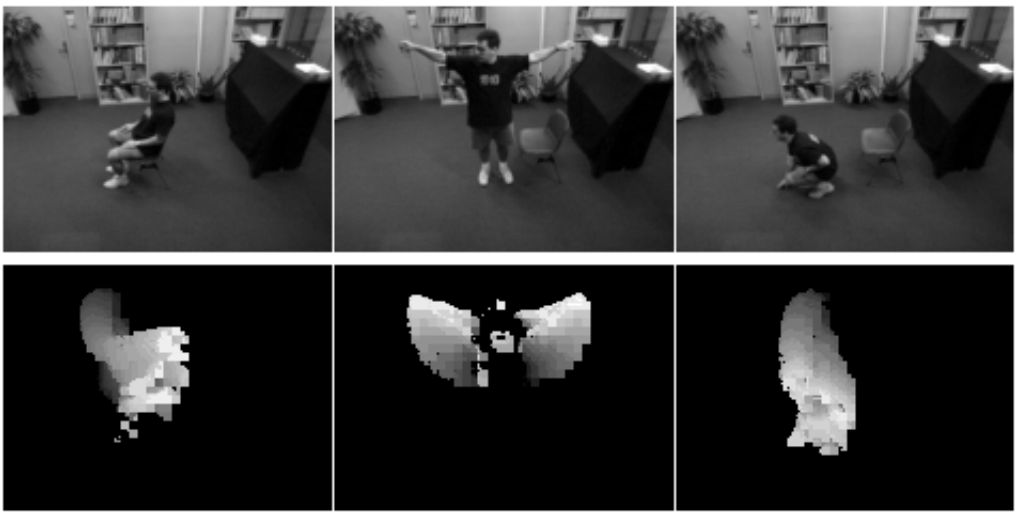

Figure 2.3: Sample Outputs for Motion History

gain control of camera or noise. To achieve this goal, image normalization given in Eqn. 2.4 is used where $\overline{I(\tau)}$ is the mean value of the current image and std(.) is the standard deviation.

$$I^{'}(\tau) = \frac{I(\tau) - \overline{I(\tau)}}{std(I(\tau))} \tag{2.4}$$

2.1.1.1.4.2 MHI Generation

MHI shows how the motion evolves historically by incorporating a temporal decay term. The forward and backward MHI, are denoted as $H_F(x, y, t)$ and $H_B(x, y, t)$ respectively; and iteratively calculated starting from the furthest frame to the current in which they are set to zero $H_F(x, y, t - (L - 1)) = 0$ and $H_B(x, y, t + (L - 1)) = 0$. For both H_F and H_B, calculation is achieved in a similar way, hence showing the calculation of H_F, Eqn. 2.5,would be enough.

$$H_{\mathrm{F}}(x, y, \tau) = \begin{cases} max(0, H_{\mathrm{F}}(x, y, \tau - 1) - d), & |D(x, y, \tau)| < T \\ 255, & |D(x, y, \tau)| \geq T \end{cases} \tag{2.5}$$

where $d = 255/L$ is the decay term, L is the buffer length and T is the distance threshold. In [10], distance D is defined as the absolute difference between

frames as in Eqn. 2.6, since [10] is designed for IR images that contains 1 channel only. However, in this work 3-channel rgb images are also used. Therefore, the distance measure is modified and Eqn. 2.7 is used for 3-channel images instead of Eqn. 2.6.

$$D(\tau) = |I(\tau) - I(\tau \pm \Delta)| \ , \tag{2.6}$$

$$D(\tau) = \sqrt{(I_r(\tau) - I_r(\tau \pm \Delta))^2 + (I_g(\tau) - I_g(\tau \pm \Delta))^2 + (I_b(\tau) - I_b(\tau \pm \Delta))^2} \tag{2.7}$$

2.1.1.1.4.3 Object Localization

After calculation of H_F and H_B, this information is used for object localization. First, these values are smoothed via median filter which results in elimination of small components which are actually noise. Then, $min(.)$ operator is used in order to suppress the gradient trail behind the object in forward MHI and ahead for the backward MHI.

$$Mask_{gray(t)} = min(MedianFilter(H_F(t)), MedianFilter(H_B(t))) \ . \tag{2.8}$$

Using Eqn. 2.8 yields strong response only for pixels within the current object boundary. Then the obtained gray level image is threshold to obtain foreground mask. Finally, the foreground mask is morphologically closed to get rid of holes that may occur inside of the object and the moving object mask is extracted.

2.1.1.2 Background Subtraction

The main purpose of this type of algorithms is to generate a background model which is to be compared with the current video frame. After the comparison, pixels which are dissimilar from the background model are marked as pixel of the moving object. The diversity of this type of algorithms comes from the usage of different solutions for background model extraction and dissimilarity measurement.

2.1.1.2.1 Averaging

As the name refers, the averaging method takes the average of lately video frames to construct background model. The averaging may have different weights between previously obtained background model and the new frame which is specified by learning rate. Although, these algorithms have low computational cost, they tend to produce tails behind the moving object and ghosts after objects that have wait long period of time. Both artifacts occur due to the condemnation of the background model. In order to solve this problem [11, 12] suggest not update the pixels of background model where in the current frame object is detected. A similar approach with adaptive thresholding methodology is also used in [7]. The background model update scheme is given in Eqn. 2.9 where α is the learning rate B_t background model at frame t and I_t is the current frame.

$$B_{t+1}(x,y) = \begin{cases} \alpha B_t(x,y) + (1-\alpha) I_t(x,y), & (x,y) \text{ not moving} \\ B_t(x,y), & (x,y) \text{ moving} \end{cases} \tag{2.9}$$

Note that moving not moving classification is achieved simply differencing corresponding background and image pixels and comparing the difference with threshold identifying dissimilarity. Different from [11, 12], [7] also includes a simple adaptive threshold selection scheme that is specified in Eqn. 2.10:

$$T_{t+1}(x,y) = \begin{cases} \alpha T_t(x,y) + (1-\alpha)(c|I_t(x,y) - B_t(x,y)|), & (x,y) \text{ not moving} \\ T_t(x,y), & (x,y) \text{ moving} \end{cases} \tag{2.10}$$

There exist other papers using averaging strategy due its low computational complexity such as [13–15].

2.1.1.2.2 Single Gaussian

To achieve better performance, more detailed background model than simple averaging is required. To answer this requirement a single Gaussian, N(μ,σ), may be used for each pixel. The mean value of the Gaussian corresponds to average of frames whereas the variance of pixels is new information that does not exist in averaging methodologies. Therefore, dissimilarity is calculated by using both as in Eqn. 2.11, which is actually nothing but a kind of dynamic

thresholding. The Eqn. 2.12 and Eqn. 2.13 show the update procedure for the Gaussian.

$$Mask_{\mathrm{t}}(x,y) = \begin{cases} 1, & |\mathrm{I_t(x,y)}\text{-}\mu_{\mathrm{t}}(\mathrm{x,y})| > 2.5\sigma_{\mathrm{t}} \\ 0, & \text{else} \end{cases} \tag{2.11}$$

$$\mu_t(x,y) = \begin{cases} \alpha I_t(x,y) + (1-\alpha)\mu_{t-1}(x,y), & \mathrm{Mask_t(x,y)}{=}1 \\ \mu_{t-1}(x,y), & \text{else} \end{cases} \tag{2.12}$$

$$\sigma_t^2(x,y) = \begin{cases} \alpha(I_t(x,y) - \mu_t(x,y)) + (1-\alpha)\sigma_{t-1}^2(x,y), & \mathrm{Mask_t(x,y)}{=}1 \\ \sigma_{t-1}^2(x,y), & \text{else} \end{cases} \tag{2.13}$$

where α is learning rate and $\mathrm{I_t(x,y)}$ is the intensity value. [16, 17] use a single Gaussian background model.

2.1.1.2.3 Gaussian Mixture Models

After the usage of single Gaussian, it is seen that individual pixel values frequently have more complex distributions so requirement of more elaborate models is emerged. Hence, the concept of Gaussian Mixture Model (GMM) is introduced with [18] in which more than one Gaussians are used to model the background. The main idea of GMM is to represent the background model of each pixel as a weighted sum of Gaussians as shown in Eqn. 2.14 and then detect pixels as foreground whenever they do not fit the model. The model construction is based on a reasonable assumption which states that the background is visible more frequently than the foreground. This assumption inherently states that the background model should be constituted by the repetitive pixel values. Therefore, for background model generation, the first need is to acquire a training set, X_T, which is updated over time. At each new sample together with the training data, the density estimation is also updated as in Eqn. 2.14

$$\hat{p}(x^{(t)}|X_T, BG + FG) = \sum_{m=1}^{K} w_m \eta(x^{(t)}; \mu_m, \Sigma_m) \ , \tag{2.14}$$

where $x^{(t)}$ is pixel value at time t, m is the mode (component) number of Gaussian, μ_m is the estimated mean value, Σ_m is the estimated covariance matrix, and w_m is non-negative mixing weight. Note that training samples include both foreground and background pixels from which the density estimation is achieved. However, one should remember the assumption which states that background pixels are visible more frequently than the foreground pixels. Considering the assumption, intruding foreground objects will be usually represented with small weights w_m. Therefore, the background model can be approximated as the first B largest weights as in Eqn. 2.15

$$\hat{p}(x^{(t)}|X_T, BG) \approx \sum_{m=1}^{B} w_m \eta(\vec{x}; \hat{\mu}_m, \sigma_m^2 I) \ . \tag{2.15}$$

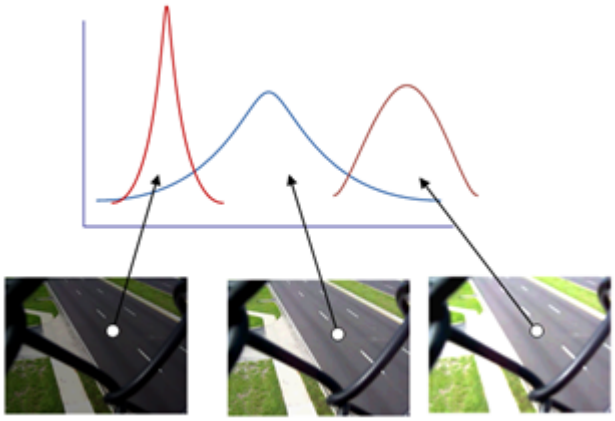

Figure 2.4: Multiple Modes are Generated in GMM by Considering Various Conditions to Represent Background Model Better

B is modes selected according to c_f which measures the maximum portion of the data that can belong to foreground objects without influencing the background model.

$$B = \arg\min_{b} \sum_{m=1}^{b} \hat{w}_m > (1 - c_f) \ . \tag{2.16}$$

Although, most of the moving object detection algorithms based on GMM use previously mentioned equations to estimate background model, they mainly differ in the parameter (μ_m, Σ_m, w_m) update techniques. Since usage of Gaussian

Mixture Model is another alternative for the overall system, update mechanism of [18] is expanded on together with the update mechanisms of its variants, [23] and [24].

2.1.1.2.3.1 Background Model Using GMM [18]

All GMM variants, updates their background model after obtaining every new pixel value. For this procedure, the first step is to check new pixel value against all existing Gaussian distributions to reveal whether the pixel value matches to a mode. If the Euclidean Distance of the new pixel to the mean values of the modes is smaller than 2.5σ, the pixel is stated to be matched. If the new pixel value cannot be matched, the least probable distribution is replaced with a new distribution whose mean is the new value. The variance of the new distribution is selected to be high to give better chance for a match with the pixel value is to come. Expectedly, the initial weight is given low since it is the first time for that mode is observed. The update equations for Strauffer and Grimson's GMM [18] are given below:

$$w_m^{(t)} = (1-\alpha)w_m^{(t-1)} + \alpha o_m^{(t)} \ , \tag{2.17}$$

$$\mu_m^{(t)} = \mu_m^{(t-1)} + o_m^{(t)}(\rho\delta_m) \ , \tag{2.18}$$

$$\sigma_m^{2\ (t)} = \sigma_m^{2\ (t-1)} + o_m^{(t)}(\delta_m^T\delta_m - \sigma_m^{2\ (t-1)}) \ , \tag{2.19}$$

where $x^{(t)} = [x_1, x_2, x_3]^T$, $\mu^{(t)} = [\mu_1, \mu_2, \mu_3]^T$, $\delta_m = x^{(t)} - \mu_m$; and $\rho = \alpha\eta(x^{(t)}; \mu_m, \sigma_m)$.

The symbol α in the update equations is the learning rate and $o_m^{(t)}$ takes the value 1 for the mode it is matched and 0 others. Examining the update equations it is seen that when the new pixel is not matched, μ and σ is not altered but it is weight is lowered. This actually means the observed background model is not destroyed; however if the data of mode is not frequently seen its weight is lowered and may be replaced with a new mode if data from that mode cease its existence. On contrary, after a short period of time, if data from the same mode is observe its weight is increased and mode preserves itself to be one of the most probable K modes. This is actually a kind of low-pass filtering which

enables algorithm to deal with repetitive motion as in the case of swaying tree and suppress sudden illumination changes for a while. From now on then, these update equations will be referred as standard update equations.

2.1.1.2.3.2 Background Model Using GMM [23]

The standard GMM update equations were extended to improve the speed and adaptation rate of the model [19–22]. Although, these algorithms become successful in coping with slow lightening changes, repetitive motions as tree leafs and long-term scene changes; they the usage of fixed number of GMM modes is found to be inadequate for more complex scenes. In the work of Zivkovic and Heijden [23], the algorithm proposed in [18] is improved by using a recursive computation to constantly update the parameters of a GMM, which adaptively chose the appropriate number of Gaussian components to model each pixel online, from a Bayesian perspective. Knowledge of the number of modes is important, since too many components may lead over fitting to data whereas too few may lead to under fitting. Moreover, update equations are modified in order to accept presence of a mode when enough evidence from the data exists for this mode.

$$w_m^{(t)} = (1-\alpha)w_m^{(t-1)} + \alpha(o_m^{(t)} - c_T) \ , \tag{2.20}$$

$$\mu_m^{(t)} = \mu_m^{(t-1)} + o_m^{(t)}(\alpha/w_m)\delta_m \ , \tag{2.21}$$

$$\sigma_m^{2\ (t)} = \sigma_m^{2\ (t-1)} + o_m^{(t)}(\alpha/w_m)(\delta_m^T\delta_m - \sigma_m^{2\ (t-1)}) \ , \tag{2.22}$$

where $x^{(t)} = [x_1, x_2, x_3]^T$, $\mu^{(t)} = [\mu_1, \mu_2, \mu_3]^T$, $\delta_m = x^{(t)} - \mu_m$ for a 3-channel image. The symbol α in the update equations is the learning rate and $o_m^{(t)}$ takes the value 1 for the mode it is matched and 0 others. In this method the important thing is that after calculation of the weights, $w_m^{(t)}$, the irrelevant components ($w_m^{(t)} < 0$) are discarded and the remaining mixture weights are re-normalized. Steps for model update can be summarized as:

1. Obtain new data $x^{(t)}$
2. Calculate ownership, $o_m^{(t)}$

3. Update mixture weights using Eqn. 2.20

4. Check irrelevant components $w_m^{(t)} < 0$, discard these modes and re-normalize remaining weights

5. Update $\mu_m^{(t)}$ using Eqn. 2.21

6. Update $\left(\sigma_m^{(t)}\right)^2$ using Eqn. 2.22

2.1.1.2.3.3 Self-Adaptive GMM

Another important parameter of Gaussian Mixture Models is the learning rate and should be adjusted properly. For a low learning rate, a very wide and inaccurate model, resulting in low detection sensitivity, can be produced. On the other hand, for a high learning rate, the model updates too quickly, and slow moving objects will be absorbed into the background model, resulting in a high false negative rate. Based on [23], the SAGMM, [24], is introduced as a solution for the problem. SAGMM adjust the learning rate online by considering number of data points which results in better adaptation to dynamic scenes. The update equations of the Self-Adaptive GMM are given below:

$$w_m^{(t)} = (1-\alpha)w_m^{(t-1)} + \alpha(o_m^{(t)} - c_T) \ , \tag{2.23}$$

$$\beta_m = \alpha(1+c_m)/c_m \ , \tag{2.24}$$

$$\mu_m^{(t)} = \mu_m^{(t-1)} + o_m^{(t)}(\beta_m/w_m)\delta_m \ , \tag{2.25}$$

$$\sigma_m^{2\ (t)} = \sigma_m^{2\ (t-1)} + o_m^{(t)}(\beta_m/w_m)(\delta_m^T\delta_m - \sigma_m^{2\ (t-1)}) \ , \tag{2.26}$$

$$c_m = c_m + 1 \ , \tag{2.27}$$

where $x^{(t)} = [x_1, x_2, x_3]^T$, $\mu^{(t)} = [\mu_1, \mu_2, \mu_3]^T$, $\delta_m = x^{(t)} - \mu_m$ for a 3-channel image. Note that each mode has different learning rate,βm, and calculated based on the basic learning rate α,which defines the limit of the influence of the old data $\alpha = 1/T$, and current accumulator counter c_m. The counter is increased at each time when a data is assigned to the mode and it is reset to 1 when a mode is perished. Assignment of a data to mode is referred as ownership and $o_m^{(t)}$ is set to 1 if sample is close to that component and otherwise it is set to 0. Closeness is measured with Mahalonobis distance and a sample is said to

be close if its Mahalonobis distance is smaller than a certain threshold. The squared distance from the mth component is calculated as:

$$D_m^2(x^{(t)}) = \delta_m^T \Sigma_m^{-1} \delta_m \ . \tag{2.28}$$

If there are no close components, a new component is generated with $w_{m+1}^{(t)} = \alpha$, $\mu_{m+1}^{(t)} = x^{(t)}$, $\sigma_{m+1}^{(t)} = \sigma_0$, $c_{m+1} = 1$. Lastly the c_T is a negative bias component which is used to suppress the components that are not supported by the data.

Although background subtraction provides the most complete feature data, it is extremely sensitive to illumination changes which yield misdetection of all pixels as foreground in the presence of sudden illumination change.

2.1.1.2.4 Eigenbackground Subtraction

Eigenbackground subtraction [25] proposes an eigenspace model for moving object segmentation. In this method, dimensionality of the space constructed from sample images is reduced by Principal Component Analysis (PCA). The main idea is stated as the reduced space after PCA should represent only the static parts of the scene, remaining moving objects, if an image is projected on this space. The main steps of the algorithm can be summarized as follows [26]:

- A sample of N images of the scene is obtained; mean background image, $\mu_{(b)}$, is calculated and mean normalized images are arranged as the columns of a matrix, A.
- The covariance matrix, $C = AA^T$, is computed.
- Using the covariance matrix C, the diagonal matrix of its eigenvalues, L, and the eigenvector matrix, Φ, is computed
- The M eigenvectors, having the largest eigenvalues (eigenbackgrounds), is retained and these vectors form the background model for the scene.
- If a new frame, I, arrives it is first projected onto the space spanned by M eigenvectors and the reconstructed frame I$^{'}$ is obtained by using the projection coefficients and the eigenvectors.

- The difference I - I$^{'}$ is computed. Since the subspace formed by the eigenvectors well represents only the static parts of the scene, outcome of the difference will be the desired change mask including the moving objects.

Although this method has an elegant theory, it cannot model dynamic scenes as expected. The reported achievements are generally restricted to some specific environments. Moreover, its computational complexity is high comparing to previously mentioned methodologies. All in all, eigenbackground subtraction is not very suitable for outdoor surveillance tasks; hence not preferable for our application.

2.1.1.3 Optical Flow Estimation

Optical flow methods make use of the flow vectors of moving objects over time to detect moving regions. Optical flow is the estimation of pixel-level motion, which is mainly based on the constancy assumption of brightness. If $I(x, y, t)$ is the image intensity at coordinate (x, y) at time t, then the optical flow constraint equation is as given in Eqn. 2.29.

$$I_x u + I_y v + I_t = 0 \ , \tag{2.29}$$

where I_x=∂I/∂x, I_y=∂I/∂y, I_t=∂I/∂t are image gradients and u=dx/dt and v=dy/dt are optical flow components. As it is seen, Eqn. 2.29 includes two unknown (u,v) which means one more equation is required for solving for these unknowns. This additional constraint is called as smoothness constraint and generally assumes smoothness in motion field as in [31,32]. In this sense, optical flow techniques can be viewed conceptually in terms of three stages of processing:

- Pre-filtering or smoothing with low-pass/ band-pass filters in order to extract signal structure of interest and to enhance the signal-to-noise ratio,
- Extraction of basic measurements, such as spatio-temporal derivatives (to measure normal components of velocity) or local correlation surfaces, and
- Integration of these measurements to produce a 2-D flow field, which often involves assumptions about the smoothness of the underlying flow field.

Optical flow method can detect independent moving object without knowing any prior information of scene which is suitable for dynamic background [28, 33, 34]. However, optical flow methods are computationally expensive due to iterative scheme of flow vector calculation which leads them to be over complex and inappropriate for our application.

2.1.2 Moving Shadow Detection

Considering overall system, detection of moving objects is important for track initialization and determining track window size which specifies tracking parameters. However, for many moving object detection algorithms, shadows tend to be classified as part of moving object since they share the same movement pattern. Therefore, to achieve discrimination between target and its shadow, various types of algorithms are proposed in the literature. One of the most recent shadow detection review, [35], categorizes the literature in a different manner, feature based, rather than done in previous survey [36] in which division is achieved in algorithm based. Based on [35], the methods published during the last decade are divided into four feature based categories which is illustrated in Fig. 2.5.

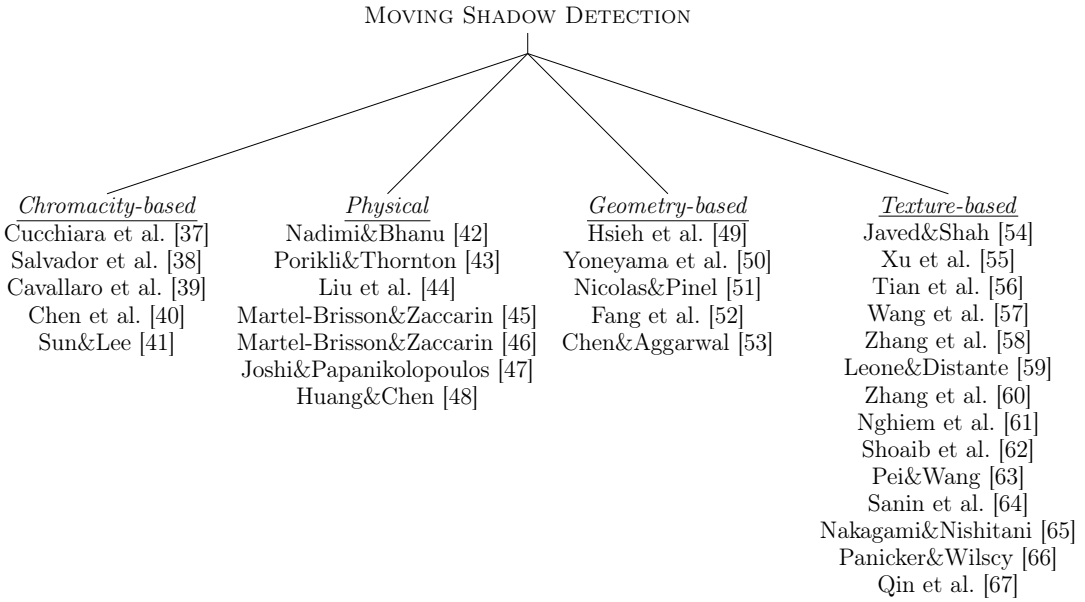

Figure 2.5: Taxonomy of Shadow Detection Methods Published in Last Decade (based on [68])

2.1.2.1 Chromacity Based Shadow Detection

These algorithms are based on the assumption that in the presence of shadow, shadowed pixels become darker but retain their chromacity. This color transition model is referred as color constancy. Since the chromaticity is a measure of color that is independent of intensity, the change in two metrics should be calculated separately. Most commonly used metrics are introduced by [69], and referred as brightness distortion (BD) and chromacity distortion (CD) and calculated using Eqn. 2.30 and Eqn. 2.31:

$$BD(x,y) = \frac{\frac{F_R(x,y)\mu_R(x,y)}{\sigma_R^2(x,y)} + \frac{F_G(x,y)\mu_G(x,y)}{\sigma_G^2(x,y)} + \frac{F_B(x,y)\mu_B(x,y)}{\sigma_B^2(x,y)}}{[\frac{\mu_R(x,y)}{\sigma_R(x,y)}]^2 + [\frac{\mu_G(x,y)}{\sigma_G(x,y)}]^2 + [\frac{\mu_B(x,y)}{\sigma_B(x,y)}]^2} \quad , \tag{2.30}$$

$$CD(x,y) = \sqrt{\left[\frac{F_R(x,y) - BD \cdot \mu_R(x,y)}{\sigma_R(x,y)}\right]^2 + \left[\frac{F_G(x,y) - BD \cdot \mu_G(x,y)}{\sigma_G(x,y)}\right]^2 + \left[\frac{F_B(x,y) - BD \cdot \mu_B(x,y)}{\sigma_B(x,y)}\right]^2} \quad , \tag{2.31}$$

where F represents foreground pixels and μ stands for the pixel value of the background model. By imposing thresholds over BD and CD, foreground pixels are classified as shadow or highlight as in Eqn. 2.32:

$$\begin{cases} highlight, & \text{CD} < \tau_{\text{CD}} \ and \ \text{BD} > \tau_{\text{B1}} \\ shadow, & \text{CD} < \tau_{\text{CD}} \ and \ \tau_{\text{Blow}} < \text{BD} < \tau_{\text{B2}} \end{cases} \tag{2.32}$$

τ_{CD} is a threshold to distinguish between chromacity of the background model and current pixel. The threshold τ_{Blow} is introduced to prevent pixels of having low RGB values to be misclassified to shadow. Similarly, τ_{B1} is applied to detect highlights. To achieve better understanding Eqn. 2.32 is visualized in Fig. 2.6.

Most of the chromacity based methods are simple to implement and computationally inexpensive comparing to other methodologies. However, they are very dependent of threshold selection and susceptible to noise. Moreover, they are also sensitive to strong illumination changes and fail with strong shadows.

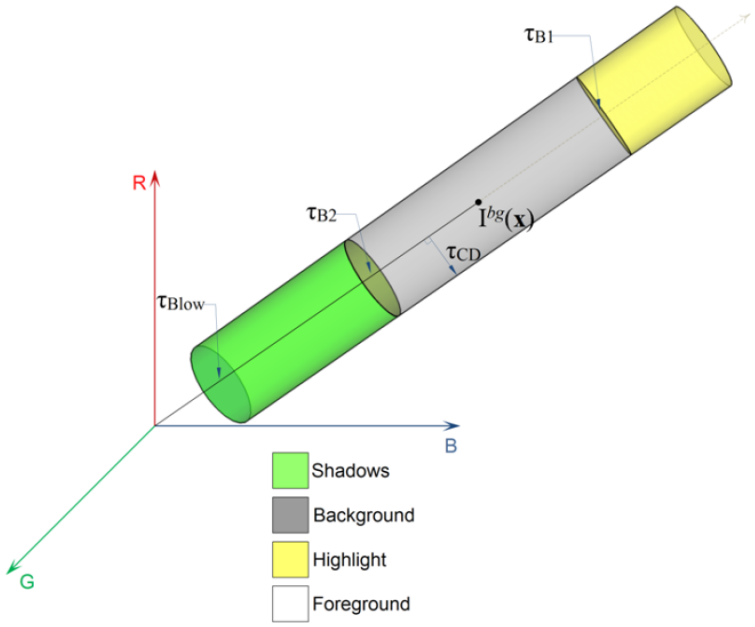

Figure 2.6: Brightness and Chromacity Distortion Levels Proposed (Source [69])

2.1.2.2 Shadow Detection using Physical based Methods

The physics-based methods use the illumination models. According to these models shadow is formed by change of illumination conditions. Hence shadow detection becomes the problem of finding illumination invariant features. In outdoor environments, two major illumination sources exist which are the sun (white light) and the light reflected from the sky (blue light). However, linear attenuation models assume illumination source to produce only white light. Hence, problem arises when the effect of the sun light decreases and the sky illumination becomes more apparent. To handle this problem, dichromatic models are proposed [42]. Apart from that, non-linear attenuation models are designed to create more general models for various illumination conditions in both indoor and outdoor scenarios [48]. Alternative approaches try to model or learn the appearance of every pixel under shadow to address attenuation problem [44–48]. Usage of learning mechanism can improve performance; however physical based methods are still limited to spectral properties which yields disadvantage in dealing with objects having similar chromacity to that of the background.

2.1.2.3 Shadow Detection using Geometry based Methods

For the cases in which the prior knowledge exists about illumination source, object shape and ground plane; the orientation, size and even shape of the shadows can be predicted. The main advantage of these methodologies is that they do not require for background model; hence can be directly executed on the input frame. Since the geometry based shadow detection is appropriate for specific object types whose shadows have different orientation than the object itself (pedestrians [49, 53]; vehicles [50, 52]), it seems to be appropriate for our problem. However, the difficulties in modeling the scene and the objects (i.e., the size of the vehicles has a wide range as car to truck; and the position of the illumination source (the sun) changes throughout the day) turns geometry based approach inappropriate for our system. Moreover, current geometry based moving shadow detection methods are not designed to deal with the shadows of multiple objects united in one foreground blob. By the nature of our problem this case frequently occurs, so overcoming this problem is our main expectation of shadow detection phase. Therefore, this type of methods is not suited practically for the problem we are dealing with.

2.1.2.4 Shadow Detection using Texture based Methods

Texture correlation is powerful method for detecting shadows; as textures are highly distinctive, do not depend on color, and robust to illumination changes. On the other hand they require more computational power than other methodologies. The main idea behind the texture based methods is the fact that the regions under the shadow retain most of their texture. In general, texture based shadow detection consists of two steps: selection of candidate pixels and regions, and classification of candidate pixels or regions as object or shadow. Candidate generation is achieved by usage of simple shadow detectors. Then, each shadow candidate is classified by correlating the texture in the frame and in the background model. If candidate and background model are similar in texture, candidate is classified as shadow or vice versa. To measure texture similarity various methods are proposed in the literature including normalized

cross-correlation [56], gradient or edge correlation [54, 55, 64], orthogonal transforms [58], Markov or conditional random fields [57, 67], Gabor Filtering [59]. Texture based moving shadow detection methods can be divided into two different approaches according to correlation levels as small region (or neighborhoods) and large region. The problem of using small regions is that they are not guaranteed to contain significant textures [59]. However, in LR texture methods as in [64]; large candidate regions of shadows are used (ideally whole shadow), which would include enough texture to discriminate between object and shadow.

2.2 Object Tracking

Object tracking is an important task within the field of computer vision. In its simplest form, tracking can be defined as the problem of estimating the trajectory of an object in the image plane as it moves around the scene. In other words, a tracker assigns consistent labels to the tracked objects in different frames of a video. In a tracking scenario, an object can be defined as anything that is of interest requiring further analysis, i.e. vehicles on a road, pedestrians in the street, planes in the air etc. This is actually a challenging task due to loss of information caused by projection of the 3D world on a 2D image, partial and full objects occlusions, complex object shapes and motion, scene illumination changes, imperfections and noise in imaging systems.

Since object tracking is a classical problem, there exist diverse set of algorithms in the literature [70,71]. The proposed solutions are primarily differing from each other based on how they model appearance, shape, and motion of the object. In this sense, taxonomy of tracking methods are proposed in [70] as in Fig. 2.7 and will be discussed in the following sections.

2.2.1 Point Tracking

This approach starts with the detection of objects in the scene by an external object detection mechanism in each frame. Then, tracking is achieved by revealing correspondence of the points (i.e. centroids of the detected objects) in

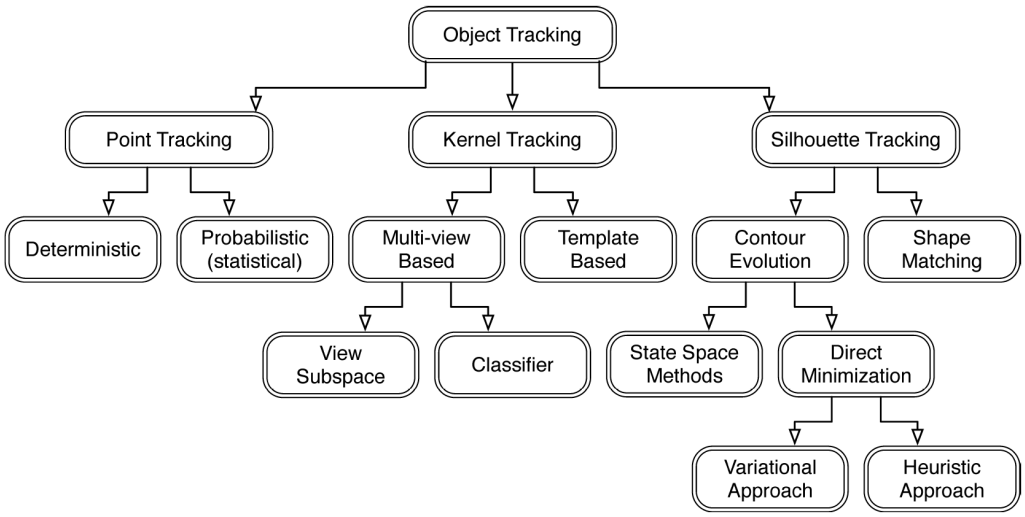

Figure 2.7: Taxonomy of Tracking Methods (Source [70])

consecutive frames. This correspondence problem is generally solved based on object states which can include object properties such as position, motion, and acceleration. The point correspondence can be accomplished by using deterministic and probabilistic methods. The deterministic methods use qualitative motion heuristics [72] whereas probabilistic methods explicitly take the object measurement and also considers uncertainties to establish correspondence as in Kalman Filter [73] , particle filter [74], and Joint Probability Data Association Filter [75].

2.2.1.1 Deterministic Methods for Point Tracking

The purpose of the deterministic methods for point tracking is to find association of each object in frame $t-1$ with a single object in frame t. As stated in [70], the correspondence problem is usually solved by using qualitative motion heuristics. Some exemplary heuristics are:

- Maximum Velocity: This constraint has the assumption that the target cannot move more than certain amount of pixels within a single frame. Hence, the possible match is searched in a limited circular neighborhood.
- Small Velocity Change (smooth motion): Based on the assumption that direction and speed of the target does not change drastically in

consecutive frames.

- Common Motion: The constraint is designed for representation of a single target with multiple key points. It is stated that the velocity of the key points is to be similar.

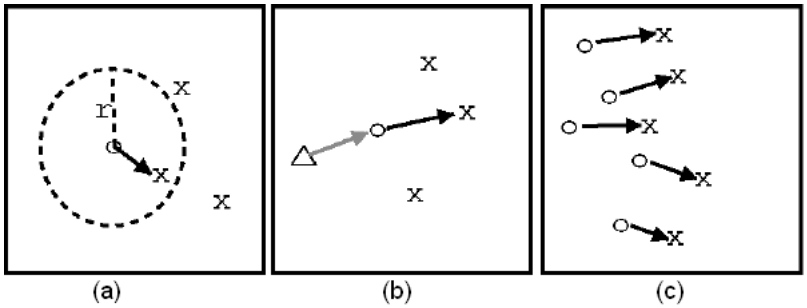

Figure 2.8: Motion Heuristics (a) Maximum Velocity, (b) Small Velocity Change, (c) Common Motion of Key Points (Source [70])

The disadvantages of deterministic methods are that they can be easily misguided when the measurements contain noise, and their performance decreases drastically when target motion undergoes random perturbations, i.e. maneuvering, dissatisfying the model assumed for target. Although deterministic point tracking methods are based on motion heuristics, one should note that usage of motion heuristics does not restricted to deterministic methods.

2.2.1.2 Statistical Methods for Point Tracking

The statistical methods are designed to solve the correspondence problem by not only taking care of the measurement which can be misleading but also considering probabilistic models for target motion by benefiting from state space approach for modeling the target properties such as position, velocity, and acceleration. From the measurements, which are usually target positions in the image, the state variables are estimated. In the literature these estimation techniques are grouped into two classes in [70] namely: single object state estimation and multiobject data association and state estimation.

The most well-known estimation techniques for single object state estimation are Kalman and particle filters. The Kalman filter is used to estimate state of

a linear system where state is assumed to be distributed normally and consists of two steps. The first step is the prediction step and achieves the new state of variables by using state model. The second step is known as correction step in which prediction is corrected by using the current observation and the state is updated with the correction. Although Kalman filter is widely used, the Gaussian assumption for state variables is an important limitation which results in poor estimations when assumption is not fulfilled. To deal with this problem, particle filter is designed. In particle filtering, the conditional state density is modeled by a set of samples with different weights. According to weight, all samples have different importance and all samples are used for estimation new value of state variables. Although, high number of particles would yield better estimation, the computational complexity also increases with the number of particles.

One should note that both kalman and particle filters are designed for estimating state of a single object; hence to achieve multiple target tracking with kalman or particle filter, the correspondence problem needs to be solved beforehand applying filter which is a kind of contradiction. However, our goal is multiple target tracking, which actually requires joint solution of data association. The most widely used two techniques for this problem are Joint Probability Density Association Filters (JPDAF) [75] and Multiple Hypothesis Tracking (MHT) [76]. The JPDAF is actually uses soft data assignment. To be more precise, JPDAF consider the probability of one measurement to belong to more than one track which results in a single hypothesis for summarizing all the past. The main limitation of JPDAF is the assumption on number of targets which is assumed to be fixed. Hence, it is not capable of handling new targets entering/leaving the scene. In MHT, this problem is solved by integrated track initiation. Association algorithm of MHT is a hypothesis based brute force implementation which aims to generate all possible hypotheses and requires high computational load. Moreover, MHT also requires large memory; since the different hypothesis from the past are kept in the memory.

In general, statistical based point tracking methods benefits from motion models of the targets and achieves a compromise between the model, i.e. constant

velocity, constant acceleration etc., and the measurements. However, for our application we cannot explicitly state any motion model. Moreover, point correspondence becomes a complicated problem especially in the presence of occlusions, misdetections, entries, and exits of objects; and can be misled by erroneous measurements which occur due to fails in moving object detection phase. Since mentioned conditions are highly probable in our problem, point tracking is not preferred for our system.

2.2.2 Kernel Tracking

Kernel tracking is typically performed by computing the motion of the object, which is represented by a primitive object region, from one frame to the next. In this sense, kernel refers to the object shape and appearance. For example, the kernel can be a rectangular template or an elliptical shape. These algorithms differ mainly based on appearance representation used. Tracking using templates and density appearance models are commonly used due to their relative simplicity which results in low computational cost. The most common approach in this category is template matching based approaches which are to be explained in detail since descendent of template matching algorithms [1, 77–79] introduces more robustness than point tracking approaches. Moreover, these algorithms does not require prior knowledge of motion model and also some of these algorithms introduce solutions for handling occlusions, which is frequently encountered in exemplary scenes for our problem.

2.2.2.1 Template Matching in General

Template matching is a straightforward concept which is designed for finding smaller parts of image which matches with the template. The matching process relies on the similarity of template and image section which is measured mainly by using a simple dot product. If the vector form of the template is defined as h and the image section as f, similarity is calculated as in Eqn. 2.33.

$$S(f,h) = f \cdot h = \sum_i f_i h_i \ , \tag{2.33}$$

where i is the pixel index in the templates. In theory, the result of dot product should be high if the resemblance of the pixels is high or vice versa. Therefore, the existence of the object can be verified by comparing the dot product result with a certain threshold. One should note that, dot product can only reveal the existence of the object for just one specific location which is the center coordinate of image section. However, template matching cannot only look for a specific location for tracking since the location of the target at the new frame is not known. Therefore, the image should be searched over for each possible location by calculation of dot products for each possible alignment. Actually, this is the definition of the correlation which is taken between template and image. The output of the correlation is a new image whose pixels are filled with the result of dot product between translated versions of the template and the image as illustrated in Fig. 2.9. The image coordinates where the local maxima occurs in the output image, C, is referred as correlation peaks. If the global maximum exceeds a certain threshold, this coordinate is indicated as the new location of the template in the image. Despite of many known weaknesses, which are to be explained; usage of correlation is very popular since it is fast, simple and easy to implement. Formal definition of the correlation is given in Eqn. 2.34:

$$C = f(x,y) \otimes h(x,y) \coloneqq \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) h(x+m, y+n) \ . \tag{2.34}$$

It should be noted that correlation methodology can produce clear peaks as long

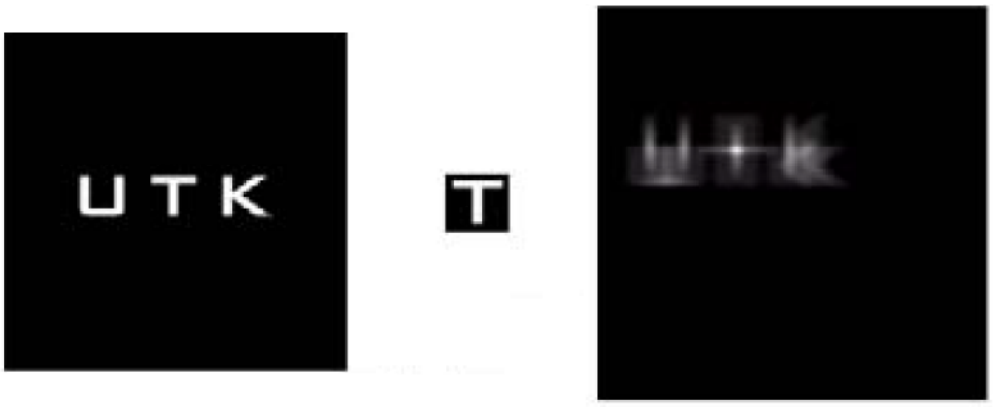

Figure 2.9: Visualization of Correlation Result between the Image and Template

as the template (tracked object model) preserves its similarity with the target in the input frame. Unfortunately, appearance of the target changes across im-

ages, due to variations among target instances and changes in imaging condition i.e. lighting, pose. In order to deal with the illumination change, normalized cross correlation concept is introduced in which the image and templates are normalized beforehand the correlation operation. The formal definition of NCC is given in Eqn. 2.35:

$$NCC(f,h) \coloneqq \frac{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(f(m,n)-\mu_f)(h(x+m,y+n)-\mu_h)}{\sigma_f \sigma_h} \quad . \tag{2.35}$$

Although, normalized cross correlation deals with illumination changes better, it is not sufficient for adaptation to appearance changes. More clearly, usage of the intensity values of the image and template may not be sufficient in complex scenes. Moreover, usage of simple template matching techniques brings also the disadvantage which is stated in [80]: while their response to a perfect example of the template pattern will always be high, the relative strength of responses to alternative patterns can be unpredictable. To overcome these problems, the Correlation Filters (CF) are proposed. The basic idea behind this technique is to learn filters that optimally map input images to their ideal output. The ideal output is a peak (or a value of one) at position of the target and zeroes for all other locations in the image (Dirac Delta Function). By training filters in this way, they do not only produce high responses for targets but also learn to suppress the response to common background distracters. To be clearer, correlation filters are designed to identify patterns that are consistent through the video sequence. Hence, they are more tolerant of common appearance changes than simple template matching and produce more prominent peaks in the target locations.

An exemplary illustration is given in Fig. 2.10. Fig. 2.10(a) shows the image, template and their correlation result. When the target (car) is slightly rotated, the correlation output of the perturbed image cannot produce as sharp peak as it used to be. On the other hand, usage of a correlation filter extracts a template shown in Fig. 2.10(c), different from the appearance of the targe, which is capable of producing a sharp correlation output.

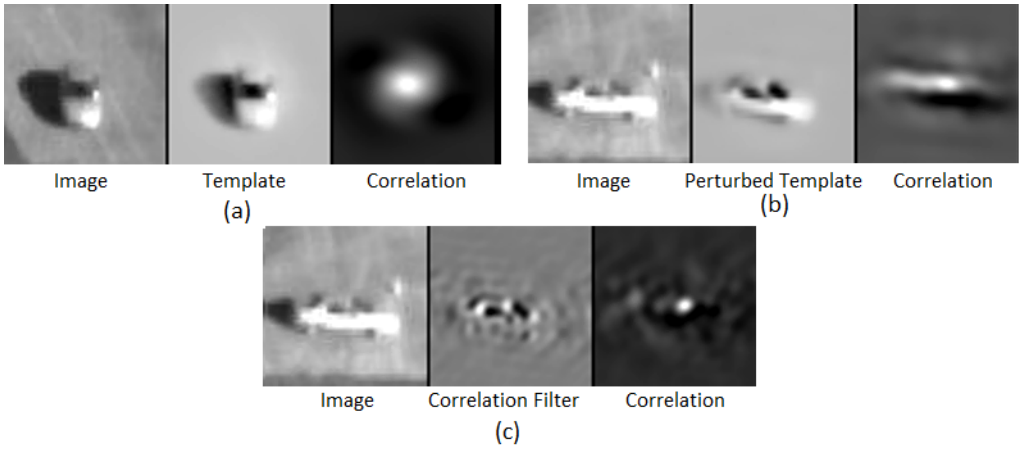

Figure 2.10: Resultant Correlation Outputs of (a) Image is same with Template; (b) Image is Slightly Rotated, (c) Template obtained via CF

2.2.2.2 Template Matching with Correlation Filters

Correlation filters have been investigated for three decades due to their attractive properties (shift invariance, robustness to graceful degradation, distortion tolerance) and employed in many applications including face localization [80], pedestrian localization [81], object localization and tracking [1], biometric recognition [82–84], and vehicle recognition [85] in the literature.

In the correlation filter approach for tracking, a carefully designed template (referred as filter), $h(x, y)$, and query image, $f(x, y)$, is cross-correlated. The output of the cross-correlation, $g(x, y)$, is searched for the most prominent peak by using a relevant metric, such as peak-to-sidelobe ratio (PSR) or peak-to-correlation energy (PCE), which is designed to indicate probability of the presence of the target. More prominent the peak, the value of the calculated metric is higher which means that the probability of the existence of the target is higher at the indicated location. On contrary, when target loses its visibility in the scene, no significant peak appears in the correlation output which results in lower values for the calculated track match quality metric. To be clearer, if the calculated match quality is above a certain threshold, the location of the prominent peak reveals the location of the target in the query image. If not, according to value of the quality metric the target is stated to be occluded or lost. For both efficiency and ease in calculation, the correlation operation is achieved in frequency

domain as in Eqn. 2.36:

$$G(u,v) = F(u,v)H^*(u,v) \ , \tag{2.36}$$

where * denotes the conjugate, and $G(u,v)$, $F(u,v)$ and $H(u,v)$ are the 2-D discrete Fourier transforms (FFTs) of the correlation output, query image, and the filter, respectively. The flow chart of tracking with correlation filter is illustrated in Fig. 2.11 together with exemplary correlation outputs for match/no match cases.

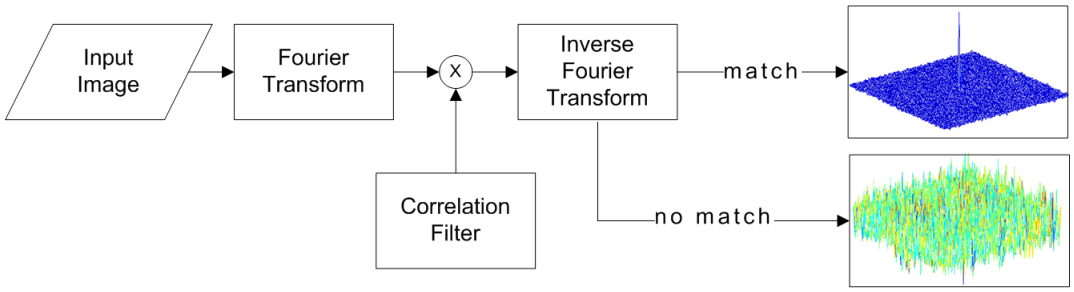

Figure 2.11: Flow Diagram of Simplest Template Matching with Correlation Filters

2.2.2.3 Related Work on Correlation Filters

As it is mentioned before, correlation filters are primarily designed to produce sharp peak at the target location while suppressing responses to zero for all other locations in the image. This actually means a search for a complete control of the shape of cross-correlation output. In order to achieve this goal, many correlation filter designs [1, 80, 86] applies Mean Square Error (MSE) between desired and calculated correlation outputs. In simplest form, cost function can be defined as in Eqn. 2.37.

$$\arg\min_{h} \|f \otimes h - g\|_2^2 \ . \tag{2.37}$$

Actually Eqn. 2.37 is the definition of the simplest correlation filter which is known as the Matched Filter. Although the matched filter is optimal for detecting a known image in the presence of additive white Gaussian noise, its detection performance decreases drastically even with small distortions in the object appearance. Therefore, usage of match filter requires samples for every

possible distortion of the object would possibly encounter making their usage impractical.

In 1980, Hester and Casasent introduced the design to take advantage of multiple training images called the Equal Correlation Peak Synthetic Discriminant Function (ECPSDF) filter [87] which is also known as the conventional SDF. To capture possible distortions the filters are designed to be weighted average of training images where weights are selected in a manner to achieve peaks filter response at match locations. Although, this filter is not used in practice due to the produced side-lobs in the correlation output, it was an initiative for the new strategies in correlation filters. The Minimum Variance Synthetic Discriminant Function (MVSDF) filter [88] was introduced in 1986 which is designed to minimize the output noise variance in the correlation plane. In practice, this filter is not used commonly due to same complication with ECPSDF.

The Minimum Average Correlation Energy (MACE) filter [89] was introduced in 1987 and it was a great breakthrough since it is the first filter that is designed to control not only the peak value of the correlation plane but the shape of whole plane. To be more precise, this filter is designed to minimize the average correlation energy (ACE) resulting in sharp peaks with minimum sidelobes. Although this filter is very sensitive to additive high frequency noise, its variations (e.g., Optimal Tradeoff Synthetic Discriminant Function (OTSDF) filter [90], Gaussian MACE [91], Minimum Squared Error Synthetic Discriminant Function (MSESDF) [92], and the Minimum Noise and Correlation Energy (MINACE) [93]) achieves better performance in presence of noise. The importance of MSESDF is its ability for allowing users to specify desired correlation output.

All of the previously mentioned filters belong to a subset of correlation filters called constraint correlation filters. The reason why they are named as constraint is in addition to minimizing the MSE between desired and calculated correlation outputs, they also constrain the correlation value at the target location. To be

more formal constrained correlation filters can be expressed as:

$$\begin{aligned} &\arg\min_{h} \|f \otimes h - g\|_2^2; \\ &s.t.\ h^T f = u \end{aligned} \tag{2.38}$$

where u denotes the value of constraint at target location. In latter works, this kind of hard constraint has found to be unnecessary and sometimes even deteriorating for generation of robust filters since it results in over-fitting to data. Therefore, removing the correlation peak constraint was a great breakthrough in correlation filters history. This new generation of filters is named as unconstrained correlation filters. The first examples of these filters are the Maximum Average Correlation Height (MACH) filter [86], the Unconstrained MACE (UMACE) filter [86], and the Unconstrained MSESDF (UMSESDF) filter [86]. More recent designs include the Average of Synthetic Exact Filter (ASEF) [80] and the Minimum Output Sum of Squared Error (MOSSE) filter [1].

Non-linear types of correlation filters also exist in the literature. Typically they exhibit superior performance than linear filters in exchange of high computational complexity. Some typical examples are the Polynomial Correlation Filters (PCFs) [94] and the Quadratic Correlation Filter (QCF) [95].

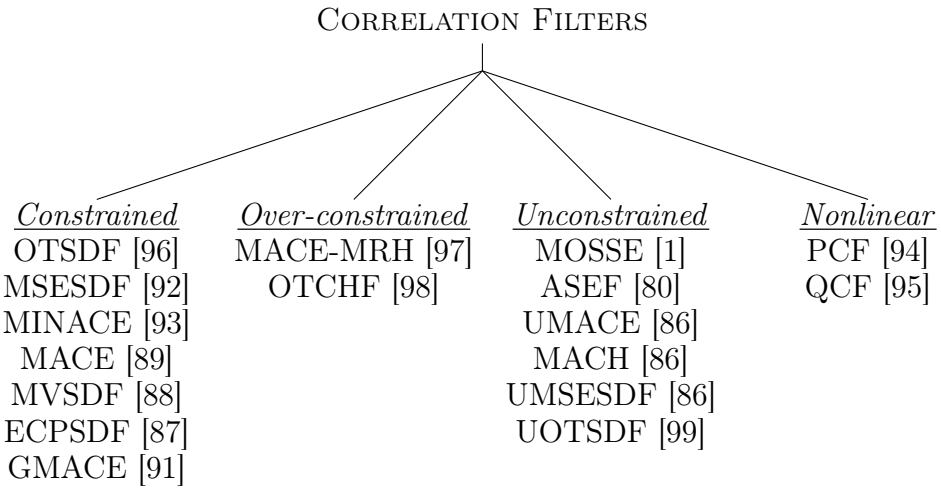

Figure 2.12: A Mind Map Consisting Various Correlation Filters Proposed in the Literature

The various types of correlation filters from the literature are gathered together in the mind map and illustrated in Fig. 2.12. In the following section each class of correlation filters is explained in brief. Since the unconstrained correlation

filters are more commonly used for target localization due to their superior performance, some recent examples of this class will be further analyzed.

2.2.2.3.1 Constrained Correlation Filters

In this class of filters, constraints on correlation output value is specified for every training image. The constraint is defined on desired output of filter u_i such that for positive training samples u_i=*1* and for negative samples u_i=*0*. For a set of training image x_i, and a filter h the corresponding constraint for a single training image is given in Eqn. 2.39

$$u_i = h^T f_i \ . \tag{2.39}$$

Note that usage of N training images results in N constraints which are typically much less than the dimensionality of the filter (number pixels in the filter). Therefore, there exist multiple filters that can satisfy these constraints. To produce single filter, different additional constraints are imposed by different algorithms. The general form of a constrained linear filter h is given in Eqn. 2.40

$$h = T^{-1}F(F^T T^{-1} T)^{-1} u \ , \tag{2.40}$$

where F is a matrix whose N columns contain frequency domain representations of N training images in vectorized form; T is a diagonal preprocessor matrix; and u is an Nx1 vector of the specified correlation output values for each training image as explained previously.

Special cases of preprocessor T lead to different filter designs. For example, in MVSDF design suppression of noise frequencies is achieved by defining T=C where C is a diagonal matrix such that each element of the diagonal corresponds to the power spectrum of the noise. A special case of MVSDF is ECPSDF in which noise spectrum is assumed to be white noise; hence C is assigned to be identity matrix. Another filter type MACE is designed to produce sharp detectable (prominent) peaks by minimizing the average correlation plane energy for the training set. Hence MACE takes preprocessor matrix T=D where D is a diagonal matrix containing average power spectrum of training images (D=$\frac{1}{N}\sum_{i=1}^{N}D_\mathrm{i}$). The OTSDF includes a trade-off parameter α that allows user

to emphasize low output noise variance (ONV) or low average correlation energy (ACE). Setting α=1 the OTSDF converges to MVSDF whereas setting α=0 converges to MACE filter. The MINACE filter achieves an alternative trade of between these two extremes. The summary of the different cases for T is summarized in Table 2.1

Table2.1: Designed Filter and the Corresponding Value of Preprocessor T

Designed Filter	Value of Processor T
ECPSDF [99]	I
MVSDF [88]	C
MACE [89]	D
OTSDF [96]	αC+(1- α^2)$^{0.5}$D
MINACE [93]	max(αC,(1- α^2)$^{0.5}$D$_1$,...,(1- α^2)$^{0.5}$D$_N$)

The most important drawback of the constrained correlation filters is that they can over fit the data due to limitations introduced by constraints. Hence their responses for test data may be different than the training data due to lack of generalization. For this reason, other type of linear correlation filters (over-constrained and unconstrained) are introduced which are more appropriate for target tracking purposes. Hence, this class of filter is not considered as an option for our system.

2.2.2.3.2 Over-constrained Correlation Filters

As the name implies, this type of filters have fewer degrees of freedom parameters (i.e., the number of pixels in the filter) than the number of constraints (i.e., the number of input images) as opposed to constrained correlation filters. Introducing that many constraints results in filters to lie in a smaller subspace than spanned by training images. Although restricting filters in a smaller subspace seems unreasonable, [100,101] proved that this approach would yield better generalization than the constrained filters.

The generalization of the filters yields better performance in the cases of certain type of distortions including in-plane rotation and scale changes. These types of distortions can be represented by some harmonic functions, i.e. for in-plane rotation: circular harmonic functions (CHF [102]) and for scale changes: Mellin radial harmonic functions (MRHF [103]). These harmonic functions are used

for filter design; and the optimal trade-off circular harmonic function (OTCHF [98]) filters, and minimum average- correlation-energy Mellin radial harmonic (MACE-MRH [97]) are introduced.

The OTCHF is introduced to design in-plane rotation invariant filters, giving a constant output for rotated versions of the filters. To achieve this goal, a response function within the interval [0, 2π] in which each value in the range corresponds to the desired correlation peak on the training image at that angle needs to be designed by the user. A typical response function is equal to one over some angular interval and zero everywhere else, which results in a rotation-tolerant filter with respect to the corresponding interval. The OTCHF filter design approximates this response function while minimizing the ACE criterion. The methodology of the MACH-MRH filter design is similar to that of the OTCHF design, except that scale is considered rather than rotation. More detailed explanation about MACH-MRH can be found in [97].

In [104], the computational complexity of this class of filters are stated to be significantly more than unconstrained correlations filters which makes unconstrained filters more preferable for our application.

2.2.2.3.3 Unconstrained Correlation Filters

By relaxing or removing the hard constraints of constraint correlation filters more generalized filters, namely unconstrained correlation filters, are proposed. The MACH filter [86] is the first unconstrained correlation filter which achieves distortion tolerance by maximizing the similarity of the shapes of true class correlation outputs over training image. This maximization problem is solved by minimizing a dissimilarity measure known as average similarity measure (ASM). While maximizing similarity, the filter also minimizes the ACE of false class images and maximizes the average correlation peak height for true class images.

For formal definition of MACH filter ,let f_i i=1,...,N_x denote the vectorized frequency-domain true-class training images; D_x and D_y denote diagonal matrices containing the average power spectra of the true and false classes; and m

is average training image. Then, formal definition of MACH filter becomes:

$$h = \left[\alpha D_y + (1-\alpha^2)^{0.5} S_x\right]^{-1} m \ , \tag{2.41}$$

$$S_x = \frac{1}{N_x} \sum_{i=1}^{N_x} [F_i - M] [F_i - M]^* \ , \tag{2.42}$$

where F_i and M are diagonal matrices containing the vectors f_i and m, respectively, along their diagonals.

Another filter example of this class is UMACE filter. UMACE only requires a high average response to the training examples which is defined as:

$$h = D'^{-1} m \ , \tag{2.43}$$

$$D' = \left(D\alpha + C\sqrt{(1-\alpha^2)}\right)^{-1} \ . \tag{2.44}$$

Here the filter for $\alpha = 0$ becomes the average training images and for $\alpha = 1$ has similarities to MACE in that it produces sharp correlation peaks. An illustration for UMACE filter training is given. First the provided training samples are averaged (avoids over fitting to data) and then the frequencies are reweighted in order to produce sharp peaks.

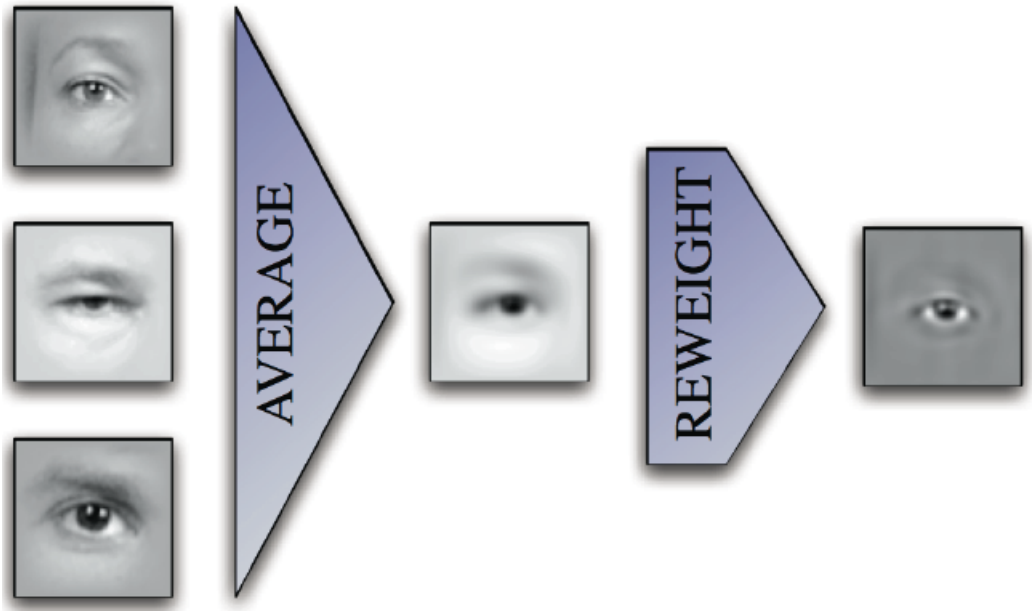

Figure 2.13: UMACE Filter Training (retrieved from [105])

The more recent examples of unconstraint correlation filters are introduced in [1, 80]. These filters are referred as optimized correlation output filters. Since their

successes are also reported in target tracking, further explanations on optimized correlation output filters will be given in the following section.

Optimized Correlation Output Filters: The optimized correlation output filters follows a different pattern in training phase of the filter. Unlike prior training methods that recombine templates, OCOFs consider image to image mapping that is performed during correlation with synthetic outputs and inverts this mapping to produce ideal filters. In this section three recent examples of OCOF (exact filters, ASEF, MOSSE) and their update mechanisms will be explained.

1. Exact Filters: The main goal of the exact filter is to obtain the filter that can map the input image to the ideal (desired) output by using only one input image. To fulfill this requirement, a training phase is required. Filters are trained from synthetically generated desired output for mapping training samples to the desired output where desired output actually represents the likelihood of the existence of the target. For this purpose, the desired output g_i is assumed to be sum of Gaussians which are centered at target positions as seen in Eqn. 2.45:

$$g_i = \sum_{j=each\ target\ location} exp\left(\frac{(x-x_j)^2 + (y-y_j)^2}{\sigma^2}\right) , \tag{2.45}$$

where (x_j, y_j) is the target location and σ specifies the radius of the peak. One should note that in the former versions of correlation filters as [88, 89, 96], delta Dirac function, zero/one (target/non-target) is used. However, in latter works [86] it is found restrictive yielding less susceptibility to noise. To achieve trade-off between robustness to noise and spatial resolution the variance of the Gaussian can be adjusted.

The required exact filter is achieved by solving Eqn. 2.46 where the subscript i stands for to indicate association of particular desired output (g_i), exact filter (h_i) and training image (f_i). Note that the $*$ symbol represents convolution operation.

$$g_i = h_i * f_i \ . \tag{2.46}$$

Solving Eqn. 2.46 in spatial domain requires lots of calculation. However, in Fourier domain correlation is nothing but an element-wise multiplication, Eqn.

2.47, which means solution for H requires only element-wise division, which is stated as in [106], as in Eqn. 2.48

$$G_i = H_i^* \odot F_i \ , \tag{2.47}$$

$$H_i^* = \frac{G_i}{F_i} \ . \tag{2.48}$$

H_i^* is called as exact filter since, it achieves exact transformation from f_i to g_i. To avoided the complex division, the Eqn. 2.48 is reinterpreted as Eqn. 2.49 where ϵ is a small constant to prevent zero division.

$$H_i^* = \frac{G_i \odot F_i^*}{F_i \odot F_i^* + \epsilon} \ . \tag{2.49}$$

In Eqn. 2.49, not only denominator is turned to be a real valued function but also both denominator and nominator becomes functions having physical meanings. To be clearer, the nominator becomes correlation of desired output with image, and denominator becomes the energy spectrum of the image.

2. Average of Synthetic Exact Filters (ASEF): Although the idea behind the exact filter is well-reasoned and achieve better performance than typical filters that are created by cropping templates out of input images, the reports indicates their susceptibility to noise [88, 96]. This is mainly because they apply exact transformation which results in over fitting to the data. To avoid this complication the idea of averaging the multiple exact filters is provided in ASEF. The averaging results in emphasizing of the consistent features across the image sequence and wipes out the inconsistent features. This idea is actually barrowed from the aggregation theory [107] which suggests that performance can be greatly improved by averaging the outputs of weak classifiers. However; instead of averaging the correlation outputs [80] averaged the filters themselves since filtering is a linear operation. The averaging is achieved in Fourier domain as shown in Eqn. 2.50

$$H^* = \frac{1}{N}\sum_{i=1}^{N} \frac{G_i \odot F_i^*}{F_i \odot F_i^* + \epsilon} \ . \tag{2.50}$$

The training procedure of the ASEF is illustrated in Fig. 2.14 which is taken from [80]. In this example the filter is trained to detect left eye in the image which can be seen by looking at the synthetic output generated which produces

peak for left eyes. After averaging of 7500 exact filter, the resultant ASEF filter is obtained. Actually the disadvantage of the ASEF filter also lies in the Fig. 2.14. ASEF requires many training samples to converge which. Hence, its low convergence rate would bring the disadvantage of not responding appearance changes quickly.

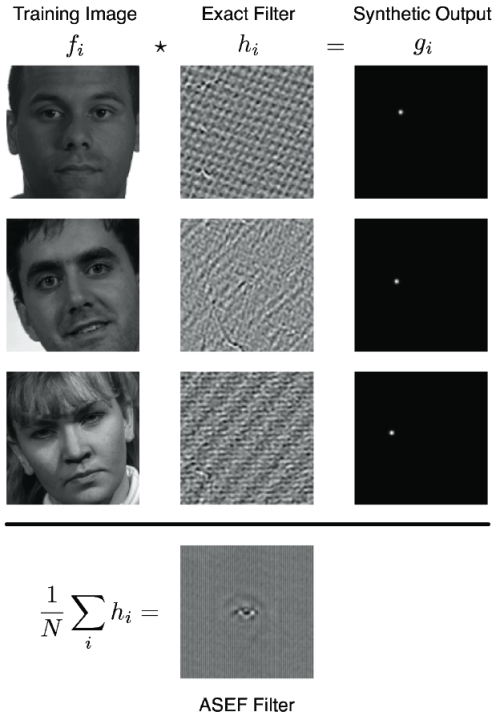

Figure 2.14: ASEF Training procedure and Obtained Filter by Usage of 7500 Exact Filters (retrieved from [80])

3. Minimizing the Output of Sum of Squared Error (MOSSE): The notion behind the MOSSE is similar to ASEF, however it requires a small number of training images to converge which makes MOSSE [1] an outstanding option for our application. In MOSSE, the filter is calculated by minimizing the output sum of squared error between calculated and desired outputs as in Eqn. 2.51

$$H^* = \min_{H^*} \sum_i |F_i \odot H^* - G_i|^2 \ . \tag{2.51}$$

Minimization of H in Eqn. 2.51 can be considered as minimization of its each element independently, hence Eqn. 2.51 can be written as in Eqn. 2.52 where w and v are index frequencies.

$$H_{wv} = \min_{H_{wv}} \sum_i |F_{iwv} H^*_{wv} - G_{iwv}|^2 \ . \tag{2.52}$$

Solving minimization problem actually means achieving the optimum point by taking derivative of the Eqn. 2.52 and setting it to zero and solving for variable of interest. However, this function is a real valued function of a complex variable and should be solved carefully. According to [108], first the Eqn. 2.52 should be rewritten in terms of both H^*_{wv} and H_{wv} and then the partial derivative w.r.t H^*_{wv} should be set to equal zero while treating both H^*_{wv} and H_{wv} independently.

$$0 = \frac{\partial}{\partial H^*_{wv}} \sum_i (H^*_{wv} F_{iwv} - G_{iwv})(H^*_{wv} F_{iwv} - G_{iwv})^* \ , \tag{2.53}$$

$$0 = \frac{\partial}{\partial H^*_{wv}} \sum_i F_{iwv} F^*_{iwv} H_{wv} H^*_{wv} - F_{iwv} G^*_{iwv} H^*_{wv} - F^*_{iwv} G_{iwv} H_{wv} + G_{iwv} G^*_{iwv} \ , \tag{2.54}$$

$$0 = \sum_i \left[F_{iwv} F^*_{iwv} H_{wv} - F_{iwv} G^*_{iwv} \right] \ , \tag{2.55}$$

$$H_{wv} = \frac{\sum_i F_{iwv} G^*_{iwv}}{\sum_i F_{iwv} F^*_{iwv}} \ , \tag{2.56}$$

Finally, solution of the optimization problem, leads to the Eqn. 2.57

$$H^* = \frac{\sum_i G^*_i \odot F_i}{\sum_i F_i \odot F^*_i + \epsilon} \ , \tag{2.57}$$

where ϵ is a small constant to prevent zero division. By looking at the Eqn. 2.57, it is seen that MOSSE is the generalized version of UMACE. Remember that UMACE is defined as $h = D'^{-1} m$ where m is a vector containing FFT of the average of the centered and cropped training images and D' is a diagonal matrix whose inverse is nothing but element wise division. Hence UMACE can be written as:

$$H^* = \frac{\sum_i F^*_i}{\sum_i F_i \odot F^*_i + \epsilon} \ . \tag{2.58}$$

In other words, Eqn. 2.58 is nothing but a specific version of Eqn. 2.57 in which g_i is defined as delta dirac function centered at image center. Hence, it can be simply stated that power of MOSSE comes from its ability to train on images with multiple uncentered images in which a certain uncertainty introduced by usage of Gaussian instead of deterministic delta dirac function. The summary of filter equations is given in Table 2.2.

Table2.2: Designed Filter and their mathematical representations

Designed Filter	Mathematical Representation
UMACE [86]	$H^* = \frac{\sum_i F_i^*}{\sum_i F_i \odot F_i^* + \epsilon}$
Exact Filter [80]	$H_i{}^* = \frac{G_i \odot F_i^*}{F_i \odot F_i^* + \epsilon}$
ASEF [80]	$H^* = \frac{1}{N} \sum_{i=1}^{N} \frac{G_i \odot F_i^*}{F_i \odot F_i^* + \epsilon}$
MOSSE [1]	$H^* = \frac{\sum_i G_i \odot F_i^*}{\sum_i F_i \odot F_i^* + \epsilon}$

2.2.2.3.4 Nonlinear Correlation Filters

Another type of CFs is non-linear CFs. Typically they exhibit superior performance but require much more computational power. The first type of this class of filter, quadratic correlation filters (QCFs), is characterized by solving for a quadratic discriminant function in d-dimensional space, where d is the number of pixels in the image. The second type, polynomial correlation filters (PCFs), are sets of linear filters applied to multichannel input images, whose outputs are subsequently summed to form a single output. Both the QCF and PCF designs solve for the set of linear filters jointly in order to optimize performance criteria.

Due their high computational complexity [109] the nonlinear correlation filters are not our interest.

2.2.2.4 Correlation Similarity Metrics

As mentioned before, the output of the cross-correlation result of filter and test image implies the existence of target. Actually, this metric represents the quality of the match and used for deciding between target states as tracked or lost. The aim is to identify prominent peaks which is not only the signature of a good match but also reveals the location of the target. More prominent the peak, the value of the correlation similarity metric should be higher. Hence, match/no match decision is made by comparing the peak sharpness (calculated metric) with a certain threshold (τ_{track}).

This section is devoted to explanation of two correlation similarity metrics form

the literature namely peak to correlation energy (PCE) and peak to sidelobe ratio (PSR).

2.2.2.4.1 Peak to Correlation Energy (PCE)

The PCE is introduced in [110] and designed to detect correlation peak and computed as in the Eqn. 2.59

$$PCE = \frac{g_{max}}{\sqrt{\sum_{m,n} |g(m,n)|^2 - |g_{max}|^2}} \ , \tag{2.59}$$

where g_{max} is the maximum value of the correlation plane. PCE is a good metric to use when there is one valid target in the image; however, when there is more than one target or significant illumination variation PSR becomes a better option.

2.2.2.4.2 Peak to Sidelobe Ratio (PSR)

The PSR [83] is the most commonly used correlation similarity metric and defined as:

$$PSR = \frac{peak - \mu}{\sigma} \ , \tag{2.60}$$

where μ and σ represents mean and standard deviation values of a small window centered at the correlation peak. Usually, a small subwindow is excluded from this calculation to measure sharpness of peak with respect to its surrounding.

2.2.3 Silhouette Tracking

Objects may have complex shapes, i.e., hands, head etc., that cannot be well described by simple geometric shapes. For this kind of objects, silhouette based methods provide an accurate shape description. The goal of a silhouette-based object tracker is to find the object region in each frame by means of an object model generated using the previous frames. Given the object models, silhouettes are tracked by either shape matching [111,112] or contour evolution [113]. Both of these methods can essentially be considered as object segmentation applied in the temporal domain using the priors generated from the previous frames.

Silhouette and contour representations are suitable for tracking complex non-rigid shapes; hence it is over complex and does not appropriate for our goals.

2.3 Visual Saliency

For human vision system identifying important regions/events s and focusing attention on important part is a routine task which is accomplished rapidly and accurately. It is widely accepted that this unconscious human visual attention is guided by the saliency concept. According to researches conducted by multiple disciplines including psychology, neurobiology, and computer vision; saliency originates from visual uniqueness, unpredictability, rarity or surprise. More formally definition of saliency is made by L. Itti as the distinct subjective perceptual quality which makes some items in the world stand out from their neighbors and immediately grab our attention. In this manner, Fig. 2.15 stands as a proof that visual attention is directly attracted to distinctive subjects.

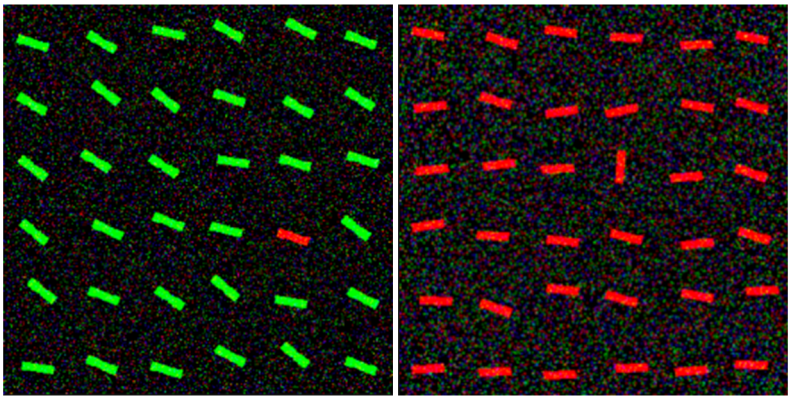

Figure 2.15: Uniqueness in Color or Direction can Effortlessly and Immediately Attracts the Attention of Human Visual System

Although humans can effortlessly detect visual distinctiveness by judging the previously mentioned attributes, computationally detection of such attributes remains a challenging goal. The desired properties of the saliency detectors in computer vision are listed in [114] as in below:

- Emphasize salient objects of all sizes
- Uniformly highlight whole salient regions

- Establish well-defined boundaries of salient objects
- Disregard high frequencies arising from texture, noise, and blocking artifacts, if any
- Compute saliency efficiently
- Output full resolution saliency maps
- Use minimal or no free parameters

To achieve these tasks, early works of Treisman and Glade [115]; Koch and Ullman [116]; and subsequent attention theories proposed by Itti, Wolfe and others proposed two different approaches namely, top-down and bottom-up saliency. While top-down [117] signals are derived from tasks demands (e.g., searching a toy bin for a red dragon); bottom-up signals are the core of visual saliency and they are stimulus driven. Actually, the bottom-up saliency is nothing but the declaration of distinctiveness of an image location from its neighbor. We focus on bottom-up approaches in the scope of this thesis; since saliency is used parallel with tracking and tracking does not reveal any contextual information.

Since saliency is defined according to its neighboring, and due to absence of high level knowledge on the neighboring; all bottom-up saliency methods rely on assumption on either objects or the neighboring (background). In this manner, the most common assumption is the contrast assumption stating that the salient object should have high contrast within a certain context. This very intuitive assumption is used nearly all saliency detection methods with different context definitions. According to context definition where the contrast is calculated these methods can be categorized as local methods [118–122] or global methods [117, 123–126]. Local methods uses pixel/patch in the local neighborhood and uses contrast measures including edge contrast [120], center-surround discriminative power [121], center-surround differences [118, 120, 127], curvature [122] and self-information [119]. The algorithm proposed by Itti [118] is also one of the fundamental frameworks in which multiple features are combined to obtain the saliency map. The general flow of the algorithm is explained in the Figure 16.

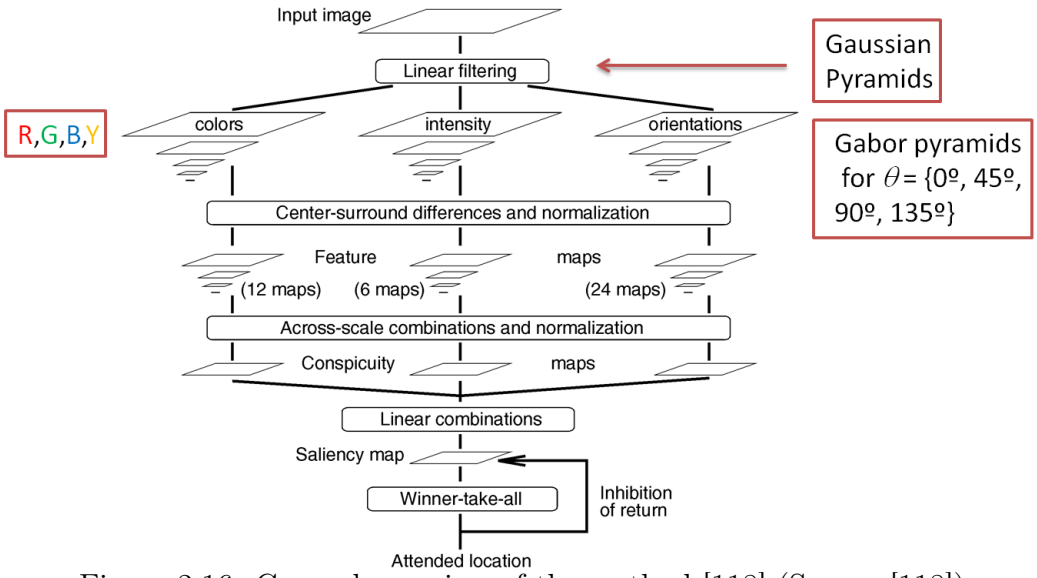

Figure 2.16: General overview of the method [118] (Source [118])

Rather than using local patches, global methods use the entire image and compare each pixel/patch with all others in the image and calculate the average dissimilarity [125, 126]. Based on contrast difference assumption; [125] uses graph representation while [126] uses the histogram of the image to reveal dissimilarity. Although contrast assumption, which is used in previously mentioned methodologies, is a strong assumption; it still has limitations especially in highlighting the interior parts of salient object. Therefore, other strategies also include usage of different domains as frequency domain for dissimilarity detection [123, 124]. In frequency-tuned saliency detection [124], salient objects are considered to have very low frequencies in the original image. However, to have well defined boundaries, some high frequencies are also desired to be included in the salient object. Hence, band pass filters are designed with cut-off frequencies excluding the noise and texture frequencies but including the object boundaries. In [123] (spectral residual approach) salient objects are detected by searching for discrepancies in the frequency spectrums. To detect the discrepancies in the spectrum, log spectrum curves are used and residual between log spectrum and averaged log spectrum is obtained. Then, the residual frequencies are marked as salient regions. Another solution for coping with the inadequacy of the contrast assumption is to support the algorithms with other assumptions. In [128], the problem is tackled from different perspective and two other assumptions are pro-

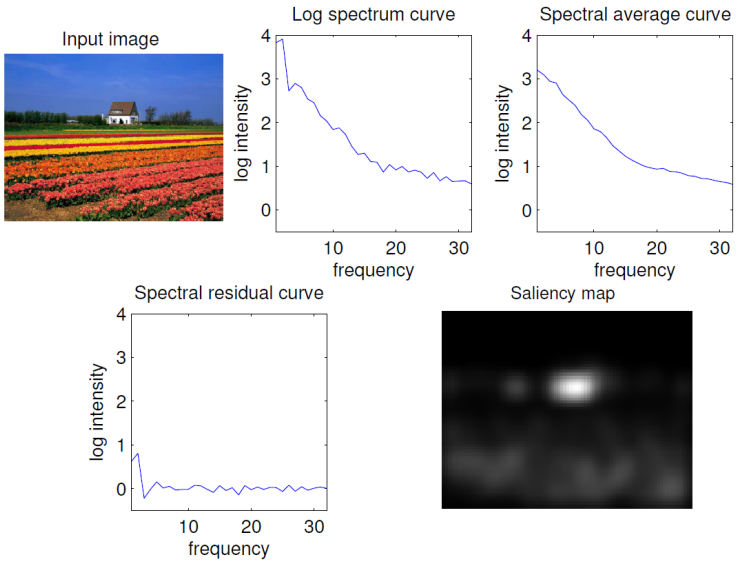

Figure 2.17: Spectral Residuals are used in [123] for Salient Region Detection (Source [123])

posed for the background rather than the salient object. The first assumption comes from the fundamental principle of the photography which is the important object is not generally cropped along the view frame. The second prior comes from the general appearance characteristics of the backgrounds in images, that is, background regions are usually large and homogeneous.

According to reported results in [128], these two additive assumptions on background significantly increase the performance as illustrated in Fig. 2.18. Moreover, the report also states that proves the effective calculation of these additional assumptions. Actually, according to reported results in [128], GS_GD [128] achieves the superior performance than many algorithms including Itti's method (IT) [118]; frequency based approaches frequency-tuned approach (FT) [124] and spectral residual approach (SR) [123]; graph based visual saliency (GB) [125]; histogram based contrast (HC) and region based contrast (RC) [126]; and context aware saliency (CA) [117]. The time performances of these algorithms are given in Table 2.3 which is retrieved from [128].

For our application all three assumptions are easy to fulfill; since we know that the salient object exist in the track window and we also have a rough estimate on background pixels from the moving object detector output. Therefore, in the proposed system geodesic saliency is preferred to be used to achieve target

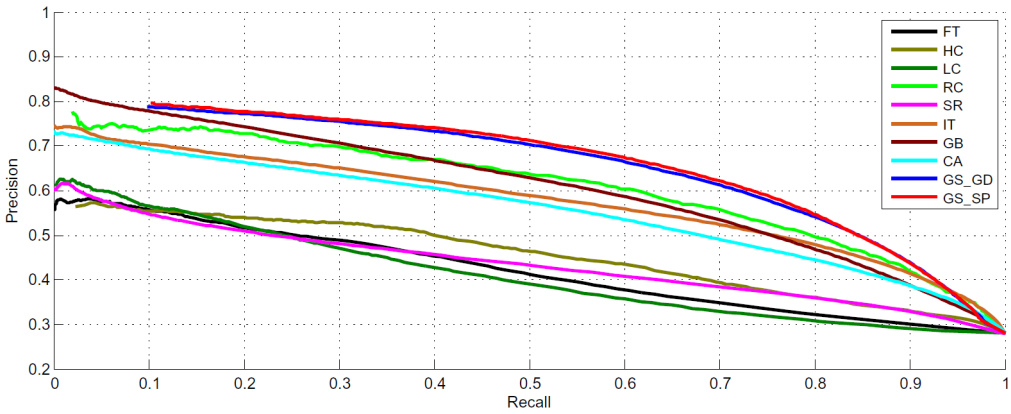

Figure 2.18: Usage of Background Priors Improved the Performance (Retrieved from [128])

silhouette which is actually the salient object in the track window.

Table2.3: Time Performances (in milliseconds) of Different Saliency Detection Algorithms (Retrieved from [128])

GS_GD	IT [118]	SR [123]	GB [125]	FT [124]	CA [117]	HC [126]	RC [126]
2.0	483	34	1557	8.5	59327	10.1	134.5

CHAPTER 3

PROPOSED SOLUTION

This chapter is devoted to the presentation of the proposed work which is designed to handle low level tasks (target detection and tracking) for an autonomous video surveillance system. It is better to emphasize that the scope of this work is limited to low level tasks and the usage of the extracted information in higher level tasks such as activity recognition, object identification is not included in this work. Since full autonomy is desired, the targets of interests should be detected by the system without any user interaction. For surveillance applications with stationary cameras, all moving objects are considered as targets of interest. Therefore, the first step of such a system should be moving object detection. The moving object detector produces a mask indicating locations of all the moving pixels which is benefited for the track initialization. Thus, the main objective of the moving object detection system is to ensure track initialization for each object.

Once the track is initiated, the goal becomes maintaining the track until the target leaves the scene. To accomplish this goal, literature [70, 71] offers 3 different classes of solutions, namely: point tracking, kernel tracking and silhouette tracking. In point tracking methods, tracking is achieved by revealing correspondence of the measurement points in consecutive frames. The measurements are generally obtained from an external object detection mechanism as moving object detector. It should be noted that the point correspondence problem becomes very complicated in the presence of occlusions, entries and exits of objects which results in varying number of data points. Moreover, this type of methods

also benefit from assumptions on target motion model and their performance decreases drastically when target motion undergoes random perturbations, i.e. maneuvering, dissatisfying the model assumed for target. Since occurrence of the mentioned conditions is highly probable for our problem, point tracking is not preferred. Another group of methodology is the silhouette tracking which is designed for tracking of non-rigid shapes; hence they are over complex and not appropriate for our goals. The last remaining category is kernel tracking which is divided into two classes as multi-view based and template based. Multi-view methods require offline training of target appearance which limits targets into specific object types and requires a lot of effort on training. Therefore, we preferred template based kernel tracking methods. Tracking using template matching is very commonly used in the literature due to their relative simplicity which results in low computational cost. However, simple template matching is susceptible of appearance variations due to pose changes or alteration in lightening conditions of the scene. Moreover, it does not provide full control of the correlation plane. More clearly, while their response to a perfect example of the template pattern will always be high, the relative strength of responses to alternative patterns can be unpredictable. To overcome all these problems, the Correlation Filters (CF) are proposed which do not count on the similarity of template images but instead learn patterns in the pixels produced by targets that are consistent even under varying imaging conditions. Moreover, they are also capable of maintaining tracking in the presence of similar objects by suppressing responses to distractive patterns. Benefiting from these two properties, they are more tolerant of common appearance changes than conventional template matching and produce more prominent peaks in the target locations.

Due to attractive properties of correlation filters most recent examples are investigated and the proposed solution is based on one of the most recent examples of correlation filters, MOSSE. Although MOSSE inherently has the ability of updating filter, it is not capable of responding to both abrupt and gradual changes of the target appearance. To overcome this problem, a multiple model visual target tracking methodology is proposed. Moreover, a possible remedy for a general disadvantage of correlation filters is also suggested. In general, correla-

tion filter uses static target widow sizes to utilize FFT and cross-correlation in successive frames. This actually results in tracking without indicating the actual sizes of the target, minimum sized bounding box encapsulating the target, which is valuable information for higher level tasks in surveillance applications. Moreover, not knowing the sizes can also decrease the tracker performance in the presence of scale changes; since some of the track parameters of the proposed tracker depend on the size estimation of the target. To fulfill this goal, a mechanism is proposed for target silhouette extraction which is based on learning of the geodesic saliency map of the track window throughout the tracking with an adaptive learning rate. Although target silhouette is only used for minimum sized target bounding box generation, this significant information can be used for higher level tasks as classification. The details of the procedure will be explained in target bounding box generation Section 3.2.2.

Considering all these, a framework is proposed as a solution whose overview is illustrated in Fig. 3.1. The remaining of the chapter is organized as follows.

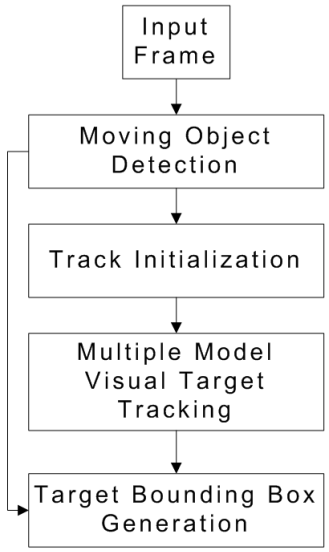

Figure 3.1: System Overview

The moving object detection is introduced in Section 3.1 together with track initialization. Section 3.2 covers the proposed tracking algorithm and the target model extraction.

3.1 Moving Object Detection

Identifying moving objects is an important task for monitoring systems; since the possible targets of interests are disclosed. In the proposed solution, generated moving object mask is used for two purposes: track initialization and target bounding box generation. For track initialization, each connected component in the moving object mask is tried to be identified whether it is being tracked or not. This goal is simply achieved by checking each track center for being within the borders of any connected component of the moving object mask. If a connected component does not possess a track window, track initialization is achieved by considering its size information. It should be noted that an exact boundary match for moving objects is not required by the track initialization system; since track window size is selected from possible discrete size levels as 16, 32, 64, 128, etc. Since this size estimation is not very precise and may misguide the selection of track parameters, they are tuned during the tracking with the collaborative work of moving object detector and target bounding box generation procedure. The target bounding box generation step is cued by the moving object detector to obtain background priors required by [128] and produces feedback information about target shape.

To sum up, what is expected from moving object detector is rough approximation of target size and location for every possible target of interest. To achieve the goal, SAGMM introduced in [24] is utilized due to three significant advantages:

1. If object is allowed to be a part of background, it does not destroy the existing background model. The original background values remain in the GMM if the object remains static for long enough, and its weight becomes larger than c_f. If the object then moves, the distribution describing the previous background still exists with the same estimated mean and variance.

2. Since each mode has its own learning rate the model can quickly achieve good estimation of mean and variance. To be more precise if background changes quickly newly introduced modes with smaller c_m will increase

learning rate β_m. Otherwise, if the background is stable, as more data samples are included in its parameter estimation resulting in high c_m, which leads β_m to converge the basic learning rate α.

3. Global illumination compensation makes the algorithm more robust to sudden illumination changes.

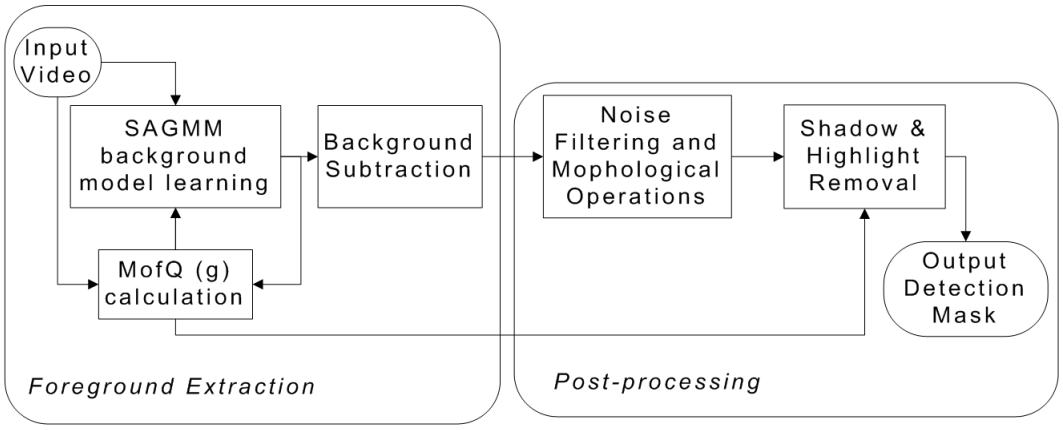

Figure 3.2: Flowchart of the Utilized Moving Object Detection

3.1.1 Foreground Extraction Module

Although SAGMM has significant advantages, it also shares the common drawback of statistical background subtraction methods, which is susceptibility to sudden illumination changes. To minimize the effect of illumination changes, the background model generation steps begin with global illumination factor (g) calculation. To achieve the goal, the Median of Quotient (MofQ) method is utilized. Actually, this procedure is intertwined with SAGMM background model learning since MofQ requires not only the usage of current (i_c) and but the reference image (i_r) pixels from the background model. The global illumination factor is calculated as in Eqn. 3.1 where S is the predefined sample pixels set.

$$g = median_{s \in S} \left(\frac{i_{c,s}}{i_{r,s}} \right) \quad . \tag{3.1}$$

Then, the foreground extraction continues with the core element, SAGMM background model learning. The calculated global illumination factor is used for

image normalization and integrated in the the SAGMM while calculating the difference as indicated below:

$$\delta_m = g \cdot x^{(t)} - \mu_m \ , \tag{3.2}$$

where $\mathrm{x}^{(\mathrm{t})} = [\mathrm{x}_1,\mathrm{x}_2,\mathrm{x}_3]^{\mathrm{T}}$, $\mu^{(\mathrm{t})} = [\mu_1,\mu_2,\mu_3]^{\mathrm{T}}$ for a 3-channel image.

After the learning of the background model each pixel is classified as foreground or background with a simple rule: if the weight of the Gaussian is greater than (1- c_f) and the distance is smaller than $T_b * \sigma^2$ (T_b is a scalar), pixel is considered to be background.

3.1.2 Post-Processing

Since most of moving object detection algorithms produce foreground masks contaminated with some scattered foreground pixels, general tendency is to benefit from some post-processing techniques. In this work, to achieve unstained foreground mask and complete the missing parts of the objects median filtering operation is followed by morphological closing and filling operations for the holes of the object mask.

Although binary operations are important, the post-processing part is dominated by shadow /highlight detection; since for many moving object detection algorithms, shadows tend to be classified as part of moving object. Due to cast shadows, several targets may be merged together which yields track initialization failure. Moreover, since estimation of the target size is used to determine track window size, processing time is also affected. Therefore, shadow removal is an important post processing step for moving object detectors.

Removing shadow pixels should be achieved carefully since misdetection of shadow pixels can cause a single target blob to be split up into multiple blobs which results in multiple track initializations for a single target. Based on the comparison results reported in shadow detection survey [35], large region texture based shadow detection algorithm [64] is preferred to eliminate shadows and highlights. The first step of this algorithm is shadow/highlight candidate generation which is answered by the corporation of the SAGMM, global illumination

factor calculation and Horprasert's algorithm, [69]. Horprasert's algorithm, [69], benefits from the assumption that in the presence of shadow, the intensities of the shadowed pixels decreases by retaining their chromacity. This assumption is referred as color consistency and measured with brightness distortion (BD) and chromacity distortion (CD) metrics. The formal definition of the utilized shadow/highlight candidate generation is given below:

$$BD(x,y) = \frac{gI(x,y)E(x,y)}{E^2(x,y)} \ , \tag{3.3}$$

$$CD(x,y) = \sqrt{\frac{gI(x,y) - BD(x,y)E(x,y)}{\sigma^2(x,y)}} \ , \tag{3.4}$$

where $I(x,y) = [I_R, I_G, I_B]$ represents the foreground pixels from current frame and $E(x,y) = [\mu_R, \mu_G, \mu_B]$ stands for the pixel value of the background model. By imposing thresholds over BD and CD, foreground pixels are classified as shadow or highlight as in Eqn. 3.5:

$$\begin{cases} highlight, & \text{CD} < \tau_{\text{CD}} \ and \ \text{BD} > \tau_{\text{B1}} \\ shadow, & \text{CD} < \tau_{\text{CD}} \ and \ \tau_{\text{Bl}} < \text{BD} < 1 \end{cases} \tag{3.5}$$

Following steps of the large region texture based shadow detection algorithm [64] is explained in detail in the next section.

Large Region (LR) Texture Based Shadow/Highlight Detection

As is mentioned previously, the first step of LR texture based method is to achieve shadow/ highlight candidates. The candidate pixels are generated by using the previously mentioned method. This method is used for only candidate generation (not for shadow detection); since the performance is highly depends on selection of proper thresholds which can be tricky. If the thresholds are selected too restrictive (high), it may fail to mark some shadow or highlight pixels. However, if these thresholds are kept low, object pixels are also marked as shadow/highlight. In candidate detection, misclassification does not constitute a problem since pixels belonging to foreground is expected to be eliminated when texture features are considered. However, if algorithm fails to detect shadow

candidates, these pixels are not processed further and will be classified as object which is not desired. Therefore, nonrestrictive (low) thresholds are used to prevent negative imposes on the upper bound of the detection accuracy. After generation of candidate mask, connected components are extracted. This labeling is actually important since it has the advantage of not breaking textures which results in better discrimination. Then for each connected component, gradient magnitude and gradient directions are calculated at each pixel by using Eqn.3.6 and Eqn.3.7:

$$|\nabla_p| = \sqrt{\nabla_x^2 + \nabla_y^2} \ , \tag{3.6}$$

$$\theta_p = arctan2\left(\frac{\nabla_x}{\nabla_y}\right) \ , \tag{3.7}$$

where the ∇_y and ∇_x are the vertical and horizontal gradients; and $arctan2(.)$ returns angle in radian between $[-\pi, \pi]$ which allows gradient to be treated as circular variable. The gradient magnitude is calculated to avoid effects of noise and give weight to pixels near edges that have more robust information about the textures. Then for each pixel which is above gradient magnitude threshold, the angular difference between in gradient direction of background model and the candidate is calculated as in the Eqn. 3.8:

$$\Delta\theta_p = arccos\left[\frac{\nabla_x^F \nabla_x^B + \nabla_y^F \nabla_y^B}{\left(\nabla_x^{F\,2} + \nabla_y^{F\,2}\right)\left(\nabla_x^{B\,2} + \nabla_y^{B\,2}\right)}\right] \ . \tag{3.8}$$

Gradient direction correlation is calculated based on angular gradient difference as in the Eqn. 3.9 and Eqn. 3.10:

$$c = \frac{\sum_{p=1}^{n} H(\tau_a - \Delta\theta_p)}{n} \ , \tag{3.9}$$

$$H(x) = \begin{cases} 1, & |x| \leq \tau_a \\ 0, & \text{otherwise} \end{cases} \tag{3.10}$$

where n is the number of the pixels selected in candidate shadow region. The correlation value c gives the ratio of the pixels having similar texture in the current frame and background model. Hence, high correlation value than a certain threshold (τ_c) for a region implies the high texture resemblance; the pixel group is classified as non-object (either shadow or highlight) region and it is removed from the moving object mask. The consecutive steps of the shadow/highlight removal method are summarized below:

1. Candidate generation of shadow/highlight pixels based on chromacity invariance,

2. Grouping of candidate pixels into candidate regions,

3. Selection of pixels with significant gradient magnitude for each region,

4. Calculation of gradient direction distance between the selected pixels from current frame and the pixels corresponding in background model,

5. Calculation of correlation score

6. Classification of pixels as object or non-object via comparison of correlation score with threshold τ_c

3.2 Multiple Model Visual Target Tracking with Target Size Feedback

As well as many computer vision problems, visual target tracking has also trade-offs between computational complexity and robustness to various conditions including pose and illumination changes, occlusion, noise etc. To achieve target tracking with less computational burden, usage of trackers based on correlation filters is an outstanding option. In this manner, a recent example [1] reveals an efficient method for visual tracking based on correlation filters. Since the proposed methodology in this work is based on [1], a brief explanation will be given.

The method in [1] starts with a pre-processing step which is designed to satisfy three goals: deal with artifacts produced by torus topological structure of FFT, put emphasis of pixels near target center for background suppression, and improve robustness for low contrast lightening situations. To achieve these goals, first the pixels of the template are transformed by using log function and then they are normalized to have zero mean and unit norm. Finally, the template is multiplied with a windowing function which gradually reduces pixel values as drifted from center resulting in emphasis of target pixels.

After achieving preprocessed input image patch, the patch and filter are cross correlated. Using the produced correlation output peak to side lobe ratio (PSR), which indicates quality of the match, is calculated. PSR is computed as in Eqn. 3.11:

$$PSR = \frac{CorrelationPeak - \mu}{\sigma} \ , \tag{3.11}$$

where μ and σ represents mean and standard deviation values of a small window centered at the correlation peak. If PSR is above a certain threshold, target is stated to be matched and the location of target is indicated with the location of the peak in the correlation output. If PSR is not high enough, the target is said to be occluded. If the occlusion case is repeated more than a certain number of consecutive frames, the target is stated to be lost. The filter update is also achieved according to PSR and updated if and only if PSR is high enough for indicating target match.

Actually, the key point of [1] is filter update mechanism which is based on MOSSE correlation filter and calculated as below:

$$H^* = \frac{A}{B} \ , \tag{3.12}$$

$$A = \sum_i G_i \odot F_i^* \ , \tag{3.13}$$

$$B = \sum_i F_i \odot F_i^* + \epsilon \ , \tag{3.14}$$

where F_i is the i^{th},template which is preprocessed and then transformed to the frequency domain; and the G_i is the frequency domain representation of desired output which is generated synthetically. An exemplary illustration of G_i is given in Fig. 3.3 where its distribution, representing target likelihood, and is selected to be Gaussian. On the left of Fig. 3.3, the visualization of the desired output can be seen which is generated for a stationary target; whereas on the right, the desired output is generated for a target which moved to 5 pixels to the west and south in one frame.

Since the target can undergo appearance changes; such as rotation, scale, pose and lightening variations; filters should adapt to these changes quickly to maintain tracking. Therefore, the filter update starts from filter initialization and

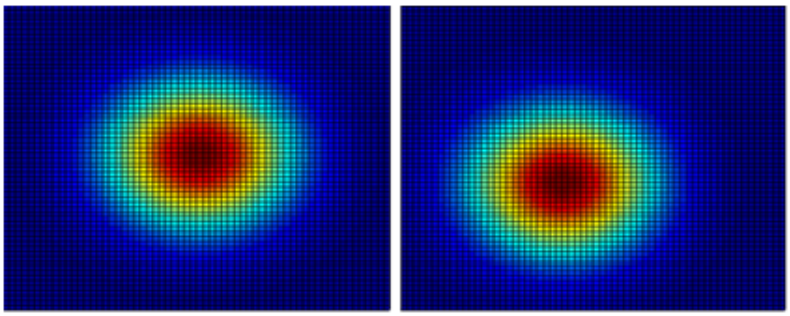

Figure 3.3: 63x63 Desired Outputs Having Same Size with Target Template for Stationary Target (Left); a Moving Target whose Displacement is (-5,-5)

achieved in each frame in which PSR is above a certain threshold. The filter update is achieved by calculating A_i and B_i as in Eqn. 3.15 and Eqn. 3.16 where the symbol η represents the learning rate.

$$A_i = \eta(G_i \odot F_i^*) + (1-\eta)A_{i-1} \ , \tag{3.15}$$

$$B_i = \eta(F_i \odot F_i^*) + (1-\eta)B_{i-1} \ . \tag{3.16}$$

According to update equations, recent frames have more weight while the past frames are also still in consideration with an exponential decay. Since the filter is constantly updated, gradual changes of the target appearance is captured which allows accurate target tracking through long video sequences. The overview of the tracking system is given in Fig. 3.4.

Obviously, the learning rate is an important parameter. Slower learning rates decrease probability of tracker to drift from the target and adapt to non-target objects; however it also limits tracker to respond quick appearance changes or vice versa. Therefore, single learning rate approach have limits on adaptation to changes in the scene; hence the performance of the algorithm is confined in a limited range especially for the cases where changing rate of appearances for different targets show wide variety.

Another disadvantage of the method suggested in [1] is the fixed object size assumption, which is shared by all correlation filters by their nature. This assumption actually yields constant track window size which is determined in track initialization and throughout the track this size is preserved to achieve FFTs and correlation matches. This would result in tracking the without knowing actual

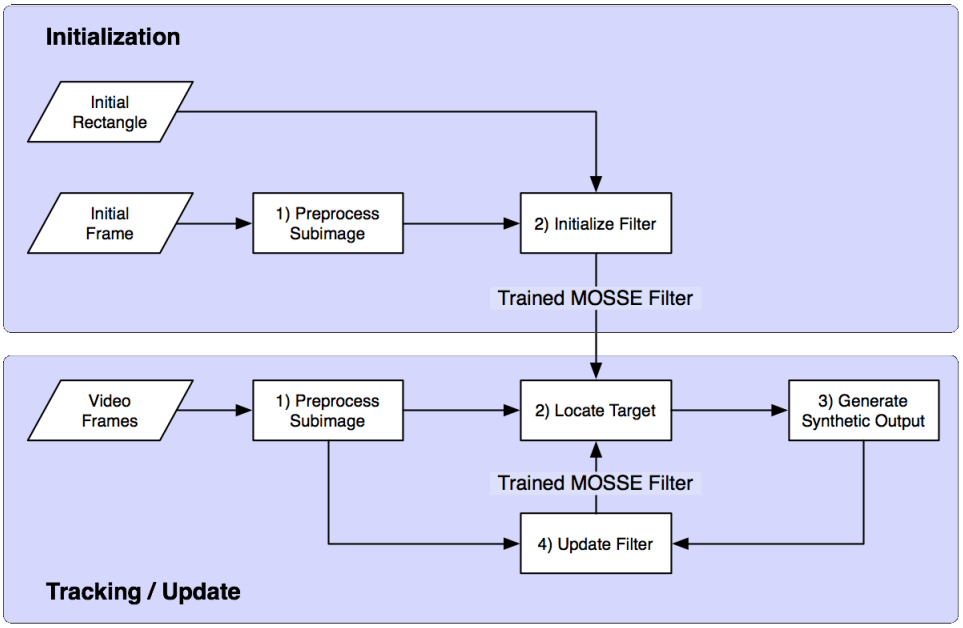

Figure 3.4: Overview of the Tracker Algorithm with MOSSE Correlation Filter (retrieved from [105])

size of the target which is a valuable information for higher level tasks. Moreover, scale change can also degrade track performance since suppression window size and PSR subwindow size is determined with respect to initial size of the target.

In this work, a solution is proposed to overcome these problems. First, a multiple model learning methodology is presented to improve adaptation limits of the system. Second, a method is introduced for generation of minimum sized bounding box for the target, Fig. 3.6 red rectangle, and to adjust tracker parameters according to target size to improve robustness for scale changes. To achieve target bounding box, the silhouette of the target is extracted via learning saliency maps in consecutive frames which is illustrated at the bottom right of the Fig. 3.6. The general overview of the proposed tracking system is given in Fig. 3.5.

3.2.1 Multiple Model Visual Tracking

Although the main flow of the algorithm in [1] is followed in this thesis, a different update mechanism is required to respond successfully to both abrupt and gradual changes of the target appearance. To accomplish the goal, two filter groups are

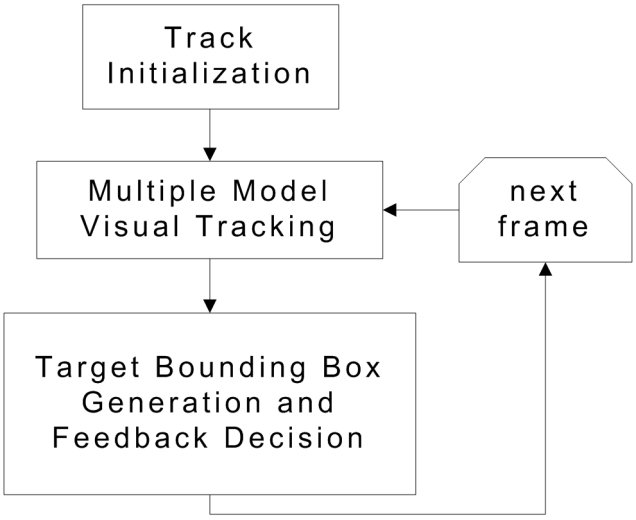

Figure 3.5: Flowchart of the Proposed Tracking System

constructed which are capable of interacting with each other as illustrated at the bottom row of Fig. 3.6 (2^{nd} and 3^{rd} sub-windows). The motivation behind this methodology is that, the filter group which is designed for adapting to gradual changes has the priority to be used as the actual response. When the rapid changes start to occur, the filter group designed to adapt to fast changes becomes active if the filter group for gradual changes cannot satisfy PSR quality requirements. In order to maintain the sustainability of the two filter groups, the filter group for gradual changes starts to be updated with the update parameters of the filter group for fast changes. Hence, interaction between the multiple filter groups is used to tolerate the errors of each other in different conditions by using their corresponding learning rates when one of the filter groups starts to give low quality tracking results. More formal definition of filter update scheme using two filters is given below:

$$F_1(t+1) = \eta_1 F_1(t) + (1-\eta_1) F_{current} \ , \tag{3.17}$$

$$F_1(t+1) = \eta_2 F_1(t) + (1-\eta_2) F_{current} \ , \tag{3.18}$$

$$F_2(t+1) = \eta_2 F_2(t) + (1-\eta_2) F_{current} \ , \tag{3.19}$$

where the F_1 represents the first filter group responsible for gradual changes and F_2 stands for the second filter group designed for abrupt appearance changes. The indexes, t and $t+1$ denote previous and the next filters, respectively. $F_{current}$

represents the currently calculated value of the filter using the current frame. The symbols η_1 and η_2 stands for learning rates for gradual and abrupt changes.

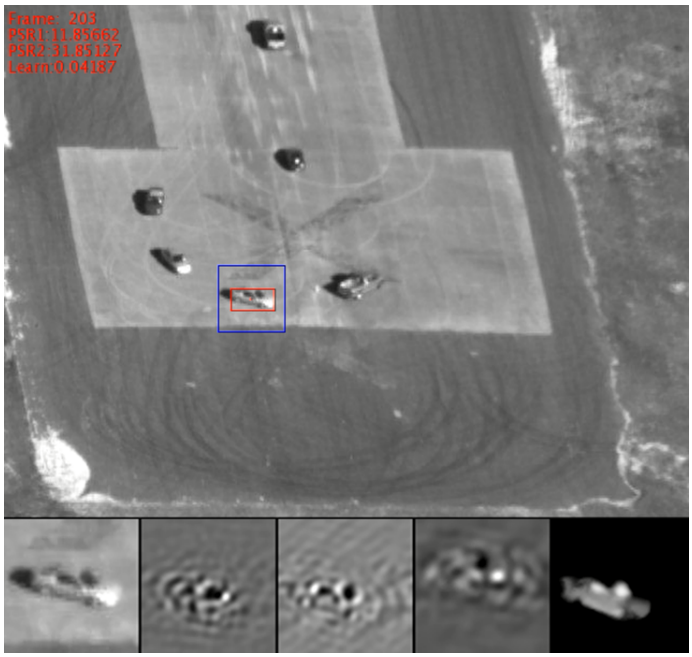

Figure 3.6: Elements of the Multiple Model Target Tracking with Target Size Feedback. Top: Track Window and Target Bounding Box; Bottom: Image patch, Filter 1, Filter 2, Correlation Output, Target Silhouette

The first filter group takes the preprocessed image as input and used for correlation matching in first row since it has the priority. By using the correlation output obtained, PSR is calculated. If the calculated PSR is higher than a predefined threshold, the target location is updated according to the first filter group and the first filter group is updated with low learning rate whereas the second filter group is updated with high learning rate as in Eqn. 3.17 and Eqn. 3.19. However, if PSR of the first filter is less than the predefined threshold, the second filter group generates the target location output by correlation matching with the second filter group. If the quality of the response of the second filter group in the current frame is higher than a predefined threshold, the target location is updated with respect to the second filter group and both of the filter groups are updated with the learning rate of a high learning rate as in Eqn. 3.18 and Eqn. 3.19. If PSR of the second filter response is not high enough compared to the predefined threshold, the system detects occlusion. This procedure is illustrated in Fig. 3.7.

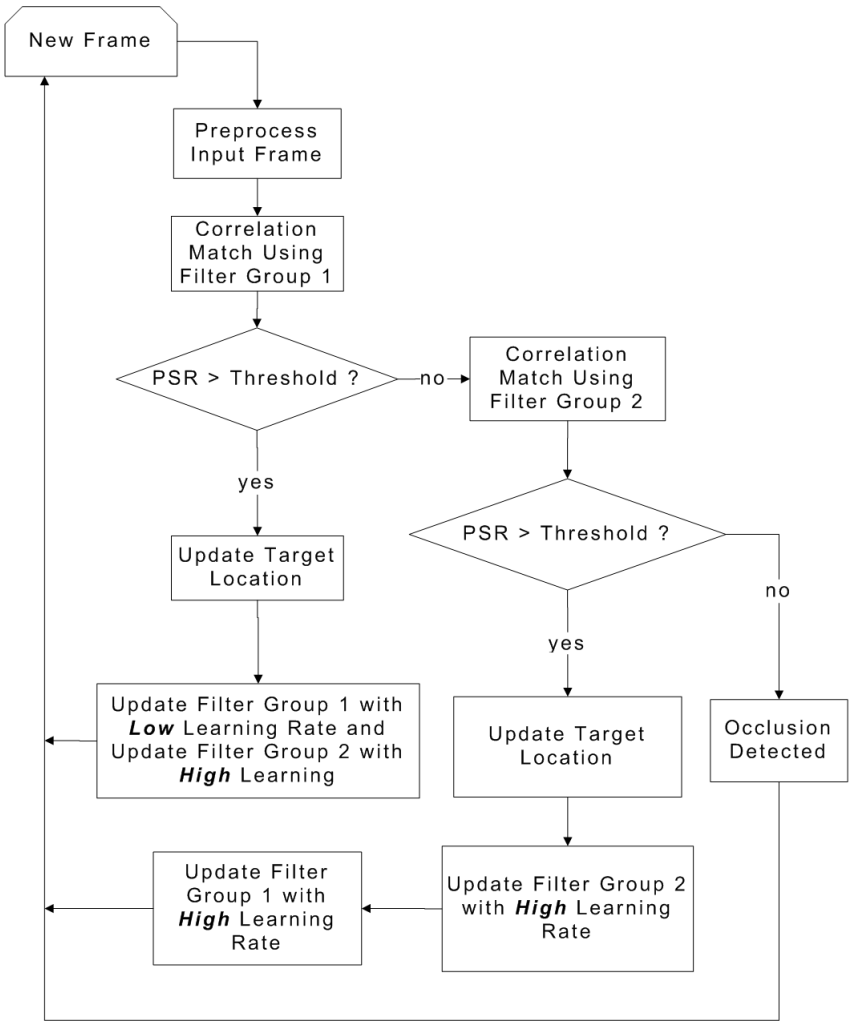

Figure 3.7: Flowchart of Multiple Model Visual Tracking Algorithm

3.2.2 Target Bounding Box Generation

Target bounding box generation actually means target size estimation and provides the ability of detecting and adapting the scale changes more appropriately. Target of interest is known be in the track window during tracking; since track window is selected to be larger than the calculated target size in the target initialization step. In other words, the target of interest is the salient object in the track window. Therefore, the proposed methodology for target size estimation is based on saliency detection and its temporal consistency. The proposed method is divided into three substages: saliency map generation, learning rate calculation and temporal refinement of the saliency map, and target selection.

3.2.2.1 Saliency Map Generation

Despite existence of various saliency detection algorithms [117,118,123–126], the work of [128] is selected in which the saliency problem is tackled from different perspective by focusing on background more than the object. The reason of its selection is its capability of extracting a saliency map within few milliseconds. However, it has two basic assumptions for the input image that should be guaranteed, namely boundary and connectivity. The boundary assumption is reflection of a basic tendency that photographer/cameraman do not crop salient objects among the frames. Therefore, the image boundary is usually background. The connectivity assumption comes from the fact that background regions are generally tend to be large and homogenous, i.e. sky, grass. In other words, most image patches can be easily connected to each other piecewisely. In our case, these two assumptions are fulfilled by simply selecting initial target window including target, roughly centralized, and keeping the target roughly centralized in the track window throughout the tracking. Moreover, the assumption of non-target pixels as the boundary pixels is relaxed by determining non-target pixels from the output of the moving detection mask for the target window.

Satisfying these two conditions, the salient regions are assumed to be the patches, which are extracted by downscaling or by any super pixel extraction algorithm, with high geodesic distance from the boundaries of the image which are assumed to correspond to piecewise-connected background regions. The geodesic saliency of a patch p is the accumulated edge weights along the shortest path from p to virtual background node b in an undirected weighted graph $p \in \{v, \xi\}$,

$$\begin{aligned} Saliency(p) = \arg \min_{p_1=p,p_2,\ldots,p_n=B} \sum_{i=1}^{n-1} weight(p_i, p_{i+1}); \\ s.t.(p_i, p_{i+1}) \in \xi \end{aligned} \tag{3.20}$$

In order to calculate the shortest distance to the background nodes from each patch, the shortest path algorithm [68] is exploited. For shortest path calculation, the original gray-level image *G(x)* is used and the distance map *F(x)* is obtained. In [68] two round iteration is executed. In the first iteration direct video order is used (from top to bottom, and left to right) and F_e^* is calculated

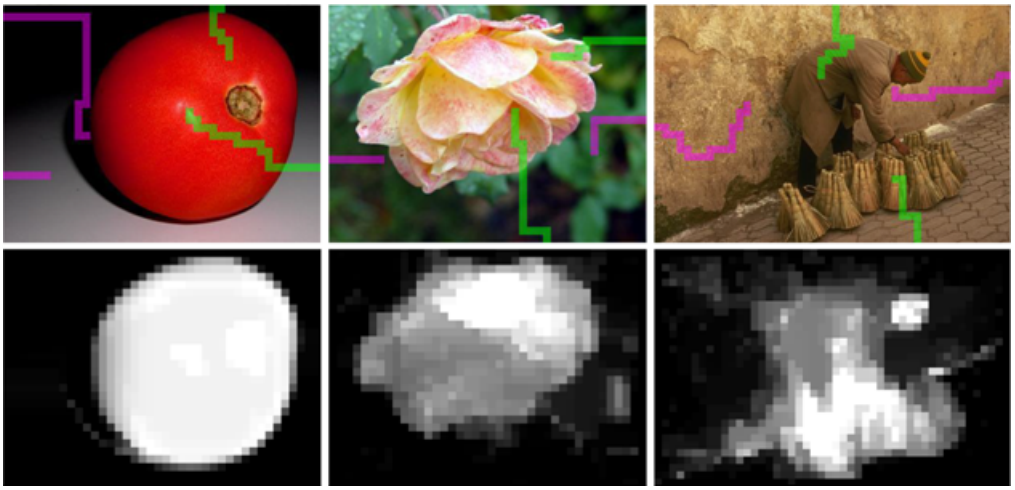

Figure 3.8: Shortest Paths for some Foreground (green) and Background (magenta) Image Patches (retrieved from [128])

according to Eqn. 3.21 and Eqn. 3.22 where * stands for already calculated points. Moreover, the pixel mapping used in Eqn. Eqn. 3.22 and Eqn. Eqn. 3.24 is illustrated in Fig. 3.9.

a	b	c
d	e	f
g	h	k

Figure 3.9: Pixel Mapping used for Shortest Path Calculation

$$F_e^* = min\left[F_e, min\left(1 + da + F_a^*, 1 + db + F_b^*, 1 + dc + F_c^*, 1 + dd + F_d^*\right)\right] \ , \tag{3.21}$$

$$da = \alpha\left|G(e) - G(a)\right|; db = \alpha\left|G(e) - G(b)\right|; dc = \alpha\left|G(e) - G(c)\right|; dd = \alpha\left|G(e) - G(d)\right| \ . \tag{3.22}$$

Also the second iteration process is similar to the first one but in inverse video order (from bottom to up, and from right to left) and the new point F_e^* is calculated according to Eqn. 3.23 and Eqn. 3.24

$$F_e^* = min\left[F_e, min\left(1 + df + F_f^*, 1 + dg + F_g^*, 1 + dh + F_h^*, 1 + dk + F_k^*\right)\right] \ , \tag{3.23}$$

$$df = \alpha\left|G(e) - G(f)\right|; dg = \alpha\left|G(e) - G(g)\right|; dh = \alpha\left|G(e) - G(h)\right|; dk = \alpha\left|G(e) - G(k)\right| \ . \tag{3.24}$$

For solving the Eqn. 3.20 using [68], it is enough to assign weights of background nodes to zero. After two round iterations proceed, shortest path for each patch to background node is calculated which means that saliency map is obtained.

The patches with higher distance measure indicate more salient patches. In this manner it should be noted that, since patches close to the center of the image requires a longer path in order to reach the background, accumulation of weights tend to be larger in the center patches meaning that this method also favors the center image regions as more salient which is very reasonable since salient regions tend to occur near the center of the image.

As it may be noticed, the method of [128] is proposed for one channel images; however our data type also includes 3 channel RGB images. Therefore, this method is adjusted for 3 channel input simply by selecting the maximum distance from each channel as in Eqn. 3.25:

$$F^*_{3-channel} = max\left[F^*_{red}, F^*_{green}, F^*_{blue}\right] \quad . \tag{3.25}$$

3.2.2.2 Learning Rate Calculation and Temporal Refinement of the Saliency Map

Since target tracking is a continual process, it can provide temporal information which can be used for target size estimation. For this purpose, saliency maps of the track windows are calculated at each frame, and referred as current saliency map. Since size estimation based on a single frame could be erroneous; using weighted average of current and previously calculated saliency maps, another saliency map is learnt which is referred as updated saliency map. The important thing is each saliency map may not represent the target with the same quality. Therefore, samples of higher quality should be weighted more in the updated saliency map to better estimate the target size and adapt changes in target appearance. Calculation of the weight for the current saliency map is referred as adaptive learning rate calculation in this context and has two significant benefits: First, due to noise or any imperfection of the sensor data saliency map may deviate from frame to frame. However, learning updated saliency map from deviated versions will extract the common structure of the target silhouette; which means tolerance to the noise. Second, when the target is fully or partially occluded, the abrupt change in the saliency map would be known and learning rate is adjusted in a way to prevent target model that exists before

occlusion.

For learning rate calculation, two parameters namely saliency ratio and correlation score are used which are calculated by using both updated and current saliency maps. Firstly, current saliency map is calculated and binarized for each image. Then the first parameter for learning rate, saliency ratio ($r_s(t)$), is calculated as in Eqn. 3.26 for current saliency map, where dominant components represents the saliency values greater than binarization threshold and $S_{current}(x)$ is the current saliency map.

$$r_s(t) = \frac{\sum_{x \in DominantComponents} S_{current}(x)}{\sum_{x \in SaliencyMap} S_{current}(x)} \quad . \tag{3.26}$$

To be clearer, this metric is designed to measure distinctiveness of the target within the track window. In the cases where only the target has high saliency values, the saliency ratio will be 1 which means the target in scene is very distinctive. Hence, for extraction of target model this frame is very reliable and should be learnt with high learning rate.

The second metric is correlation score, $D_{NCC}(t)$, which is a very strong cue for detection of abrupt changes in the updated target model. To achieve this goal, normalized cross correlation between the target models, selected from updated saliency map using the target selection procedure that will be discussed, and current saliency map is taken as in Eqn.3.27.

$$D_{NCC}(t) = max\left\{NCC\left(S_{current}, TargetModel\right)\right\} \quad . \tag{3.27}$$

Using the saliency ratio and the correlation score, the learning rate is calculated at each frame as in the Eqn. 3.28 and represented with symbol $\lambda(t)$ at time t. Note that ranges of both metrics extends from 0 to 1 and if both are 1 the current target is overwritten to the updated target which is not desired since it clears out all temporal information. In order to prevent this, maximum learning rate is restricted to α. Moreover, in order to prevent mislearning of target, a penalization constant, β, is used whenever target model and the best possible match has resemblance below the feedback threshold, $FB_{threshold}$, which simply means system updates target model whenever the measurement is considered to

be secure.

$$\lambda(t) = \begin{cases} \alpha \cdot r_s(t) \cdot D_{NCC}(t), & D_{NCC}(t) > FB_{threshold} \\ \beta \cdot \alpha \cdot r_s(t) \cdot D_{NCC}(t), & D_{NCC}(t) \leq FB_{threshold} \end{cases} \tag{3.28}$$

After calculation of learning rate, the saliency map is updated, $S_{updated}(t)$, according to the calculated learning rate at each frame as in the Eqn. 3.29. The natural response of such a learning framework is to learn more if the current salient component is worth considering. Moreover, the components which are consistent with the learnt saliency map are also considered to be learnt more.

$$S_{updated}(t) = \lambda(t) \cdot S_{current} + (1 - \lambda(t)) \cdot S_{updated}(t-1) \ . \tag{3.29}$$

Since correlation metric shows resemblance between target and current saliency map, it is also used to answer to the question when the feedback should be given to the visual tracking system. The feedback is used for adjusting size dependent parameters of the tracker that are suppression window size and PSR subwindow size. The formulation in Eqn. 3.30 is used for querying tracking feedback.

$$isFeedBack = (D_{NCC}(t) > FB_{threshold}) \ . \tag{3.30}$$

In Eqn. 3.30, the $IsFeedBack$ variable is a binary variable controlling the decision of giving feedback or not (Give Feedback if 1 else 0). If the correlation score is high enough, then the current saliency map is consistent with the previous behavior of the region of interest. This results in giving feedback to the visual tracking system since it is the signature of a secure measurement.

3.2.2.3 Target Selection

After saliency map update, the target in the saliency map should be extracted. To achieve this goal, target selection procedure is utilized in two steps: binarization and maximization of the regularization energy. The binarization step is intended to obtain the pixels having significant contrast difference with the background in the track window. It should be noted that all of the obtained pixels do not necessarily belong to target since the track window may also contain other moving objects or parts of moving objects that are close to main target.

This is the reason why the second stage is required. In the second stage, the target of interest in each track window is selected by maximizing regularization energy. Then, the target bounding box is outputted as the bounding box of the selected connected component.

Although minimum computational cost is desired in each step, using static threshold or suboptimal methods for binarization may be problematic. Thus, Otsu's method [129] is used with slight refinement. The method of Otsu can be either defined as an exhaustive search for the threshold that either minimizes the within-class variance or maximizes between-class variance. The between-class variance is often calculated as given in Eqn. 3.31:

$$\sigma_B^2 = w_b w_f \left(\mu_b - \mu_f\right)^2 \ , \tag{3.31}$$

where w_b, w_f is referred as class probabilities and μ_b, μ_f are class means. After some manipulations Eqn. 3.31 can be written as in Eqn. 3.32.

$$\sigma_B^2 = w_b \mu_b^2 + w_f \mu_f^2 - \mu^2 \ , \tag{3.32}$$

where μ is the mean value of the histogram. Since the purpose is to calculate the optimal threshold value T that maximizes , the problem can be solved by either inserting Eqn. 3.31 or Eqn. 3.32 into the Eqn. 3.33.

$$\hat{T} = argmax\left\{\sigma_B^2\right\} \ , \tag{3.33}$$

Note that using Eqn. 3.31 and Eqn. 3.32 directly results in Eqn. 3.34 and Eqn. 3.35 respectively;

$$\hat{T} = argmax\left\{\left(\sum_{i=1}^{T} f_i\right)\left(\sum_{i=T+1}^{L} f_i\right)\left(\frac{\sum_{i=1}^{T} i f_i}{\sum_{i=1}^{T} f_i} - \frac{\sum_{i=T+1}^{L} i f_i}{\sum_{i=T+1}^{L} f_i}\right)^2\right\} \tag{3.34}$$

$$\hat{T} = argmax\left\{\frac{\left(\sum_{i=1}^{T} i f_i\right)^2}{\sum_{i=1}^{T} f_i} + \frac{\left(\sum_{i=1}^{T} i f_i\right)^2}{\sum_{i=1}^{T} f_i}\right\} \tag{3.35}$$

where the number of pixels with gray level $i(1 \leq i \leq L)$ is given with f_i As it can be seen using Eqn. 3.35 becomes slightly advantageous since constant μ term is dropped out. This slight modification results in one less multiplication in Eqn. 3.35 than Eqn. 3.34 which results in L less multiplication in exhaustive search used in Otsu's methodology.

After thresholding the saliency map, the connected component maximizing the regularization energy given by Eqn. 3.36, i.e. the most salient region with minimum distance to the center, is selected as the target.

$$argmax_{C_i} \frac{C_i S}{\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}} \ , \tag{3.36}$$

where C_i is the vectorized form obtained by raster scanning the 2D label matrix with values 1 and 0 as foreground and background respectively, S is the saliency map vectorized similarly and (x_i, y_i), (x_c, y_c) are the centers of each connected component and the initial window respectively.

3.2.2.4 Case Study

A possible scenario for target silhouette extraction and bounding box generation is visualized with the aid of Fig. 3.10, Fig. 3.11 and Fig. 3.12. In these figures, the top-left window is dedicated for the original image in which track bounding box, larger rectangle determining region of interest, is visualized together with target bounding box, small rectangle revealing target location and size. The top-middle figure shows the updated saliency map and the top-right figure illustrates the binarization result of the updated saliency map. The bottom-middle figure shows the current saliency map and the bottom-right figure shows the binarization of the current saliency map. The bottom-left figure shows the normalized cross correlation result of the updated and current saliency maps.

The scenario starts with manual target initialization for the pedestrian in the IR image. Using 68 frames the updated saliency map, Fig. 3.10 (top-middle), is learnt from the current saliency maps, bottom-middle. When the target is partly or fully occluded as in the case shown in Fig. 3.11, the updated and current saliency maps would be dissimilar. This is the case, when the temporal consistency is spoilt. The dissimilarity would yield low cross correlation between target model and current saliency map which prevents giving the location feedback. Moreover, the learning rate is decreased with the penalty term β as in Eqn. 3.28 to prevent target model. After 510 frames (Fig. 3.12), the occlusions coming from the trees and the moving person disappear and the system firstly

starts to increase the learning rate due to effect of the saliency ratio metric. Then, this yields increase in correlation score and when the upper condition is satisfied in Eqn. 3.28 system both omits the penalization term in Eqn. 3.28, β, and starts to feedback to the tracker. By this way, the algorithm presented here prevents itself from adverse effects of occlusion and clutter. Fig. 3.12 is also a good example for explanation of target selection mechanism. After, binarization of current saliency map, there exist pixels which do not belong the target of interest but other pedestrian. If the other pedestrian passing by would stay in the scene for longer, his silhouette would be seen in also the updated saliency window. If we assume the current saliency map of the Fig. 3.12 as updated saliency map, the target whose center is near the window center would be selected, since sum of their saliency values are comparable.

Figure 3.10: Updated and Current Saliency Maps with their Binarized Images at the Beginning of the Track

One should note that all these saliency calculations are utilized in the track window which is the output of the visual tracking algorithm and illustrated as outer bounding box in Fig. 3.13. After applying the presented target bounding box generation method, the inner bounding box is obtained.

The whole summary of target bounding box generation and feedback decision steps are illustrated in in Fig. 3.14.

Figure 3.11: Updated and Current Saliency Maps with their Binarized Images after 380 Frames

Figure 3.12: Updated and Current Saliency Maps with their Binarized Images after 510 Frames

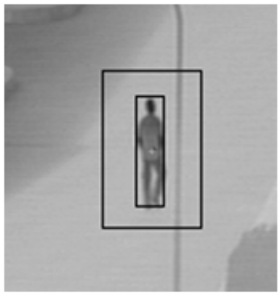

Figure 3.13: Target Bounding Box (inner) together with Track Window (outer)

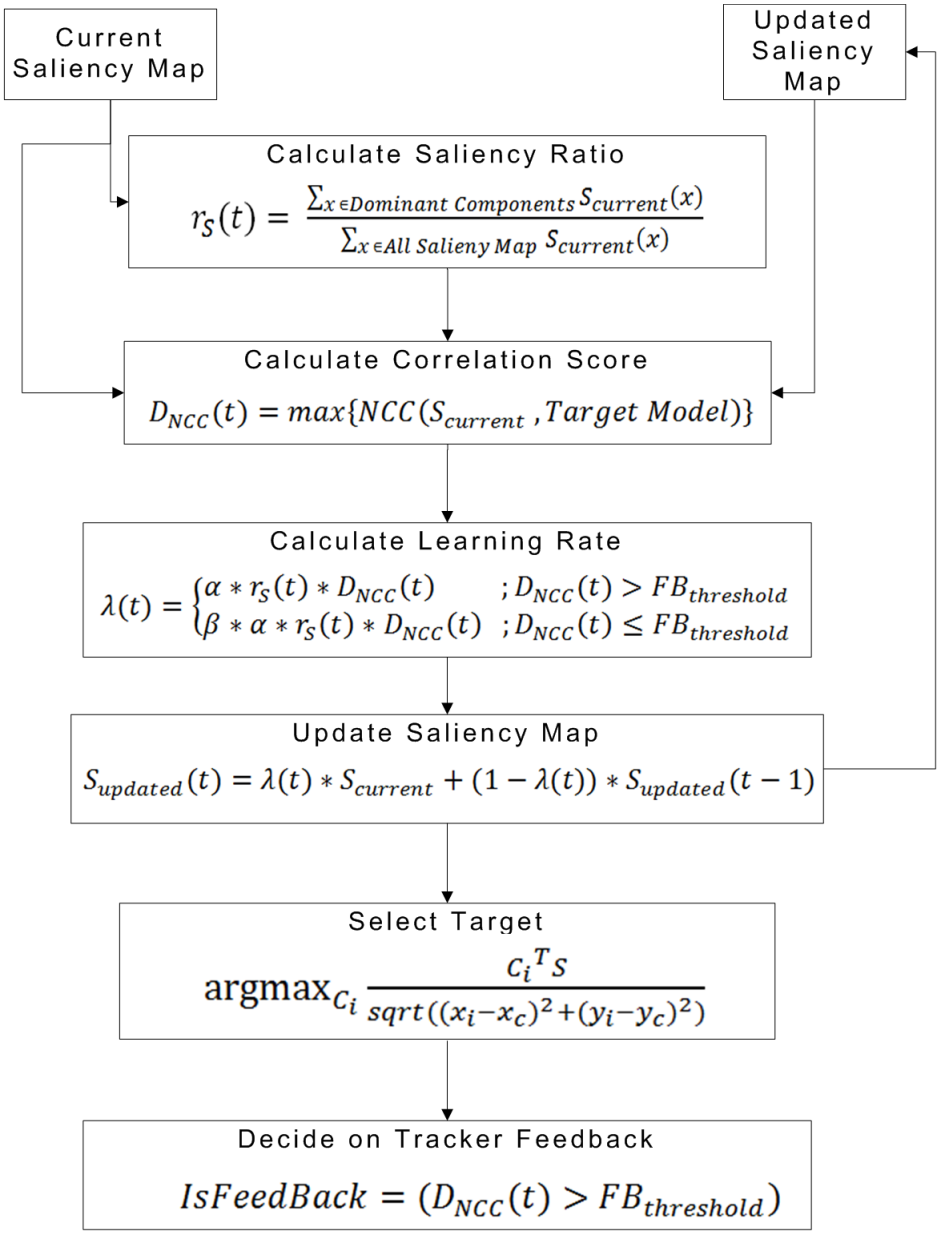

Figure 3.14: Target Bounding Box Generation and Feedback Decision Steps

CHAPTER 4

EXPERIMENTS

The proposed system consists of collaborative operations of two subsystems; moving object detection and object tracking. Therefore, to achieve detailed analysis on system dynamics and evaluate the overall performance, subsystem level tests are as important as integrated system tests. In this manner, the experimental procedure is divided into three stages covering two subsystem and a system test stages.

At the first stage, the performance of the utilized moving object detection subsystem is evaluated quantitatively and compared with other possible options from the literature to reveal strengths and drawbacks of each possible methodology in a detailed fashion. Since time efficiency is an important criteria, the detailed analysis for the time requirements of each possible algorithm is also provided for CPU implementations. Furthermore, the GPU implementation of the winner algorithm is achieved in order to emphasize its extreme suitability for the parallel coding (up to 10x speed ups achieved for SAGMM) which allows winner algorithm to be used in more complex real-time applications.

At the second stage, a very similar procedure is followed for the proposed tracking system. Firstly, the performance of the proposed methodology is compared with other trackers based on correlation filters. The results revealed the success of the proposed methodology in challenging scenarios. Moreover, the required processing times are also given to discuss about their suitability for real-time applications.

Finally, the winner algorithms from previous stages are integrated to achieve vehicle detection and tracking in the urban traffic scenes. To measure overall performance, a real life scenario is practiced which aims to count number of vehicles for each branch road entering and leaving the crossroad separately.

4.1 Results for Moving Object Detection

The moving object detector is designed to fulfill two primary objectives: to obtain approximate size and location of the target for automatic track initialization; and to inform the tracking algorithm with background pixels for satisfying background prior for calculation of the geodesic saliency map which is used for target silhouette extraction. Note that, both of these goals do not necessarily require very precise target boundaries. Thus, all we need is to achieve well separation between targets and enough number of background pixels to use as background prior.

Considering our goals, the preliminary assessment revealed the possible algorithms for our purposes. In this manner, three different algorithms from two different moving object detection categories (frame differencing and background subtraction) are selected for performance comparison. Although frame differencing methods can achieve successful moving object detection for only objects having motion in a predefined speed limits (double/no detection for fast/slow moving objects), first possible option is selected to be a frame differencing method, MHI, due to its simplicity resulting in fast detection. The MHI methodology benefits from the combinations of the object movements from different frames to achieve better target localization than many frame differencing methods. Actually, this algorithm performs sufficient in the scenes without shadow; and with the moving objects having comparable speeds, since the motion history length, L, is selected in accordance with object speed and size. Moreover, due to lack of internal mechanism for background model generation in MHI, post-processing for shadow/highlight detection can only be achieved with more simplistic methods that can decrease the performance.

Two other possible options are selected from statistical background subtraction techniques are ZHGMM [23] and SAGMM [24]. Although both of these algorithms are based on [18], it is not selected for comparison; since superiority of [23] is reported more than once [23, 27]. ZHGMM is one of the most popular background subtraction algorithms and its success is reported many times. Since SAGMM includes additional extensions to ZHGMM which can result in better performance, it is also selected for performance comparison. Both of these methods also share the advantage of background model generation which is the requirement of many high performance shadow/highlight detection algorithms. Due to high computational cost and absence of background model generation, no algorithm from optical flow category is selected for comparison.

4.1.1 Evaluation Data Set

For performance evaluation of moving object detector, 4 different test scenarios from [130] are used. The sequences of the data set were rendered by Mental Ray, a ray tracer provided by Autodesk Maya, while the GT data was generated by Maya Vector. Each sequences is designed to mimics the imagery captured from a stationary camera having resolution of 800x600. To obtain realistic footage, sensor noise is simulated by adding Gaussian distributed ($\mu = 0$, $\sigma = 0.0001$) noise to each frame.

In all sequences, there exists a swaying tree as an uninteresting moving object (background) which is desired to be suppressed. Moreover, the moving shadows of the moving objects also stand as a challenge for each sequence. The selected four test scenarios differ from each other with other additional challenges and named according to their dominant challenges.

Basic: This sequence was generated for achieving general performance overview.
Bootstrap: This scenario is designed for measuring performance of moving object detection without any training frames for learning background. Since we do not use any training frames in any scenario, this sequence is similar for "Basic" scenario for our evaluation.
Camouflage: In the test scenes some objects are poorly differ from the appear-

ance of the background, which makes correct detection difficult. To generate a similar situation car colors and the clothing of the pedestrians are selected to be gray which is similar to road.

No Camouflage: As opposed to camouflage case, more distinctive objects were used.

4.1.2 Performance Measure

In the literature, performance of moving detection algorithms is generally evaluated in pixel level, since the process is considered to be pixel level classification. Thus, we followed the general tendency and expressed the performance of the algorithms by means of true positive (TPR) rate, false positive rate (FPR), precision, and F-Measure.

$$TPR(recall) = \frac{\#\,of\ correctly\,classified\,foreground\,pixels}{\#\,of\ foreground\,pixels\,in\,GT}\ , \tag{4.1}$$

$$FPR = 1 - \frac{\#\,of\ correctly\,classified\,background\,pixels}{\#\,of\ background\,pixels\,in\,GT}\ , \tag{4.2}$$

$$Precision = \frac{\#\,of\ correctly\,classified\,foreground\,pixels}{\#\,of\ pixels\,classified\,as\,foreground}\ , \tag{4.3}$$

$$F - Measure = 2\frac{Recall \cdot Precision}{Recall + Precision}\ . \tag{4.4}$$

4.1.3 Post-Processing

In all moving object detection algorithms the produced foreground masks can be contaminated with some scattered foreground pixels which are generally generated due to noise. Therefore, to achieve unstained results some simple post processing methods are applied after achieving the moving object mask. The procedure followed in this work is summarized in Fig. 4.1. The first row of Fig. 4.1 illustrates the input frame and the stained moving object mask with noise. In order to get rid of salt & paper like noise median filtering is used and followed

by closing operation to achieve close contours for filling the holes inside of the object in the next step. Usage of median filtering and morphological closing operations result in the image at the bottom left of Fig. 4.1. The morphological enhancement step is finalized with filling the holes in the mask. The final output is illustrated at the bottom right of the Fig. 4.1.

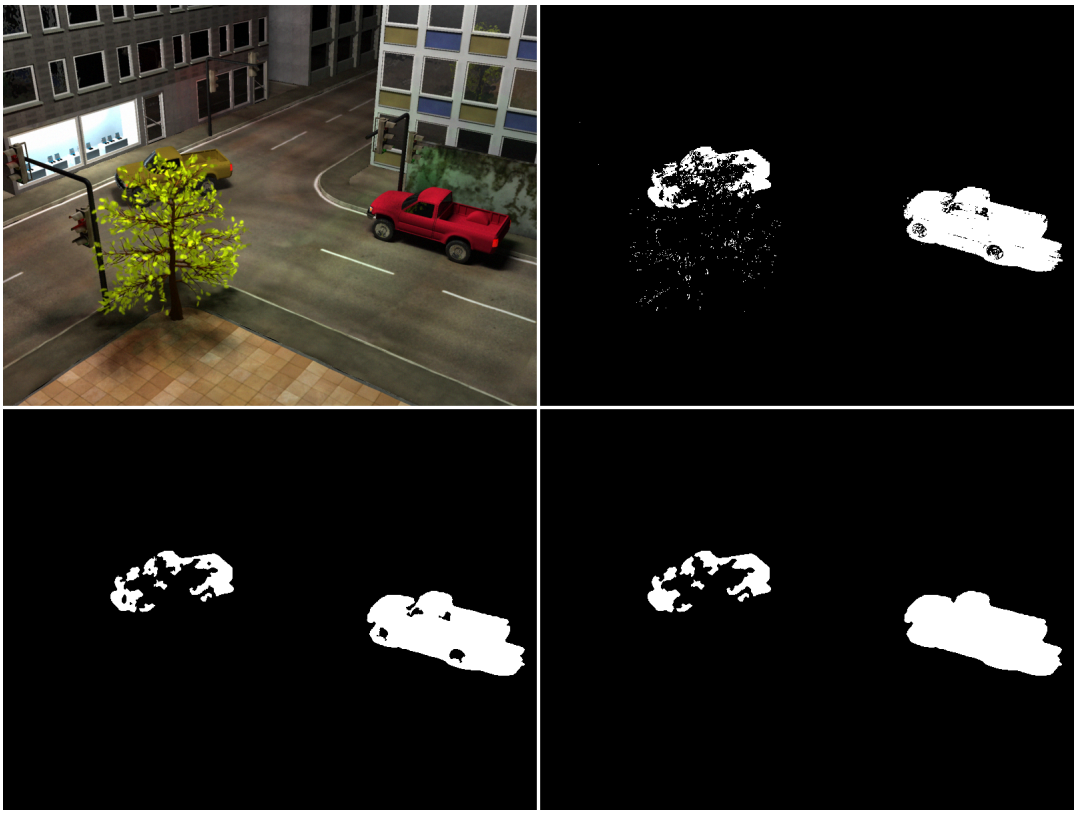

Figure 4.1: First Row: Input Image and the SAGMM Output; Second Row: Output of Median Filtering + Morphological Closing, and Output of Filling Process

As mentioned previously, existence of shadow/highlight pixels can degrade the performance of the moving object detector system. Therefore, another important issue is shadow/highlight removal from the moving object mask. After obtaining the cleaned out moving object mask, a shadow detection algorithm is utilized. Some exemplary outputs for shadow detection can be seen in Fig. 4.2. As it can be seen, shadow detection helps to unmerge targets which mean successful track initialization for both of the targets.

For selection of shadow detection algorithm, we benefit from [35] in which not only the shadow detection taxonomy is introduced, but also five algorithms

Figure 4.2: Input Image, together with Moving Object Detection Mask and Detection Mask after the Shadow Removal Algorithm Utilized

from different classes (two from texture-based and one from each remaining categories) are compared. In comparison, texture based algorithms are divided into two categories as: small region (SR) and large region (LR) texture based method. The comparison is executed with 7 different data sets with various types of moving cast shadows in size, strength and direction. The quantitative results are obtained by using two different metrics, namely shadow detection rate (η) and shadow discrimination rate (ξ), proposed in [36].

$$\eta = \frac{TP_S}{TP_S + FN_S} \ , \tag{4.5}$$

$$\xi = \frac{TP_F}{TP_F + FN_F} \ , \tag{4.6}$$

where TP and FN stand for true positive and false negative pixels with respect to either shadows(S) or foreground objects(F). The shadow detection rate is concerned with labeling the maximum number of cast shadow pixels as shadows. The shadow discrimination rate is concerned with maintaining the pixels that belong to the moving object as foreground. Considering our overall system, misclassifying foreground pixels as shadow may result in splitting the moving object as two which is not desired. Hence, discrimination metric becomes more important for our application. According to the reported results, in [35] large region (LR) texture based method outperforms others in all sequences by obtaining high values of both detection and discrimination rates. The evaluation result of [35] is illustrated in Fig. 4.3. It should be noted that for LR texture based method the discrimination rate is always over *90%* which makes this algorithm suitable for our application. Moreover, by nature of the algorithm the LR texture method is not sensitive to pixel level noise which can be encountered in many scenes.

In [35], comparison of the time performances for 7 different scenarios is also provided which is shown in Table 4.1. Considering both processing time and performance, [64] is selected for post processing step in the proposed system.

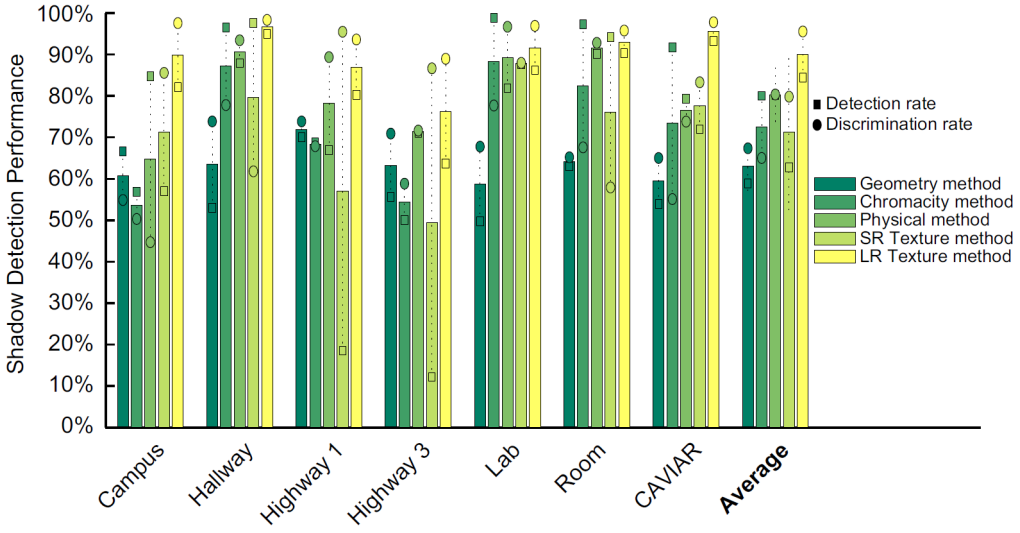

Figure 4.3: Comparison of Shadow Detection Results in Different Scenarios (retrieved from [35])

Table4.1: Average Frame Processing Time (in milliseconds), Reported in [35] for each Sequence

	Geometry [131]	Chromacity [37]	Physical [48]	SR Texture [59]	LR Texture [64]
Campus (352x288)	9.44	8.72	10.00	156.46	20.76
Hallway (320x240)	8.91	11.28	12.81	223.64	21.59
Highway1(320x240)	24.75	10.73	16.93	341.32	34.71
Highway3(320x240)	6.49	6.82	7.15	120.36	11.75
Lab(320x240)	17.68	8.95	15.34	253.82	22.73
Room(320x240)	8.41	7.14	8.51	144.87	16.25
CAVIAR(384x288)	13.94	10.82	14.08	243.07	24.70
Average	**12.80**	**9.21**	**12.12**	**211.93**	**21.78**

4.1.4 Performance Evaluation

For the comparison of three algorithms, the final moving object detection masks which are obtained after post-processing with [64]. However, MHI cannot produce background model which is required by [64]. Hence, for shadow detection algorithm [37] is used instead.

In experiments, the parameters of MHI that are MHI length (L) and distance threshold(T used in the Eqn. 2.5) are selected to be 7, 15 respectively. For evaluation of background subtraction based methodologies, the parameters of ZHGMM and SAGMM are selected to be exactly same for fair comparison. In

both implementations, maximum number of modes is set to 4. The c_f parameter, which measures the maximum portion of the data that can belong to foreground objects without influencing the background model (used in Eqn. 2.16), is set to 0.1 and the complexity reduction prior constant (c_T) is set to 0.05. The distance threshold for ownership is selected to be 9 and the initial variance of the each new mode is given as 11. Finally, the basic learning rate is set to 0.005 for both ZHGMM and SAGMM; however SAGMM automatically alters this parameter during the process. All the defined parameters are adjusted experimentally.

The evaluation results for each scenario are given in the Table 4.2.

Table4.2: Performance Comparison of Algorithms in Different Scenarios

		Basic	*Bootstrap*	*Camouflage*	*No Camouflage*
SAGMM	*TPR*	0.9182	0.8388	0.8609	0.8919
	FPR	0.0070	0.0044	0.0104	0.0052
	Prec.	0.7367	0.8641	0.6647	0.8756
	F-Meas.	***0.8175***	***0.8513***	***0.7502***	***0.8837***
ZHGMM	*TPR*	0.8734	0.8118	0.8386	0.8815
	FPR	0.0075	0.0054	0.0119	0.0060
	Prec.	0.7231	0.8363	0.6465	0.8602
	F-Meas.	0.7912	0.8239	0.7301	0.8707
MHI	*TPR*	0.7306	0.8294	0.8225	0.8592
	FPR	0.0239	0.0074	0.0206	0.0078
	Prec.	0.5495	0.6923	0.5949	0.8199
	F-Meas.	0.6272	0.7547	0.6904	0.8391

According to experimental results SAGMM is achieves the best performance. Considering both experiments and the philosophy of the SAGMM algorithm, two significant advantages are observed to be the key point in the successes. Firstly, if object is allowed to be a part of background, it does not destroy the existing background model. The original background values remain in the GMM if the object remains static for long enough, and its weight becomes larger than c_f. If the object then moves, the distribution describing the previous background still exists with the same estimated mean and variance. Therefore, the objects that are stopped and moved are redetected quickly. Secondly, since each mode has its own adaptive learning rate the model can quickly achieve good estimation of mean and variance. To be more precise if background changes quickly newly introduced modes with smaller c_m will increase learning rate β_m. Otherwise, if the background is stable, as more data samples are included in its parameter

estimation resulting in high c_m, which leads β_m to converge the basic learning rate α. The benefit of using adaptive learning rate is illustrated in Fig. 4.4 and Fig. 4.5. In these figures, the top row is dedicate to visualization of input image and the ground truth while the second row includes the moving object detection masks of SAGMM, ZHGMM and MHI respectively. In Fig. 4.4, while the SAGMM achieves suppression for the swaying tree, ZHGMM cannot accomplish the task at the same number of the frames. However, after 65 frames, Fig. 4.5, it is seen that ZHGMM also achieves the same performance for suppressing tree which emphasizes the ability of SAGMM for faster adaptation. One should also note that neither in Fig. 4.4 nor in Fig. 4.5, MHI cannot successfully suppress the swaying tree due to the erratic motion of the leaves.

Table4.3: Time Measurements for Algorithms and Post-Processing in milliseconds

		Basic	*Bootstrap*	*Camouflage*	*No Camouflage*
SAGMM	*Algorithm*	39.314	38.851	38.754	40.015
	Post-Proc.	131.420	129.741	132.125	138.428
ZHGMM	*Algorithm*	36.981	37.521	37.125	36.512
	Post-Proc.	128.932	132.411	131.045	130.134
MHI	*Algorithm*	11.215	12.047	11.515	11.110
	Post-Proc.	51.413	49.126	47.113	49.278

Another important factor which deserves to be examined is processing time. Although processing time of the SAGMM is proportional to the number of pixels in the frame (video resolution), the shadow/highlight detection algorithm depends on the number of candidate shadow pixels and their connectivity. Thus, the processing time of the shadow detection algorithm can vary from scenario to scenario. However, since the tested data set includes same shadow pattern for each scenario, post processing times are roughly equal for each scenario. The Table 4.3 illustrates time comparison of each algorithm and utilized post processing step. All measurements are obtained with an Intel i7-2670QM CPU @ 2.20 GHz processor.

The timing results indicated in Table 4.3 is high enough to restrain the overall system from real- time execution. To achieve more efficient implementation, we benefitted from the nature of moving object detection algorithms. In all these algorithms, each pixel is processed independently which makes them extremely appropriate for parallel implementations. To both improve time efficiency and

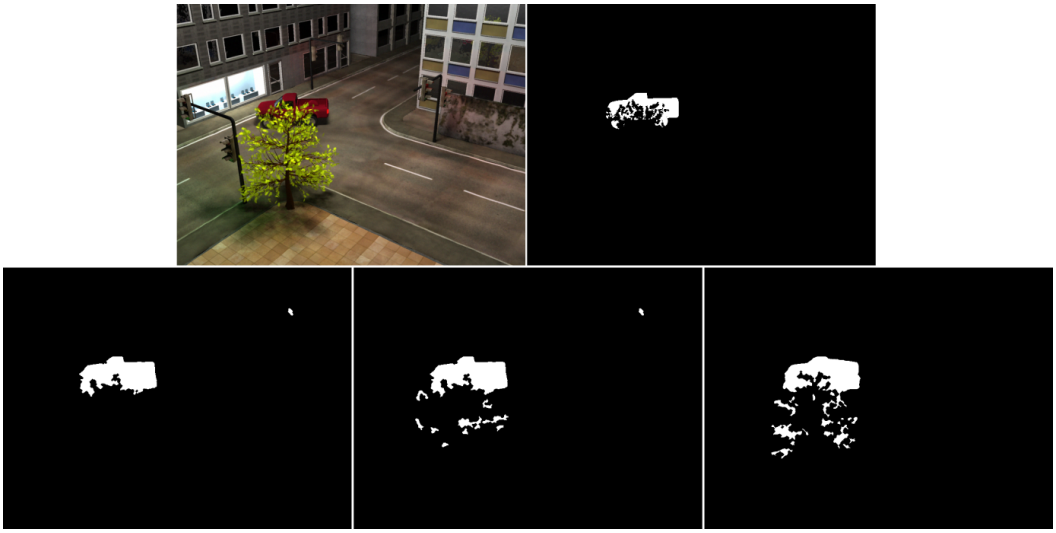

Figure 4.4: Visualization of Input Image, Ground Truth Masks and Algorithm Results for 235th frame of Basic Scenario (Bottom Row: SAGMM, ZHGMM, MHI detection masks respectively)

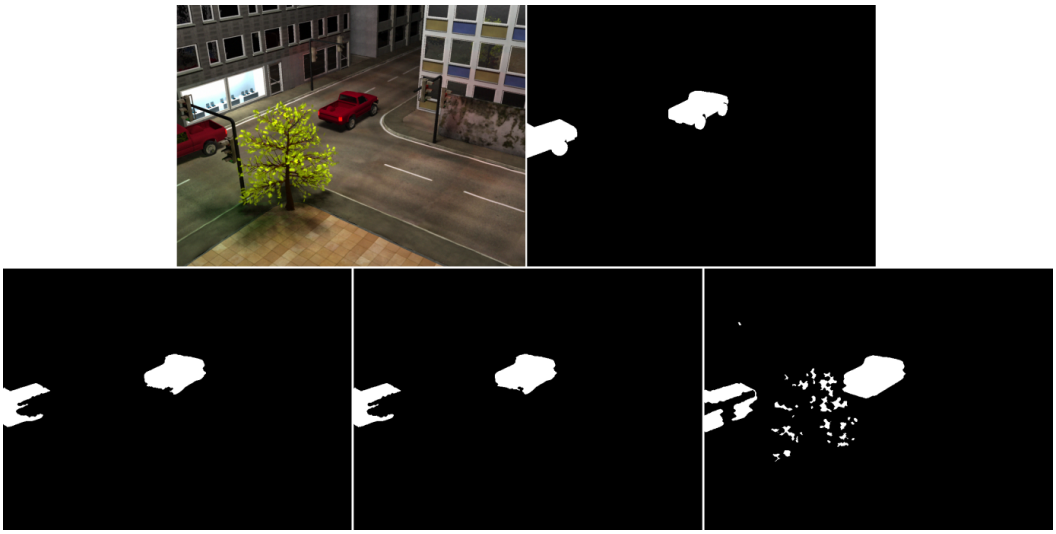

Figure 4.5: Visualization of Input Image, Ground Truth Masks and Algorithm Results for 301th frame of Basic Scenario (Bottom Row: SAGMM, ZHGMM, MHI detection masks respectively)

show the capability of this algorithm in real-time applications, we have implemented SAGMM in an efficient manner by using CUDA. The achieved speeds up for GPU implementation on different image resolutions are illustrated in Table 4.4. The measurements are obtained with NVidia GeForce 660 GTX.

According to results, the GPU implementation achieved up to 10x speed up, which enables usage of SAGMM in real-time applications. The performance comparison of CPU and GPU is also illustrated in Fig. 4.6 to emphasize the

Table4.4: Achieved Time Performances for Different Image Resolution with GPU Implementation of SAGMM

Image Size	*Platform*	*Time(ms)*	*Mpixels/s*	*FPS*	*Speedup*
320x320	*CPU*	7.102	13.750	140.803	6.766
	GPU	1.049	93.037	952.698	
640x640	*CPU*	29.811	13.103	33.544	8.028
	GPU	3.714	105.189	269.284	
960x960	*CPU*	63.626	13.813	15.717	8.788
	GPU	7.240	121.392	138.117	
1280x1280	*CPU*	130.171	12.003	7.682	10.786
	GPU	12.062	124.541	82.906	
1920x1920	*CPU*	260.499	13.495	3.839	10.255
	GPU	25.402	138.396	39.366	

improvement in time performance.

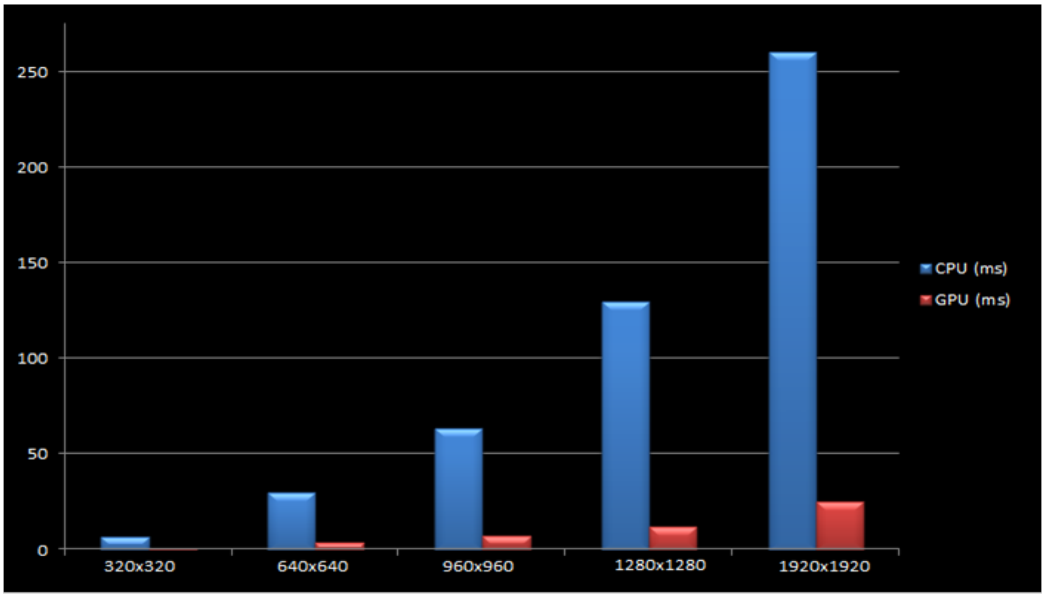

Figure 4.6: Comparison of Time Performances of CPU and CPU Implementations of SAGMM

4.2 Results for Target Tracking

The literature review for object tracking revealed the attractive properties of correlation filters such as shift invariance, robustness to graceful degradation, distortion tolerance, rejection of distractive patterns in complex scenes and low computational complexity. Therefore, the optimized correlation filters are decided to be utilized in the tracking system.

This part is intended to compare the performance of the proposed tracking

algorithm with the 3 most recent correlation filters namely UMACE, ASEF, MOSSE in addition to naïve methodology (match filter).

4.2.1 Evaluation Data Set

For subsystem level test, tracking evaluation datasets are preferred rather than traffic scenes; since tracking datasets are constituted from relatively longer footages. These footages generally includes various challenges for the targets throughout the scene which are designed to disclose strengths and drawbacks of the tracker. Hence, both to achieve better observations on alterations of the correlation filters and not to break the linkage with the intended tracking problem, the VIVID dataset [132] is preferred. The selected dataset includes footages of moving vehicles which frequently experiences similar challenges to those in traffic applications as out of plane rotation, pose variation, occlusions, existence of similar targets in the vicinity. Brief explanations about the main challenges of the test scenarios used from VIVID dataset are given below:

EgTest01: All the vehicles make a U turn which result in rapid appearance changes at the beginning of the scene. Then, one vehicle speeds up and passes nearby others with similar appearance.
EgTest02: Scene include scale change, rotation, a 90 degree sharp turn and very similar vehicles to passing by each other.
EgTest03: This scenario includes low contrast images, and different sized targets which are occluded by very similar vehicles.
EgTest04: Main challenges in this scenario are defocusing of the camera and multiple occlusions by the trees.
pkTest01: The track is interrupted with frequent occlusion by trees, and similar vehicles come close the tracked target at the end of the occlusions.
pkTest02: Vehicles pause at the cross-road and then continue to move with 90 degree sharp turn. Camera gain adjustment also stands as a challenge together with some occlusions by trees.

4.2.2 Performance Measure

Despite existence of multiple possible target vehicles in each scenario in the VIVID dataset [132], the ground truth is only provided for the primary target in each scenario. To increase the number of scenarios that are tested and achieve better evaluation other existing targets in the scenes are also included in the tests. Due to lack of ground truth data for secondary targets, we followed the same procedure used in [1] for performance evaluation. In [1], tracking quality is evaluated by manually labeling the results as good tracking; tracking had drifted off center, or a lost track. The track is described as good track when the track center is within the object; the frame is labeled as drifted track when the track center is located at outside of the object boundary; and the track is stated to be lost whenever track quality (PSR) decreases below the predefined threshold (PSR < 7.0). One exemplary illustration is given in Fig. 4.7 for each situation (good track, drifted track, and lost respectively).

Figure 4.7: Illustrative Examples for Good Track (Left), Drifted Track (Middle), and Track Lost (Right)

4.2.3 Performance Evaluation

In this section, performance of the tracking subsystem is evaluated by comparing its performance with the most recent examples of correlation filters MOSSE, ASEF, UMACE. Moreover, all these algorithms are compared with the match filter in order to observe the robustness introduced by correlation filters.

For the experimental setup, for each filter PSR occlusion threshold is selected to be 7.0, desired output variance is set to 0.2 and the Cosine window is preferred

as windowing function. Moreover, learning rates are selected to be 0.06 for all filters. This learning rate is used as fundamental learning rate (filter group 1) of the proposed tracker and the high learning rate (filter group 2) is selected to be 0.08. In target bounding box generation procedure, the feedback threshold is set to 0.85; the maximum learning rate (α) and the penalization constant (β) is set to 0.1 and 0.3 respectively.

Although we wish to explain the detailed comparison results to disclose strengths and weaknesses of each filter, starting with a general overview would be beneficial for discussing further details. In order to expose the big picture, the overall results are given in the Table 4.5 where the naive filter is obtained by the averaging preprocessed tracking windows with the same online update rate of other filters. All the tests are achieved with fixed sized manual track window initializations.

Table4.5: General Overview of Track Performances of Different Filters

Methodology	*Total Frames*	*Good Track*	*Drifted Track*	*Track Loss*	*Coverage*
Naive	19543	8140	2841	8562	0.5619
UMACE [86]	19543	9340	5150	5053	0.7414
ASEF [80]	19543	11642	3094	4807	0.7540
MOSSE [1]	19543	12785	2825	3933	0.7988
Proposed	19543	16057	1147	2339	***0.8803***

As expected, usage of naïve filter yields the worst performance; since the usage of only image intensity values for correlation can achieve clear correlation peaks as long as the target preserves same structural pattern. Therefore, naïve filter can be easily misguided in the presence of external disturbances which are encountered in the test datasets frequently. Although UMACE and ASEF filter introduces significant increase in track performance and achieves comparable successes, each filter suffers from different issues. UMACE mainly differs from ASEF and MOSSE with the type of desired synthetic output used in the training (filter learning). Thus, the major drawback of UMACE filter comes from the selection of desired output as Kronecker delta which results in learning the target without introducing any perturbations which results in better generalization. On the other hand, MOSSE and ASEF filters select the desired output as a Gaussian to achieve better generalization and focus on more general features. Therefore, UMACE cannot response the relatively fast appearance changes with

the same effectiveness. Although ASEF learns more general features, it also suffers from responding fast appearance changes such as steep turns. However, the reason is quite different. This ill-posed nature of the ASEF is also stressed by the inventor of the MOSSE algorithm in [80]. The convergence of the ASEF filter requires relatively long number of frames than other filters, since it uses averages of the exact filters. Therefore, MOSSE filter achieves better performance by responding changes in a better fashion than ASEF and UMACE by achieving more general features and converging faster. However, its adaptation rate may not be sufficient to capture the relatively fast changes in the scene. To overcome this problem, the adaptation rate may be increased; however this would result in drifting problems since the uncorrelated data can be learnt quickly. In the Fig. 4.8, 4.9, 4.10, 4.11, this situation is illustrated by visualizing the track window and the corresponding filters. The information row at the bottom of Fig. 4.8, 4.9, 4.10 visualizes the image patch in the track window, the filter and the resulting correlation output respectively. As it can be observed from these figures, UMACE, ASEF and MOSSE cannot adapt the changing target appearance during the sharp turn which results in tracking with drifted center.

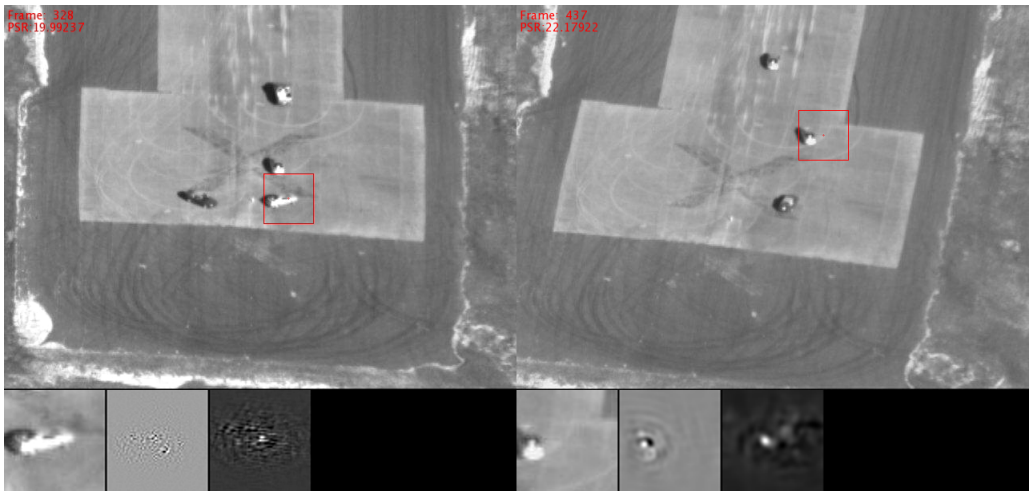

Figure 4.8: Effect of Fast Appearance Change (Sharp Turn) on UMACE filter Update and Corresponding Tracking Result

To overcome this problem, the proposed tracker benefits from two filters with different adaptation rates as illustrated in Fig. 4.11. Since the information row also contains both of the filters (2^{nd} and 3^{rd} image patches) in Fig. 4.11, the benefit of usage of two filters becomes more apparent. The information

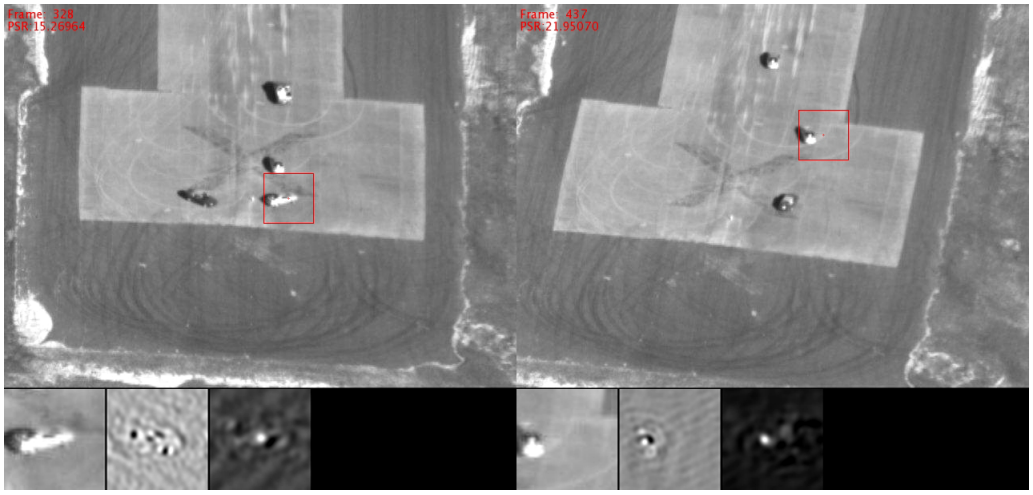

Figure 4.9: Effect of Fast Appearance Change (Sharp Turn) on ASEF filter Update and Corresponding Tracking Result

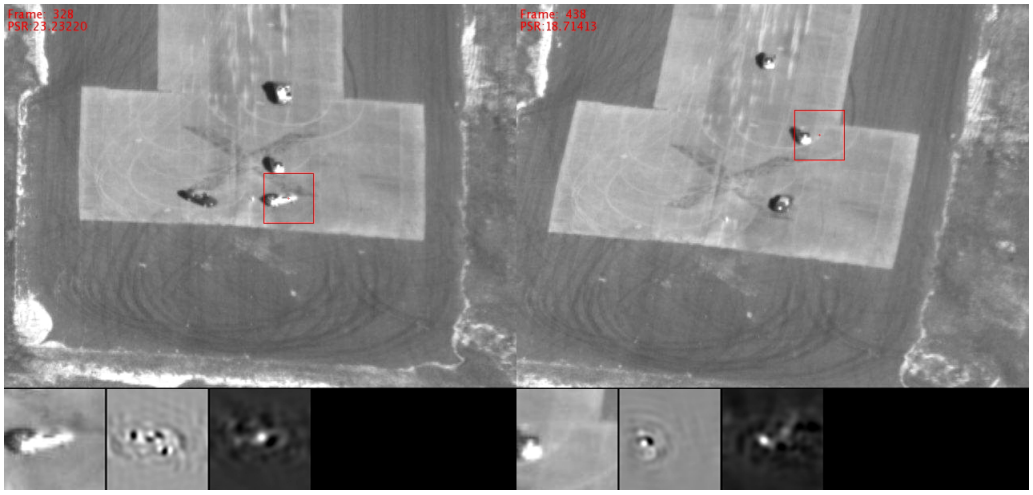

Figure 4.10: Effect of Fast Appearance Change (Sharp Turn) on MOSSE filter Update and Corresponding Tracking Result

row of Fig. 4.11, also illustrates another advantage of the proposed tracker by visualizing the updated saliency map (5^{th} image patch) which is used for target bounding box generation (red rectangle). The target bounding box does not only provide extra information for the overall system, but also helps to improve the tracking performance by adjusting tracking parameters according to size estimation.

To achieve more detailed performance comparison, the tracking results for each target in each scenario are illustrated in Fig. 4.13. In this figure all frames are manually labeled as good track (green), drifted track (yellow), and track lost

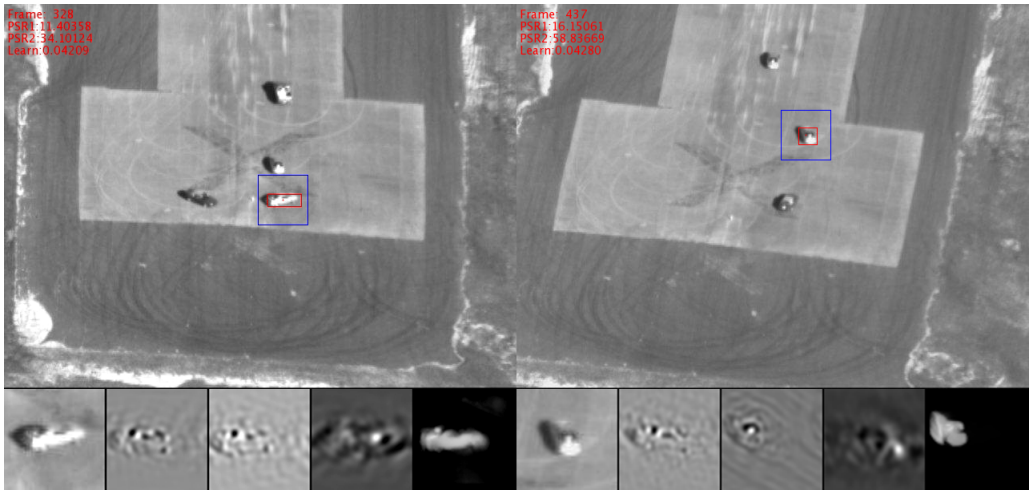

Figure 4.11: Proposed Tracker can Adapt Fast Appearance Change Better by Using Two Filters with Different Learning Rates

(red). The scenario name and target number can be found at the title of each sub graph. Examination of the Fig. 4.13 reveals an important fact. The major difference of the proposed methodology and others is its redetection performance. This situation becomes apparent in the scenarios egTest04-02, pktest02-01, and pktest02-02. This achievement is based on usage of multiple filters with different learning rate which prevents drifting while learning most recent examples of the target appearance. Hence, in the presence of occlusion, more recent examples are searched for which improves the redetection chance of the target.

Another observation from Fig. 4.13 is the unexpected failure of the proposed methodology in scenario egTest03-01. In this scenario, a small sized target (motorcycle) is tried to be tracked. Due to fixed window sized track initialization a much larger track window is used for tracking, Fig. 4.12, which can cause perturbations of the track center by the nature of the algorithm. When this oscillation is combined with the feedback from the size estimator, which limits the suppression window according to size, the filter starts to learn from image patches in which whole target may not exists. This yields mislearning of the target appearance resulting in premature track losses. Therefore, track should be initialized with a window proportional target size. To overcome this problem, general system benefits from the output mask of the moving object detector to achieve track initialization with proportional window sizes.

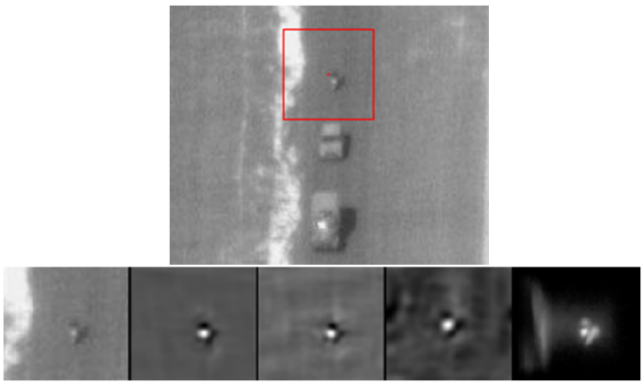

Figure 4.12: Target and Initial Track Sizes should not be Disproportional

Figure 4.13: Tracking Performances of each Filter Type for each Scenario. Each frame is labeled as good track (green), drifted track (yellow), and lost (red).

Since the processing time is considered as an important factor the corresponding timing results are measured from the C++ implementations of the algorithms. All measurements are obtained from the single core of an Intel i7-2670QM CPU @ 2.20 GHz processor and assembled in Table 4.6.

Examining Table 4.6 reveals the fact that increasing the track window size re-

sults in logarithmic increase for the filter training where as target bounding box generation follows a linear path. According to timing results, the tracker allows up to 15 targets when the 64x64 track windows are employed. Although the tracker is not as highly parallelable as the moving object detector case, multiple cores of processors can be used to increase number of tracks.

4.3 Performance Evaluation of Integrated System

After investigating two core elements of the system separately, this section is dedicated to performance analysis of the integrated system. In the light of previous analysis, the integrated system is constituted from the collaborative work of the moving object detector based on SAGMM and the proposed multiple model visual tracker.

In order to measure the performance of the integrated system, a real life scenario is practiced. In this scenario, footages taken from crossroads are automatically analyzed by the proposed system to reveal statistical information about the traffic density of different branch roads entering and leaving a junction. To be more precise, the aim is to detect all vehicles entering the junction and maintain tracking until their exit. When the vehicle is decided to be exiting, its route is revealed, i.e. from branch 1 to branch 3; and vehicle is counted to disclose the information which direction is employed by how many vehicles within a specific time period.

Table4.6: Processing Time (ms) and Number of Possible Tracks for Real-time Performance in 10 fps Video Streams

Track Window Size	*Filter Training MOSSE, ASEF, UMACE*	*Dual Filter Training*	*Target BB Generation*	*# of Multiple Tracks for 10 fps*
32x32	1.112	1.428	1.368	40.3
64x64	1.902	2.372	4.671	15.2
128x128	3.983	4.523	17.688	4.6
256x256	8.669	11.609	75.144	1.2

4.3.1 Evaluation Data Set

For performance evaluation of the system, datasets form two different sources are used. The first dataset [133] was prepared by Institute for Algorithm and Cognitive Systems department of Karlsruhe Institute for Technology and released as publicly available. The selected part of the dataset includes 6401 frames of real data taken with standard PTZ camera containing both grayscale and color images with two similar resolutions that are 768x576, 740x560. The footages are taken in various weather conditions including fog, heavy snow fall and winter time when the surrounding is covered with snow. Moreover, [133] also contains challenges as partial or full occlusions by trees, static objects or other vehicles.

Figure 4.14: Image Sequences Used for Performance Evaluation [133]

The second test image sequence contains 30 minutes of continuing stream of a cross-road that is taken with a fisheye camera. One sample frame with resolution 1920x1920 is illustrated in Fig. 4.15. Although moving object detector does not affected by this type of data, performance of the tracker would be decreased

with the instant appearance changes caused by the fisheye distortion. Therefore, before execution of the data, the fisheye distortion is corrected with the usage of a transform function which is not our interest.

4.3.2 Performance Evaluation

Since the test scenario is defined as counting number of vehicles that are following each possible route, the ultimate success is defined as counting all of the vehicles without producing any false alarm. Therefore, false alarm rate, recall, and precision metrics are selected to be used for the interpretation of the success and calculated in same manner to Eqn. 4.1, 4.2, 4.3.

In order to make this analysis more comprehensible, all the resultant frames are enhanced with symbology. As illustrated in Fig. 4.15, each junction node is marked with an ID starting from 1. The IDs that are illustrated with red represent the entrance nodes (where the vehicle comes from) whereas the blue IDs interpret the exit nodes (where the vehicle heads to). The symbology also contains red and blue loops which are the visualization of the only input required by the user. These loops remark the entrance (red) and exists (blue) of the cross-roads. In this manner, the track initialization is achieved at the entrance node and terminated by counting and revealing the route of the vehicle at the exit node. The number of counted vehicles is shown with a chart (see Fig. 4.15) whose rows and columns represent the IDs of the entrance and exit nodes respectively.

Since the result of the analysis is recorded at the chart, it is used for evaluation as is expected. For evaluation, ground truth charts are obtained for each image sequences. Then, comparing the resultant charts with the ground truth, performance of the system is analyzed. The resulting evaluation scores are visualized in Table 4.7.

Further analysis of the results revealed strengths and weaknesses of the proposed system. The main advantages and disadvantages can be summarized as follows.

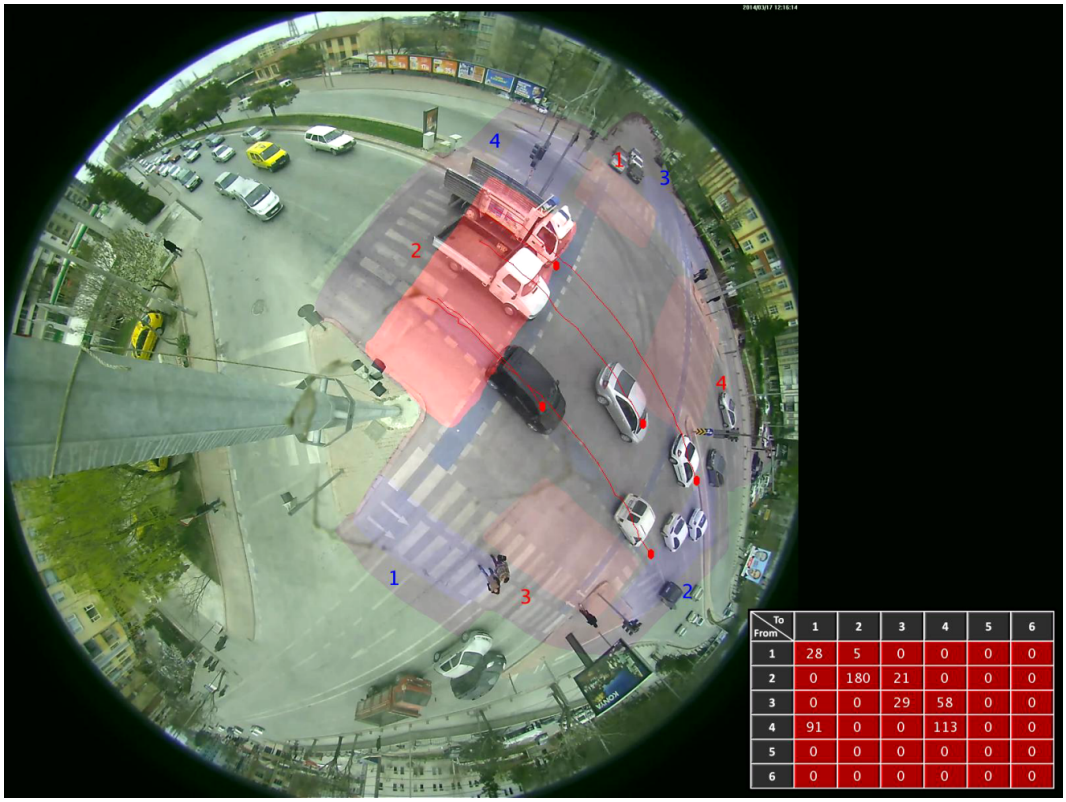

Figure 4.15: An Exemplary Frame from Fisheye Camera Enhanced with Symbology

The first virtue of the system is achieved by the usage of multiple target models. Since the multiple models can respond to appearance changes in different rates, robust tracking is successfully maintained with the targets having different rate of appearance changes as in the case of turning vehicles with different angles as in Fig. 4.16.

The second advantage can be stated as the insuseptibility of the tracker against

Table4.7: Performance Evaluation of each Scenario in Separate and the Overall Result

Scenario	*Vehicles in Total*	*Detected*	*Correctly Counted*	*Mis-detected*	*FAR*	*Recall*	*Prec.*
dtPassat	30	23	23	0	0	0.7667	1.0000
dtPassat03	18	14	14	0	0	0.7778	1.0000
dtneu Nebel	6	5	5	0	0	0.8333	1.0000
dtneu Schnee	9	8	8	0	0	0.8889	1.0000
dtneu Winter	8	8	8	0	0	1.0000	1.0000
kwbB	23	18	16	2	0.0870	0.7826	0.8889
Stau02	23	23	19	4	0.1739	1.0000	0.8261
Fisheye	616	506	464	20	0.0325	0.8214	0.9167
Overall Results	***733***	***605***	***557***	***26***	***0.0355***	***0.8254***	***0.9207***

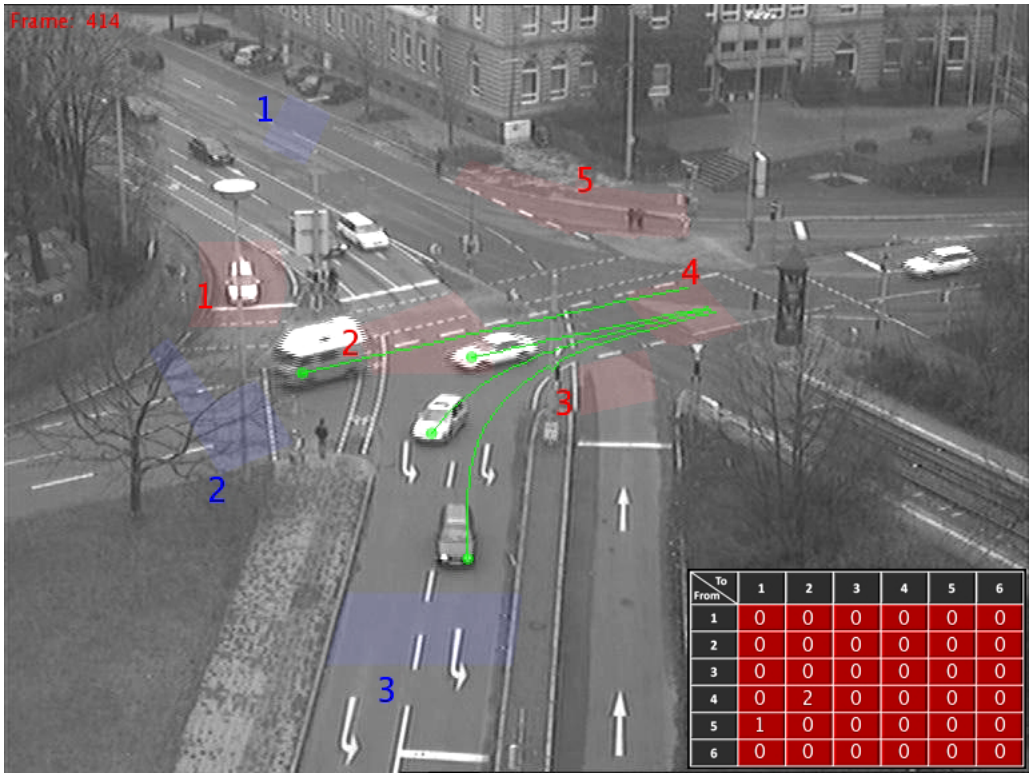

Figure 4.16: Multiple Model Target Tracking can Successfully Respond to Appearance Changes in Different Rates

low contrast which is achieved by the courtesy of preprocessing step introduced in [1]. Hence, the system can successfully deal with targets that are colored similar to surrounding or maintain tracking through the foggy scenes (see Fig. 4.17). Another important topic is occlusion tolerance. Although partial occlusion is observed to be dealt with, the full occlusion cases usually resulted in track losses. The main reason is the winding nature of the branch roads in the junctions. Therefore, the target undergoes appearance change while it is not being tracked resulting in failure of redetection. Some exemplar cases are illustrated in Fig. 4.18.

The experimental results exposed low false alarm rate of the system. However, in severe weather conditions the system could achieve multiple track initializations for single target as in Fig. 4.19. The main reason is the inefficient response of moving object detector on suppression of erratically moving snowflakes which allows them to be piled into an object. Another failure of target initialization manifests itself in crowded scenes in which vehicle to vehicle occlusion occurs.

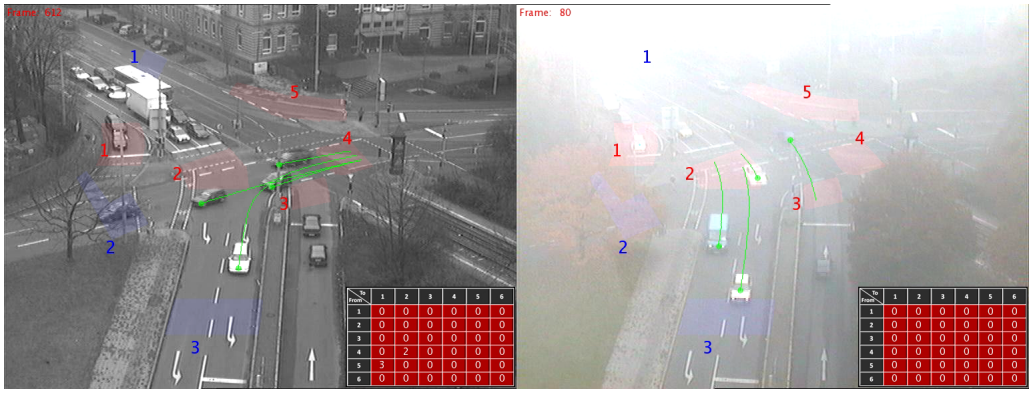

Figure 4.17: Tracking can be Maintained throughout the Scenes Having Low Contrast

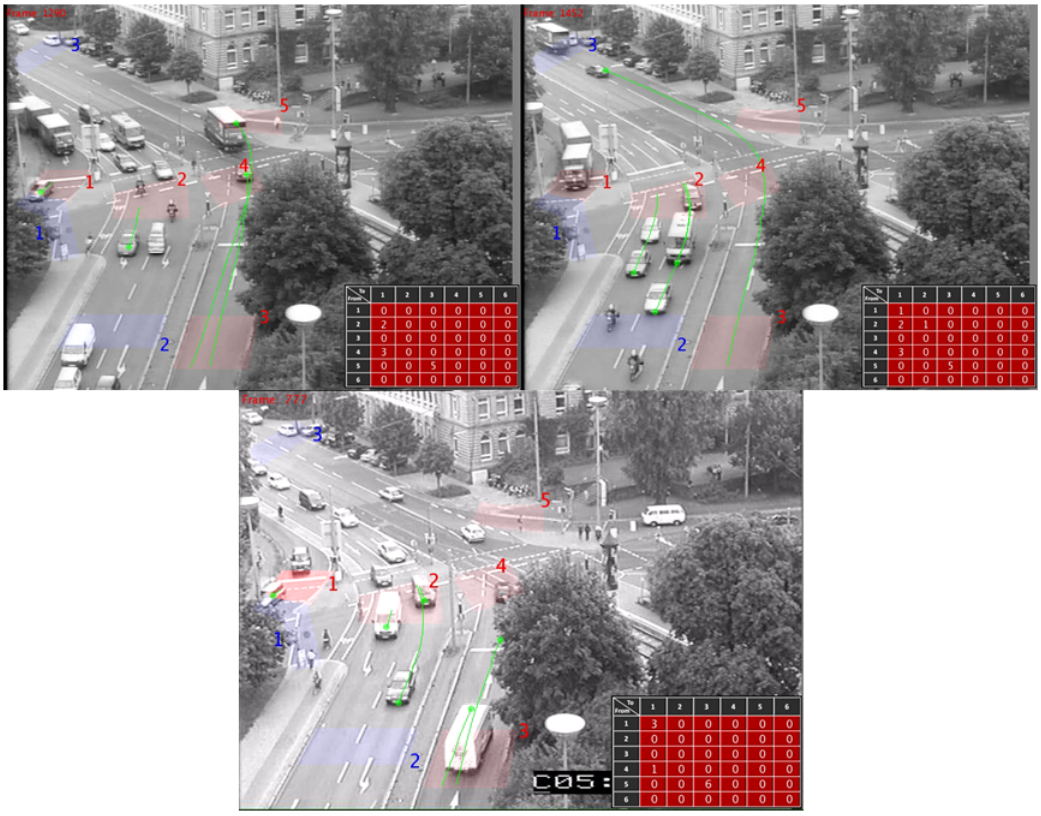

Figure 4.18: Although Partial Occlusion can be Handled (1st row), Full Occlusion Results in Track Failures (2nd row)

Since the target initialization is cued from moving object detection, failure in separation of two close targets would result in single target initialization as shown in Fig. 4.20.

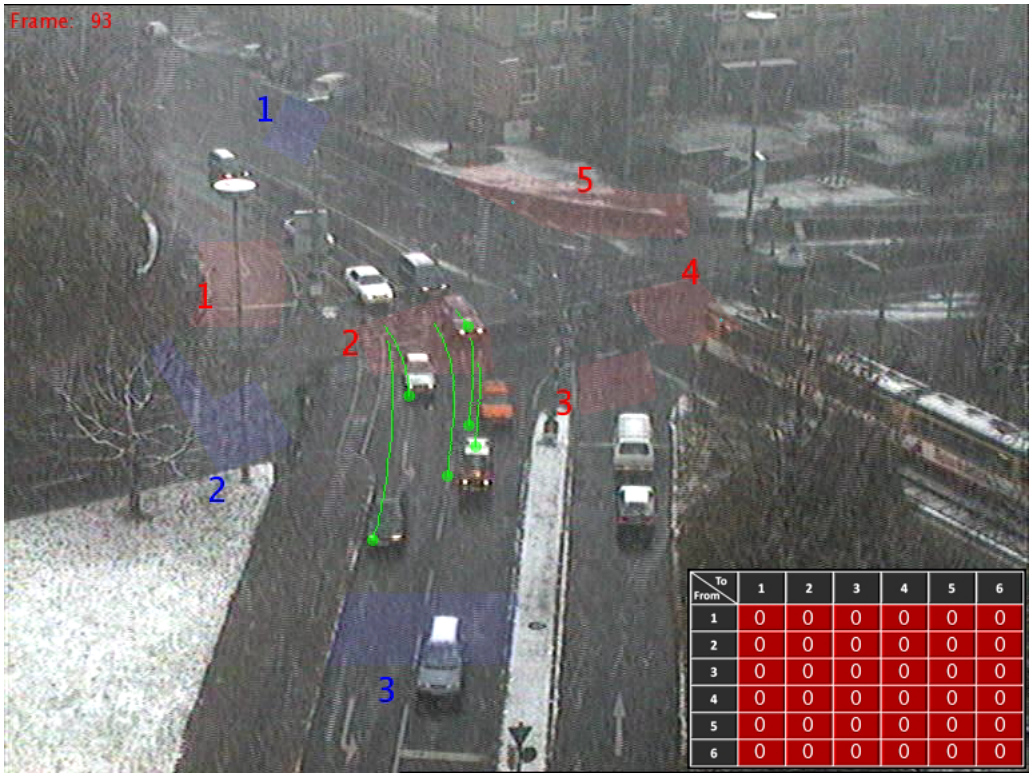

Figure 4.19: Severe Weather Conditions may Result in Multiple Track Initializations for Single Target

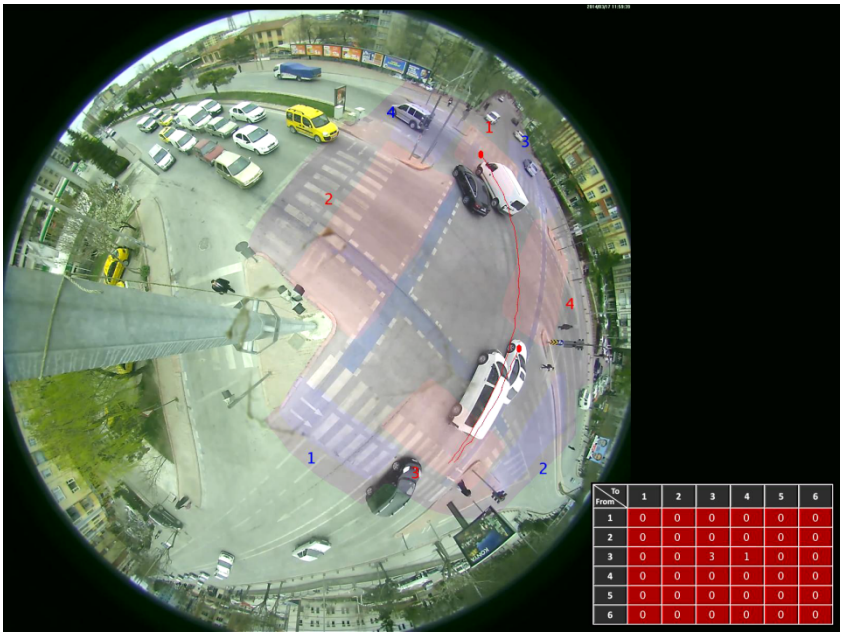

Figure 4.20: Single Track is Initialized for Multiple Targets due to Occlusion

The optimal solution of this problem should be usage of an efficient segmentation technique. However, instead of segmentation our system seeks for a single frame in which two targets are instantly separated. This solution generally works since the targets enter the junction in an order fashion. After track initialization, tracking could be maintained since our tracker is not effected from the presence similar objects in the vicinity. This case is illustrated in Fig. 4.21. However, efficient segmentation of the occluded targets remains as a necessity and noted as a future work.

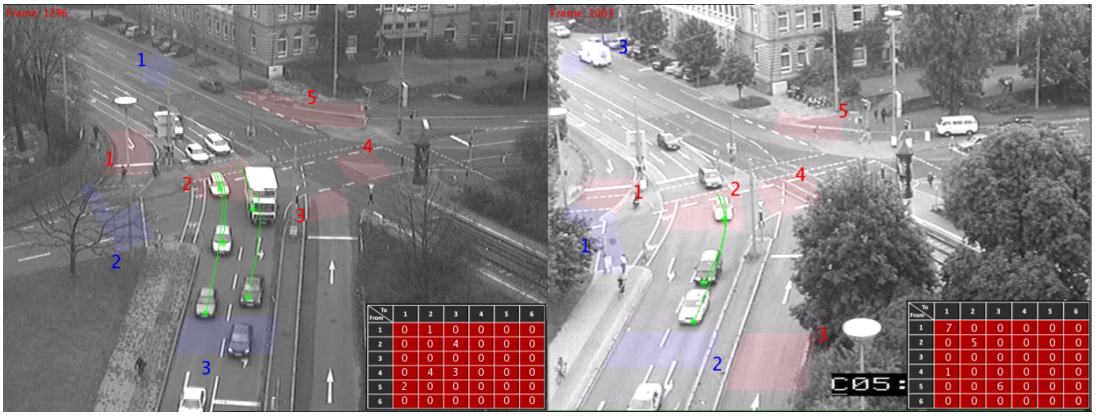

Figure 4.21: Ordered Entrance of the Cross Road Helps Successful Initialization (left); Tracking is not Effected with by Similar targets in the vicinity (Right)

CHAPTER 5

CONCLUSIONS

5.1 Summary

There is an increasing demand for automated intelligent transportation systems in order to manage and improve the traffic flow; since the number of vehicles in the traffic continue to increase. To satisfy the needs, this study presents a solution for main building blocks of automated ITS systems which are moving vehicle detection and tracking. Although the proposed system is tested on the crossroads in urban traffic scenes, no experimental findings exists for restricting its usage to urban scenes.

As expectedly, the presented algorithm consists of two subsystems as moving object detector and target tracker that are working collaboratively. For moving object detection, the Self-Adaptive GMM is employed which is capable of not only selecting the number of modes but also the learning rate of the each mode online. Thus, the highly adaptive nature of the SAGMM resulted in efficiently dealing with illumination changes and repetitive clutter. However, like the output of all moving object detectors the output of the SAGMM is also stained by the noise. Hence, to minimize the effect of noise median filtering is followed by the morphological closing and filling.

Another post processing requirement for shadow/highlight detection also arises; since the shadow/highlight pixels are also classified as moving objects. To fulfill the goal, large region texture based moving shadow/highlight detection is utilized which benefits the fact that shadows do not change the texture. The main

reason of the utilization of this methodology is its capability of shadow/highlight removal in region based which prevents separation of single vehicle into multiple segments resulting in multiple track initializations for a single vehicle.

After post processing, obtained moving object detection mask is used for track initialization. The track is initialized for each vehicle entering the scene and intended to be maintained until exiting the scene. For target tracking, a multiple model visual target tracking algorithm is proposed based on the correlation filters. Since some of the track parameters depend on the target size in correlation filters, the tracker is also feedback with the target bounding box information which is extracted based on the consistency of the visual saliency of the track window. Throughout the tracking, saliency maps of the track window at each frame are learned with an adaptive learning rate which yields the target silhouette; since the target is generally the salient object in the track window. This extra information is very valuable and can also be used in higher level activities such as classification.

5.2 Discussions

The proposed algorithm is tested in 3 stages including separate subsystem level tests for moving object detector and tracker; and test for integrated system. Based on these experiments many observations are obtained.

The first subsystem level test is conducted for moving object detection and results revealed that methods based on statistical background subtraction techniques achieves better performance than the frame differencing techniques for our problem due to their capability of adapting the scene better and excluding repetitive motions. Moreover, based on the experiments SAGMM proved to be more successful than the ZHGMM for adapting scene changes which is illustrated in Section 4.1.4; since SAGMM is capable of adjusting its adaptation rate. Another important fact obtained at the first stage of the experiments is about shadow/highlight detection techniques. Since chromacity based methods achieves classification in pixel level, they can result in splitting the targets re-

sulting in multiple track initialization. However, the LR Texture based method can preserve targets as one blob since the shadow/highlight decisions are made for connected components of shadow candidates. Hence, this approach makes usage of LR Texture based shadow detection to be safer. Another important result is the fact that SAGMM can be speed up to the 10 times when they are implemented on GPU. This speedup enables their usage in real-time applications either with higher resolutions or together with more complex algorithms.

At the second stage, the proposed target tracking algorithm is tested on over challenging sequences containing 19500 frames and compared with the naïve and three most recent examples of correlation filters UMACE, ASEF, MOSSE. According to results the naive filters showed the worst performance since it cannot produce clear correlation peaks with the perturbed templates. Moreover, incapability of controlling the whole correlation plane also makes the naive method vulnerable to distructive patterns in the scene. Based on experimental results it is proved that selection of the correlation filter type effects the tracking performance significantly. In this manner, usage of UMACE cannot respond to relatively fast appearance changes efficiently due to selection of the desired output as Kronecker delta function which limits the generalization. The other filter type ASEF also shares the same disadvantage with UMACE and cannot respond the appearance changes as well as MOSSE due to its convergence issue. The experimental results also showed that adaptation success of the MOSSE is improved with the proposed algorithm and the proposed algorithm is achieved responding appearance changes in different rates as illustrated in Section 4.3.2. Although the experiments cannot fully prove, our findings also give the insights of the improved redetection performance.

At the final stage of the experiments, the integrated system is experienced on a real life scenario which aims to count number of vehicles for each branch road entering and leaving a crossroad separately. The tests are conducted on challenging scenarios including severe weather conditions. The experiments disclosed some advantages including responding different rates of appearance changes, insusceptibility of the low contrast and ability of handling partial occlusions.

All in all, the proposed system successfully achieves the low level tasks of automatic moving object detection and tracking starting from the entrance nodes of the scene until the exit nodes. The only user input required by the system is the definition of the entrance and exit nodes. The proposed solution is applicable to different tasks such as vehicle counting, congestion detection, average traffic flow speed estimation.

5.3 Future Work

As some future work, shadow/highlight removal process can be achieved in a faster way since the timing results showed that it is the bottleneck of the process. More importantly, the detection performance is degraded significantly due to occlusion which results in one track initialization for multiple targets. To cope with this issue an efficient segmentation methodology should be integrated to the system.

REFERENCES

[1] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2544–2550. IEEE, 2010.

[2] Thakore Kinjal A Joshi, Darshak G. A. survey on moving object detection and tracking in video surveillance system. *International Journal of Soft Computing and Engineering (IJSCE)*, 2(3), July 2012.

[3] J. Anitha Merin Antony A. A. survey of moving object segmentation methods. *International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)*, 1(4), October 2012.

[4] Khaled M. El-Sayed Shireen Y. Elhabian and Sumaya H. Ahmed. Moving object detection in spatial domain using background removal techniques - state-of-art. *Recent Patents on Computer Science*, 1:32–54, 2008.

[5] Alan J Lipton, Hironobu Fujiyoshi, and Raju S Patil. Moving target classification and tracking from real-time video. In *Applications of Computer Vision, 1998. WACV'98. Proceedings., Fourth IEEE Workshop on*, pages 8–14. IEEE, 1998.

[6] Alexander Strehl and JK Aggarwal. Detecting moving objects in airborne forward looking infra-red sequences. In *Computer Vision Beyond the Visible Spectrum: Methods and Applications, 1999.(CVBVS'99) Proceedings. IEEE Workshop on*, pages 3–12. IEEE, 1999.

[7] Robert T Collins, Alan Lipton, Takeo Kanade, Hironobu Fujiyoshi, David Duggins, Yanghai Tsin, David Tolliver, Nobuyoshi Enomoto, Osamu Hasegawa, Peter Burt, et al. *A system for video surveillance and monitoring*, volume 2. Carnegie Mellon University, the Robotics Institute Pittsburg, 2000.

[8] Aaron F. Bobick and James W. Davis. The recognition of human movement using temporal templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(3):257–267, 2001.

[9] G.R. Bradski and J. Davis. Motion segmentation and pose recognition with motion history gradients. In *Applications of Computer Vision, 2000, Fifth IEEE Workshop on.*, pages 238–244, 2000.

[10] Zhaozheng Yin and Robert Collins. Moving object localization in thermal imagery by forward-backward motion history images. In *Augmented Vision Perception in Infrared*, pages 271–291. Springer, 2009.

[11] Surendra Gupte, Osama Masoud, Robert FK Martin, and Nikolaos P Papanikolopoulos. Detection and classification of vehicles. *Intelligent Transportation Systems, IEEE Transactions on*, 3(1):37–47, 2002.

[12] Chung-Lin Huang and Wen-Chieh Liao. A vision-based vehicle identification system. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 4, pages 364–367, 2004.

[13] N.K. Kanhere and S.T. Birchfield. Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features. 9(1):148–160, 2008.

[14] N.K. Kanhere, S.J. Pundlik, and S.T. Birchfield. Vehicle segmentation and tracking from a low-angle off-axis camera. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 1152–1157, 2005.

[15] Xin Chen and Chengcui Zhang. Vehicle classification from traffic surveillance videos at a finer granularity. In *Advances in Multimedia Modeling*, pages 772–781. Springer, 2006.

[16] B. Morris and M. Trivedi. Robust classification and tracking of vehicles in traffic video streams. In *Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE*, pages 1078–1083, 2006.

[17] Xiaoyuan Su, Taghi M Khoshgoftaar, Xingquan Zhu, and Andres Folleco. Rule-based multiple object tracking for traffic surveillance using collaborative background extraction. In *Advances in Visual Computing*, pages 469–478. Springer, 2007.

[18] Chris Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, 1999.

[19] Pakorn KaewTraKulPong and Richard Bowden. An improved adaptive background mixture model for real-time tracking with shadow detection. In *Video-Based Surveillance Systems*, pages 135–144. Springer, 2002.

[20] D.-S. Lee. Effective Gaussian mixture learning for video background subtraction. 27(5):827–832, 2005.

[21] C. Kamath S.S. Cheung. Robust techniques for background subtraction in urban traffic video. *Video Communications and Image Processing*, 5308(1):881–892, 2004.

[22] M. Haque, M. Murshed, and M. Paul. Improved Gaussian mixtures for robust object detection by adaptive multi-background generation. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4, 2008.

[23] Zoran Zivkovic and Ferdinand van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780, 2006.

[24] Zezhi Chen and Tim Ellis. Self-adaptive gaussian mixture model for urban traffic monitoring system. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 1769–1776. IEEE, 2011.

[25] Nuria M Oliver, Barbara Rosario, and Alex P Pentland. A bayesian computer vision system for modeling human interactions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):831–843, 2000.

[26] Massimo Piccardi. Background subtraction techniques: a review. In *Systems, man and cybernetics, 2004 IEEE international conference on*, volume 4, pages 3099–3104. IEEE, 2004.

[27] Sebastian Brutzer, Benjamin Hoferlin, and Gunther Heidemann. Evaluation of background subtraction techniques for video surveillance. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1937–1944. IEEE, 2011.

[28] H. Jin F. L. Bu, R. Wang. Moving objects detection and tracking based on optical flow. *Journal of Chinese Peoples Public Security University(Science and Technology)*, 2:58–60, 2009.

[29] Masayuki Yokoyama and Tomaso Poggio. A contour-based moving object detection and tracking. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pages 271–276. IEEE, 2005.

[30] J.Bu M.Zhao and C.Chen. Semi- automatic video object segmentation basing on hierarchy optical flow. *SPIE: Electronic Imaging and Multimedia Technology III*, 4925:307–316, October 2002.

[31] Berthold K Horn and Brian G Schunck. Determining optical flow. In *1981 Technical Symposium East*, pages 319–331. International Society for Optics and Photonics, 1981.

[32] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *Proc. 7th Int. Joint Conf. Artificial Intelligence*, pages 674–679, 1981.

[33] Stefan M Karlsson and Josef Bigun. Lip-motion events analysis and lip segmentation using optical flow. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 138–145. IEEE, 2012.

[34] Andrés Bruhn, Joachim Weickert, Christian Feddern, Timo Kohlberger, and Christoph Schnorr. Variational optical flow computation in real time. *Image Processing, IEEE Transactions on*, 14(5):608–615, 2005.

[35] Andres Sanin, Conrad Sanderson, and Brian C Lovell. Shadow detection: A survey and comparative evaluation of recent methods. *Pattern recognition*, 45(4):1684–1695, 2012.

[36] Andrea Prati, Ivana Mikic, Mohan M Trivedi, and Rita Cucchiara. Detecting moving shadows: algorithms and evaluation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(7):918–923, 2003.

[37] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts, and shadows in video streams. 25(10):1337–1342, 2003.

[38] Elena Salvador, Andrea Cavallaro, and Touradj Ebrahimi. Cast shadow segmentation using invariant color features. *Computer vision and image understanding*, 95(2):238–259, 2004.

[39] Andrea Cavallaro, Elena Salvador, and Touradj Ebrahimi. Shadow-aware object-based video processing. In *Vision, Image and Signal Processing, IEE Proceedings-*, volume 152, pages 398–406. IET, 2005.

[40] Chun-Ting Chen, Chung-Yen Su, and Wen-Chung Kao. An enhanced segmentation on vision-based shadow removal for vehicle detection. In *Green Circuits and Systems (ICGCS), 2010 International Conference on*, pages 679–682. IEEE, 2010.

[41] Bangyu Sun and Shutao Li. Moving cast shadow detection of vehicle using combined color models. In *Pattern Recognition (CCPR), 2010 Chinese Conference on*, pages 1–5. IEEE, 2010.

[42] Sohail Nadimi and Bir Bhanu. Physical models for moving shadow and object detection in video. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(8):1079–1087, 2004.

[43] Fatih Porikli and Jay Thornton. Shadow flow: A recursive method to learn moving cast shadows. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 891–898. IEEE, 2005.

[44] Zhou Liu, Kaiqi Huang, Tieniu Tan, and Liangsheng Wang. Cast shadow removal combining local and global features. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, 2007.

[45] N. Martel-Brisson and A. Zaccarin. Learning and removing cast shadows through a multidistribution approach. 29(7):1133–1146, 2007.

[46] N. Martel-Brisson and A. Zaccarin. Kernel-based learning of cast shadows from a physical model of light sources and surfaces for low-level segmentation. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[47] A.J. Joshi and N.P. Papanikolopoulos. Learning to detect moving shadows in dynamic environments. 30(11):2055–2063, 2008.

[48] Jia-Bin Huang and Chu-Song Chen. Moving cast shadow detection using physics-based features. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2310–2317, 2009.

[49] Jun-Wei Hsieh, Wen-Fong Hu, Chia-Jung Chang, and Yung-Sheng Chen. Shadow elimination for effective moving object detection by gaussian shadow modeling. *Image and Vision Computing*, 21(6):505–516, 2003.

[50] A. Yoneyama, C.H. Yeh, and C.-C.J. Kuo. Moving cast shadow elimination for robust vehicle extraction based on 2D joint vehicle/shadow models. In *Advanced Video and Signal Based Surveillance, 2003. Proceedings. IEEE Conference on*, pages 229–236, 2003.

[51] Henri Nicolas and Jean-Marie Pinel. Joint moving cast shadows segmentation and light source detection in video sequences. *Signal processing: Image communication*, 21(1):22–43, 2006.

[52] Liu Zhi Fang, Wang Yun Qiong, and You Zhi Sheng. A method to segment moving vehicle cast shadow based on wavelet transform. *Pattern Recognition Letters*, 29(16):2182–2188, 2008.

[53] Chia-Chih Chen and Jake K Aggarwal. Human shadow removal with unknown light source. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 2407–2410. IEEE, 2010.

[54] Omar Javed and Mubarak Shah. Tracking and object classification for automated surveillance. In *Computer Vision,ECCV 2002*, pages 343–357. Springer, 2002.

[55] Dong Xu, Xuelong Li, Zhengkai Liu, and Yuan Yuan. Cast shadow detection in video segmentation. *Pattern Recognition Letters*, 26(1):91–99, 2005.

[56] Ying-Li Tian, Max Lu, and Arun Hampapur. Robust and efficient foreground analysis for real-time video surveillance. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 1182–1187. IEEE, 2005.

[57] Yang Wang, Kia-Fock Loe, and Jian-Kang Wu. A dynamic conditional random field model for foreground and shadow segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(2):279–289, 2006.

[58] Wei Zhang, Xiang Zhong Fang, and Yi Xu. Detection of moving cast shadows using image orthogonal transform. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 1, pages 626–629. IEEE, 2006.

[59] Alessandro Leone and Cosimo Distante. Shadow detection for moving objects based on texture analysis. *Pattern Recognition*, 40(4):1222–1233, 2007.

[60] Wei Zhang, Xiang Zhong Fang, Xiaokang K Yang, and QM Jonathan Wu. Moving cast shadows detection using ratio edge. *Multimedia, IEEE Transactions on*, 9(6):1202–1214, 2007.

[61] Anh-Tuan Nghiem, François Bremond, and Monique Thonnat. Shadow removal in indoor scenes. In *Advanced Video and Signal Based Surveillance, 2008. AVSS'08. IEEE Fifth International Conference on*, pages 291–298. IEEE, 2008.

[62] Muhammad Shoaib, Ralf Dragon, and Jörn Ostermann. Shadow detection for moving humans using gradient-based background subtraction. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 773–776. IEEE, 2009.

[63] Lizhi Pei and Runsheng Wang. Moving cast shadow detection based on pca. In *Natural Computation, 2009. ICNC'09. Fifth International Conference on*, volume 2, pages 581–584. IEEE, 2009.

[64] Andres Sanin, Conrad Sanderson, and Brian C Lovell. Improved shadow removal for robust person tracking in surveillance scenarios. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 141–144. IEEE, 2010.

[65] Kazuki Nakagami and Takao Nishitani. The study on shadow removal on transform domain gmm foreground segmentation. In *Communications and Information Technologies (ISCIT), 2010 International Symposium on*, pages 867–872. IEEE, 2010.

[66] JV Panicker and M Wilscy. Detection of moving cast shadows using edge information. In *Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on*, volume 5, pages 817–821. IEEE, 2010.

[67] Rui Qin, Shengcai Liao, Zhen Lei, and Stan Z Li. Moving cast shadow removal based on local descriptors. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 1377–1380. IEEE, 2010.

[68] Pekka J Toivanen. New geodosic distance transforms for gray-scale images. *Pattern Recognition Letters*, 17(5):437–450, 1996.

[69] L. Davis T. Horprasert, D. Harwood. A statistical approach for real-time robust background subtraction and shadow detection. *IEEE ICCV'99 Frame-Rate Workshop*, 1999.

[70] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13, 2006.

[71] Emanuele Trucco and Konstantinos Plakas. Video tracking: a concise survey. *Oceanic Engineering, IEEE Journal of*, 31(2):520–529, 2006.

[72] Cor J Veenman, Marcel JT Reinders, and Eric Backer. Resolving motion correspondence for densely moving points. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(1):54–72, 2001.

[73] Ted J Broida and Rama Chellappa. Estimation of object motion parameters from noisy images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (1):90–99, 1986.

[74] David JC MacKay. Introduction to monte carlo methods. In *Learning in graphical models*, pages 175–204. Springer, 1998.

[75] Y. Bar-Shalom and T. Foreman. Tracking and data association. *Academic Press Inc.*, 1988.

[76] Donald B Reid. An algorithm for tracking multiple targets. *Automatic Control, IEEE Transactions on*, 24(6):843–854, 1979.

[77] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *Computer Vision–ECCV 2012*, pages 702–715. Springer, 2012.

[78] Amit Adam, Ehud Rivlin, and Ilan Shimshoni. Robust fragments-based tracking using the integral histogram. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 798–805. IEEE, 2006.

[79] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual tracking with online multiple instance learning. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 983–990. IEEE, 2009.

[80] David S Bolme, Bruce A Draper, and J Ross Beveridge. Average of synthetic exact filters. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2105–2112. IEEE, 2009.

[81] David S Bolme, Yui Man Lui, Bruce A Draper, and J Ross Beveridge. Simple real-time human detection using a single correlation filter. In *Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on*, pages 1–8. IEEE, 2009.

[82] M. Savvides. Biometric identification using advanced correlation filter methods. Springer-Verlag Lecture Notes in Computer Science, Ambient Intelligence, 2005.

[83] Marios Savvides and BVK Vijaya Kumar. Efficient design of advanced correlation filters for robust distortion-tolerant face recognition. In *Advanced Video and Signal Based Surveillance, 2003. Proceedings. IEEE Conference on*, pages 45–52. IEEE, 2003.

[84] Jason Thornton, Marios Savvides, and V Kumar. A bayesian approach to deformed pattern matching of iris images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(4):596–606, 2007.

[85] Andres Rodriguez and BVK Vijaya Kumar. Automatic target recognition of multiple targets from two classes with varying velocities using correlation filters. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 2781–2784. IEEE, 2010.

[86] Abhijit Mahalanobis, BVK Vijaya Kumar, Sewoons Song, SRF Sims, and JF Epperson. Unconstrained correlation filters. *Applied Optics*, 33(17):3751–3759, 1994.

[87] Charles F Hester and David Casasent. Multivariant technique for multiclass pattern recognition. *Applied Optics*, 19(11):1758–1761, 1980.

[88] BVK Kumar. Minimum-variance synthetic discriminant functions. *JOSA A*, 3(10):1579–1584, 1986.

[89] Abhijit Mahalanobis, BVK Kumar, and David Casasent. Minimum average correlation energy filters. *Applied Optics*, 26(17):3633–3640, 1987.

[90] Ph. Refegier and J. Figue. Optimal trade-off filters for pattern recognition and their comparison with the wiener approach. *Optical Computing and Processing*, 1(3):245–266, 1991.

[91] David Casasent, Gopalan Ravichandran, and Srinivas Bollapragada. Gaussian–minimum average correlation energy filters. *Applied optics*, 30(35):5176–5181, 1991.

[92] Bhagavatula Vijaya Kumar, Abhijit Mahalanobis, Sewoong Song, S Richard F Sims, and Jim F Epperson. Minimum squared error synthetic discriminant functions. *Optical Engineering*, 31(5):915–922, 1992.

[93] Gopalan Ravichandran and David Casasent. Minimum noise and correlation energy optical correlation filter. *Applied Optics*, 31(11):1823–1833, 1992.

[94] B. V. K. Vijaya Kumar Abhijit Mahalanobis. Polynomial filters for higher order correlation and multi-input information fusion, September 2001.

[95] Abhijit Mahalanobis, Robert R Muise, and S Robert Stanfill. Quadratic correlation filter design methodology for target detection and surveillance applications. *Applied optics*, 43(27):5198–5205, 2004.

[96] BVK Kumar, Abhijit Mahalanobis, and Daniel W Carlson. Optimal trade-off synthetic discriminant function filters for arbitrary devices. *Optics Letters*, 19(19):1556–1558, 1994.

[97] Ryan A Kerekes and BVK Vijaya Kumar. Correlation filters with controlled scale response. *Image Processing, IEEE Transactions on*, 15(7):1794–1802, 2006.

[98] BVK Vijaya Kumar, Abhijit Mahalanobis, and Alex Takessian. Optimal tradeoff circular harmonic function correlation filter methods providing controlled in-plane rotation response. *Image Processing, IEEE Transactions on*, 9(6):1025–1034, 2000.

[99] Marios Savvides, Krithika Venkataramani, and BVKV Kuman. Incremental updating of advanced correlation filters for biometric authentication systems. In *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*, volume 3, pages III–229. IEEE, 2003.

[100] D. Casasent R. Shenoy and E. G. Zelnio. Eigen-minace sar detection filters with improved capacity. *Proc. SPIE*, pages 435–447, 1998.

[101] Bhagavatula Vijaya Kumar and Mohamed Alkanhal. Eigen-extended maximum average correlation height (eemach) filters for automatic target recognition. In *Aerospace/Defense Sensing, Simulation, and Controls*, pages 424–431. International Society for Optics and Photonics, 2001.

[102] Ronald Wu and Henry Stark. Rotation-invariant pattern recognition using a vector reference. *Applied optics*, 23(6):838–840, 1984.

[103] D Mendlovic, E Marom, and N Konforti. Shift and scale invariant pattern recognition using mellin radial harmonics. *Optics communications*, 67(3):172–176, 1988.

[104] Ryan A Kerekes and BVK Vijaya Kumar. Selecting a composite correlation filter design: a survey and comparative study. *Optical Engineering*, 47(6):067202–067202, 2008.

[105] David S. Bolme. *Theory and Applications of Optimized Correlation Output Filters*. PhD thesis, Department of Computer Science, Colorado State University, 2011.

[106] Saul A. Teukolsky William H. Press, Brian P. Flannery and William T. Vetterling. *Numerical Recipes in C++*. Cambridge University Press, 2 edition, 1988. C - 13.1 Convolution and Deconvolution Using the FFT, pages 538-545.

[107] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[108] David G. Messerschmitt. Stationary points of a real-valued function of a complex variable. Technical report, EECS Department, University of California, Berkeley, Jun 2006.

[109] Andres Rodriguez, Vishnu Naresh Boddeti, BVK Vijaya Kumar, and Abhijit Mahalanobis. Maximum margin correlation filter: A new approach for localization and classification. *Image Processing, IEEE Transactions on*, 22(2):631–643, 2013.

[110] BVK Kumar and Laurence Hassebrook. Performance measures for correlation filters. *Applied optics*, 29(20):2997–3006, 1990.

[111] Jinman Kang, Isaac Cohen, and Gerard Medioni. Object reacquisition using invariant appearance model. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 4, pages 759–762. IEEE, 2004.

[112] Daniel Cremers. Dynamical statistical shape priors for level set-based tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(8):1262–1273, 2006.

[113] Weiming Hu, Xue Zhou, Wei Li, Wenhan Luo, Xiaoqin Zhang, and Stephen Maybank. Active contour-based visual tracking by integrating colors, shapes, and motions. *Image Processing, IEEE Transactions on*, 22(5):1778–1792, 2013.

[114] Radhakrishna Achanta. *Finding objects of interest in images using saliency and superpixels*. PhD thesis, ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE, 2011.

[115] Anne M Treisman and Garry Gelade. A feature-integration theory of attention. *Cognitive psychology*, 12(1):97–136, 1980.

[116] Christof Koch and Shimon Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. In *Matters of Intelligence*, pages 115–141. Springer, 1987.

[117] S. Goferman, L. Zelnik-Manor, and A. Tal. Context-aware saliency detection. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2376–2383, 2010.

[118] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11):1254–1259, 1998.

[119] K. Tsotsos N.D.B. Bruce. Saliency based on information maximization. *NIPS*, 2005.

[120] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(2):353–367, 2011.

[121] N. Vasconcelos D. Gao, V. Mahadevan. The discriminant center-surround hypothesis for bottom-up saliency. *NIPS*, 2007.

[122] Roberto Valenti, Nicu Sebe, and Theo Gevers. Image saliency by isocentric curvedness and color. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2185–2192. IEEE, 2009.

[123] Xiaodi Hou and Liqing Zhang. Saliency detection: A spectral residual approach. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[124] Radhakrishna Achanta, Sheila Hemami, Francisco Estrada, and Sabine Susstrunk. Frequency-tuned salient region detection. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1597–1604. IEEE, 2009.

[125] P. Perona J. Harel, C. Koch. Graph-based visual saliency. *NIPS*, 2006.

[126] Ming-Ming Cheng, Guo-Xin Zhang, N.J. Mitra, Xiaolei Huang, and Shi-Min Hu. Global contrast based salient region detection. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 409–416, 2011.

[127] Dominik Alexander Klein and Simone Frintrop. Center-surround divergence of feature statistics for salient object detection. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2214–2219. IEEE, 2011.

[128] Yichen Wei, Fang Wen, Wangjiang Zhu, and Jian Sun. Geodesic saliency using background priors. In *Computer Vision–ECCV 2012*, pages 29–42. Springer, 2012.

[129] N. Otsu. A threshold selection method from gray-level histogram. *IEEE Transactions on System Man Cybernetics*, SMC-9, 1979.

[130] `http://www.vis.uni-stuttgart.de/~hoeferbn/bse/`. [last accesed on 2014-08-08].

[131] Chia-Jung Chang, Wen-Fong Hu, Jun-Wei Hsieh, and Yung-Sheng Chen. Shadow elimination for effective moving object detection with Gaussian models. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 2, pages 540–543, 2002.

[132] Vivid dataset,. `http://vision.cse.psu.edu/data/vividEval/datasets/datasets.html`. [last accesed on 2014-08-08].

[133] `http://i21www.ira.uka.de/image_sequences/`. [last accesed on 2014-08-08].