Research on Moving Target Detection and Tracking Methods for Intelligent Traffic Surveillance

(高度交通監視のための移動体検出と追跡に関する研究)

January 2014

Doctoral Course in Computer Science

Graduate School of Science and Technology

Nihon University

Xiaofeng LU

Abstract

Visual surveillance in dynamic scenes has become a very important research area of image processing and computer vision techniques in recent years, which attempts to detect, recognize, and track certain objects from image sequences, and more generally to understand and describe target behaviors. Visual traffic surveillance system provides the most efficient traffic information for traffic control and management, and assistance for safe driving in Intelligent Transport Systems (ITS). Moving target detection and tracking methods are the most basic and important technologies in the area of intelligent visual surveillance, and are the key to realizing real time intelligent visual surveillance. However, due to the short history of development, some important problems are still unresolved, and new methods of techniques are needed. Thus, the research of moving target detection and tracking has great theoretical significance and practical value. This dissertation does the researches focused on the key technical problems about intelligent traffic surveillance, and the major works include as follows:

A novel background subtraction method is proposed to detect the moving target. Due to the dynamic conditions and interference factors, the color statistical background model based on YCbCr color space is presented. We propose a multiple feature similarity fusion using Choquet integral to class the foreground and background, and provide a new idea for high precision target detection. By analyzing of the traditional blind and selective background maintenance process, an adaptive background maintenance method is proposed to adapt the complex condition.

For single target tracking, we present a multiple feature fusion algorithm based on the Particle filter(PF). The proposed mechanism not only fuses multiple features to represent the tracking target, but dynamically balances the effect of feature similarity and feature discriminability among target model, candidate and adjacent background to obtain the adaptive feature weight. Bhattacharyya coefficient is adopted to represent the similarity, and the variance of the log-likelihood ratio is used to describe the discriminability between the target model and the adjacent background.

For the tracking problems of complex conditions, such as large scale change, rotation and mutual osculation, etc. A robust vehicle tracking based on the Speeded-Up Robust

Features(SURF) in a particle filter framework is presented. What's more, we propose the dynamic update mechanisms of target template to capture the appearance change. Specifically, adopting new feature points and discarding bad feature points. The size of tracking window is also modified dynamically by balancing the weights of three feature distributions. Furthermore, the weights of each particle are allocated by an improved distance kernel function method.

In order to robustly track multiple target, we firstly analyze the traditional data association methods, and then propose a multiple target tracking method based on feature measurement Probability Hypothesis Density (PHD) filter. In this method, the feature measurement is used to approximate the posterior density. And, we adopt an adaptive weight to fuse the color and LBP features which are extracted by Monte Carlo technology, and implement the tracking method using Gaussian mixture.

The experimental results corresponding to each method are presented and the effectiveness of the methods are evaluated and discussed under the criterion of accuracy. The researches of this thesis will make a contribution to the technology of moving target detection and tracking in intelligence visual surveillance.

Key Words: Intelligent visual surveillance, Moving target detection, Moving target tracking, Background subtraction, Particle filter, Probability hypothesis density

Acknowledgements

This research and the work described in this thesis have been made possible by the continuous support and encouragement of many people.

First and foremost, I would like to express my sincere gratitude to my supervisor, Prof. Takashi Izumi. I have tremendously benefitted from his immense knowledge, patience, enthusiasm, and experience. Especially, I would like to thank Prof. Izumi for giving me the freedom to explore idea, and continuously motivating me to carry out our common goals. His guidance and consistent support helped me become independent in my scholarly endeavors. I am extremely grateful to two other devoted examiners: Prof. Hideo Nakamura and Prof. Takashi Ono for reviewing this thesis and giving me very helpful advice and encouragement. I would like to thank other teachers in the Department of Electronics and Computer Science, College of Science and Technology, Nihon University. In addition, I wish to thank Japanese Government for awarding Monbukagakusho-MEXT scholarship that supported this work.

I would like to thank Prof. Lei Wang, Prof. Xinhong Hei and all teachers of College of Computer Science and Engineering of Xi'an University of Technology. They gave me a lot of valuable advice and consistent support on my research.

I would also like to express my thanks to all members of Izumi Laboratory, who helped me a lot both on my research and life in Japan. Especially, Asst. Tomoaki Takahashi, Dr. Lin Teng, Tadahiro Horie, Hiroto Seki. Although I cannot list all of their names, I would like to express my thanks to all of them.

I am also grateful to my friends for their patient help. They include Yan Wang, Jun Wang, Tong Zhao, Yingwei Liu, Zongfan Duan, Guo Xie, Yuanyuan Yang and all friends in Yakuendai Kouminkan. They gave me consistent help in study and life in Japan.

Finally, the biggest thanks go to my family, especially my wife, Xingping Wang, and my parents for their unconditional love and support. I am eternally grateful to them, without whom this work would not have been possible.

Contents

List of Figures

List of Tables

Abbreviations

EKF	Extended Kalman Filter
EOH	Edge Orientation Histogram
FISST	Finite Set Statistics
GNN	Global Nearest Neighbor
ITS	Intelligent Transport Systems
JPDA	Joint Probabilistic Data Association
LBP	Local Binary Patterns
LKF	Linear Kalman Filter
MHT	Multiple Hypothesis Tracking
MoG	Mixture of Gaussian
MTT	Multiple Target Tracking
NN	Nearest Neighbor
NNDR	Nearest Neighbor Distance Ratio
NNSF	Nearest Neighbor Standard Filter
OSPA	Optimal Sub-Pattern Assignment
PDA	Probabilistic Data Association
PF	Particle Filter
PHD	Probability Hypothesis Density
RANSAC	Random Sample Consensus
RFS	Random Finite Set
SIFT	Scale Invariant Feature Transform
SIS	Sequential Importance Sampling
SMC	Sequential Monte Carlo
SURF	Speeded Up Robust Features
UKF	Unscented Kalman Filter
ULBP	Uniform Local Binary Patterns

List of Symbols

Chapter	Symbols	Meaning
2	$I(x,y,t)$	Gray value of pixel (x,y) at time t
2	$f_1(x,y)$	Frame at time t_1
2	$\Delta f(x,y)$	Moving region between two frames
2	$D_i(x,y)$	Difference between any two frames
2	BW	Binary image of movement region
2	$I_{background}(x,y)$	Value of background model frame at point (x,y)
2	$I_{current}(x,y)$	Value of current frame
2	$I_{object}(x,y)$	Foreground point set
2	$I^n_{background}(x,y)$	Background model from n frames
2	$B_t(x,y)$	Background image at time t.
2	$f(I_t = u)$	Pixel distribution of Mixture of Gaussians
2	$LBP(x_c,y_c)$	Original LBP operator
2	$LBP_{P,R}(x_c,y_c)$	Extended LBP operator with number P and radius R
2	$LBP^{ri}_{P,R}$	Rotation invariance LBP operator with number P and radius R
2	$ULBP^{ri}_{P,R}$	Uniform rotation invariance LBP operator
2	$H^{Y^*}_{(x,y)}$, $H^{Cb^*}_{(x,y)}$ and $H^{Cr^*}_{(x,y)}$	Histograms of three color components at pixel (x,y)
2	$f_{(x,y)}(Y,Cb,Cr)$	Frequency of background pixel in YCbCr color space.
2	$S_k(x,y)$	Color similar measure at pixel (x,y)
	$S_T(x,y)$	Texture similar measure at pixel (x,y)
2	C_h	Choquet integral value
2	CB_{n+1}	Current background frame
2	$IB_n(x,y)$	Instantaneous background frame

2	a_n	Variable learning rate
2	a_inst_n	Adaptive weight
2	$area\_unmov_{n,n-1}$	Number of pixel in non-moving target areas
2	$sum\_unmov_{n,n-1}$	Number of change pixel between two consecutive frame F_n and F_{n-1}
2	$S(A,B)$	Similarity measure between two frames
3	x_k	State of particle at time k
3	y_k	Measurement of particle at time k
3	$p(x_k \mid y_{1:k-1})$	Prediction result of particle
3	$p(x_k \mid y_{1:k})$	Update result of particle
3	$p(y_k \mid x_k)$	Likelihood function
3	$P_N(dx_{0:k} \mid y_{1:k})$	Monte Carlo approximates of particle
3	$w_k^*(x_{0:k})$	Importance weight
3	$w_k(x_{0:k}^{(i)})$	Normalized importance weight
3	X $= \{x, \dot{x}, y, \dot{y}, h_x, \dot{h_x}, h_y, \dot{h_y}\}^T$	Dynamic state of tracking target
3	A and B	Coefficient matrix of tracking model
3	w_{t-1}	Gaussian noise matrix of tracking model
3	$p_f(y_t \mid x_t)$	Feature likelihood function of tracking target
3	$p(y_t \mid x_t)$	Fusion feature likelihood function
3	$SC_f[q_f, p_f]$	Similarity coefficient between object model and candidate model
3	$DC_f[L; q_f, p_{f(b)}]$	Discriminability between object model and adjacent background.
3	W_f	Adaptive weight of likelihood function
3	q_f	Feature distribution of target object
3	$p_f(x_t)$	Feature distribution of candidate at time t
3	$p_{f(b)}$	Feature distribution of adjacent background
3	$\rho[q_f, p_f(x_t)]$	Bhattacharyya coefficient between object model and candidate model
3	$d_f[q_f, p_f(x_t)]$	Hellinger distances between object model and candidate model

3	$G_x(x_i, y_i),\ G_y(x_i, y_i)$	Gradients at point $I(x_i, y_i)$
3	$\theta(x_i, y_i)$	Orientation of the edge
3	$G(x_i, y_i)$	Edge strength at the point (x_i, y_i)
3	C	Normalization factor
4	$I_\Sigma(\mathrm{X})$	Integral image
4	$\mathcal{H}(\mathrm{X}, \sigma)$	Hessian matrix in X at scale σ
4	$S = \{x, y, s, o, hist\}$	SURF feature descriptor
4	$q_c^{(u)}$	Color distribution of target object
4	$p_c(x_t)$	Calculate color distributions of target object
4	w_t^C	Weight of color likelihood function
4	w_t^L	Weight of texture likelihood function
4	w_t^F	Fusion weight
4	$P_{i,t+1}$	Probability of i th feature point in target template at time $t+1$
4	$UP_{i,t+1}$	Probability of the point in matching set UM_t
4	U_t	Un-matching sets at time t
4	UM_{t+1}	New matching set at time $t+1$
4	h_t	Rectangle region with height and width half axis
4	h_{SURF}	Region of SURF matches distribution
4	h_{color}	Region of color feature distribution
4	$h_{texture}$	Region of texture feature distribution
4	$M_t = \{m_1, m_2, \cdots, m_p\}$	Matching candidate set of SURF feature
4	C(x, y)	Object center point
4	$k_S(d_i)$	Exponential kernel function
4	$w_i^{(t)}$	Updated weight of particles
4	$\widetilde{w}_i^{(t)}$	Normalized weight of particles
5	$X_k = \{x_{k,1}, \cdots, x_{k,N_k}\}$	State set of targets
5	$Z_k = \{z_{k,1}, \cdots, z_{k,M_k}\}$	Received Observation set of targets
5	$\mathcal{F}(\mathcal{X}),\ \mathcal{F}(\mathcal{Z})$	Respective collections of all finite subsets of $\mathcal{X}$ and $\mathcal{Z}$
5	$B_{k\|k-1}(x)$	RFS of targets spawned at time k

5	$S_{k\|k-1}(x)$	RFS of survival targets at time k
5	Γ_k	RFS of spontaneous birth target at time k
5	Z_k	RFS of set-valued observation at time k
5	$\Theta_k(x)$	RFS of measurements generated by the single-target state x at time k
5	K_k	RFS of clutter measurements or false alarms at time k
5	$f_{k\|k-1}(X_k\|X_{k-1})$	Multi-target transition density
5	$g_k(Z_k\|X_k)$	Multi-target measurement likelihood
5	$p_k(\cdot\|Z_{1:k})$	Multi-target posterior density at time k
5	$v_{k\|k-1}(x)$	Predicted intensity function from time $k-1$ to time k
5	$v_k(x)$	Updated posterior intensity at time k
5	$p_{S,k}(\mathrm{x})$	Survival probabilities of target model
5	$p_{D,k}(x)$	Detection probabilities of target model
5	$\gamma_k(x)$	Intensities of birth RFS s
5	$\beta_{k\|k-1}(x\|\zeta)$	Intensities of spawn RFS s
5	$h(x)$	Real-valued function.
5	$L_{k,z}(x)$	Sensor likelihood function at time k
5	$\hat{v}_k(x)$	Approximate posterior intensity
5	$Z_{F,k}$	Fusion feature measurement set at time k
5	$Z_{C,k}$	Color measurement set at time k
5	$Z_{T,k}$	Texture measurement set at time k
5	$\overline{d}_p^{(c)}(X,Y)$	OSPA metric of order p with cut off at c

Chapter 1. Introduction

1.1. An overview of visual surveillance systems

Vehicle traffic is one of the most important symbols of modern society, but, with the development of economy, the sharp increase in number of vehicles, serious traffic congestion, deteriorating traffic conditions, and frequent traffic accidents, these conditions greatly put the pressure on urban transport, and seriously affect the economic development and daily life. Although the building, expansion of highway can alleviate this problem, however, the growing population and rapid growth of the urban construction land result in a decrease of available land for road construction. And restricting the number of vehicles is infeasible. Therefore, only considering the road and vehicles is very difficult to solve the problem. In this context, it was envisaged using a variety of new technologies, considering the factor of road and vehicle, to systematically solve the traffic problems. Thus, Intelligent Transport Systems (ITS) have become a new research field. ITS is advanced applications which without embodying intelligence as such aim to provide innovative services relating to different modes of transport and traffic management and enable various users to be better informed and make safer, more coordinated and 'smarter' use of transport networks[1].

ITS commonly includes the following nine research areas: advances in navigation systems, electronic toll collection systems, assistance for safe driving, optimization of traffic management, increasing efficiency in road management, support for public transport, increasing efficiency in commercial vehicle operations, support for pedestrians, support for emergency vehicles operations[2].

Nowadays, there is an urgent need for the robust and reliable traffic surveillance system to improve traffic control and management and assist for safe driving. Vehicle detection technique appears to be the weakest link in traffic surveillance and control system [3]. Although many detect devices such as supersonic, radar, sonar and inductive loop are exist and widely used, the most important drawback of these equipment is their

limitation in measuring some important traffic parameters and accurately assessing traffic condition. The first reason is that blind type of detection technology is employed. They are expensive to install and maintain. These sensors cannot provide full traffic scene information. Intelligent visual surveillance is one of the most attractive alternative technologies as it offers opportunities for performing substantially more complex tasks and provides more information than other sensors. Intelligent visual surveillance techniques include the extraction of moving vehicle, description, tracking, identification and behavior analysis and other aspects. Intelligent visual surveillance provides most efficient traffic information for traffic control and management, assistance for safe driving in ITS. Comparing with other sensors, video sensors offer a relatively low installation cost with little traffic disruption during maintenance. Furthermore, they provide wide area monitoring for traffic control and management, and evaluate the traffic conditions by the following parameters: traffic flow rate, average traffic speed, the length of queue, traffic density and so on. Therefore, developing real-time traffic parameter surveillance systems based on video aiming to derive reliable and robust traffic state information has attracted a lot of attention during the past decade. As a result, vehicle detecting and tracking by a stationary video camera is one of the most promising new techniques for large scales traffic information data collection and analysis.

In this thesis, we propose four methods for moving target detection and object tracking with application in visual surveillance and consider robustness and accuracy as the major design goals of our work.

1.2. Target detection

Moving target detection is the basic step for intelligent visual surveillance. It can provide the size of moving target, position and velocity, and other basic information. The performance of this step is particularly significant because subsequent processing of the video data is greatly dependent on this step. In addition, moving target detection is also one of the important research areas in computer vision. Moving target detection aims at extracting moving targets that are of interest in video sequences. Commonly used techniques for moving target detection are optical flow [4][5][6][7][8], temporal frame differencing[9], and background subtraction[10][11][12]. Background subtraction,

in particular, is a commonly used technique for foreground segmentation in static scenes because it has a very low computational cost. The goal of background subtraction is to remove the background in a scene by describing an adequate model of the background. The result is that only interesting targets are left in the scene for tracking.

Detection target is usually the pedestrians and vehicles. Accurate detection of moving targets is the key to the analysis and understanding of video scenes. Due to the application of intelligent video surveillance system, the camera often is fixed, so the background of an image sequence captured is static, which provides the convenience for moving target detection. However, for moving target detection in the intelligent visual surveillance, there are many challenges, including the following two points:

1). Influence of dynamic scene: In practice, background images are often very complex in natural scene. Background images, captured by a still camera, are not static in surveillance scene. For example, illumination changes, waving tree branches and flags, vibration of camera, and so on. The moving region of background will often be detected as moving targets, and thus a large area interfering targets will be produced, which seriously affect the accuracy of moving target detection.

2). Influence of moving shadow: In a natural scene, shadow always accompanies with moving target. In other words, where there is target, there is shadow. Generally, the moving shadow is detected as a part of moving target in target detection methods, it will not only decrease the precision of moving target detection, and connect other moving targets, then seriously disturb the analysis of moving targets.

1.3. Target tracking

Visual target tracking is an important step in many applications such as intelligent transport, human-computer interaction, guidance, medical imaging, obstacle avoidance and gesture recognition and so on. Target tracking problems can be formulated as a hidden state estimation problem given available measurements. The measurements are taken at regular intervals and the task is to estimate the state of a target at each point in time, such as its position, velocity or other attributes. Successive estimates provide the tracks which describe the trajectory of a target. Over the last few decades, numerous techniques have been proposed for target tracking, such as Kalman filter[13], extended Kalman filter[24], unscented Kalman filter[14], Mean-shift [15], and Particle filter[16].

Object tracking methods can be classified into two categories [17]. The first one is a deterministic method that compares a target model with the current frame, searching for the most probable region, and selects it as the target, such as mean shift tracking. The second category is a probabilistic method which uses the state space to model the underlying dynamics of the tracking process, such as Kalman filter, particle filter, etc.

In addition, multiple target tracking is an extension of single target tracking, and can be defined as the processing of multiple measurements obtained from multiple targets in order to maintain estimates of the target current states. Many techniques have been described for multiple target tracking, such as Nearest neighbor (NN)[18], Probabilistic Data Association (PDA) [19], joint probabilistic data association (JPDA) [20], Multiple hypothesis tracking(MHT)[21][22]. These traditional approaches require data association that operate in conjunction with filtering. The data association problem makes up the bulk of the computational load in multiple target tracking method. Recent works show that the probability hypothesis density (PHD) filter is a promising approach for multiple target tracking, which propagates the PHD or the first moment of the multiple target posterior density instead of the full multiple target posterior density, incorporating track initiation and termination without consideration of measurement-to-track association [23].

1.4. Original contributions

This thesis proposes four methods for moving target detection, single target tracking, and multiple target tracking for applications in visual surveillance. We consider robustness and accuracy as the major design goals of our work. The main contributions of the thesis are as follows：

Moving target detection

In the area of moving object detection a technique robust to background dynamics using background subtraction with a color statistical background model method is described. A foreground-background pixel classification method using the Choquet Integral is presented. Another technique that is robust to sudden illumination changes using a novel adaptive background maintenance method is proposed.

Single target tracking

Once a moving object is detected, the foreground object mask generated is used to initiate object tracking using particle filtering. We propose a method for robust tracking with dynamic parameter setting for likelihood model of particle filtering. We propose an adaptive multiple-feature fusion mechanism that not only improves the represent-ability of tracking target, but dynamically trades off the effects of feature similarity and discriminability.

For the fast moving, mutual occlusion, scale change and clutter condition, we propose a novel tracking method based on SURF feature matching using particle filtering. In this method, we also propose an adaptive on-line feature point update mechanism, which includes the discarding bad feature points and adopting new feature points method. In addition, we propose a new updating method for weight of particle and tracking window modification method to improve the tracking performance.

Multiple target tracking

The likelihood function and feature measurement random set are applied for the PHD approximation. In the implementation parts, Gaussian Mixture for the probability hypothesis density filter is used. We propose the method for calculating the feature measurement random set by Monte Carlo technique. In addition, we propose the adaptive weight to fuse color measurement and texture measurement.

1.5. Thesis outline

This thesis is organized as follows:

Chapter 1 Introduction

Chapter 1 provides the overview and development of object detection and object tracking. In particular, it outlines the motivation for research considered in this thesis and summarizes the major contributions.

Chapter 2 Moving target detection with background subtraction

Chapter 2 provides an overview of target detection. In particular, summaries of background subtraction based on background model are included. In order to appreciate the problems associated with traditional background subtraction approaches, a brief

description of statistical background model and foreground classification is provided. In this thesis, fusion feature similarity measures based on the Choquet integral are used as foreground classification. In addition, an adaptive background maintenance method is proposed, which addresses the traditional problem, such as ghosts, etc. Furthermore, this chapter also presents the comparing experiments with conventional background subtraction, Gaussian mixture model, and other main background maintenance methods.

Chapter 3 Single target tracking with particle filter and multiple features fusion

Chapter 3 introduces the Particle filter (PF) technique to target tracking. It also contains a brief literature review of the developments and application of the particle filter technique. Moreover, it outlines some of the drawbacks of PF. It presents an adaptive multiple-feature fusing mechanism, and the make-off of feature similarity between target object and candidate with feature discriminability between target object and its adjacent background. In addition, it presents the comparing experiments with other state-and-art methods, such as single feature tracking, fixed weight tracking, and mean shift method.

Chapter 4 Single target tracking with speeded up robust features (SURF) and PF

Chapter 4 introduces the SURF method for detecting the scale and rotation invariant feature point. In particular, it presents the SURF feature point update mechanism, which includes the discarding bad feature points and adopting new feature points. In addition, it also presents the calculating method for likelihood function of color and texture feature. Moreover, preliminary results of these proposed methods have been presented in this chapter.

Chapter 5 Multi-target tracking with the FM-PHD filter

Chapter 5 introduces the classical data association methods for multi-target tracking. In particular, it provides a description of the random finite set approach to multi-target tracking. It presents the PHD filter that is a tractable suboptimal approximation to the full multi-target Bayes filter based on RFS. Furthermore, this chapter also presents the approximation for PHD using feature measurement, called the feature measurement probability hypothesis density (FM-PHD) filter that performs estimate-to-track association based on the implementation of PHD filter. It also presents the extracting method of feature measurement random set by Monte Carlo technique. Experimental results of these proposed methods in visual sequence are included in this chapter.

Chapter 6 Conclusion

Finally, Chapter 6 includes the concluding remarks of this thesis. In particular, it summarizes its main contributions and outlines possible future research directions.

Chapter 2. Moving target detection with background subtraction

2.1. Introduction

Moving vehicle detection means to extract from the video stream in real time vehicle target, location, area and color characteristics, and represent the foreground object model by the static characteristics. Test results can provide area of interest for subsequent intelligent monitoring tasks, such as target recognition, tracking, behavior analysis and so on. Target detection, first of all, needs to make a video segmentation, in other words, segmenting the change region from the video sequences, and then get the interesting target through recognition techniques. Fast and accurate segmentation of moving targets has very important significance for classification, vehicle tracking and behavior analysis in the traffic scene. However, because of the effects of weather change, illumination change, clutter background and shadows, moving target segmentation and detection face a challenge.

This chapter is organized as follows: Firstly, several kinds of target detection algorithms are introduced in traffic surveillance scene, and then we propose a new background subtraction method based on the Choquet integral. It includes an improved statistical color background model based on the YCbCr color space, foreground detection using the Choquet integral, and adaptive background maintenance. Finally, experimental results and analysis are described.

2.2. Detection methods

Optical flow

Optical flow is a dense field of displacement vectors which defines the translation of each pixel in a region. It is computed using the brightness constraint, which assumes

brightness constancy of corresponding pixels in consecutive frames. Flow vectors are used to divide the image in segments with equal motion. Background motion will be different from that of moving objects, so moving objects will be segmented from the background. Many methods for computing optical flow have been proposed [4][5][6] [7][8].

Let $I(x, y, t)$ denotes the gray value of pixel (x, y) at time t. u and v represent the components of the optical flow vectors at the point x and y directions, respectively. $u = dx/dt$, $v = dy/dt$. Because $dI(x, y, t)/dt = 0$, then we obtain the optical flow equation:

$$I_x u + I_y v + I_t = 0 \tag{2.1}$$

where I_x, I_y and I_t are the partial derivatives of the gray value of pixel point with respect to x, y and t directions. $V = (u, v)$ denotes the components of the image velocity field. The distribution of the velocity in each point of the image is known as the optical flow.

The optical flow equation is not sufficient for computing two components, u and v. In practice, it is necessary to consider additional restrictions on the problem to estimate the motion at every image location. For traffic video surveillance, the camera is generally fixed on the roadside poles. This problem can be greatly simplified. For the roads background, the optical flow is ideally zero, and the moving foreground only has the optical flow.

The computational complexity of optical flow techniques is high. Real time implementation is therefore difficult or expensive. Once moving objects are segmented, only the flow of those pixels assigned to an object is necessary. But for the detection of new objects, always a global optical flow calculation is necessary, although this can be at a lower resolution or less frequent in time. Besides the computational complexity, another important disadvantage of optical flow in surveillance applications is that the flow is not always correct. It is undefined at object edges because of smoothing in the calculation of optical flow. This causes inaccurate object segmentation. The flow inside objects may also be wrong. For homogenous regions such as parts of a vehicle for example, flow will be zero.

Two frame difference

Two frame difference method is to obtain the difference image between the previous frame and back frame at a small time interval, and then to get the moving target region by the threshold binary image. Because the operation is very simple, and the binaryzation of difference image corresponds to many efficient algorithms, it is very suitable for real-time detection of moving object. Let $f_1(x, y)$ denotes the input frame at time t_1, $f_2(x, y)$ is the input frame at time t_2. If there is the moving vehicle at this period, and then $f_2(x, y) = f_1(x - \Delta x, y - \Delta y)$, assume

$$\Delta f(x, y) = |f_2(x, y) - f_1(x, y)| \tag{2.2}$$

For the static part of image, $\Delta x = \Delta y = 0$, then $\Delta f(x, y) = 0$, yet, for the moving part of image, $\Delta f(x, y) \neq 0$, thus we can obtain the moving region. However, it is prone to generate cavity in the internal region. Because the segmented region is the combine area between the two positions of the objects, is larger than the actual area of object. Secondly, it is very sensitive to noise, and the detected position of the object is not accurate. Otherwise, the sampling frequency has greatly influence to the detection performance, if the time interval is set not suitable, for a fast moving vehicle, it is easy to mistakenly identified into two different objects, and for a slow moving vehicle, the fraction of detection object can only be detected. In general, the multiple frames difference method addresses this problem.

Multiple frames difference

Multiple frames difference is proposed to overcome the problem of two frame difference. Three or more consecutive image frames make the difference between any two frames. Assume the set of binary image sequence $\{I_i\}$, the difference image D_i is defined as

$$D_i(x, y) = |I_{i+1}(x, y) - I_i(x, y)| \tag{2.3}$$

$$J_i(x, y) = \begin{cases} 1 & D_i(x, y) \geq TH \\ 0 & D_i(x, y) < TH \end{cases} \tag{2.4}$$

where TH is a threshold value. Lastly, we can obtain the binary image of movement

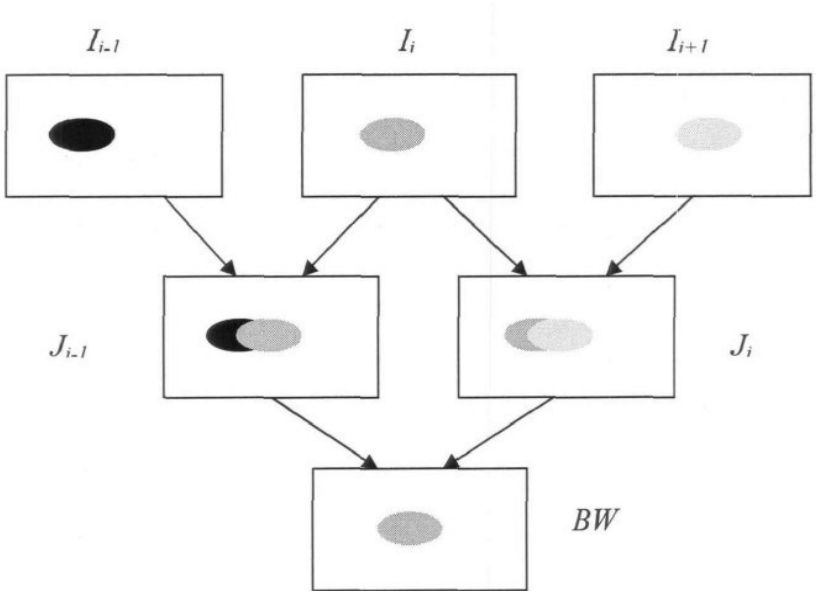


Figure 2.1 Multiple frames difference method.

region BW in Figure 2.1.

$$BW = J_i \cap J_{i-1} \tag{2.5}$$

However, the actual traffic condition is very complex, and the speed of detection object is changed faster and dramatically, such as stop-and-go traffic flow, stopped vehicles (i.e. accident). The performance of multiple frames difference method is low stability, and low reliability.

Background subtraction

Background subtraction is a method typically used to segment moving regions in image sequences taken from a static camera by comparing each new frame to a model of scene background.

Background subtraction identifies moving object from the portion of a video frame that differs significantly from a background model. There are many challenges in developing a good background subtraction algorithm. Firstly, it must be robust against changes in illumination. Secondly, it should avoid detecting non-stationary background objects such as swinging leaves, rain, snow, shadow cast by moving objects. Finally, its internal background model should react quickly to changes in background such as stop-and-go of vehicles.

At present, numerous background subtraction methods have been proposed, no matter

what kind of algorithm, these methods have the same theory. Next, we introduce the basic principle of background subtraction in following section.

The basic idea of the background subtraction is to match the current image frame and the reference image of background model, and to calculate the similar measurement of each point in the current frame and background model, and to class the foreground and background using the formula (2.6).

$$I_{object}(x,y) = \begin{cases} 1, & \left|I_{current}(x,y) - I_{background}(x,y)\right| \geq TH \\ 0, & \left|I_{current}(x,y) - I_{background}(x,y)\right| < TH \end{cases} \tag{2.6}$$

where (x,y) denotes an any point in the image frame. $I_{background}(x,y)$ is the characteristic value of background model frame at point (x,y). $I_{current}(x,y)$ represents the characteristic value of current frame. TH is a given threshold. If $I_{object}(x,y) = 1$, the point (x,y) is adjudged as a moving foreground point, conversely, it is a background point.

Because the background model often changes, the background model needs to be updated in real-time for the accuracy of subsequent calculations. Then, the updating method of background model is defined as:

$$I^{n}_{background}(x,y) = \begin{cases} (1-\alpha) I^{n-1}_{background}(x,y) + \alpha I_{current}(x,y), & I^{n}_{object}(x,y) = 1 \\ (1-\beta) I^{n-1}_{background}(x,y) + \beta I_{current}(x,y), & I^{n}_{object}(x,y) = 0 \end{cases} \tag{2.7}$$

Assume that the current image is the n *th* frame, $I^{n-1}_{background}(x,y)$ denotes the before updating background model, $I^{n}_{background}(x,y)$ represents the after updating one. We can see that this updating method adapts to the complex conditions, such as illumination change, stop-and-go moving foreground objects, and improves the classification accuracy of the foreground and background. And, learning rate α controls the fusion of stationary foreground objects, learning rate β determines the rate of adapting change in illumination conditions, α and β determine the effective degree of the current frame on the background model and updating rate of the background frame.

Even though there exist a lot of background subtraction algorithms in the literature, most of them follow a simple flow diagram shown in Figure 2.2.

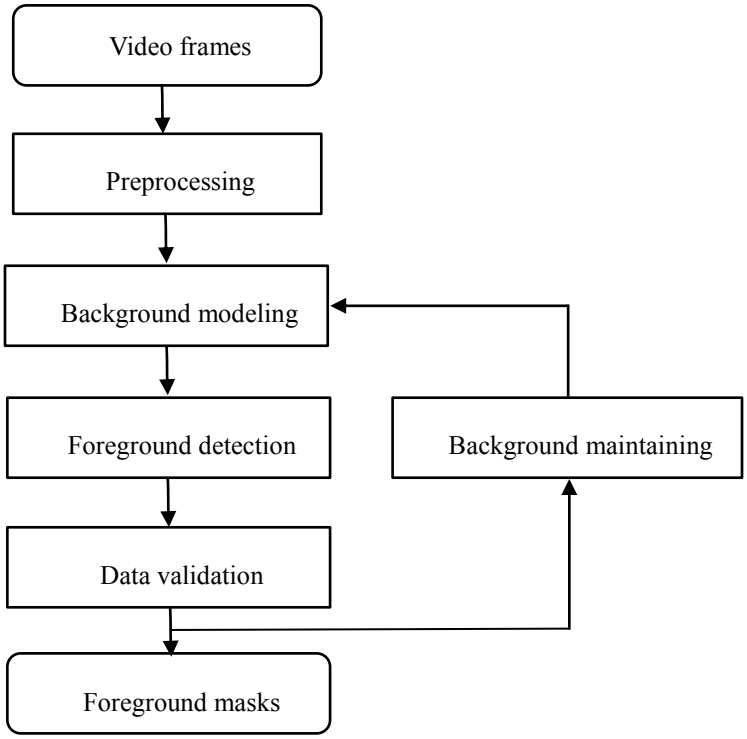


Figure 2.2 Flow diagram of a generic background subtraction algorithm.

The major steps in a background subtraction algorithm are preprocessing, background modeling, foreground detection, data validation and background maintaining (updating). Preprocessing consists of a collection of simple image processing tasks that change the raw input video into a format that can be processed by subsequence steps. Background modeling uses the new video frame to calculate and update a background model. Foreground detection then identifies pixels in the video frame that cannot be adequately explained by the background model, and outputs them as a binary candidate foreground mask. Data validation examines the candidate mask, eliminates those pixels that do not correspond to actual moving objects, and outputs the final foreground mask. Finally, background maintaining updates the current background model to meet the needs of change conditions.

Background modeling is at the heart of any background subtraction algorithm. Recent research has been devoted to developing a background model that is robust against environmental changes in the background, but sensitive enough to identify all moving objects of interest. Next, we simply introduce the common background modeling methods.

Time differentiation background modeling

Time differentiation background modeling[10] takes the difference between the current image and the current background giving the difference image for each frame of the video sequence, and updates the background by taking a weighted average of the current background and the current frame of the video sequence. It classifies the pixels as foreground and background and then uses only the background pixels from the current image to modify the current background. Through a periodic update intervals, it can obtain the background model. This method requires an iteration process for a number of image sequences, speeds some time. When the number of training images is small, trained background image is affected by the first image seriously.

Statistical background modeling

The color value of static background pixel changes very little or almost constant in the image sequence. From the statistical point of view, the gray value of background pixel can be thought as a statistic result. In other words, we can statistic the occurrence frequency of each pixel corresponding to the image sequence. This is known as statistical background modeling method [11][25][26].

Let $B_t(x,y)$ denotes the background image at time t.

$$B_t(x,y) = U(I_t(x,y), I_{t-\Delta t}(x,y), \cdots, I_{t-(n-1)\Delta t}(x,y)) \tag{2.8}$$

where $U(\cdot)$ denotes the update function, $I_t(x,y)$ is the image captured at the time t. n is the number of frame for estimating the background model. Δt is a sampling interval, then the statistical time is $n \cdot \Delta t$. For improving the update effect, the previous computed background $B_{t-1}(x,y)$ with an adequate weight w_b will be combined into the formula (2.9).

$$B_t(x,y) = U(I_t(x,y), I_{t-\Delta t}(x,y), \cdots, I_{t-(n-1)\Delta t}(x,y), w_b B_{t-1}(x,y)) \tag{2.9}$$

In the statistical method, selecting different update functions, we can obtain different background update methods. If the average will be selected, the formula (2.10) should be defined as,

$$B_t(x,y) = \frac{1}{n+1}(I_t(x,y), I_{t-\Delta t}(x,y), \cdots, I_{t-(n-1)\Delta t}(x,y), w_b B_{t-1}(x,y)) \tag{2.10}$$

Within a certain time interval, various colors vehicles run on the road, the vehicle's brightness value is higher of lower than the ones of load surface, or with the road, so this method is simple to calculate the background model, but if the high brightness or low brightness vehicle run on the road, especially, slow moving case, this method will have a great error. In order to be more close to reality background, it should be used more than enough frames to make the deviation within the allowable range.

In practice, when the frame number n is limited, the median function is commonly adopted as the update function. In the case of vehicle flow smooth, the background pixel in the continuous multiple frames accounts for the major part. The median of background pixel may be used as the pixel values of the estimated background model (see the formula (2.11)). This method counts the gray change of individual pixel in the successive video frame, and sorts of the changing gray value, and then selects the median as gray value of corresponding pixel in background model.

$$B_t(x,y) = median(I_t(x,y), I_{t-\Delta t}(x,y), \cdots, I_{t-(n-1)\Delta t}(x,y), w_b B_{t-1}(x,y)) \quad (2.11)$$

Non-parametric model

Non-parametric model was proposed in literature [27]. It uses the entire history $I_{t-L}, I_{t-L+1}, \cdots, I_{t-1}$ to form a non-parametric estimate of the pixel density function $f(I_t = u)$:

$$f(I_t = u) = \frac{1}{L}\sum_{i=t-L}^{t-1} K(u - I_i) \quad (2.12)$$

where $K(\cdot)$ is the kernel estimator which was chosen to be Gaussian. The current pixel I_t is declared as foreground if it is unlikely to come from this distribution, i.e. $f(I_t)$ is smaller than some predefined threshold. The advantage of using the full density function over a single estimate is the ability to handle multi-modal background distribution. Examples of multi-modal background include pixels from a swinging tree or near high-contrast edges where they flicker under small camera movement. The implementation uses the median of the absolute differences between successive frames as the width of the kernel. Thus, the complexity of building the model is the same as median filtering. On the other hand, the foreground detection is more complex as it needs to compute the formula (2.12) for each pixel.

Mixture of Gaussian

The MoG method tracks multiple Gaussian distributions simultaneously. MoG has enjoyed tremendous popularity since it was first proposed for background modeling in literature [12]. Similar to the non-parametric model, MoG maintains a density function for each pixel. Thus, it is capable of handling multi-modal background distributions. On the other hand, since MoG is parametric, the model parameters can be adaptively updated without keeping a large buffer of video frames. Our description of MoG is based on the scheme described in literature [28]. The pixel distribution $f(I_t = u)$ is modeled as a mixture of K Gaussians:

$$f(I_t = u) = \sum_{i=1}^{K} \omega_{i,t} \cdot \eta(u; \mu_{i,t}, \sigma_{i,t}) \tag{2.13}$$

where $\eta(u; \mu_{i,t}, \sigma_{i,t})$ is the i-th Gaussian component with intensity mean $\mu_{i,t}$ and standard deviation $\sigma_{i,t}$. $\omega_{i,t}$ denotes the portion of the data accounted for by the i-th component. Typically, K ranges from three to five, depending on the available storage. For each input pixel I_t, the first step is to identify the component $\hat{\imath}$ whose mean is closed to I_t. Component $\hat{\imath}$ is declared as the matched component if $\left|I_t - \mu_{\hat{\imath},t-1}\right| \leq D \cdot \sigma_{\hat{\imath},t-1}$, where D defines a small positive deviation threshold. The parameters of the matched component are then updated as follows:

$$\omega_{\hat{\imath},t} = (1 - \alpha)\omega_{\hat{\imath},t-1} + \alpha \tag{2.14}$$

$$\mu_{\hat{\imath},t} = (1 - \rho)\mu_{\hat{\imath},t-1} + \rho I_t \tag{2.15}$$

$$\sigma_{\hat{\imath},t}^2 = (1 - \rho)\sigma_{\hat{\imath},t-1}^2 + \rho(I_t - \mu_{\hat{\imath},t})^2 \tag{2.16}$$

where α is a user-defined learning rate with $0 \leq \alpha \leq 1$. ρ is the learning rate for the parameters and can be approximated as follows:

$$\rho \approx \frac{\alpha}{\omega_{\hat{\imath},t}} \tag{2.17}$$

If no matched component can be found, the component with the least weight is replaced by a new component with mean I_t, a large initial variance σ_o and a small weight ω_o. The rest of the components maintain the same means and variances, but

lower their weights to achieve exponential decay:

$$\omega_{i,t} = (1-\alpha)\omega_{i,t-1} \tag{2.18}$$

Finally, all the weights are renormalized to sum up to one. To determine whether I_t is a foreground pixel, we first rank all components by their values of $\omega_{i,t}/\sigma_{i,t}$. Highter-rank components thus have low variances and high probabilities, which are typical characteristics of background. If $i_1, i_2, \cdots, i_K$ is the component order after sorting, the first M components that satisfy the following criterion are declared to be the background components:

$$\sum_{k=i_1}^{i_M} \omega_{k,t} \geq \Gamma \tag{2.19}$$

where Γ is the weight threshold. I_t is declared as a foreground pixel if I_t is within D times the standard deviation from the mean of any one of the background components. Note that the above formulation can be easily extended to handle color data. The computational complexity and storage requirement of MoG is linear in terms of the number of components K.

The above described algorithms only adopt gray image for moving object detection. For the targets of the larger differences in brightness and closer hue, these methods are easy to cause failure. Therefore, Kyungnam Kim et al.[29] proposed a new background subtraction based on the code book. It can make the more accurate background model using color video, but the modeling process is more complex, and the computing speed restricts its practical application.

2.3. Color and texture feature

2.3.1. Color feature

RGB color space

The RGB color model is an additive color model in which red, green, and blue light are added together in various ways to reproduce a broad array of colors. RGB is a device-dependent color model: different devices detect or reproduce a given RGB value differently, since the color elements and their response to the individual R, G, and B levels vary from manufacturer to manufacturer, or even in the same device over time. Thus an RGB value does not define the same color across devices without some kind of color management. The RGB color model mapped to a cube as shown in Figure 2.3. The horizontal x-axis as red values increasing to the left, y-axis as blue increasing to the lower right and the vertical z-axis as green increasing towards the top. The origin, black, is the vertex hidden from view. In order to encode color, different binary digital representations are in use, commonly, 8 bits color depth is used for representing the possible values per component from 0 to 8 power of two minus one (2^8-1). For example, within this range, assigning black to the origin at the vertex (0,0,0), and with increasing intensity values running along the three axes up to white at the vertex (255,255,255), diagonally opposite black.

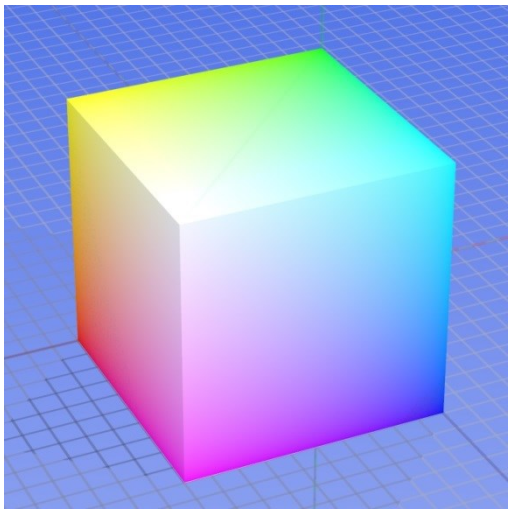


Figure 2.3 RGB color space model

However, the RGB color space has an obvious drawback: These three color components are dependent reciprocally which increase its sensitivity to the illumination changes. These features will bring great difficulties to shadow suppression. So, the significant correlation among three components is not conductive to detect and track the moving object.

HSV color space

HSV (hue, saturation, value) is one of several color systems used by people to select colors from a color wheel or palette. This color system is considerably closer than the RGB system in the way which people experience and describe color sensations. In artist's terminology, hue, saturation, and value refer approximately to tint, shade, and tone. The hue parameter has the range [0,360^o], and the saturation and value parameters values have the range [0,1]. The hue of a color refers to which pure color it resembles. All tints, tones and shades of red have the same hue. Hues are described by a degree that specifies the position of the corresponding pure color on the color wheel, as a degree between 0^o and 360^o, 0^o refers to red, 60^o is yellow, 120^o is green, and so forth around the color wheel. The saturation of a color describes how white the color is. A pure red is fully saturated, with a saturation of 1, tints of red have saturations less than 1, and white has a saturation of 0. The value of a color, also called its lightness, describes how dark the color is. A value of 0 is black, with increasing lightness moving away from black. Figure 2.4 illustrates the HSV color space [103].

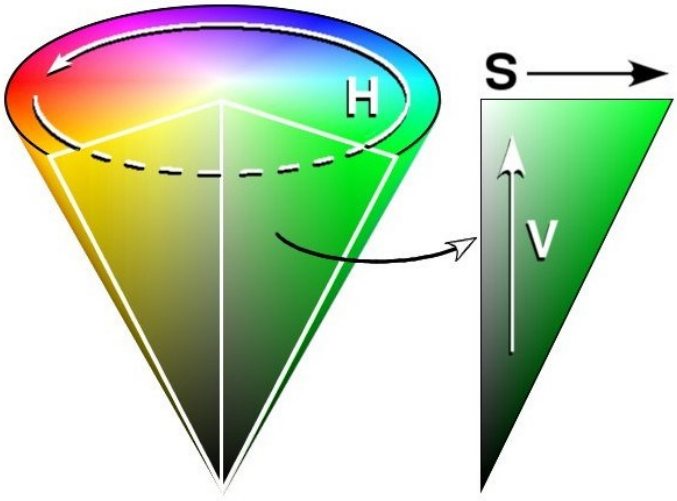


Figure 2.4 HSV color space model

Given the values of any point is (R,G,B), where $(R,G,B) \in [0,255]$. The transformation from RGB to HSV color space (H,S,V) is defined as :

$$V = max(R,G,B) \tag{2.20}$$

$$S = \begin{cases} (V - \min(R,G,B)) * 255/V\,, & V! = 0 \\ 0, & else \end{cases} \tag{2.21}$$

$$H = \begin{cases} (G-B) * 60/S & V = R \\ 180 + (B-R) * 60/S & V = G \\ 240 + (R-G) * 60/S & V = B \end{cases} \tag{2.22}$$

YCbCr color space

The YCbCr color space is used widely in digital video. In this format, luminance information is represented by a single component, Y, and color information is stored as two color-difference components, Cb and Cr. Component Cb is the difference between the blue component and a reference value, while component Cr is the difference between the red component and a reference value. The transformation used by Image Processing Toolbox to convert from RGB to YCbCr is

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112.000 \\ 112.000 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.23}$$

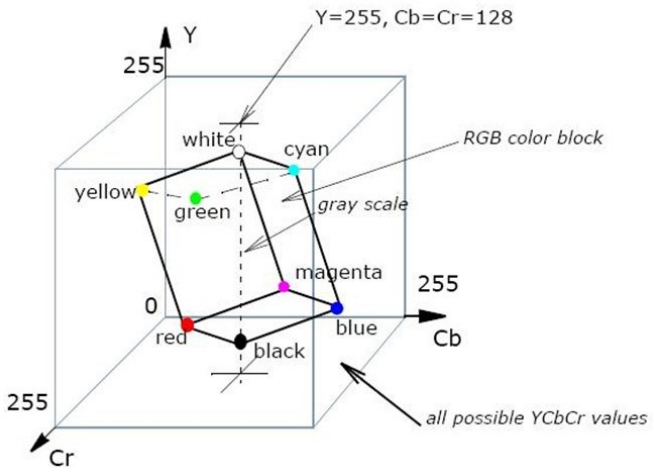


Figure 2.5 YCbCr color space model

The selection of the color space is one of the key factors for moving vehicle detection. In the background reconstruction and foreground detection, the most commonly used one is the RGB color space which is directly available from the sensor or the camera. But the RGB color space has an obvious drawback: These three color components are

dependent reciprocally, which increase its sensitivity to the illumination changes. These features will bring great difficulties to shadow suppression. A number of color space comparisons are presented in the literature [30][31][32]. After experimental observation of the effect of different color spaces on the segmentation result, the YCbCr color space was selected as an appropriate color space which reduces the influence of shadows and illumination changes.

2.3.2. Texture feature

Texture feature is defined by using the gray distribution of the neighborhoods. Therefore, it can represent the pixel value and value of surrounding pixels. There are many ways to represent the texture characteristic, such as edges, gradient, histogram, probability distribution. These ways to define the image texture characteristic, such as spatial scale, contrast, roughness, orientation and correlation. The texture feature, unlike the color, is based on the block processing, rather than pixel. The part of the texture change is detected by the block matching, and considers the effect of other surrounding pixels. T. Ojala et al. [33][34][35] present a gray-scale and rotation invariant texture operator based on local binary patterns (LBP). Recently, LBP and robustness in terms of gray-scale variations and no effect by any monotonic transformation of the gray scale have become a research hotspot. In this chapter, we define the uniform rotation invariant texture operator ULBP based on LBP to represent texture feature. The ULBP operator improves the rotation invariance of LBP code, and quantifies the occurrence statistics of individual rotation invariant patterns corresponding to certain micro-features in the image. The operators are defined as follows:

Original LBP operator

LBP is a gray-scale invariant texture primitive statistic. The operator labels the pixels of an image region by thresholding the neighborhood of each pixel with the center value and considering the result as a binary pattern [33]. The basic version of the LBP operator considers only the eight neighbors of a pixel as showed in Figure 2.6.

$$LBP(x_c, y_c) = \sum_{i=0}^{P-1} s(g_i - g_c)2^i \tag{2.24}$$

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{2.25}$$

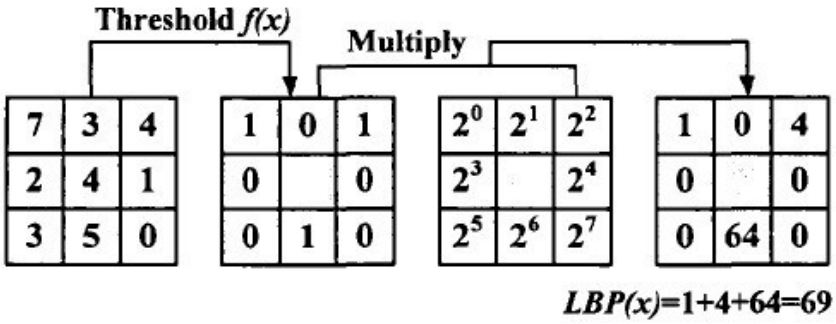


Figure 2.6 Example for calculating the original LBP code

Extended local binary pattern

The original LBP operator does not represent the large-scale structure of texture feature using the 3×3 neighborhood, so it can be easily extended to include all circular neighborhoods with any number of pixels. The gray values of the diagonal pixel are determined by interpolation. The extended LBP is showed in Figure 2.7.

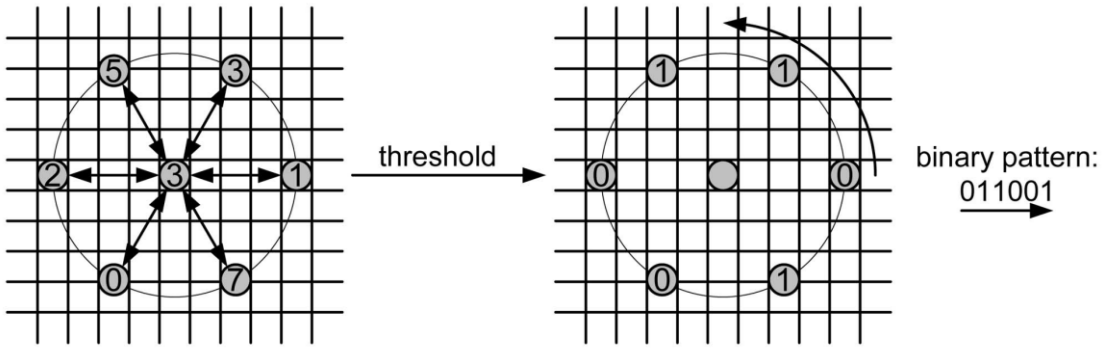


Figure 2.7 Calculating the extended binary pattern

$$LBP_{P,R}(x_c, y_c) = \sum_{p=0}^{P-1} s(g_p - g_c) 2^p \tag{2.26}$$

$$s(x) = \begin{cases} 1, & x \geq th \\ 0, & x < th \end{cases} \tag{2.27}$$

where g_c corresponds to the gray value of the center pixel (x_c, y_c) of a local

neighborhood and g_p to the gray values of P equally spaced pixels on a circle of radius R. In order to make the LBP more robust against these negligible changes in pixel values, we adopt the threshold value th to replace the original scheme. Note that the bigger the value of th is, the bigger changes in pixel values are allowed without affecting the thresholding results. With the modified version, the proposed background subtraction method consistently behaves more robustly and thus should be preferred over the original one.

Rotation invariance LBP operator

The $LBP_{P,R}$ operator produces 2^p different output values, corresponding to the 2^p different binary patterns that can be formed by the P pixels in the neighborhood set. When the image is rotated, the gray values g_p will correspondingly move along the perimeter of the circle around g_c. Thus, the different $LBP_{P,R}$ value will be obtained. To address the effect of rotation, the rotation invariant LBP is defined as:

$$LBP_{P,R}^{ri} = min\{ROR(LBP_{P,R}, i)|i = 0,1,\cdots,P-1\} \quad (2.28)$$

where $ROR(x,i)$ performs a circular bit-wise right shift on the P-bit number x and i times. So, the eight basic patterns, 00000001, 00000010, 00000100, 00001000, 00010000, 00100000, 0100000, and 10000000, can be obtained by the rotation of uniform pattern 00000001. The 36 unique rotation invariant binary patterns of $LBP_{8,R}^{ri}$ are showed in Figure 2.8. For example, pattern 0 detects bright spots, pattern 8 dark spots and flat areas, and pattern 4 edges.

Uniform LBP operator

However, the $LBP_{P,R}^{ri}$ operator does not provide very good discrimination according to practical experience [34]. The main reason is that the occurrence frequencies of the 36 individual patterns incorporated in $LBP_{P,R}^{ri}$ vary greatly. Thus, it will cause the sparse histogram, and easily affected by noise. So, in our experiment, we adopt the following uniform rotation invariant operator to represent the texture feature.

Definition 2.1 If local binary pattern operator $LBP_{P,R}$ satisfies the following conditions,

1). $LBP_{P,R}$ is the rotation invariance operator $LBP_{P,R}^{ri}$;

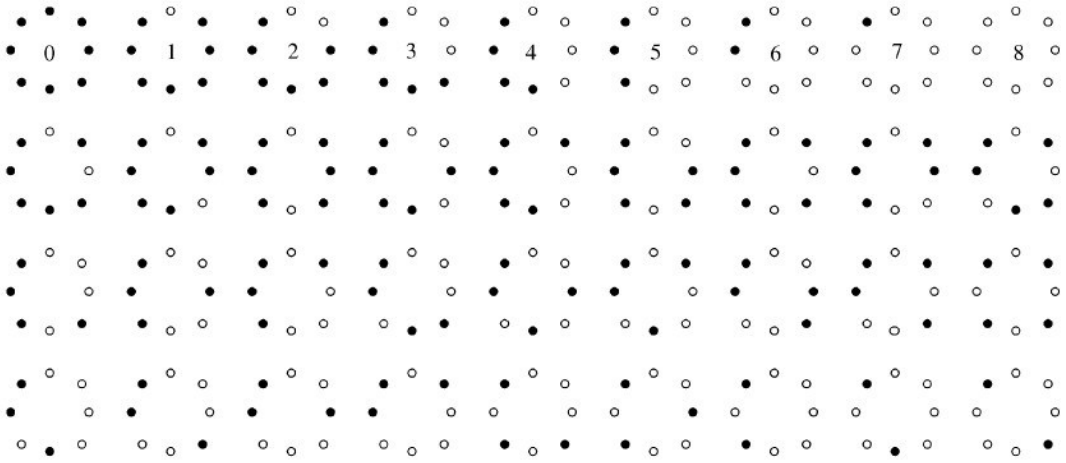


Figure 2.8 The 36 unique rotation invariant binary patterns.

2). $U(LBP_{P,R}^{ri}) \leq 2$ (2.29)

Then $LBP_{P,R}$ is known as a uniform pattern $ULBP_{P,R}^{ri}$ (referred as ULBP), corresponding to patterns are called as basic texture feature.

Here, the function $U(\mathrm{x})$ denotes the number of spatial transitions (bitwise 0/1 changes) in the binary pattern. For example, pattern x=00000000 have U value of 0, pattern x=00010000 have U value of 2 as there are exactly two 0/1 transitions in the pattern. The 9 patterns in the first row of Figure 2.8, $(\mathrm{x}) \leq 2$, are the ULBP pattern. The other 27 patterns have U value of at least 4. Thus, the ULBP operator is defined as:

$$ULBP_{P,R}^{ri} = \begin{cases} \sum_{i=0}^{P-1} s(g_i - g_c) & if \quad U(LBP_{P,R}) \leq 2 \\ P+1 & otherwise \end{cases} \tag{2.30}$$

where,

$$U(LBP_{P,R}) = |s(g_{P-1} - g_c) - s(g_0 - g_c)| + \sum_{i=1}^{P-1} |s(g_i - g_c) - s(g_{i-1} - g_c)| \tag{2.31}$$

By definition, exactly $P+1$ uniform binary patterns can occur in a circularly symmetric neighbor set of P pixels. Corresponding to the number of "1" bits in the pattern, each of them will be assigned a unique label from 0 to P.

The ULBP operator improves the rotation invariance of Local Binary Pattern (LBP) code, and quantifies the occurrence statistics of individual rotation invariant patterns

corresponding to certain micro-features in the image. Furthermore, the ULBP textures within the local area is not under the influence of the monotonous change of brightness, between the shaded area and the corresponding location of background in a gray-scale image have a similar texture feature. So, when detecting moving target the shadow influence is very small, but the ULBP textures can effectively suppress shadows interference on motion detection.

2.4. Background subtraction with Choquet integral

2.4.1. Background reconstruction

We propose a probabilistic algorithm to reconstruct the background of a traffic scene by eliminating the moving object information. According to it, the background color information of crowded scenes is dynamically retrieved by assessing color variation per pixel through a series of frames. The overall idea is based on the notion that a specific location is occupied by moving objects for a time period shorter than that for which it remains unoccupied.

The implementation of the algorithm has been applied in the YCbCr color system. The selection of the color space is one of the key factors for moving vehicle detection. In the background reconstruction and foreground detection, the most commonly used one is the RGB color space that is directly available from the sensor or the camera. But the RGB color space has an obvious drawback: These three color components are dependent reciprocally which increase its sensitivity to the illumination changes. These features will bring great difficulties to shadow suppression. A number of color space comparisons are presented in the literature [30][31][32]. After experimental observation of the effect of different color spaces on the segmentation result, the YCbCr color space was selected as an appropriate color space which reduces the influence of shadows and illumination changes. The overall idea of background reconstruction is based on the notion that a specific location is occupied by moving objects for a time period shorter than that for which it remains unoccupied [31].

Suppose $\emptyset = \{Y^*, Cb^*, Cr^* \in R^3\}$ is continuous YCbCr color space, and let $\overline{\emptyset}$ refer to the discrete color space. Then $\overline{\emptyset} = \{Y = \lfloor Y^*/h \rfloor, Cb = \lfloor Cb^*/h \rfloor, Cr = \lfloor Cr^*/h \rfloor\} \in Z^3$, where '$\lfloor \quad \rfloor$' is the floor operator, and h is the chromatic distance defined by the bin

dimensions. Let $b_{(Y,Cb,Cr)}\{Y, Cb, Cr\} \in N^3$ correspond to cubic bins, whose edges have length h, and each discretization element $b_{(Y,Cb,Cr)}$ denote a continuous chromatic range of colors. The discrete color parameters are $Y = \lfloor Y^*/h \rfloor$, $Cb = \lfloor Cb^*/h \rfloor$ and $Cr = \lfloor Cr^*/h \rfloor$, where $Y \le Y^*/h \le Y + 1$, $Cb \le Cb^*/h \le Cb + 1$ and $Cr \le Cr^*/h \le Cr + 1$. Without loss of generality, we can assume that a video consists of sequential frames of resolution $m \times n$. Let $B = \{B_{(x,y)}\}$ be a background color map. At time *t*, we will use $I_{(x,y)}(t) = (I^{Y^*}_{(x,y)}(t), I^{Cb^*}_{(x,y)}(t)$ to denote the color vector at pixel (x, y) of the frame, where $I^{Y^*}_{(x,y)}(t), I^{Cb^*}_{(x,y)}(t), and\ I^{Cr^*}_{(x,y)}(t)$ refer to Y^*, Cb^* and Cr^* elements of $I_{(x,y)}(t)$, respectively.

The color components of pixel (x, y) with respect to time *t* are estimated by a sampling procedure. Assume that $I_{(x,y)}(t)$ is calculated for *T* consecutive frames, where the sample size *T* is based on the density of vehicles on the road; if the density is greater, the sample size is greater, and vice versa. Assume that sampling begins at time t_0. Thus, the temporal sample $S_{(x,y)}(t_0) = (I_{(x,y)}(t_0), I_{(x,y)}(t_0 + 1), \cdots, I_{(x,y)}(t_0 + T - 1))$ of pixel (x, y) defines the frequency $f_{(x,y)}(Y, Cb, Cr)$ of the examined pixel having a color component belonging to the bin $b_{(Y,Cb,Cr)}$:

$$f_{(x,y)}(Y, Cb, Cr) = \sum_{t=t_0}^{t_0+T-1} \delta\left(Y - \left\lfloor \frac{I^{Y^*}_{(x,y)}(t)}{h} \right\rfloor\right)\delta\left(Cb - \left\lfloor \frac{I^{Cb^*}_{(x,y)}(t)}{h} \right\rfloor\right)\delta\left(Cr - \left\lfloor \frac{I^{Cr^*}_{(x,y)}(t)}{h} \right\rfloor\right) \quad (2.32)$$

where $Y, Cb, Cr \in N$, and $\delta()$ is the Kronecker delta function.

Because the frequency $f_{(x,y)}(Y, Cb, Cr)$ corresponds to the most persistent color value of pixel (x, y) in the *T* consecutive frames, the most persistent color value is the one that is most likely to represent the actual background frame. Then, when the maximum frequency $f_{(x,y)}(Y, Cb, Cr)$ is obtained, its three color components $Y, Cb, \text{and } Cr$ will compose a pixel $B_{(x,y)}$ of the background model. Thus, the background model will be reconstructed simply:

$$B_{(\mathrm{x,y})} = argmax\left[f_{(x,y)}(Y, Cb, Cr)\right] = (Y_{model}, Cb_{model}, Cr_{model}) \quad (2.33)$$

which represents the anticipated color values of the background model.

In order to reduce the memory requirements and computational complexity, we calculate the three color-component values in Equation (2.32). Thus, Equation (2.33) will be simplified as follows:

$$B_{(x,y)} = \left(argmax\left[f_{(x,y)}^{(Y)}(Y)\right], argmax\left[f_{(x,y)}^{(Cb)}(Cb)\right], argmax\left[f_{(x,y)}^{(Cr)}(Cr)\right]\right) \quad (2.34)$$

From the histograms $H_{(x,y)}^{Y^*}$, $H_{(x,y)}^{Cb^*}$ and $H_{(x,y)}^{Cr^*}$ of pixel (*x,y*) in *T* consecutive frames, we obtain the maximum frequencies $f_{(x,y)}^{(Y)}(Y)$, $f_{(x,y)}^{(Cb)}(Cb)$ and $f_{(x,y)}^{(Cr)}(Cr)$ and reconstruct the virtual background model.

Background reconstruction is a statistical method, therefore, the identification of the sample size is important. In general, the sample size should be large enough to contain enough information for extracting the background color in each image pixel. To achieve this goal, the sample size, in terms of time, should exceed the average time that a passing vehicle occupies any pixel in the image. In our experiment, we chose a 600 frames sample, translated in terms of time to a 20s exposure correspondingly.

2.4.2. Color and texture similarity measures

In this chapter, we define color and texture similar measures between pixels in current and background images. In this case, pixels corresponding to background should be similar in the two images while pixels corresponding to foreground should not be similar. And then we use the Choquet integral to fuse the color and texture similar measures to segment the foreground region by the predetermined threshold. The color and texture similar measures are defined in detail next section.

Color feature similar measure

About color space choices, as described above. The color similar measure $S_k(x,y)$ at the pixel (x,y) is computed as in literature [36].

$$S_k(x,y) = \begin{cases} \frac{C_k^I(x,y)}{C_k^B(x,y)} & if \quad C_k^I(x,y) < C_k^B(x,y) \\ 1 & if \quad C_k^I(x,y) = C_k^B(x,y) \\ \frac{C_k^B(x,y)}{C_k^I(x,y)} & if \quad C_k^I(x,y) > C_k^B(x,y) \end{cases} \quad (2.35)$$

where $k \in \{1,2,3\}$ is one of the color components Y, Cb and Cr, $C_k^I(x,y)$ and $C_k^B(x,y)$ represent the color value of current frame and background frame at time t, respectively. Note that $S_k(x,y)$ between 0 and 1. Furthermore, $S_k(x,y)$ is close to one if $C_k^I(x,y)$ and $C_k^B(x,y)$ are very similar.

Texture feature similar measure

The $ULBP_{K,R}^{ri}$ operator is an excellent measure of the spatial structure of local image texture. For each pixel texture in the current image and the background image, the $ULBP_{K,R}^{ri}$ operator is less sensitive to illumination changes and can be used to derive an accurate local texture difference measure. To reduce the influence of illumination changes, the texture similarity measure $S_T(x,y)$ at the pixel (x,y) is defined as follows:

$$S_T(x,y) = \begin{cases} \frac{I_T(x,y)}{B_T(x,y)} & if \quad I_T(x,y) < B_T(x,y) \\ 1 & if \quad I_T(x,y) = B_T(x,y) \\ \frac{B_T(x,y)}{I_T(x,y)} & if \quad I_T(x,y) > B_T(x,y) \end{cases} \tag{2.36}$$

where $I_T(x,y)$ and $B_T(x,y)$ denote the texture $ULBP$ code of pixel (x,y) in the current and background frames, respectively. Note that $S_T(x,y)$ is close to 1 if $I_T(x,y)$ and $B_T(x,y)$ are very similar.

2.4.3. Fusion of feature similarity measures using Choquet Integral

The Choquet integral[104] is adopted to fuse the similarity measure for foreground detection.

Let h be a fuzzy measure on a finite set X of criteria, and a non-additive measure on a subset of X is any function μ: X→ [0, 1].

Definition 1: The Choquet integral of μ with respect to *h* is defined by:

$$C_h = \sum_{i=1}^{n} (\mu(x_{\sigma(i)}) - \mu(x_{\sigma(i-1)}))h\,(A_{\sigma(i)}) \tag{2.37}$$

where σ is a permutation of the indices such that $0 \le \mu(x_{\sigma(1)}) \le \mu(x_{\sigma(2)}) \le \cdots \le \mu(x_{\sigma(n)}) \le 1$, $\mathrm{X} = \{x_1, x_2, \cdots, x_n\}$ and $A_{\sigma(i)} = \{\sigma(1), \sigma(2), \cdots, \sigma(n)\}$.

For each pixel, a similarity measure is computed in different dimensions from the background and current frame, as explained in Section 4. We define the set of criteria $\mathrm{X} = \{x_1, x_2, x_3, x_4\}$ with $\{x_1, x_2, x_3\}$ being the three color-component features of the chosen color space, and x_4 being texture features obtained from the ULBP code. For every x_i, let $h(x_i)$ be the importance that the feature x_i takes in the decision of the foreground detection process. The fuzzy functions $\mu(x_i)$ are defined in [0,1], so

that $\mu(x_k) = S_k(x,y)$ (where $k \in \{1,2,3\}$) and $\mu(x_4) = S_T(x,y)$. To compute the value of the Choquet integral for each pixel, we first use permutation function σ to rearrange the features x_i in the finite set X with respect to the order: $\mu(x_1) \leq \mu(x_2) \leq \mu(x_3) \leq \mu(x_4)$.

The pixel (x,y) is considered as the foreground if its Choquet integral value is less than a predetermined threshold TH, as follows:

If $C_h(x,y) < TH$ then *(x, y)* is foreground

2.4.4. Background maintenance

The movement and stopping of a vehicle can lead to changes in the foreground and background. For example, a moving vehicle stops at a certain time t, and then the next moment, the vehicle will change from the foreground to the background frame. Furthermore, the illumination of the entire video may change over time. Therefore, the background maintenance process is a critical step in moving target detection. Background maintenance determines how the background adapts itself to take into account the critical situations that can occur.

In a traditional blind background maintenance method, a learning rate α is defined. In the selective background maintenance method, based on a pixel that belongs to the foreground or background, two learning rates are defined: α and β. In this chapter, we propose an adaptive background maintenance algorithm based on literature [10][37]. Our adaptive algorithm can solve the traditional problems in the blind and selective background maintenance, such as ghosts, etc. More specifically, the values of pixels classified as the foreground are taken into account in the computation of the new background and therefore pollute the background image [30].

The adaptive background maintenance algorithm is defined as follows:

$$CB_{n+1} = a_n \times IB_n + (1 - a_n) \times CB_n \tag{2.38}$$

where CB_n is the current background, and IB_n is an instantaneous background that is computed as follows:

$$IB_n(x,y) = \begin{cases} F_n(x,y) & if\ MP(x,y) = 0 \\ \mathrm{CB}_n(\mathrm{x},\mathrm{y}) & if\ MP(x,y) = 1 \end{cases} \tag{2.39}$$

where a_n is a variable learning rate in the interval [0,1]. The value of a_n is based on changes in the frame structure and illumination between the current frame and the background frame, and can be computed as follows:

$$a_n = 0.9 \times a_{n-1} + 0.1 \times a\_inst_n \tag{2.40}$$

In theory, when there are rapid changes in illumination, the weight a_n should be set to a higher value, and when the changes are slow, it should be set to a lower value, because a high weight allows adaptation to rapid changes in illumination, and a low weight reduces the effect of moving targets on background estimation.

Here, a_inst_n is defined as an adaptive weight on illumination normalization between frame F_n and frame F_{n-1}, as follows:

$$a\_inst_n = \frac{sum\_unmov_{n,n-1}}{area\_unmov_{n,n-1}} \tag{2.41}$$

Because the moving objects' coverage areas in this scene do not reflect illumination changes, the weight on illumination normalization is only calculated in non-moving target areas:

$$area\_unmov_{n,n-1} = \sum_{(x,y)}(1 - MP(x,y)) \tag{2.42}$$

where $MP(x,y) \in MP_n \cup MP_{n-1}$.

Here, $area\_unmov_{n,n-1}$ represents the number of pixels of the non-moving area in the current frame. In other words, it is the area defined by the current frame minus the union of MP_n and MP_{n-1}, and MP_n and MP_{n-1} are the moving pixels in the frame F_n and the frame F_{n-1}, respectively. They are binary map, where a target pixel value is 1, and a non-target pixel value is 0.

Here, $sum\_unmov_{n,n-1}$ corresponds to a change in color or gray-scale between two consecutive frames F_n and F_{n-1}:

$$sum\_unmov_{n,n-1} = \sum_{(x,y)} \frac{|F_n(x,y) - F_{n-1}(x,y)|}{256} \tag{2.43}$$

2.5. Experimental results

In order to verify the effectiveness of the proposed algorithms, we analyzed the experimental results obtained from the three phases, namely, background initialization, foreground detection, and background maintenance.

In the background initialization phase, we created a test process involving in capturing videos of four scenes, as shown in Figure.2.9, to validate whether a color that is more frequent at a specific pixel in a series of frames is more likely to belong to the background rather than the foreground. Scene I and Scene II are actual traffic scenes from a static camera. Scene III is from the PETS2001 dataset [38] widely used in video surveillance evaluation, and Scene IV is from test data in the ATON project [39].

In each scene, with different illumination and traffic conditions, two pixel positions (point A and point B) were selected. In 200 consecutive frames, we calculated the mean and standard deviation of the selected pixels, and we compare our experimental results to verify the process. The results are shown in Table 2.1. From a comparison of the results for the mean and standard deviation of actual consecutive frames and a virtual background model created with our proposed method, we found that the background model results were more stable.

In the foreground detection phase, we conducted an experiment to compare three different methods for the same video. Experimental results are showed in Figure 2.10. Quantitative evaluation was based on the similarity measure derived in literature [40].

Let A be a detected region and B be the corresponding ground truth. The similarity between A and B can be defined as:

$$S(A,B) = \frac{A \cap B}{A \cup B} \tag{2.44}$$

This quantity approaches 1 when A and B are similar, and 0 otherwise. The results are showed in Table 2.2.

In the background maintenance phase, because different background maintenance methods will produce different background model frames which are segmented by the same segmentation methods, the moving object is not the same. Therefore, we take the average ratio of the moving object image and the ground truth, with the aim of reflecting the effect of different background maintenance methods. Experimental results

(a1) current frame (a2) background model (a3) moving object

Scene I

(b1) current frame (b2) background model (b3) moving object

Scene II

(c1) current frame (c2) background model (c3) moving object

Scene III [20]

(d1) current frame (d2) background model (d3) moving object

Scene IV [21]

Figure 2.9 Snapshots taken from the four captured scenes, and experimental results. In each scene, two pixels, point A and point B, were selected. Evaluation results are presented in Table 2.1.

Table 2.1 Results for background model, showing comparison of other experiment [26] in which mean and standard deviation were obtained from 200 frames and by employing our proposed method.

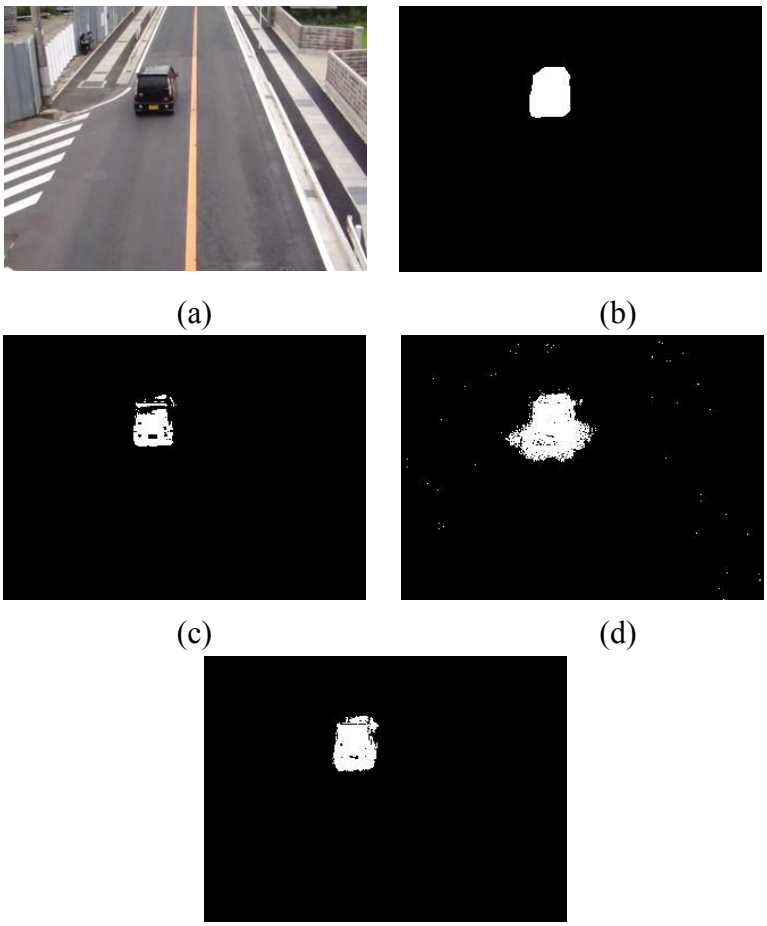
			Y		Cb		Cr	
			Experimental	Our method	Experimental	Our method	Experimental	Our method
Scene I	Point A	Mean	129.50	132.60	128.70	128.70	127.99	127.99
		SD	4.46	0.90	0.49	0.49	0.26	0.26
	Point B	Mean	117.02	111.78	129.78	129.66	127.88	128.12
		SD	31.07	0.75	1.04	0.69	1.38	0.39
Scene II	Point A	Mean	104.50	106.58	128.89	128.92	129.14	129.77
		SD	31.63	0.83	3.65	0.57	3.54	0.91
	Point B	Mean	128.75	122.84	128.83	128.91	129.78	130.09
		SD	27.79	1.13	1.16	0.48	1.10	0.51
Scene III	Point A	Mean	145.20	145.00	131.82	131.72	127.25	127.26
		SD	1.62	1.12	0.87	0.72	0.37	0.31
	Point B	Mean	132.17	132.04	133.92	133.92	127.41	127.41
		SD	1.31	0.99	0.22	0.22	0.09	0.09
Scene IV	Point A	Mean	127.52	126.89	131.19	131.66	128.50	128.50
		SD	1.86	1.14	1.18	0.90	0.54	0.54
	Point B	Mean	116.70	116.20	131.05	130.81	128.77	128.80
		SD	1.47	1.00	1.31	1.10	0.57	0.50

for different background maintenance methods are showed in Figure 2.11. The quantitative evaluation is showed in Table 2.3.

Note that the effects of different background updating methods on the experimental results are significant.

Table 2.2 Quantitative evaluation of object detection.

Method	*S(A,B)* %
Conventional background subtraction	66.00
MoG[12]	50.72
Proposed method	81.20



(a) (b) (c) (d) (e)

Figure 2.10 Results of vehicle detection, showing (a) the current frame, (b) the ground truth, and the results obtained with (c) conventional background subtraction, (d) MoG[12], and (e) the proposed method.

Table 2.3 Evaluation of background maintenance methods.

Maintenance method	*S(A,B)* %
Blind Maintenance	76.65
Selective Maintenance	77.74
Adaptive Maintenance	81.20

2.6. Summary

In this chapter, a novel background subtraction approach for detecting moving

vehicles from video frames is proposed. For background initialization, an algorithm based on statistical color sampling per pixel over time was presented. The algorithm

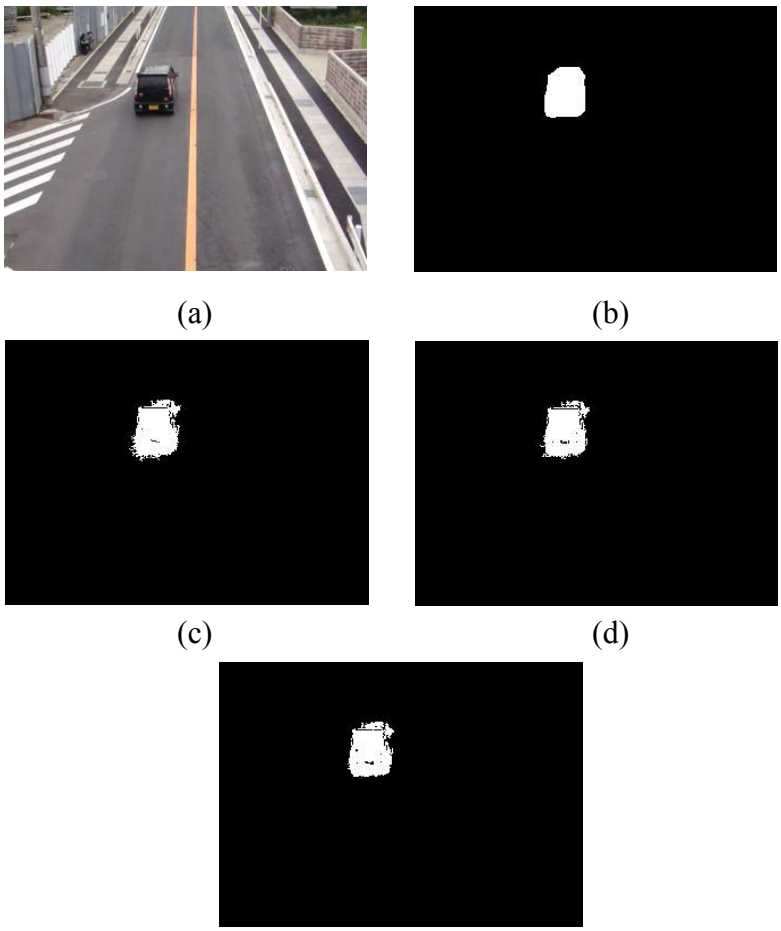


Figure 2.11 Comparison of maintenance methods, showing (a) the current frame, (b) the ground truth, and results obtained with (c) blind maintenance, (d) selective maintenance, and (e) adaptive maintenance.

was robust in reconstructing the actual background. For foreground detection, we presented a novel fuzzy background model. This method uses the Choquet integral for fusing color features and texture features in YCbCr color space. This color space was chosen instead of RGB, which removes most of the shadows, and aggregates ULBP texture features, which have better descriptive abilities than LBP textures, allowing more effective distinction of textures and adapting to illumination changes. For background maintenance, we proposed a novel adaptive background maintenance

method that is more adaptive to scene and illumination changes. The proposed algorithm was tested against multiple media and several standard video surveillance datasets containing different illuminations and moving object scenes. The experimental results showed that the proposed method was more robust and efficient in a quantitative evaluation.

Chapter 3. Single target tracking with PF and multiple feature fusion

In this chapter, firstly, we introduce the overview of target tracking techniques, and survey the particle filter techniques. Secondly, multiple feature fusion algorithm for single target tracking based on particle filter is proposed, especially multiple feature fusion mechanism which improves the represent-ability of tracking target and dynamically balances the effect of feature similarity and feature discriminability among target object, candidate and adjacent background. Lastly, we compare the state-and-art methods to demonstrate the effectiveness of this proposed method.

3.1. Introduction

Visual target tracking is an important step in many applications such as intelligent transport, human-computer interaction, guidance, obstacle avoidance and gesture recognition. Target tracking problems can be formulated as a hidden state estimation problem given available measurements. The measurements are taken at regular intervals and the task is to estimate the state of a target at each point in time, such as its position, velocity or other attribute.

Many different techniques have been proposed for tracking object in literatures. For example, in which Mean shift algorithm was proposed [15][41], the feature histograms based on target representations are regularized by spatial masking with an isotropic kernel. The template of target is achieved via detection and a cost function is established to describe the similarity between target candidate and template. According to the maximum of the cost function, target location is determined. Mean shift is quick and effective in some scenes. However, it cannot be adopted in targets with large scale variation.

A number of Bayesian stochastic filters have been used with success in object tracking applications. The most commonly used are Linear Kalman Filter (LKF)[13][42][43], Extended Kalman Filter (EKF)[24][44], and Unscented Kalman Filter (UKF)[14][45].

These filters have been utilized to relax the linearity assumption of the system model and thus to improve the tracking capabilities. However, these methods are limited to linear or Gaussian systems, and single modal probability distribution. In more complex scenes, the detectability of moving object will significantly reduce or track failure.

The particle filter methods [16] have been introduced for object tracking in nonlinear, non-Gaussian, multi-modal situations. These algorithms are based on sequential Monte Carlo sampling methods and represent the required posterior density function by a set of random samples with associated weights and to compute an estimation based on these samples and weights [46].

To represent target objects, a common solution is to use various features, such as color, edges, texture and motion etc. Color histograms have been widely used in the particle filter for likelihood estimation [47][48][49]. Color histograms are robust to partial occlusions, unrestrictive about the type of objects being tracked and can be computed efficiently. However, the main problem in tracking objects with a single color feature is ambiguous in scene with objects or regions with similar color properties to those of target objects. To improve the target model representation, a combination of features is commonly used. For instance, spatial information based on multiple color histograms computed on semi-overlapping image areas can be introduced [50]. The combination of shape and color is used [51]. The likelihood of each feature is calculated and weighted before Bayes' rule is applied to obtain the resultant posterior. Multiple multimodal features such as color, motion and sound can be fused non-adaptively assuming conditional independence of the features given the state [52]. A consistent histogram for the analysis of color, edges, and texture features was proposed [53]. The features' noise parameters are changed adaptively and the features are adaptively weighted. Combining color and the local scale invariant feature transform (SIFT), Harris and SIFT methods were proposed in recent year [54][55]. In the literature [54], the particle weight is calculated firstly by color similarity measurement and then is updated according to the distribution of SIFT matches. However, only use the average values of the matches points of SIFT features to approximate the object center. In the literature [55], Harris and SIFT feature are adopted to establish the object model, and Harris corner is applied to generate the SIFT feature vector which has significantly reduced the time complexity, and effectively tracked the object under the simple scene.

3.2. Particle filter techniques

Bayesian theory was originally discovered by the British researcher Thomas Bayes in a posthumous publication in 1763 [56]. However, Bayes theory has not gained a lot of attention in the early days until Bayesian inference was developed by the French mathematician Laplace [57]. Bayesian inference, which applies Bayesian statistics to statistical inference, has become one of the most important branches in statistics and has been successfully applied in statistical decision, detection and estimation, pattern recognition, and machine learning. The early idea of Monte Carlo was designed to estimate the number π, but the modern formulation of Monte Carlo methods started from physics in the 1940s and came to statistics later in the 1950s. Roughly speaking, Monte Carlo technique is a kind of stochastic sampling approach aiming to tackle the complex systems which are analytically intractable. In recent years, sequential Monte Carlo (SMC) approaches have attracted more and more attention to the researchers from different areas with many successful applications in signal processing, machine learning, tracking and many others.

It is a technique for implementing a recursive Bayesian filter by MC simulations. The key idea is to represent the required posterior density function by a set of random samples with associated weights and to compute estimates based on these samples and weights. As the number of samples becomes very large, this MC characterization becomes an equivalent representation to the usual functional description of the posterior probability density function (pdf), and the sequential importance sampling (SIS) filter approaches the optimal Bayesian estimate.

A number of techniques for this type of filtering have been proposed and successfully applied, but our main focus is the SMC estimation, collectively referred to as particle filter (PF).

The following sections present a general review of particle filter theory from a Bayesian perspective. To our interest, we present the Bayesian estimation. After the Bayesian estimation, we show the mathematical formulations of a recursive Bayesian filter using the Monte Carlo method.

3.2.1. Bayesian estimation

To define the problem of tracking, the state space equations of targets are defined by:

$$x_k = f_k(x_{k-1}, w_{k-1}) \tag{3.1}$$

$$y_k = h_k(x_k, v_k) \tag{3.2}$$

where $f_k: \Re^{n_x} \times \Re^{n_w} \to \Re^{n_x}$ and $h_k: \Re^{n_x} \times \Re^{n_v} \to \Re^{n_y}$ are two possibly nonlinear function. $\{w_{k-1}, k \in \mathbb{N}\}$ is an independent and identically distributed (iid) process noise sequence. $\{v_k, k \in \mathbb{N}\}$ is an iid measurement noise sequence. n_x, n_w are dimensions of the state and process noise vectors, respectively; n_y,n_v are dimensions of the measurement and measurement noise vectors, respectively. In particular, we seek filtered estimates of x_k based on the set of all available measurements $y_{1:k} = \{y_i, i = 1, \cdots, k\}$ up to time k.

From a Bayesian perspective, the tracking problem is to recursively calculate some degree of belief in the state x_k at time k, taking different values, given the data $y_{1:k}$ up to time k. Thus, it is required to construct the pdf $p(x_k|y_{1:k})$. It is assumed that the initial pdf $p(x_0|y_0) \equiv p(x_0)$ of the state vector, which is also known as the prior. Then, the pdf $p(x_k|y_{1:k})$ may be recursively obtained in following two stages: prediction and update.

$$p(x_k|y_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|y_{1:k-1})dx_{k-1} \tag{3.3}$$

$$p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_k|y_{1:k-1})}{p(y_k|y_{1:k-1})} \tag{3.4}$$

where $p(x_k|x_{k-1})$ is defined by the system equation of target (3.1), $p(y_k|x_k)$ is defined by the measurement equation of target (3.2). The normalizing constant

$$p(y_k|y_{1:k-1}) = \int p(y_k|x_k)p(x_k|y_{1:k-1})dx_k \tag{3.5}$$

depends on the likelihood function $p(y_k|x_k)$. Thus, starting from the initial density $p(x_0)$ one can, at least in principle, recursively arrive at the desired density $p(x_k|y_{1:k})$. After getting $p(X_k|Z_{1:k})$, we can calculate the optimal estimation of a minimum mean square error sense

$$\hat{x}_k = \int x_k p(x_k|y_{1:k}) dx_k \tag{3.6}$$

3.2.2. Monte Carlo simulation

From the previous section, in solving prediction equation (3.3) and update equation (3.4), the integral operation is required, generally very difficult to solve for high-dimensional variable integral operation. In addition to linear, Gaussian assumptions, an analytical solution can be obtained, namely the Kalman filter algorithm, the other must be solved by the numerical approximation method. Monte Carlo method for solving the problem of high dimensional variable integral provides a simple and effective method.

The recursive estimation given above, leads to expressions that are impossible to evaluate analytically unless strong assumptions are made. In a more general framework, we use the Monte Carlo methods to overcome this problem.

Suppose we are able to sample N iid random variables, $x_{0:k}^{(i)} \sim p(x_{0:k}|y_{1:k})$ for $i = 1, \cdots, N$. Then the Monte Carlo method approximates $p(x_{0:k}|y_{1:k})$ by the empirical measure [46].

$$P_N(dx_{0:k}|y_{1:k}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_{0:k}^{(i)}}(dx_{0:k}) \tag{3.7}$$

where $\delta_{x_{0:k}^{(i)}}(dx_{0:k})$ denotes the Dirac-delta mass located in $x_{0:k}^{(i)}$. Subsequently one can approximate the marginal $p(x_k|y_{1:k})$ as

$$P_N(dx_k|y_{1:k}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_k^{(i)}}(dx_k) \tag{3.8}$$

And expectations of the form

$$I(g_k) = \int g_k(x_{0:k}) p(x_{0:k}|y_{1:k}) dx_{0:k} \tag{3.9}$$

$$I_N(g_k) = \int g_k(x_{0:k}) P_N(dx_{0:k}|y_{1:k}) = \frac{1}{N}\sum_{i=1}^{N} g_k(x_{0:k}^{(i)}) \tag{3.10}$$

This estimate is unbiased and according to the law of large numbers, I_N will almost surely converge to I [58]. Moreover, if the variance of $g_k(x_{0:k})$,

$$\sigma^2 = \int (g_k(x_{0:k}) - I)^2 p(x_{0:k}|y_{1:k}) dx_{0:k} \tag{3.11}$$

is finite, the central limit theorem will hold and the estimation error will converge in distribution as

$$\lim_{N\to\infty} \sqrt{N}(I_N - I) \sim \mathcal{N}(0, \sigma^2) \tag{3.12}$$

The error of the estimate of the stochastic integration, $e = (I_N - I)$, is of the order $O(N^{-\frac{1}{2}})$, thus the rate of convergence of this estimate is independent of the dimension of the integrand. In contrast, any deterministic integration method has a rate of convergence that decreases as the dimension of the integrand increase. This is one of the mina advantages of Monte Carlo integration methods, although the rate of convergence is very slow.

Unfortunately, it is usually impossible to sample effectively from the posterior pdf $p(x_{0:k}|y_{1:k})$, which is typically multivariate, nonstandard, and only known up to a proportionality constant. An effective solution is to introduce a known probability density distribution $q(x_{0:k}|y_{1:k})$, usually referred as importance density or proposal density.

3.2.3. Bayesian importance sampling

Suppose we cannot generate samples directly from $p(x_{0:k}|y_{1:k})$ to estimate I_N using equation (3.10). One can introduce an arbitrary distribution $q(x_{0:k}|y_{1:k})$ of importance probability density, from which it is easy to sample and whose support covers the support of $p(x_{0:k}|y_{1:k})$. This distribution is known as importance distribution (also often referred to as proposal distribution) and the integration in equation (3.9) can be rewritten as

$$\begin{aligned} I(g_k) &= \int g_k(x_{0:k}) \frac{p(x_{0:k}|y_{1:k})}{q(x_{0:k}|y_{1:k})} q(x_{0:k}|y_{1:k}) dx_{0:k} \\ &= \int g_k(x_{0:k}) \frac{p(y_{1:k}|x_{0:k}) p(x_{0:k})}{p(y_{1:k}) q(x_{0:k}|y_{1:k})} q(x_{0:k}|y_{1:k}) dx_{0:k} \\ &= \int g_k(x_{0:k}) \frac{w_k^*(x_{0:k})}{p(y_{1:k})} q(x_{0:k}|y_{1:k}) dx_{0:k} \end{aligned} \tag{3.13}$$

where $w_k^*(x_{0:k})$ is known as the importance weight, which is given by

$$w_k^*(x_{0:k}) = \frac{p(y_{1:k}|x_{0:k})p(x_{0:k})}{q(x_{0:k}|y_{1:k})} \tag{3.14}$$

The equation (3.13) is the integral of $x_{0:k}$, so $y_{1:k}$ can be treated as a constant. Thus, $I(g_k)$ can now be written as

$$I(g_k) = \frac{\int g_k(x_{0:k})w_k^*(x_{0:k})q(x_{0:k}|y_{1:k})dx_{0:k}}{p(y_{1:k})} \tag{3.15}$$

And because

$$\begin{aligned} p(y_{1:k}) &= \int p(y_{1:k}, x_{0:k})\, dx_{0:k} \\ &= \int \frac{p(y_{1:k}|x_{0:k})p(x_{0:k})q(x_{0:k}|y_{1:k})}{q(x_{0:k}|y_{1:k})} dx_{0:k} \\ &= \int w_k^*(x_{0:k})\, q(x_{0:k}|y_{1:k})dx_{0:k} \end{aligned} \tag{3.16}$$

Replace formula (3.16) to formula (3.15)

$$I(g_k) = \frac{\int g_k(x_{0:k})w_k^*(x_{0:k})q(x_{0:k}|y_{1:k})dx_{0:k}}{\int w_k^*(x_{0:k})q(x_{0:k}|y_{1:k})dx_{0:k}} \tag{3.17}$$

Accordingly, one can simulate N independent and identically distributed samples (also known as particles), $x_{0:k}^{(i)} \sim q(x_{0:k}|y_{1:k})$, for $i = 1, \cdots, N$ and the Monte Carlo approximation of the integration in equation (3.17) can be given by

$$I(g_k) \approx \hat{I}_N(g_k) = \frac{\frac{1}{N}\sum_{i=1}^N g_k\left(x_{0:k}^{(i)}\right)w_k^*\left(x_{0:k}^{(i)}\right)}{\frac{1}{N}\sum_{i=1}^N w_k^*\left(x_{0:k}^{(i)}\right)} = \sum_{i=1}^N g_k\left(x_{0:k}^{(i)}\right)w_k(x_{0:k}^{(i)}) \tag{3.18}$$

where $w_k(x_{0:k}^{(i)})$ is the normalized importance weight, it is given by

$$w_k\left(x_{0:k}^{(i)}\right) = \frac{w_k^*\left(x_{0:k}^{(i)}\right)}{\sum_{i=1}^N w_k^*\left(x_{0:k}^{(i)}\right)} \tag{3.19}$$

Bayesian importance sampling is to use a series of random samples and their weights to approximate the posterior probability density, and then to get an estimate of the required statistics. The posterior probability density of random variable can be approximated by a series of discrete samples and the weights, and the degree of

approximation depends on the number of samples N. Typically, the posterior probability density of the random variable can not be directly obtained, but Bayesian importance sampling provides an effective solution to this problem.

3.2.4. Sequential importance sampling

The Bayesian importance sampling method as described above, has the following defects. As with the arrival of new measurement y_k, we need to re-draw samples $\left\{x_{0:k}^{(i)}, i = 1, \cdots, N\right\}$ from the importance probability density $q(x_{0:k}|y_{1:k})$, and have to re-compute the importance weights $w_k\left(x_{0:k}^{(i)}\right)$ over the entire state sequence (3.14). As a result, the computational complexity increases with time. It is not conductive to the practical application. Sequential importance sampling (SIS) addresses this problem in the context of sequential estimation. SIS rewrites the posterior density using Bayesian theorem, and the weights are also recursively updated. This leads to a particle filter that is often referred to as SIS.

To recursively estimate the probability density, the importance density is chosen to factorize such that

$$q(x_{0:k}|y_{1:k}) = q(x_k|x_{0:k-1,}y_{1:k})q(x_{0:k-1}|y_{1:k-1}) \tag{3.20}$$

Then one can obtain samples $x_{0:k}^{(i)} \sim q(x_{0:k}|y_{1:k})$ by augmenting each of the existing samples $x_{0:k-1}^{(i)} \sim q(x_{0:k-1}|y_{1:k-1})$ with the new state $x_k^{(i)} \sim q\left(x_k\middle|x_{0:k-1,}y_{1:k}\right)$. Replace the formula (3.21) to the formula (3.14)

$$w_k^*(x_{0:k}) = \frac{p(y_{1:k}|x_{0:k})p(x_{0:k})}{q(x_k|x_{0:k-1,}y_{1:k})q(x_{0:k-1}|y_{1:k-1})} \tag{3.21}$$

And according to the formula (3.14), we can obtain

$$w_{k-1}^*(x_{0:k-1}) = \frac{p(y_{1:k-1}|x_{0:k-1})p(x_{0:k-1})}{q(x_{0:k-1}|y_{1:k-1})} \tag{3.22}$$

Last, according to the formula (3.21) and the formula (3.22), we can obtain

$$w_k^*(x_{0:k}) = w_{k-1}^*(x_{0:k-1})\frac{p(y_k|x_k)p(x_k|x_{k-1})}{q(x_k|x_{0:k-1,}y_{1:k})} \tag{3.23}$$

The SIS algorithm thus consists of recursive propagation of the weights and support

points as each measurement is received sequentially. A pseudo-code description of this algorithm is given in the following section.

SIS Particle Filter algorithm

For times $k = 0,1,2,\cdots$

1). Prediction

For $i = 1,\cdots,N$, sample $x_k^{(i)} \sim q\left(x_k \middle| x_{0:k-1}^{(i)}, y_{1:k}\right)$, and set $x_{0:k}^{(i)} = \left\{x_k^{(i)}, x_{0:k-1}^{(i)}\right\}$;

2). Update

For $\mathrm{i} = 1,\cdots,\mathrm{N}$, use the formula (3.23) to evaluate the importance weights of each new sample $\mathrm{w}_\mathrm{k}^*\left(\mathrm{x}_{0:\mathrm{k}}^{(\mathrm{i})}\right)$ up to a normalizing constant:

$$\mathrm{w}_\mathrm{k}^*\left(\mathrm{x}_{0:\mathrm{k}}^{(\mathrm{i})}\right) = \mathrm{w}_{\mathrm{k}-1}^*\left(\mathrm{x}_{0:\mathrm{k}-1}^{(\mathrm{i})}\right)\frac{\mathrm{p}\left(\mathrm{y}_\mathrm{k}\middle|\mathrm{x}_\mathrm{k}^{(\mathrm{i})}\right)\mathrm{p}\left(\mathrm{x}_\mathrm{k}^{(\mathrm{i})}\middle|\mathrm{x}_{\mathrm{k}-1}^{(\mathrm{i})}\right)}{\mathrm{q}\left(\mathrm{x}_\mathrm{k}^{(\mathrm{i})}\middle|\mathrm{x}_{0:\mathrm{k}-1}^{(\mathrm{i})},\mathrm{y}_{1:\mathrm{k}}\right)} \tag{3.24}$$

The importance weights are normalized, given by

$$\mathrm{w}_\mathrm{k}\left(\mathrm{x}_{0:\mathrm{k}}^{(\mathrm{i})}\right) = \frac{\mathrm{w}_\mathrm{k}^*\left(\mathrm{x}_{0:\mathrm{k}}^{(\mathrm{i})}\right)}{\sum_{\mathrm{i}=1}^{\mathrm{N}}\mathrm{w}_\mathrm{k}^*\left(\mathrm{x}_{0:\mathrm{k}}^{(\mathrm{i})}\right)} \tag{3.25}$$

End time goes to $k+1$, and algorithm goes to importance sampling step.

Last, we can obtain the samples with the weights $\left\{\left(x_{0:k}^{(i)}, w_k\left(x_{0:k}^{(i)}\right)\right), i = 1,\cdots,N\right\}$, thus the probability density can be approximated as

$$p(x_{0:k}|y_{1:k}) \cong \sum_{i=1}^{N} w_k\left(x_{0:k}^{(i)}\right)\delta\left(x_{0:k} - x_{0:k}^{(i)}\right) \tag{3.26}$$

For facilitating description, we will simplify $w_k^*\left(x_{0:k}^{(i)}\right)$ to $w_k^*(i)$, $w_k\left(x_{0:k}^{(i)}\right)$ to $w_k(i)$ in the following thesis.

According the above description, the probability density of x_k is approximated by

$$p(x_k|y_{1:k}) \cong \sum_{i=1}^{N} w_k(i)\delta\left(x_k - x_k^{(i)}\right) \tag{3.27}$$

And the optimal estimation of minimum mean square error of x_k is given as

$$\hat{x}_k = \sum_{i=1}^{N} w_k(i)x_k^{(i)} \tag{3.28}$$

Sequential importance sampling is a recursive Bayesian importance sampling

algorithm. So far, a variety of particle filtering algorithms are based on the sequential importance sampling method.

With sequential importance sampling, we are able to avoid the difficulty of sampling directly from the posterior density by sampling from an importance or proposal distribution. In this section, we find that the posterior density function can be approximated well by drawing samples from a proposal distribution. However, this algorithm has a problem which prevents from using it. Another algorithm, sequential importance re-sampling, still needs to be introduced to solve the existing problem.

3.2.5. Particle degeneracy and resampling

The problem of the updating process is that the variance of the weights increases exponentially over time, which means that after a few iterations, the distribution of importance weights becomes more and more skewed. As time increases, all but one particle has negligible weights. This is known as degeneracy phenomenon. This degeneracy implies that a large computational effort is devoted to updating particles whose contribution to the approximation to the probability density is almost zero. The effective methods for reducing its effect are the following two methods: good choice of importance density and use of resampling. These are described next.

Good choice of importance density

The optimal importance function was introduced in the literature [16].

$$q(x_k|x_{0:k-1}^{(i)}, y_{1:k})_{opt} = p(x_k|x_{k-1}^{(i)}, y_k) \tag{3.29}$$

And proof the variance of the importance weight $var(w_k^*(i)) = 0$ conditional upon $x_{k-1}^{(i)}$ and y_k. This choice of importance density is optimal since for a given $x_{k-1}^{(i)}$, $w_k(i)$ take the same value, whatever sample is drawn from $q(x_k|x_{0:k-1}^{(i)}, y_{1:k})_{opt}$. Hence, conditional on $x_{k-1}^{(i)}$, $var(w_k^*(i)) = 0$. This is the variance of the different $w_k(i)$ resulting from different sampled $x_k^{(i)}$. This optimal importance density suffers from two major drawbacks. It requires the ability to sample from $p\left(x_k\middle|x_{k-1}^{(i)}, y_k\right)$ and to evaluate the integral over the new state. In the general case, it may not be straightforward to do either of these things.

A simple and widely used method is to choose the prior system state transition probability density as the importance density, such as

$$q\left(x_k \middle| x_{0:k-1}^{(i)}, y_{1:k}\right) = p(x_k | x_{k-1}^{(i)}) \tag{3.30}$$

Substitution of formula (3.30) into the formula (3.24) then yields

$$w_k^*(i) = w_{k-1}^*(i)\, p(y_k | x_k^{(i)}) \tag{3.31}$$

The method is often inefficient in simulations as the state space is explored without any knowledge of the observations. It is especially sensitive to outliers. However, it does have the advantage that the importance weights are easily evaluated. This would seem to be the most common choice of importance density in the several particles filter methods.

Resampling

The second method by which the effects of degeneracy can be reduced is to use resampling whenever a significant degeneracy is observed. The basic idea of resampling is to eliminate particles that have small weights and to concentrate on particles with large weights. In formal term, resampling step replaces the weighted empirical measure by the un-weighted measure

$$\tilde{P}_N(dx_{0:k} | y_{1:k}) = \frac{1}{N}\sum_{i=1}^{N} N_k^{(i)} \delta_{x_{0:k}^{(i)}}(dx_{0:k}) \tag{3.32}$$

where $N_k^{(i)}$ is the number of offspring associated to particle$x_{0:k}^{(i)}$; $N_k^{(i)}$ is an integer number such that $\sum_{i=1}^{N} N_k^{(i)} = N$. If $N_k^{(j)} = 0$ then the particles $x_{0:k}^{(j)}$ dies. The surviving particles $x_{0:k}^{(i)}$, i.e. those particles, for which $N_k^{(i)} > 0$, are approximately distributed according to $P(x_{0:k} | y_{1:k})$. Different unbiased resampling schemes have been proposed in the literature, such as multinomial resampling, stratified resampling, residual resampling and systematic resampling. For a review on their comparative performances please refer to the article by Hol et al [59]. Considering resampling quality and computational complexity, the systematic resampling is favourable.

Having discussed about all the necessary ingredients, we are now describing a generic particle filter algorithm. This is explained below.

Generic SMC algorithm

Recursively over time $k = 0,1,2,\cdots$

For $i = 1,\cdots,N$, where N is the total number of particles.

1). Sample $x_k^{(i)} \sim q\left(x_k | x_{0:k-1}^{(i)}, y_{1:k}\right)$ and set $x_{0:k}^{(i)} \triangleq (x_{0:k-1}^{(i)}, x_k^{(i)})$

2). Evaluate the corresponding importance weights $w_k^*(i)$ according to formula (3.23),

3). Normalize the importance weights $w_k(i)$ according to formula (3.25).

4). If $N_{eff} < N_{Th}$, resample from $\left\{x_k^{(i)}\right\}_{i=1}^N$ with probabilities $\{w_k(i)\}_{i=1}^N$ keeping the sample size still to be N and assign equal weights $1/N$.

where $N_{eff} = 1/\sum_{i=1}^N (w_k(i))^2$, N_{Th} is a fixed threshold. To avoid carrying the trajectories with small normalized importance weights and to concentrate upon the ones with large weights, the effective sample size N_{eff} is used to decide resampling.

3.3. Target tracking with multiple feature fusion and PF

3.3.1. Dynamic model

The model considered for the moving object provides invariance under different motions, such as translation, rotation, and scale changes. This can cover the different types of motion of the object, as well as the case where the object size varies considerably, and hence ensures reliable performance of the particle filter [53].

The moving object is modeled by a rectangular bounding box defined in terms of the dynamic state $\mathrm{X} = \left\{x, \dot{x}, y, \dot{y}, h_x, \dot{h_x}, h_y, \dot{h_y}\right\}^T$, where x and y represent the center coordinates of the rectangular box for tracking an object, $\dot{x}$ and $\dot{y}$ represent the respective velocity components, h_x, h_y denote the height and width of half axes, and $\dot{h_x}, \dot{h_y}$ represent the scale changes of h_x and h_y, respectively.

The sample set is propagated through the application of a dynamic model

$$\mathrm{X}_t = A\mathrm{X}_{t-1} + Bw_{t-1}, \tag{3.33}$$

where A and B are the deterministic components of the model, and w_{t-1} is a Gaussian

distribution of zero mean with covariance matrix $Q = diag\left\{\sigma_x^2, \sigma_y^2, \sigma_{h_x}^2, \sigma_{h_y}^2\right\}$, describing the uncertainty in the state vector.

This model has the ability to adapt to transitions of the motion, which helps to estimate the distribution area and size variations of the tracking object more accurately.

3.3.2. Feature likelihood models

This section describes how we model the separate features of the rectangular region surrounding the moving objects and the likelihood models of the features in detail. All of the models are based on histograms. Histograms have the useful property that allow some change in the object appearance without changing the histogram [53].

Color distribution model

Color distributions are used as target models since they achieve robustness against non-rigidity, rotation, and partial occlusion. In the RGB color space, because these three color components are reciprocally dependent, illumination changes have a great influence on the histogram, and the RGB space is not suitable for discrimination under all circumstances. Thus, the HSV color space is selected to represent color information. In the experiments, typical 8×8×4 bins were used to make the histograms less sensitive to intensity variations, and we assumed that the distributions are discretized into m bins. Similar color distributions are also described by Comaniciu [60]. The histograms are produced with a function $b(x_i)$ that assigns the color of location x_i to the corresponding bin. We calculate the color distribution inside a rectangular region with half axes h_x, h_y.

Now, the color distribution of a target object $q_c = \left\{q_c^{(u)}\right\}_{u=1,\cdots,m}$ at point (x_i, y_i) is defined as

$$q_c^{(u)} = \frac{1}{n}\sum_{i=1}^{n}\delta[b(x_i) - u] \tag{3.34}$$

where i is the number of pixels in the object region, and δ is the Kronecker delta function. The normalization factor $1/n$ ensures that $\sum_{u=1}^{m} q_c^{(u)} = 1$, where u is the index of the histogram bin.

In the update step, we use similarity measures to calculate color distributions $p_c(x_t) = \left\{p_c^{(u)}(x_t)\right\}_{u=1,\cdots,m}$ of the candidate at time t. According to the color distributions of the target object and candidate, the similarity measure between these two distributions is then given by the Hellinger distance [61][62].

$$d_c[q_c, p_c(x_t)] = \sqrt{1 - \rho[q_c, p_c(x_t)]}, \tag{3.35}$$

where $\rho[q_c, p_c(x_t)]$ is the Bhattacharyya coefficient that is defined as

$$\rho[q_c, p_c(x_t)] = \sum_{u=1}^{m} \sqrt{q_c^{(u)} \cdot p_c^{(u)}(x_t)} \tag{3.36}$$

The larger the measure ρ is, the more similar these distributions are. Conversely, for the distance d_c, the smaller the value is, the more similar these distributions are. For two identical normalized histograms, if we obtain $d_c = 0$ ($\rho = 1$), this indicates a perfect match.

The likelihood function for the color feature can be defined as

$$p_c(y_t|x_t) \propto exp\left(-\frac{d_c^2[q_c, p_c(x_t)]}{2\sigma_c^2}\right) \tag{3.37}$$

where the standard deviation σ_c specifies the Gaussian noise in the measurements. Note that small Hellinger distances correspond to large weights in the particle filter. The adaptive value σ_c will be updated online.

Edge distribution model

We adopt the local edge orientation histogram (EOH) [63][64][65], which employs gradient orientation information for feature extraction. The EOH is largely invariant to global illumination changes and is capable of capturing geometric properties of a moving object, which is difficult to capture with linear edge filters [64].

The gradients at point $I(x_i, y_i)$ in the target region can be calculated by convolving Sobel masks with the image as follows:

$$G_x(x_i, y_i) = Sobel_x * I(x_i, y_i) \tag{3.38}$$

and

$$G_y(x_i, y_i) = Sobel_y * I(x_i, y_i) \tag{3.39}$$

where $Sobel_x$ and $Sobel_y$ are the Sobel masks of x and y, respectively. The edge strength at the point (x_i, y_i) is defined as

$$G(x_i, y_i) = \sqrt{G_x(x_i, y_i)^2 + G_y(x_i, y_i)^2} \tag{3.40}$$

The orientation of the edge is defined as

$$\theta(x_i, y_i) = arctan\left(\frac{G_x(x_i, y_i)}{G_y(x_i, y_i)}\right) \tag{3.41}$$

We then divide the edges into m bins according to θ in the anticlockwise direction; in other words, the value of each interval is $\mathrm{bin_u}$, $u = 1, \cdots, m$. If m=16, $\theta(x_i, y_i) \in [-\pi/2 + (u-1)\,\pi/16\,, -\pi/2 + u\,\pi/16]$. Then, the edge strength that belongs to the u-th interval at point $(\mathrm{x_i}, \mathrm{y_i})$ is

$$\varphi_u(x_i, y_i) = \begin{cases} G(x_i, y_i), & if\ \theta(x_i, y_i) \in bin_u \\ 0, & otherwise \end{cases} \tag{3.42}$$

Thus, the edge orientation histogram can be defined as

$$q_e^{(u)} = \sum_{i=1}^{n} \varphi_u(x_i, y_i), \tag{3.43}$$

where n is the number of pixels in the target region. Suppose the edge distribution of the target object is $q_e = \left\{q_e^{(u)}\right\}_{u=1,\cdots,m}$, and the edge distribution of candidate regions is $p_e(x_t) = \left\{p_e^{(u)}(x_t)\right\}_{u=1,\cdots,m}$ at time t. As described above, we use the Bhattacharyya coefficient to describe the similarity measure between these two distributions:

$$d_e[q_e, p_e(x_t)] = \sqrt{1 - \rho[q_e, p_e(x_t)]}. \tag{3.44}$$

Here, the Bhattacharyya coefficient is

$$\rho[q_e, p_e(x_t)] = \sum_{u=1}^{m} \sqrt{q_e^{(u)} \cdot p_e^{(u)}(x_t)}. \tag{3.45}$$

The likelihood function for an edge feature can be defined as

$$p_e(y_t|x_t) \propto exp\left(-\frac{d_e^2[q_e,p_e(x_t)]}{2\sigma_e^2}\right), \tag{3.46}$$

where the standard deviation σ_e specifies the Gaussian noise in the measurements.

Texture distribution model

Texture is an important characteristic for describing the target. We adopt the extended LBP texture operator [66], which has recently shown excellent performance in many applications. LBP is a gray-scale invariant texture primitive statistic. It is a powerful means of texture description. The operator labels the pixels of an image region by thresholding the neighborhood of each pixel with the center value and considering the result as a binary pattern (see Figure 2.7). The LBP is described in the section 2.3.2 texture feature in Chapter 2. In our experiments, we set $P = 6, R = 3$ and *th* =3.

Thus, the texture histogram of the image $I(x_i, y_i)$ can be defined as

$$q_t^{(u)} = \sum_{i=1}^{n} F\{LBP(x_i, y_i) = u\}, \tag{3.47}$$

where $u = 0, \cdots, m-1$. As described above, the LBP operator set is separated into $m = 64$ bins. The function $F\{\cdot\}$ is defined as

$$F\{A\} = \begin{cases} 1, & A \ \ is\ True \\ 0, & A \ \ is\ False \end{cases} \tag{3.48}$$

Suppose that the texture distribution of the target object is $q_t = \left\{q_t^{(u)}\right\}_{u=1,\cdots,m}$, and the texture distribution of the candidate is $p_t(x_t) = \left\{p_t^{(u)}(x_t)\right\}_{u=1,\cdots,m}$ at time t. The similarity measure between these two distributions is described as

$$d_t[q_t, p_t(x_t)] = 1 - \sum_{u=1}^{m} min\left[q_t^{(u)}, p_t^{(u)}(x_t)\right] \tag{3.49}$$

This measure has an intuitive motivation in that it calculates the common part of two histograms. Its advantage is that it explicitly neglects features that only occur in one of the histograms. The complexity is very low as it requires very simple operations [66].

Thus, the likelihood function for the texture feature can be defined as

$$p_t(y_t|x_t) \propto exp\left(-\frac{d_t^2[q_t,p_t(x_t)]}{2\sigma_t^2}\right) \tag{3.50}$$

where the standard deviation σ_t represents the Gaussian noise in the measurements. The values of σ_t, σ_c, and σ_e will be updated on-line. The update method was proposed by Brasnett [53].

3.3.3. Multiple feature fusion

In the tracking process, there are complex conditions, such as illumination changes, partial occlusion, and similar backgrounds. Only using a single feature is inadequate for meeting the tracking needs, although the computational complexity of the weighted algorithm is low. Multiple feature fusion can better utilize information provided by the features supplementing each other and can improve the robustness of the algorithm. The relationship between different features has been treated differently by different authors. In this paper, we assume that the relationships among color, edges, and texture are independent. A similar assumption was described in literatures [52][53][67].

With this assumption, the overall likelihood function of the target object is represented as a product of the likelihoods of the separate features.

$$p(y_t|x_t) = \prod_f p_f(y_t|x_t)^{w_f} \tag{3.51}$$

where w_f is the adaptive weight value of a single feature $\sum_f w_f = 1,\ 0 < w_f < 1$, $f \in \{c, e, t\}$, where c, e and t represent color, edge, and texture, respectively.

In multiple feature fusion algorithms, there are various features fusion, and fixed weights are commonly adopted. However, in the tracking process, if the tracking feature changes but the corresponding weight value remains unchanged, this will inevitably lead to poor performance or tracking failure. Thus, in successful tracking, different feature weights should instantaneously change in response to the changing tracking conditions. In other words, the weight value of the corresponding feature will be calculated using the similarity between the object model and the tracking result. As the similarity becomes higher, the weight value becomes larger. However, for the multiple selected features, if a particular feature is unable to effectively distinguish the object and the background, even if that feature gives a higher model similarity, due to the interference of the background,

the tracking performance will also be affected by this feature. Therefore, for a particular feature, in addition to considering the similarity between the target model and the candidate, we also consider the discriminability between the target object and the adjacent background.

Based on this, we propose a method that dynamically balances the effects of the similarity and the discriminability, and calculates adaptive weights of the three features. For convenience of description, we introduce a feature called the similarity coefficient (SC) to represent the similarity between the object model and the candidate, and a feature called the discriminant coefficient (DC) to describe the discriminability between the object model and the adjacent background. In the tracking process, these two coefficients are used to dynamically calculate the weight value of the selected feature, in order to achieve dynamic multiple feature fusion tracking according to Equation (3.51).

As described above, we use the Bhattacharyya coefficient to represent the similarity of the distributions of color, edge, and texture:

$$SC_f[q_f, p_f] = \rho[q_f, p_f] = \sum_{u=1}^{m} \sqrt{q_f^{(u)} \cdot p_f^{(u)}} \tag{3.52}$$

where q_f is the feature distribution of the target model, and p_f is the feature distribution of the candidate model, where $f \in \{c, e, t\}$.

We use the variance of the log-likelihood ratio to represent the discriminability between the object model and the adjacent background. The likelihood ratio produces a function that maps feature values associated with the target to positive values and those associated with the background to negative values [68][69]. The frequency of the pixels that appear in a histogram bin is defined as $\xi_f^{(u)} = q_f^{(u)}/n_{fg}$ and $\xi_{f(b)}^{(u)} = p_{f(b)}^{(u)}/n_{fg(b)}$, where q_f and $p_{f(b)}$ are histograms, n_{fg} is the number of pixels in the target object, and $n_{fg(b)}$ is the number of pixels in the adjacent background.

The log-likelihood ratio of a feature value is given by

$$L_f^{(u)} = \max(-1, \min(1, log \frac{\max\left(\xi_f^{(u)}, \delta_L\right)}{\max\left(\xi_{f(b)}^{(u)}, \delta_L\right)})) \tag{3.53}$$

where δ_L is a very small number (δ_L is set to 0.00001 in this work), and we have

$$DC_f[L; q_f, p_{f(b)}] = var(L; q_f, p_{f(b)})$$

$$= E\left[\left(L_f^{(u)}\right)^2\right] - (E[L_f^{(u)}])^2. \tag{3.54}$$

In our experiment, the adjacent background is the outer ring area of the target, the area width is $h_{(bg)} = 0.75 \cdot max(h_x, h_y)$, where h_x, h_y denote the height and width of the half axes of the target region, respectively.

We present a multiplication rule to balance the effect of similarity and discriminability. Then, the optimization weight of the particle is defined as

$$W_f = C \cdot SC_f[q_f, p_f] \cdot DC_f[L, q_f, p_f], \tag{3.55}$$

where C is a normalization factor that ensures that $\sum_f w_f = 1$ and it is given by

$$C = \frac{1}{\sum_{f=\{c,e,t\}} SC_f[q_f,p_f] \cdot DC_f[L,q_f,p_f]}. \tag{3.56}$$

3.4. Experiment results

In order to evaluate the performance of the proposed algorithm, we test the proposed algorithm on several challenging video sequences and compare the tracking performance with some related state-of-the-art approaches.

Test conditions

The results presented in this section were obtained on a dataset of targets extracted from different test sequences (see Figure 3.1). A first vehicle (V1) was extracted from the PETS 2001 dataset [38], and another vehicle target (V2) was extracted from a traffic surveillance video. The challenging factors include clutter, translation, rotation, and scale changes.

Figure 3.1 Targets of the evaluation data sets.

The parameters of the tracker were set experimentally and were different for different datasets. Two hundred particles were used per frame. The results presented were obtained from a MATLAB implementation running on a computer with a Core 3.40GHz CPU. In the case of no optimization code, the proposed algorithm can handle about 5 frames per second.

Performance evaluation

The performance evaluation was based on a metric using the true positive pixels $\mathrm{T_p}$ in each frame. The number of true positives is the number of pixels belonging both to the ground truth and the estimated target, as well as to the tracker output. The metric P is defined as

$$\mathrm{P} = 1 - \frac{2\mathrm{T_p}}{|\mathrm{A_e}|+|\mathrm{A_g}|}, \tag{3.57}$$

where $|A_e|$ and $|A_g|$ are the ground truth and the estimated target area, respectively. This performance measure rewards candidates with a high percentage of true positive pixels, and with few false positives and false negatives, avoiding the asymmetry problem seen with other area-based measures [72]. In our experiment, we obtained evaluation curves of the tracking results of 80 frames in the four sequences described above. We also used the average tracking error as a second criterion to evaluate the tracking performance. The average tracking error was manually calculated from the distance between the position of the tracking result and that of the ground truth.

Tracker comparison

For comparison, we implemented some single-feature trackers, including color, edge, and texture trackers, a fixed weight (FW) multiple feature tracker using a particle filter framework, and an adaptive multiple feature Mean shift tracker [68] (referred to as AMF-MS).

We conducted four experiments to demonstrate the validity of our proposed tracker. In the first experiment using the V1 sequence, the tracked target was a fast-moving vehicle undergoing a large scale change in outdoor cluttered environment. The tracking results are illustrated in Figure 3.2, in which four representative frames (310, 345, 360, and 382) are shown, where rows 1, 2, 3 and 4 correspond to Texture, FW, AMF-MS, and

our tracker, respectively. We can see that our proposed tracker performed well from beg-

Figure 3.2 Tracking results of V1 sequence.

inning to end. The evaluation results of these methods are shown in Figure 3.6 (d).

The second sequence, V2, was extracted from a traffic surveillance video. We used it to test the performance of our tracker in handling significant scale changes in an outdoor cluttered environment. Some tracking results are shown in Figure 3.3, in which four representative frames (271, 283, 291, and 304) are shown, where rows 1, 2, 3 and 4 correspond to Color, AMF-MS, FW, and our tracker, respectively. The evaluation results clearly reflect the performance of the different methods in Figure 3.6(b). In particular, the AMF-MS method could not provide accurate state information for the vehicle under these conditions. From the evaluation curves, it is clear that our tracker was capable of tracking the target object successfully in the case of large scale changes.

From these experimental results, the average tracking errors of the six trackers are listed in Table 3.1. We can see that our tracker had lower tracking errors in all of the test

sequences. The AMF-MS method had a much larger tracking error in the sequence with

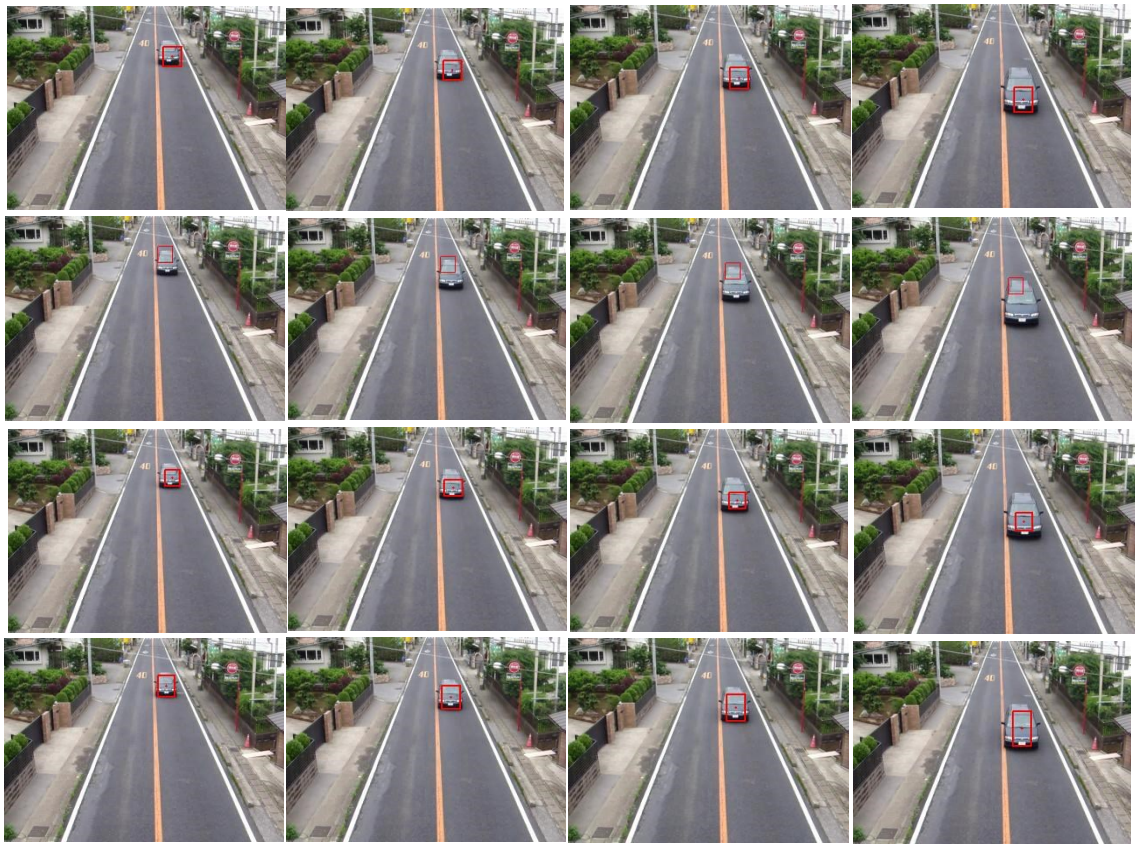


Figure 3.3 Tracking results of V2 sequence.

a large scale change. The single edge tracker had the highest average tracking error in all test sequences.

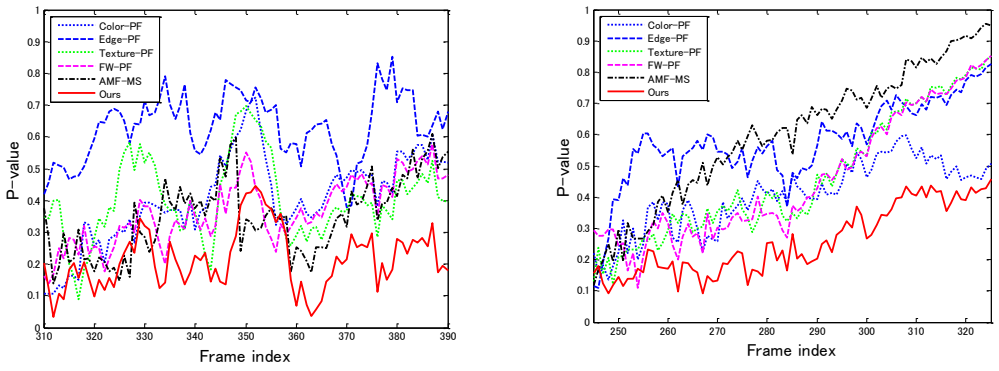


Figure 3.4 Evaluation results. (a) V1 (from frame 310 to frame 390); (b) V2 (from frame 245 to frame 325).

Table 3.1 The average tracking errors (in pixels) of the compared methods

	Color-PF	Edge-PF	Texture-PF	FW-PF	AMF-MS	Ours
V1	8.9	11.3	9.2	6.3	15.1	5.0
V2	6.2	16.3	7.9	4.8	8.4	4.4

3.5. Summary

In this chapter, firstly, a review of particle filter techniques is introduced. Secondly, we propose an adaptive multiple feature fusion mechanism using a particle filter to track moving target. The proposed mechanism not only improves the represent-ability of tracking targets, but also dynamically balances the effects of feature similarity and feature discriminability among target object, candidate and adjacent background. It can calculate dynamically the weight value of individual feature and improve the performance of tracking. Moreover, comparing with other state-and-art algorithms using four different datasets, the experimental results clearly demonstrate the effectiveness of our proposed method.

Chapter 4. Single target tracking with SURF and PF

4.1. Introduction

In the previous chapter, we introduced a multiple feature fusion method based on color, edge and texture feature for visual target tracking. These low level visual features meet the needs of an ordinary scene. However, for the complex conditions, such as large scale change, fast moving, rotation, we should adopt other more complex visual features to represent the tracking target. Nowadays, the Speeded-Up Robust Features (SURF) is a promising approach for extracting scale and rotation invariant feature points, especially, outperforms previously proposed methods, and computes much faster.

In this chapter, SURF method is firstly surveyed, including interest point detection and description. Then we propose a robust vehicle tracking method based on SURF feature. Combining the color and LBP feature improves the representation ability of tracking target. The proposed method mainly includes the dynamic update mechanism of target template, tracking window modification and particle weight allocation. Lastly, the experimental result and analysis are presented.

4.2. SURF

SURF method was proposed by Herbert Bay et al. in literatures [73][74]. SURF approximates or even outperforms previously proposed schemes with respect to repeatability, distinctiveness, and robustness, yet can be computed and compared much faster. For a traffic image, SURF algorithm has excellent matching characteristics and robustness in the target mutual occlusion, scale changes and noise condition, so it can be used to track the moving vehicle. SURF is achieved by relying on integral images for image convolutions; by building on the strengths of the leading existing detectors and descriptors; by simplifying these methods to the essential. SURF mainly includes the following three steps: Interest point detection, interest point description, and matching.

4.2.1. Interest point detection

Under interest points, we understand small image regions with high changes of the local gradient in two distinctive directions. Such points can be reliably extracted and provide a high amount of information. Furthermore, interest points are often robust to various transformations, e.g. rotation, scale and partially affine transformations.

SURF approach for interest point detection uses a very basic Hessian matrix approximation. This lends itself to the use of integral images which reduces the computation time drastically.

Integral images

The entry of an integral image $I_\Sigma(\mathrm{X})$ at a location $\mathrm{X} = (x, y)^T$ represents the sum of all pixels in the input image I within a rectangular region formed by the origin and X.

$$I_\Sigma(\mathrm{X}) = \sum_{i=0}^{i\le x}\sum_{j=0}^{j\le y} I(i,j)\,. \tag{4.1}$$

Once the integral image has been computed, it takes four additions to calculate the sum of the intensities over any upright, rectangular area (see Figure 4.1). Hence, the calculation time is independent of its size. For using big filter sizes, this is important.

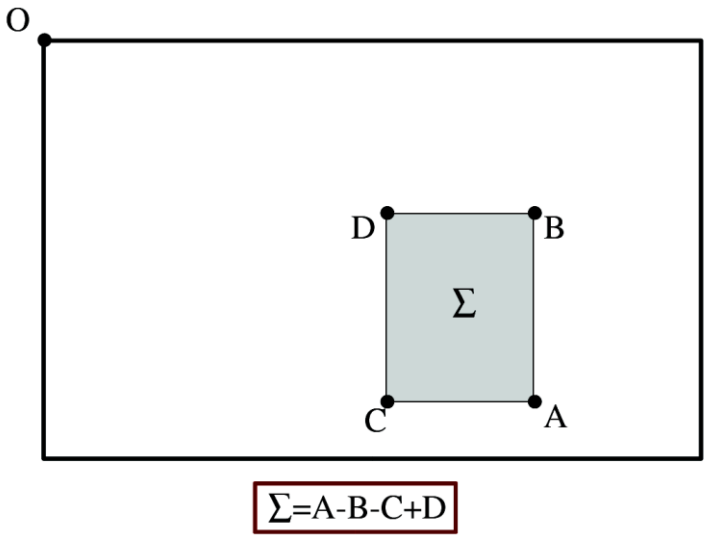


Figure 4.1 Integral image to calculate the area of a rectangular region of any size using four operations.

Hessian matrix based interest points

Because of the good performance of the Hessian matrix in computation speed and accuracy, it detects blob-like structures at locations where its determinant is maximum. SURF relies on the determinant of the Hessian also for the scale selection.

Given a point $\mathrm{X} = (x, y)$ in an image I, the Hessian matrix $\mathcal{H}(\mathrm{X}, \sigma)$ in X at scale σ id defined as follows

$$\mathcal{H}(\mathrm{X}, \sigma) = \begin{bmatrix} L_{xx}(\mathrm{X}, \sigma) & L_{xy}(\mathrm{X}, \sigma) \\ L_{xy}(\mathrm{X}, \sigma) & L_{yy}(\mathrm{X}, \sigma) \end{bmatrix} \tag{4.2}$$

where $L_{xx}(\mathrm{X}, \sigma)$ is the convolution of the Gaussian second order derivative $\frac{\partial^2}{\partial x^2} g(\sigma)$ with the image I in point X, and similarly for $L_{xy}(\mathrm{X}, \sigma)$ and $L_{yy}(\mathrm{X}, \sigma)$.

Gaussians are optimal for scale space analysis [75], but in practice they have to be discretized and cropped (Figure 4.2 left half, the Gaussian second order partial derivative in y-(L_{yy}) and xy-direction (L_{xy}), respectively). This leads to a loss in repeatability under image rotations around odd multiples of $\pi/4$. This weakness seems to hold for Hessian-based detectors in general. SURF method pushes the approximation for the Hessian matrix even further with box filter (in the right half of Figure 4.2, the approximation for the second order Gaussian partial derivative in y-(D_{yy}) and xy-direction (D_{xy}). The grey regions are equal to zero). These approximate second order Gaussian derivatives and can be evaluated at a very low computational cost using integral images. The performance is comparable or better than with the discretized and cropped Gaussians.

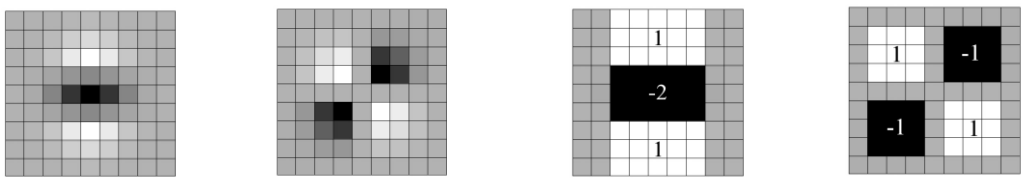


Figure 4.2 Gaussian second order partial derivative and approximation of the Hessian matrix with box filter.

The 9×9 box filters in Figure 4.2 are approximations of a Gaussian with $\sigma = 1.2$ and represent the lowest scale, which are denoted by D_{xx}, D_{yy}, and D_{xy}. The weights

applied to the rectangular regions are kept simple for computational efficiency. The relative weights of the filter responses are further balanced in the expression for the Hessian's determinant. This is required for the energy conservation between the Gaussian kernels and the approximated Gaussian kernels,

$$\frac{|L_{xy}(1.2)|_F |D_{yy}(9)|_F}{|L_{yy}(1.2)|_F |D_{xy}(9)|_F} = 0.912\cdots \cong 0.9 \tag{4.3}$$

where $|x|_F$ is the Frobenius norm. This yields

$$\det(\mathcal{H}_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2 \tag{4.4}$$

Furthermore, the filter responses are normalized with respect to their size. This guarantees a constant Frobenius norm for any filter size.

The approximated determinant of the Hessian represents the blob response in the image at location X. These responses are stored in a blob response map over different scales, and local maxima are detected.

Scale space representation

Interest points need to be found at different scales, not least because the search of correspondences often requires their comparison in images where they are seen at different scales. Scale spaces are usually implemented as an image pyramid. The images are repeatedly smoothed with a Gaussian and subsequently sub-sampled in order to achieve a higher level of the pyramid.

Due to the use of box filters and integral images, SURF method does not iteratively apply the same filter to the output of a previously filtered layer, but instead can apply such filters of any size at exactly the same speed directly on the original image and even in parallel. Therefore, the scale space is analyzed by up-scaling the filter size rather than iteratively reducing the image size. The output of the 9×9 filter is considered as the initial scale layer, and set the scale $s = 1.2$ (corresponding to Gaussian derivatives with $\sigma = 1.2$). The following layers are obtained by filtering the image with gradually bigger masks, taking into account the discrete nature of integral images and the specific structure of the filters. Specifically, this results in filters of size 9×9, 15×15, 21×21, 27×27, etc. At larger scales, the step between consecutive filter sizes should

also scale accordingly. The scale space is divided into octaves, for each new octave, the filter size increase is doubled (going from 6 to 12 to 24). Simultaneously, the sampling intervals for the extraction of the interest points can be doubled as well. In order to localize interest points in the image and over scales, a non-maximum suppression in a $3 \times 3 \times 3$ neighbourhood is applied. The maxima of the determinant of the Hessian matrix are then interpolated in scale and image space with the method proposed by Brown et al.[76].

4.2.2. Interest point description

SURF descriptor is based on features with a complexity stripped down even further. It includes the following three steps, such as fixing a reproducible orientation based on information from a circular region around the interest point, constructing a square region aligned to the selected orientation and searching for correspondences in a second image. These three steps are explained in this section.

Orientation assignment

In order to be invariant to image rotation, Bay identifies a reproducible orientation for the interest points. He first calculates the Haar wavelet responses in x and y direction within a circular neighbourhood of radius $6s$ around the interest point, with s the scale at which the interest point was detected. Also, the sampling step is scale dependent and chosen to be s. In keeping with the rest, also the wavelet responses are computed at the current scale s. Accordingly, at high scales the size of the wavelets is big. Therefore, for fast filtering, the again integral images are used. The used filters are shown in Figure 4.3. The dark parts have the weight-1 and the light part+1. Only six operations are needed to compute the response in x or y direction at any scale. The side length of the wavelets is $4s$.

Once the wavelet responses are calculated and weighted with a Gaussian ($\sigma = 2s$) centred at the interest point, the responses are represented as points in a space with the horizontal response strength along the abscissa and the vertical response strength along the ordinate. The dominant orientation is estimated by calculating the sum of all responses within a sliding orientation window of size $\pi/3$(see Figure 4.4). A sliding orientation window of size $\pi/3$ detects the dominant orientation of the Gaussian

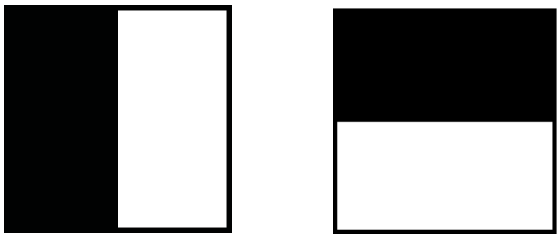


Figure 4.3 Haar wavelet filters to compute the responses in x(left) and y direction (right).

weighted Haar wavelet responses at every sample point within a circular neighborhood around the interest point. The horizontal responses within the window are summed, and also the vertical responses. The two, summed responses then yield a local orientation vector. The longest such vector lends its orientation to the interest point.

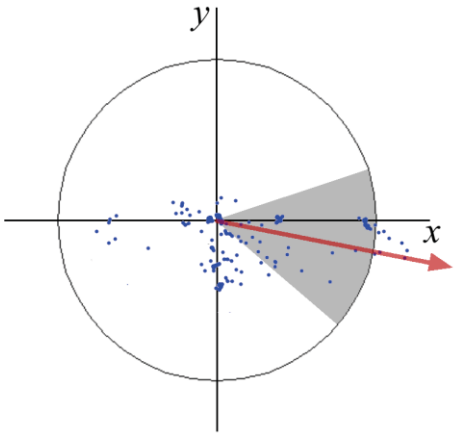


Figure 4.4 Orientation assignment.

Descriptor based on sum of Haar wavelet responses

For the extraction of the descriptor, the first step consists of constructing a square region centred around the interest point and oriented along the orientation. The region is split up regularly into smaller 4×4 square sub-regions. For each sub-region, we compute Haar wavelet responses at 5×5 regularly spaced sample points. d_x and d_y denote the Haar wavelet response in horizontal and vertical direction, respectively. “Horizontal” and “vertical” here is defined in relation to the selected interest point

orientation (see Figure 4.5). To build the descriptor, a quadratic grid with 4×4 square sub-regions is laid over the interest point (left). For each sample, the wavelet responses are computed. For this figure 2×2 vectors per sub-region for reasons of illustration, each sub-region (right), the sums of d_x, $|d_x|$, d_y, and $|d_y|$ relative to the orientation of the grid, are computed. The responses d_x and d_y are first weighted with a Gaussian ($\sigma = 3.3s$) centred at the interest point. Then, the wavelet responses d_x and d_y are summed up over each sub-region and form a first set of entries in the feature vector, and extracted the sum of absolute values of the responses, $|d_x|$ and $|d_y|$. Hence, each sub-region has a four-dimensional descriptor vector v for its underlying intensity structure $v = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$. This results in a descriptor vector for all 4×4 sub-region of length 64.

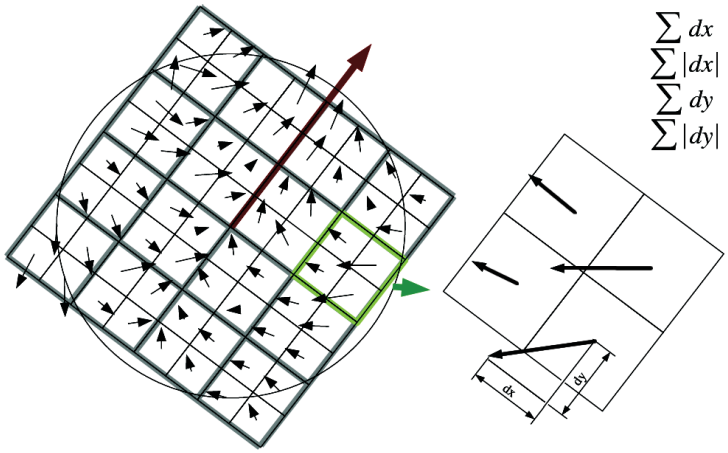


Figure 4.5 Calculate the orientation of the grid.

4.3. Target tracking based on SURF and PF

SURF algorithm, which is a novel scale and rotation invariant interest point detector and descriptor. It approximates or even outperforms previously proposed schemes with respect to repeatability, distinctiveness, and robustness, yet can be computed and compared much faster [73][74]. In interest point detection, it uses a very basic Hessian matrix approximation with box filters. These approximate second order Gaussian derivatives and can be evaluated at a very low computational cost using integral images. Also, for scale space representation, SURF algorithm applies box filters of any size at

exactly the same speed directly on the original image and even in parallel to achieve a higher level of the image pyramid. In interest point description, its descriptor is built on the distribution of the first order Haar wavelet responses in x and y direction rather than the gradient, exploit integral images for speed, and use only 64D. This reduces the time for feature computation, and has proven to simultaneously increase the robustness of the descriptor.

The SURF feature descriptor is represented as $S = \{x, y, s, o, hist\}$. Where x and y are the position of the feature in terms of the image coordinate, s is the feature scale, o is the feature vector direction, and hist is the gradient orientation distribution quantized into 64 or 128 bins. The SURF points of target vehicle are illustrated in Figure 4.6 with the size of the oriented descriptor window at different scales.

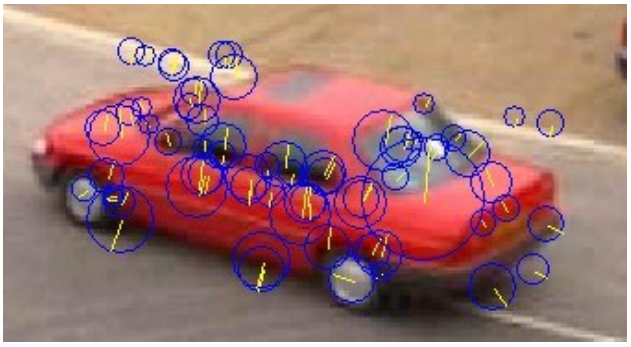


Figure 4.6 Detail of the vehicle showing the size of the oriented descriptor window at different scales.

4.3.1. PF framework and dynamic model

Particle filter is an efficient statistical simulation method based on the idea of recursive Bayesian estimation. It uses a set of weighted particles sampled randomly to approximate the posterior possibility density function. In this chapter, we adopt the PF framework to track moving target. The PF framework is described in section 3.2 particle filter techniques in Chapter 3. And we use the random walk model to represent the tracking target (referring to section 3.3.1 dynamic model in Chapter 3).

4.3.2. Color and texture feature

Color feature distribution

Color distributions are used as target models as they achieve robustness against non-rigidity, rotation and partial occlusion. In this experiment, the HSV space is selected to represent color information. Typical $8 \times 8 \times 4$ bins were used to make the histograms less sensitive to intensity variations. Suppose the distributions are discretized into m bins. The similar color distributions are also described by Comaniciu [60]. The histograms are produced with a function $b(x_i)$ that assigns the color of location x_i to the corresponding bin. We calculate the color distribution in a rectangle region with height and width half axis h_x, h_y. To increase the reliability of the color distribution when boundary pixels belong to the background or get occluded, smaller weights are assigned to the pixels that are further away from the region center by employing a weighting function.

$$k(\|r\|) = \begin{cases} 1 - \|r\|^2, & \|r\| < 1 \\ 0, & otherwise \end{cases} \tag{4.5}$$

where $k(\cdot)$ is the Epanechnikov kernel, r is the distance from the region center. Thus, the color distribution of target object $q_c = \left\{q_c^{(u)}\right\}_{u=1,\cdots,m}$ at location (x_i, y_i) is defined as

$$q_c^{(u)} = \mathrm{C} \sum_{i=1}^{n} k\left(\frac{\|x - x_i\|}{l}\right) \delta[b(x_i) - u] \tag{4.6}$$

where i is the number of pixels in the object region, δ is the Kronecker delta function, the parameter $l = \sqrt{h_x^2 + h_y^2}$ is used to adapt the size of the region, and the normalization factor C ensures that $\sum_{u=1}^{m} q_c^{(u)} = 1$.

In update step, we use similarity measures to calculate color distributions $p_c(x_t) = \left\{p_c^{(u)}(x_t)\right\}_{u=1,\cdots,m}$ of the candidate at time *t*. According to the color distributions of target object and candidate, the similarity measure between these two distributions is then given by the Bhattacharyya coefficient [61][62]. The calculation methods are described in detail in section 3.3.2 dynamic model in Chapter 3. The likelihood function for the color feature can refer to formula (3.37).

Texture feature distribution

In this chapter, we adopt the LBP texture operator, which has recently shown excellent performance in many applications. LBP is described in detail in section 3.3.2 texture distribution model in Chapter 3. Suppose the texture histogram of the image $I(x_i, y_i)$ can be defined as $q_L^{(u)}$, then, the texture distribution of target object is $q_L = \left\{q_L^{(u)}\right\}_{u=1,\cdots,m}$, the texture distribution of candidate is $p_L(x_t) = \left\{p_L^{(u)}(x_t)\right\}_{u=1,\cdots,m}$ at time *t*. The similarity measure between these two distributions and the likelihood function for the texture feature can be calculated with the same method. Thus, we calculate the each particle's similarity measure and normalize to gain the weight w_t^L. Suppose the relationships among color, texture and SURF feature are independent. So we can obtain the fusion weighting by:

$$w_t^F = \gamma w_t^C + (1-\gamma) w_t^L \tag{4.7}$$

where $\gamma \in [0,1]$ is an adaptive control parameter. And γ denotes the proportion of the number of larger weight color particle to the total number of color and texture particles. Note that the larger the value, the more similar color feature distributions.

4.3.3. SURF feature point matching

In the matching strategy, several matching methods are defined, such as fast indexing matching, nearest neighbor matching, and nearest neighbor distance ratio (NNDR) [77]. We use the NNDR approach to match the feature points. This approach is similar to the nearest neighbor matching, except that the thresholding is applied to the distance ratio between the first and the second nearest neighbor. For two regions A and B, the regions are matched if $\|D_A - D_B\| / \|D_A - D_C\| < t$, where D_B is the first and D_C is the second nearest neighbor to D_A, t is experimental threshold and is set to be 0.5 in our work. Because the precision of each SURF feature matching point may affect the tracking in our experiment, especially when the mismatched feature points are far away from the object center. So, before matching, we firstly discard the outliers of feature points using the random sample consensus (RANSAC) algorithm [78][79](see Figure 4.7).

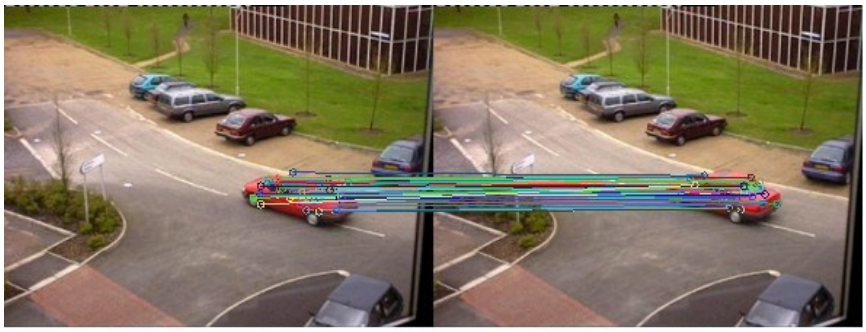


Figure 4.7 Experimental result of vehicle using RANSAC algorithm.

4.3.4. SURF feature point update mechanism

In the tracking stage, due to the deformation of moving target, we must update the feature points of the target template over time. Therefore we propose the adaptive on-line update mechanism. It includes two stages, discarding bad feature points and adopting new feature points.

Discarding bad feature points

Our hypothesis is that the repeated occurrences of a feature point within the target template in the past period of time is rare, then the probability of re-emergence of this feature point in the next period of time is almost zero. Based on this assumption, we examine the repeated occurrences of every point in target template for a while, to measure whether we want to discard or adopt this point. This similar method was introduced in references [80][81]. The mechanism is defined as

$$P_{i,t+1} = (1-\theta)\cdot P_{i,t} + \theta \cdot \delta_i \tag{4.8}$$

where $P_{i,t+1}$ is the probability of ith feature point in target template at time $t+1$, $\theta \in [0,1]$ indicates the degree of object change. The larger θ is, the faster the change of object will be, and $\delta_i \in \{0,1\}$ is a factor of matching determination. If the current ith feature point in target template match any one of candidate feature points, $\delta_i = 1$, else $\delta_i = 0$. If the probability $P_{i,t+1}$ is greater than an experimental threshold $th2$, show that the feature point is a stable feature point. In other words, the frequency of its occurrence in consecutive frames is large. Otherwise, we will discard this point from the template. In our work, $th2$ is set to be 0.0001. As shown in Figure 4.8, at time t_4, we will discard the upper left point of the target template according to this mechanism. The advantage of this mechanism is that it can quickly adapt to the appearance changes of

the tracking object, and instantly deletes the inappropriate feature points.

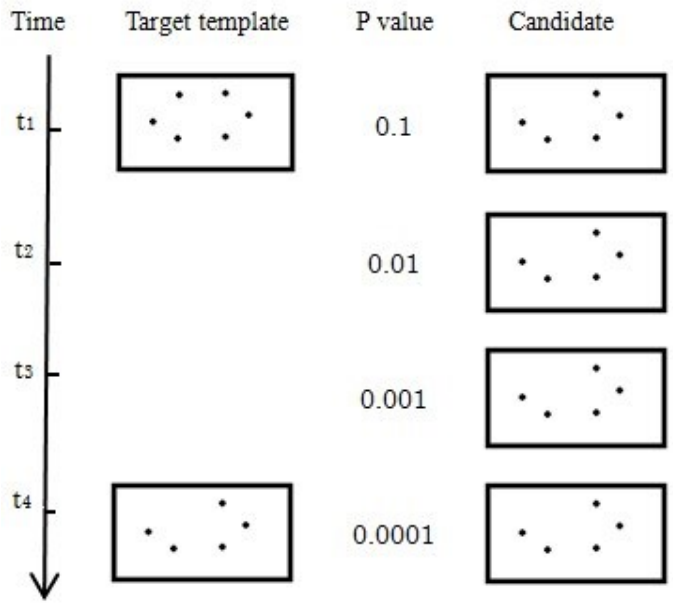


Figure 4.8 Discarding the upper left point of the target.

Adopting new feature points

In order to adapt to the quick appearance changes of target, the tracker should not only delete inappropriate old features points, but also promptly adopt new feature points reflecting the current situation of target. In the existing algorithms presented in references [80][81], with the advent of new frame, the feature points randomly selected from the new frame are added to the target template. The random selection method brings uncertainty to the object template and is not conducive to the stability of the tracking. Therefore, we present our hypothesis. The adopted new feature points should be selected from a stable non-matching set of the candidate objects with the target template for a while.

In our experiment, we need to define the object template. Given a feature point of object template set $S_{t_0} = \{s_1, s_2, \cdots, s_n\}$, as mentioned above, the weight corresponding to each feature point is $P_{i,t}$ at time$t$. Now assuming the target object has been successfully tracked in frame I_{t-1}. When frame I_t arrives, we detect the candidate feature point set $C_t = \{c_1, c_2, \cdots, c_q\}$ using the SURF detector and descriptor and get the matching candidate set $M_t = \{m_1, m_2, \cdots, m_p\}$ and un-matching set $U_t = \{u_1, u_2, \cdots, u_l\}$ with the template set S_t by the NNDR method，where $M_t \cup U_t = C_t$, $M_t \cap U_t = \emptyset$, $p + l = q$. $\mathrm{s_i}, \mathrm{c_i}, \mathrm{m_i}$ and $\mathrm{u_i}$ denote the state of the i-th feature point

from the corresponding feature points sets S_t, C_t, M_t and U_t at time t respectively. Next, when new frame I_{t+1} arrives, we can get the new candidate feature point set C_{t+1} according to the prediction step of particle filter, and, after matching the object template S_{t+1}, also get the matching candidate set M_{t+1} and un-matching set U_{t+1}. For the two consecutive un-matching sets U_t and U_{t+1}, in order to detect the stability of each point of these two sets, we define a new matching set UM_{t+1} which contains the matching points of the two consecutive un-matching sets U_t and U_{t+1}. If the two sets do not match, we use the set U_{t+1} to represent the set UM_{t+1}, and compute the each point probability by

$$UP_{i,t+1} = \begin{cases} UP_{i,t} + 1, & \text{if matching is true} \\ 1, & otherwise \end{cases} \tag{4.9}$$

where $UP_{i,t}$ represents the weight of the point in matching set UM_t which match the ith point in the un-matching set U_{t+1}. If the probability $UP_{i,t+1}$ exceeds a threshold $th3$, it represents that this point is a stable non-matching point. Thus we will adopt this point to the object template. In this experiment, $th3 = 4$, if the feature point exists in more than three consecutive frames, we will adopt this point. As shown in Figure 4.9, we will adopt the two stable center points from the non-matching set UM at time t_4 according to this mechanism. This mechanism of adopting new feature point overcomes the shortcomings of randomly selected new points, and experiments show that it has a strong adaptability for tracking condition changes.

4.3.5. Tracking window modification

The tracking window is described as a rectangle region with height and width half axis $h = (h_x, h_y)$ to cover the tracked object. In the practical tracking, the object may change its size with motion of object itself, and distance between the object and camera. To keep precise tracking, the size of tracking window should be modified automatically with the state of the object. Considering the size of color region, texture region and SURF matches distribution region in current frame, and the tracking window size in the former frame, we calculate the size of tracking windows in current frame at time t by

$$h_t = \lambda h_{t-1} + (1-\lambda)\big((\alpha+\beta)h_{SURF} + (1-\alpha)h_{color} + (1-\beta)h_{texture}\big) \tag{4. 10}$$

where h_{SURF} represents the window covering the region of SURF matches distribution, h_{color} and $h_{texture}$ denote the window region that cover the color and texture region of object, respectively. α, β are the harmonizing factor that balance the effect of

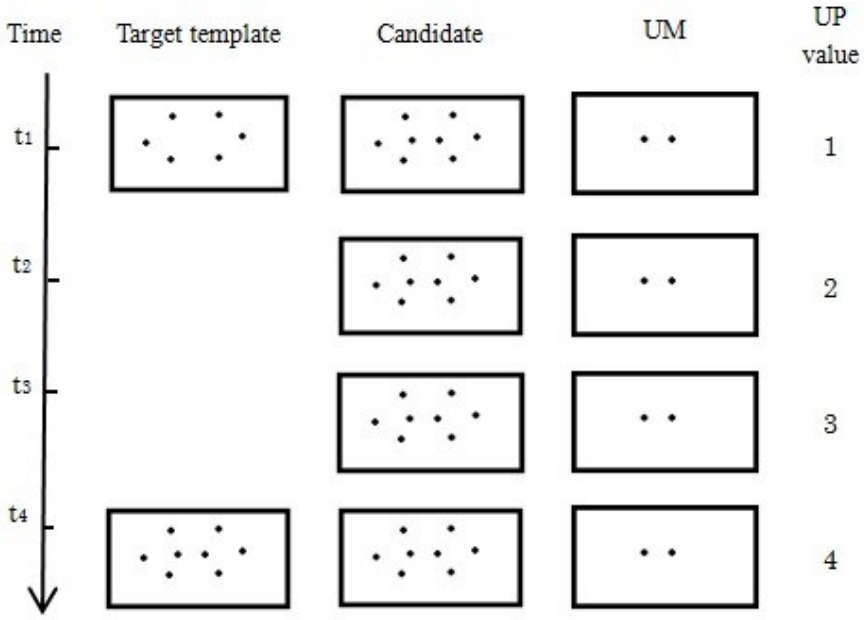


Figure 4.9 Adopting the two stable center points from the non-matching set UM at time t_4.

SURF cue, color cue and texture cue, λ is a forgetting factor that balances the window in frames at time t and time $t-1$ to avoid over-sensitive scale adaption. The similar approach is proposed in references [54][82].

4.3.6. Particle weight updating

In the tracking process, there are complex conditions such as illumination change, partial occlusion, and similar background. Single feature is unable to meet tracking needs, although the computational complexity of weighted algorithm is low. So, we combine the color feature, texture feature and the SURF feature distributions, and present an improved distance kernel function to update the weights of particles to achieve adaptive tracking.

Suppose the state of particles at time t is defined as $X_t = \left\{x_i, \dot{x}_i, y_i, \dot{y}_i, h_{x_i}, h_{y_i}\right\}$ $i = 1, \cdots, N$, where N represents the total number of particles, x and y represent the center coordinates of the rectangular box for tracking an object, $\dot{x}$ and $\dot{y}$ represent the respective velocity components, h_x and h_y denote the height and width half axis of rectangle region, respectively. $w_t = \left\{w_i^{(t)}\right\}$ represents the weight in which w_t is computed by w_t^F. The SURF feature matching candidate set is $M_t = \{m_1, m_2, \cdots, m_p\}$.

Thus, the weight of each particle is allocated as follows:

Step 1. Approximate the object center C(x, y) by

$$x = \varepsilon\left(\sum_{i=1}^{N} x_i w_i^F\right) + (1-\varepsilon)\left(\frac{1}{P}\sum_{i=1}^{P} x_i^S\right) \tag{4.11}$$

$$y = \varepsilon\left(\sum_{i=1}^{N} y_i w_i^F\right) + (1-\varepsilon)\left(\frac{1}{P}\sum_{i=1}^{P} y_i^S\right) \tag{4.12}$$

Step 2. For $i = 1, \cdots, N$

1). Calculate the Euclidean distance d_i from each particle to center point C.

2). Get the factors using exponential kernel function.

$$k_S(d_i) = \begin{cases} 1 & , \ \mathrm{d}_i \le \mathrm{T}_d \\ \exp\left((-1)\left(d_i - \frac{T_d}{max\{d_i - T_d\}_{i=1}^{N}}\right)\right), & \mathrm{d}_i > \mathrm{T}_d \end{cases} \tag{4.13}$$

3). Update the weight w_i using

$$w_i^{(t)} = k_S(d_i) \cdot w_i^{(t)} \tag{4.14}$$

Step 3. Normalize the weights

$$\widetilde{w}_i^{(t)} = w_i^{(t)} / \sum_{i=1}^{N} w_i^{(t)} \tag{4.15}$$

where, x_i^S and y_i^S are the coordinates of SURF feature points, ε is a weight of fusion three feature and is set to be 0.4. T_d is set as the half of the minimum of height and width of object rectangular region, P is the number of the SURF feature points.

Algorithm flow

To facilitate understanding, we give the description and flow chart of our method as shown in Figure 4.10. The whole process of our proposed tracking algorithm is described as follows:

Input: Video frames $F_1, F_2, \cdots, F_n$.

Output: Target regions according to target states in every frame.

Algorithm:

(1) Select the target object in the initializing frame manually, and calculate the appearance model for the first frame.

(2) Select and propagate samples at frame $\mathrm{F_t}$ from the samples at $\mathrm{F_{t-1}}$ according to their weights.

(3) Predict the state of particles according to the dynamic model.

(4) Compute the weight of each feature using the formula (4.6) and formula (3.47) according to the likelihood using formula (3.37) and formula (3.50).

(5) Extract SURF points, and compute the number of matching points.

If the number of matching points is larger than threshold, approximate the object center using formula (4.11) and (4.12), and update the weights of particles using the improved distance kernel function, else, only fuse the color and texture feature using formula (4.7).

(6) Update SURF feature points of object target using the formula (4.8) and formula (4.9).

(7) Modify size of tracking window using the formula (4.10).

(8) Output target's location, and go to step 2.

4.4. Experiment results

Test Conditions

In order to verify the effectiveness of the proposed algorithms, we demonstrate the proposed tracker on a data set from three different tracking sequences (Table 4.1). The vehicles (V1 and V2) are extracted from the PETS2001 [38] and PETS2000 [83] data set, respectively. Finally, the vehicle (V3) is extracted from a traffic surveillance video. Because of the different characteristics of target motion, such as significant deformation and target rotation under curvilinear motion, these characteristics bring larger challenges to track tasks.

The parameters of the tracker are set experimentally, the forgetting factor λ and θ are set to be 0.9. α, β and γ are set to be 0.5. Note that the SURF based weight updating could be applied only when there are many enough SURF matching points, we set the least threshold as 5. The 200 particles are used for per frame. The results presented were obtained from MATLAB implementation with a Core 3.40GHz PC under Windows.

Performance evaluation and comparison

In order to highlight the difference of tracking results with different features and different tracking algorithms, we implement some state-of-the-art trackers including the

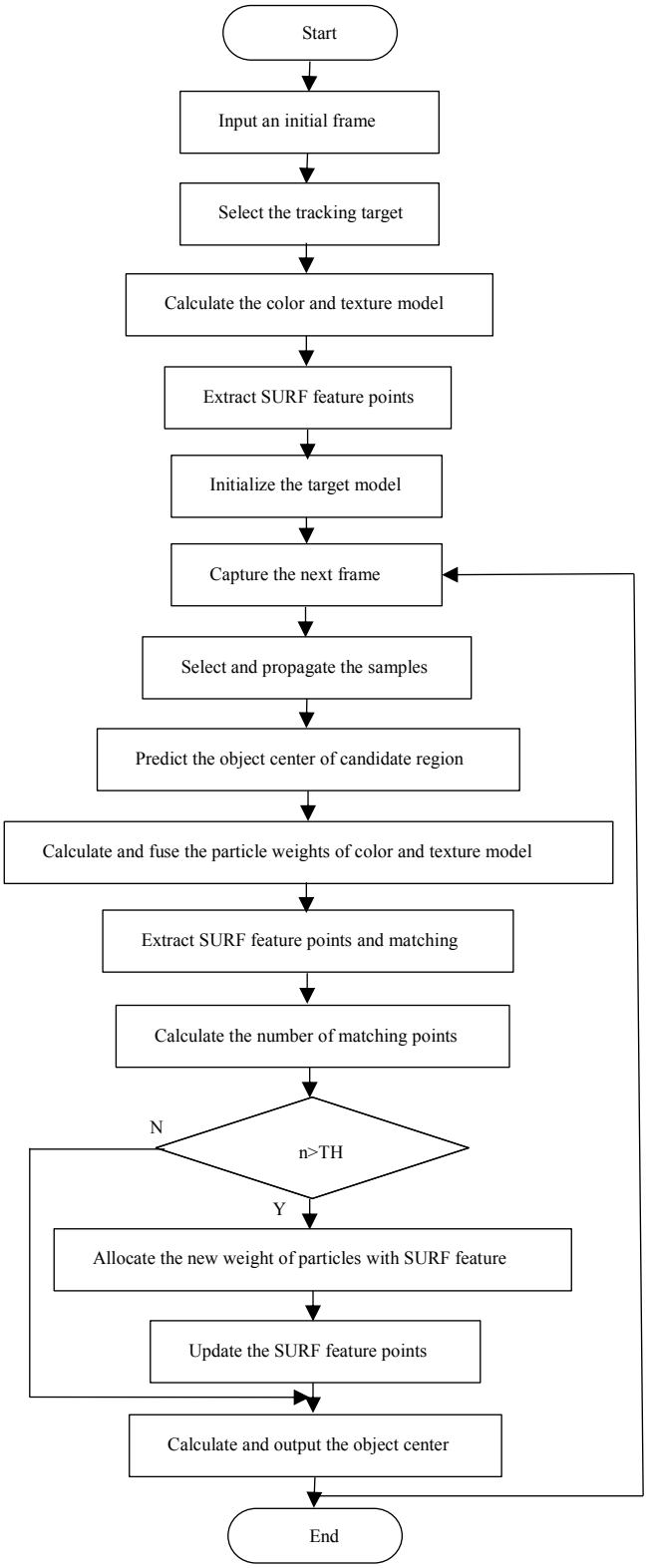


Figure 4.10 The flow chart of our proposed method.

Table 4.1 The description of the tracking data set.

Targets	Frame size	Frame rate	Target trajectory	Characteristics
V1	768*576	25fps	slash	Scale changes, clutter, fast motion
V2	768*576	25fps	curve	Scale changes, rotation
V3	800*600	30fps	Straight line	Scale changes, illumination changes

combining color and SIFT feature particle filter-based tracker [54](referred to as SIFT-PF), the standard Mean shift tracker[41] and single color feature (referred to as Color-PF) tracker. SIFT-PF is a multiple feature tracker, but, which only uses the color histogram for object representation, uses SIFT points to approximate the object center. The Mean shift tracker based on histogram analysis is a classical tracking algorithm. It is a mode-finding technique that locates the local minimum of the posterior density function. Note that, in addition to the Mean shift method, the other trackers (including ours) utilize the same particle filtering framework for tracking and use the same dynamic model in the experiments. The implementations are all parameterized according to the original paper with some tuning. Therefore, the comparison is valid because only the observation model changes between our tracker and the other two trackers. And everything else keeps consistent. We present some representative frames to show the tracking results and also give some statistical analysis to quantify the performance of the proposed tracker.

The first evaluation criteria of the tracking error are based on the relative position errors (in pixel) between the center of the tracking result and that of the ground truth. Ideally, the position differences should be around zero.

The first sequence undergoes a significant scale changes, fast motion and clutter environment. Some samples of the final tracking results are demonstrated in Figure 4.11, where rows 1, 2, 3 and 4 correspond to mean shift, Color-PF, SIFT-PF and our tracker, respectively, in which three representative frames (313, 352, 375) are shown. It shows a blue color vehicle moving from the left top corner of the image to the right bottom. Note that the vehicle is small in the image and undergoes scale changes and fast motion.

Figure 4.11 Tracking results of sequence V1 as it undergoes scale changes and clutter. Rows 1, 2, 3 and 4 correspond to Mean shift, Color-PF, SIFT-PF, and our tracker, respectively.

The number of SURF feature point is smaller, thus the performances of our proposed tracker are slightly better than the other three trackers. Since the background undergoes clutter environment, the position error of these three methods is larger between 340*th* and 360*th* frame, but all three methods can locate the vehicle position. The quantitative comparison of the tracker in terms of position errors is shown in Figure 4.12.

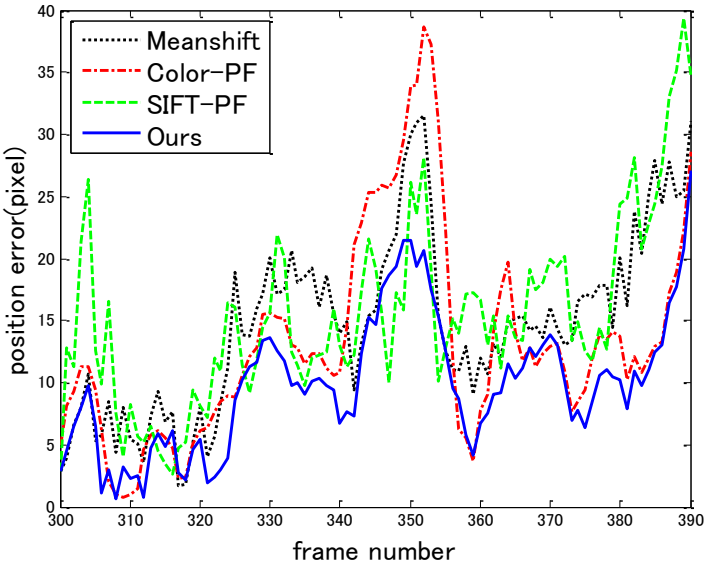


Figure 4.12 Evaluation results of V1.

For second sequence V2, the tracking results are illustrated in Figure 4.13, where rows 1, 2, 3 and 4 correspond to mean shift, Color-PF, SIFT-PF and our tracker, respectively, in which three representative frames (140, 175, 189) are shown. The characteristics of target motion are significant deformation and target rotation under curvilinear motion. The rotation invariant detectors and descriptors of SURF algorithm offer a good compromise between feature complexity and robustness to commonly occurring photometric deformations. The tracking results show the mean shift tracker (rows 1) and PF-color tracker (rows 2) have large position errors in this challenging situation. And the PF-color tracker even occur tracking failure. While SIFT-PF drifts gradually, because it only approximates the center point of tracking object using the SIFT feature points. The evaluation results of these four algorithms are shown in Figure 4.14.

The last sequence V3 is a traffic surveillance video which undergoes scale changes and illumination changes. Some samples of the tracking results are shown in Figure 4.15, where rows 1, 2, 3 and 4 correspond to Mean shift, Color-PF, SIFT-PF and our tracker, respectively, in which three representative frames (50, 85, 103) are shown. The significant deformation of the moving vehicle leads to poor performance of the mean shift tracker (rows 1). Moreover, the PF-color tracker cannot provide accurate state information for the moving vehicle. Note that the vehicle is becoming larger, more and more the SURF feature points are captured, and the performances of our proposed

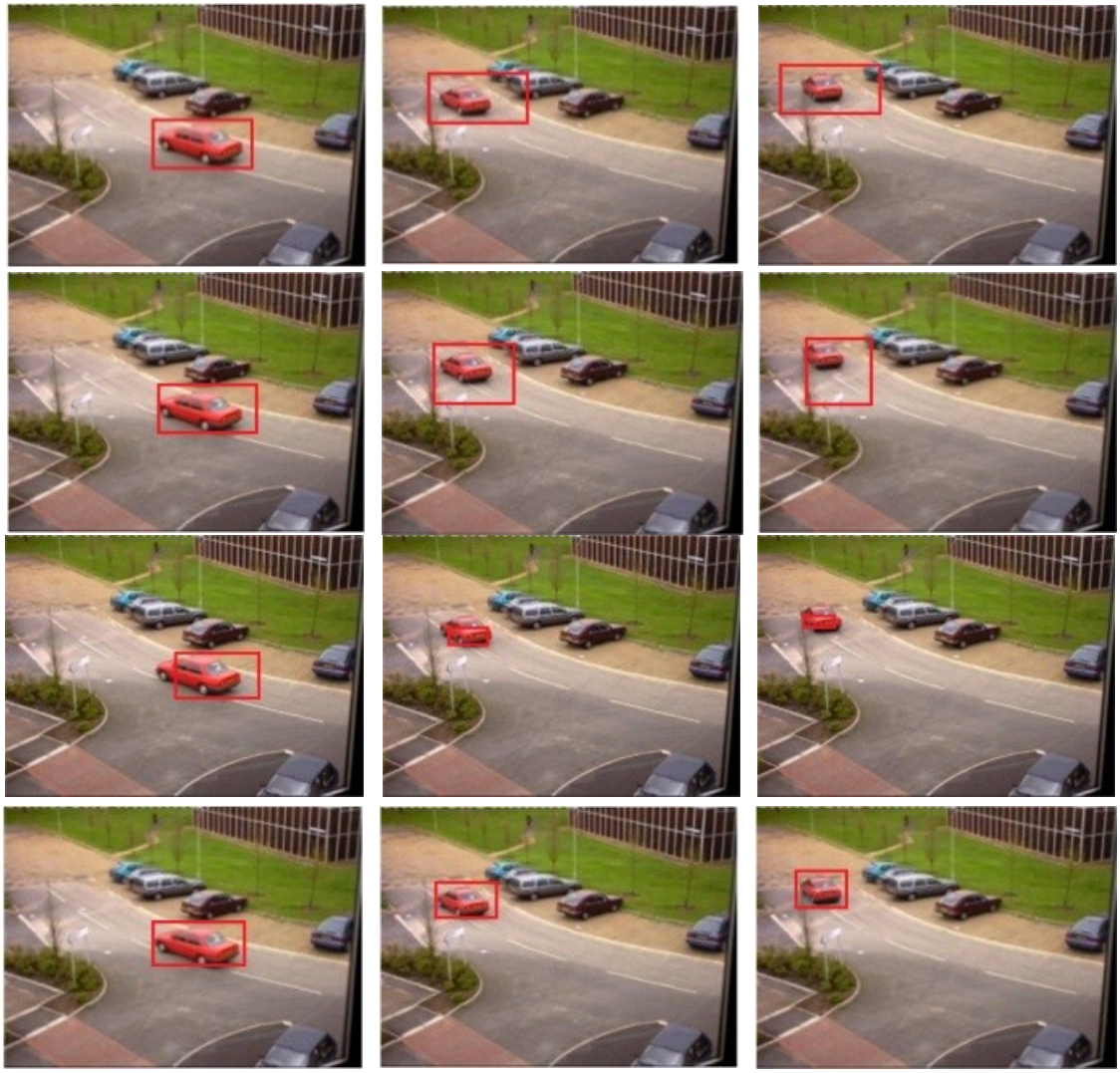


Figure 4.13 Tracking results of sequence V2 as it undergoes scale changes and rotation. Rows 1, 2, 3 and 4 correspond to Mean shift, Color-PF, SIFT-PF, and our tracker, respectively.

method are significantly improved. SIFT-PF also succeeds in tracking the target but has a larger tracking error compared to our method, and we can see obviously that SIFT-PF changes the position of target drastically because of the approximate method of the center point of object. SIFT-PF only uses the SIFT feature points. On the contrary, when the vehicle starts pulling away, it becomes smaller and smaller since the number of feature point is small. The performances of our method are similar to the classical particle filter. Figure 4.16 shows the evaluation results of comparison.

The second evaluation method is this performance measure rewards candidates with a

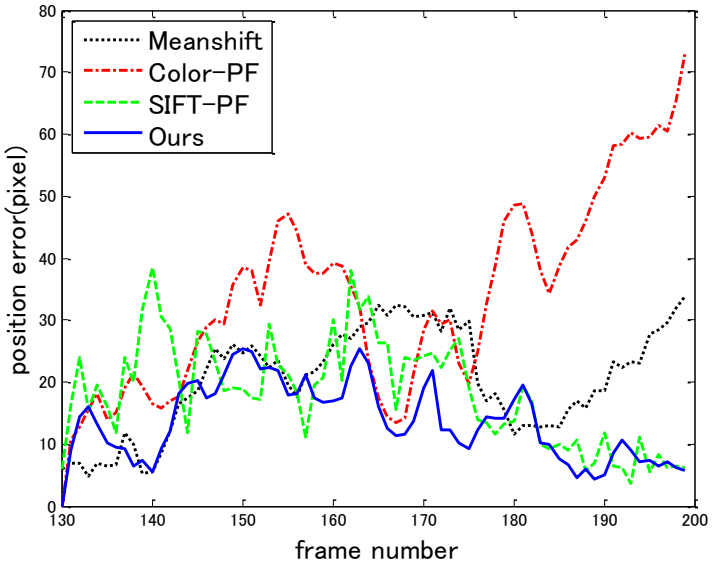


Figure 4.14 Evaluation results of V2.

high percentage of true positive pixels, and with few false positives and false negatives avoiding the asymmetry problem of other area based measures, which was described in section 3.4 experiment results in the Chapter 3.

In our experiment, the quality measure of a whole track is obtained by averaging P value over the frames where the target is visible. Since particle filter is a probabilistic algorithm, each tracker is run 50 times for each sequence. The comparison results of these four algorithms are shown in Table 4.2. We can see obviously that the target estimation of our tracker is more precise than other trackers.

For the efficiency of algorithm, the computational complexity of proposed method is mainly focused on the extracting and matching of local feature points. Under the non-optimal condition, comparing the efficiency of the proposed method and SIFT-PF, our approach of applying the incremental SURF detection takes 128.5 ms on average, the latter takes 168.3 ms. Obviously, our algorithm is superior to contrast method about the computational cost, but, the improving of efficiency will be still considered in our future work.

Table 4.2 The evaluation P-value of the tracking data set.

Target	Mean shift	Color-PF	SIFT-PF	Ours
V1	0.378	0.433	0.556	0.352
V2	0.404	0.426	0.351	0.193
V3	0.565	0.280	0.503	0.194

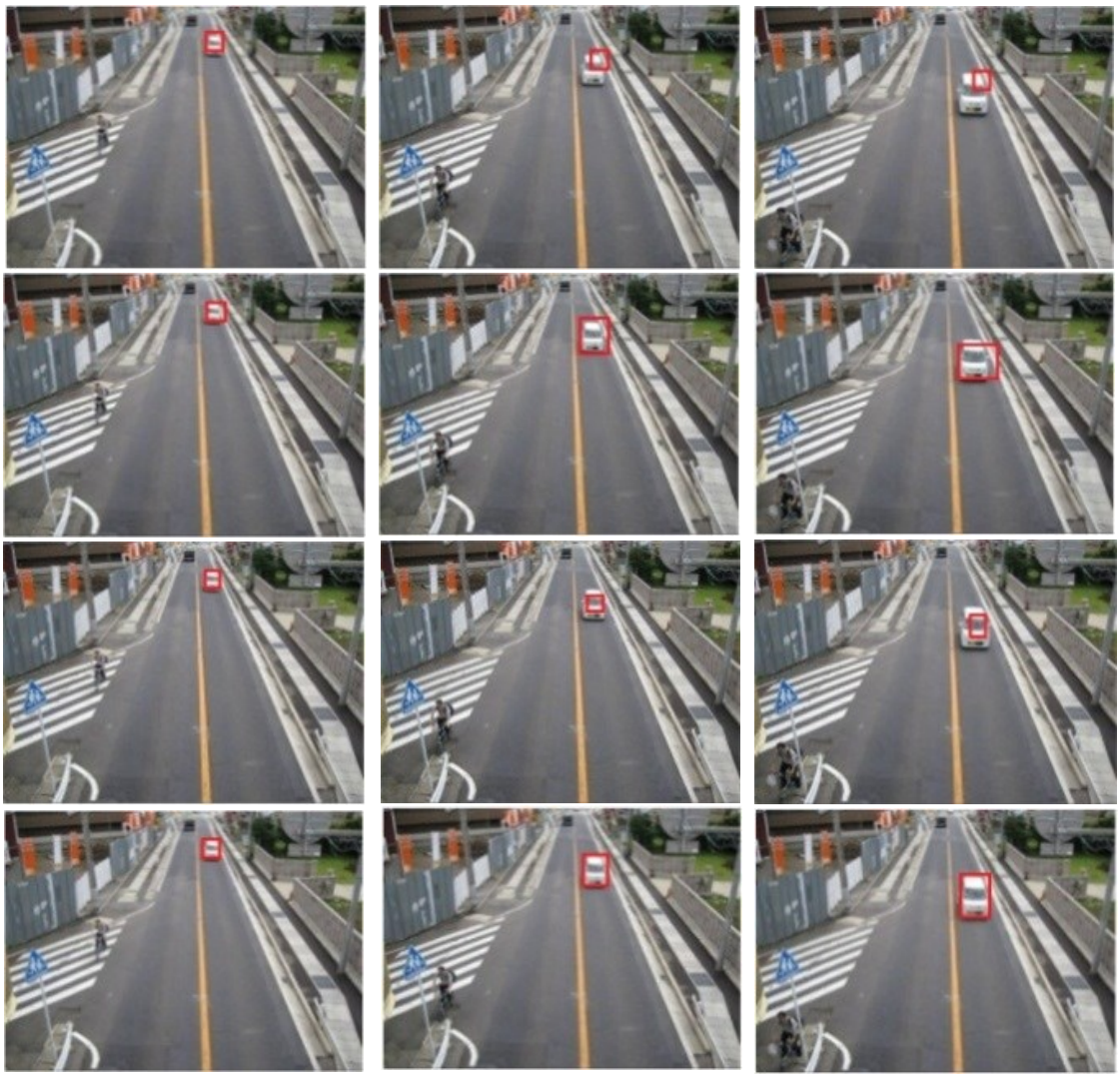


Figure 4.15 Tracking results of sequence V3 as it undergoes scale changes and illumination changes. Rows 1, 2, 3 and 4 correspond to Mean shift, Color-PF, SIFT-PF, and our tracker, respectively.

4.5. Summary

In this chapter, for the tracking problem of complex conditions, we propose a multiple features matching method based on SURF feature for robust vehicle tracking. Firstly, we introduce a review of SURF method. Secondly, for further robustness, we propose an adaptive update mechanism of feature point including discarding bad feature points and adopting the new feature points, and improved the distance kernel function method to

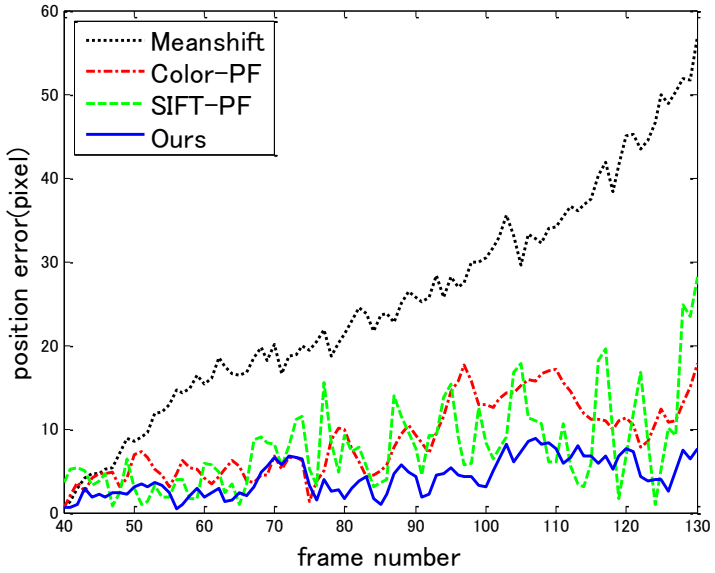


Figure 4.16 Evaluation results of V3.

allocate the weight of particle. In thorough experiments involving in challenging sequences and other state-of-the-art trackers, our approach demonstrates very promising performance. The experimental results clearly demonstrate the effectiveness of our proposed method.

Chapter 5. Multiple target tracking with FM-PHD filter

In the previous chapter, we introduced two single target tracking methods based on PF, however, multiple target tracking can provide more, useful information for decision making in the actual visual surveillance systems. In this chapter, we firstly introduce an overview of multiple target tracking. Secondly, classical data association methods are presented. Lastly, we propose a novel multiple target tracking method with feature measurement probability hypothesis density (FM-PHD), and conclude the experimental result.

5.1. Multiple target tracking

Multiple target tracking (MTT) techniques are fundamentally different from single target tracking techniques. The difference lies in the state-space model used. In a single target tracking algorithm, the state of only one target is modeled; detections from other targets are assumed to be false alarms and problems result when tracking closely-spaced or crossing targets. MTT algorithms, however, take the existence of more than one target into account simultaneously in their measurement association processes. Figure 5.1 shows the basic elements of a conventional MTT system. In theory, MTT algorithms are capable of tracking closely-spaced and crossing targets. Usually, only one measurement is assumed to be produced by each target at a given time and the targets are assumed to have independent dynamics. While these assumptions can be relaxed, it comes at a high price in terms of computational requirements.

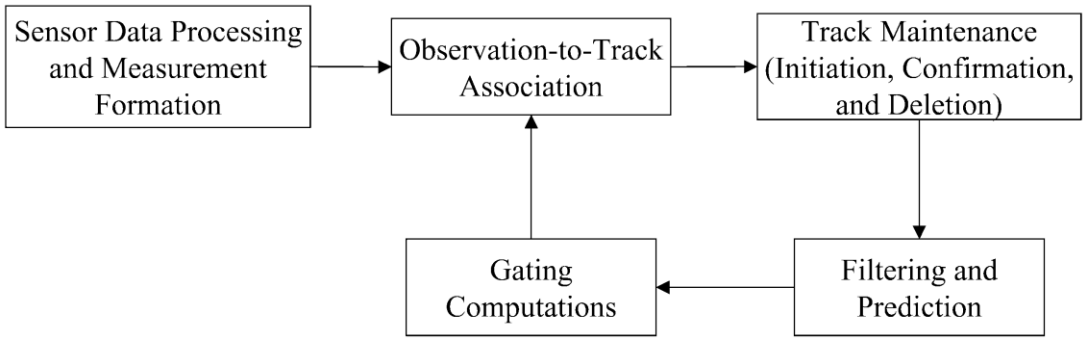


Figure 5.1 Basic elements of a conventional MTT system[22].

The multiple targets tracking problem extends the scenario to a situation where the number of targets may not be known and varies with time. The measurements which have originated from targets are not known since some of them may be due to false alarms. We are now required to estimate the positions of an unknown number of targets, based on observations of the targets corrupted by noise, with the possibilities that there may be missed detections and that observations may be false alarms due to clutter. In addition, the identities of the targets may need to be known to determine their trajectories. The usual method for solving this problem is to assign a single-target stochastic filter, such as a Kalman filter or an Extended Kalman filter, to each target and use a data association technique to assign the correct measurement to each filter.

The data association problem in multiple targets tracking usually involves in ensuring that the correct measurement is given to each stochastic filter so that the trajectories of each target can be accurately estimated, which is referred to as measurement-to-track association. The three classical approaches to it are the nearest neighbour standard filter (NNSF)[20], the joint probabilistic data association filter (JPDAF)[19], and the multiple hypothesis tracking (MHT) filter [21][22].

The NNSF simply takes the nearest validated measurement to the predicted measurement to update each of the target states. This can result in problems since the nearest validated measurement may be the same for two different targets. The JPDAF computes the joint probabilities for all the pairings between the predicted measurements and estimated target states. This technique also has to consider the false alarms from spurious measurements but is restricted to a known, fixed number of targets. The ideal MHT filter maintains probabilities of all possible associations at each time step. Unlike the NNSF and JPDAF, it does not just consider the probabilities from the previous time step, which allows for backtracking and also tracks initiation. In practice, it is not feasible to keep track of all possible hypotheses, as the computational complexity grows exponentially. Techniques for reducing the complexity include gating, to ignore irrelevant observations, pruning, to eliminate low probability hypotheses, and merging, to combine hypotheses into a single track. Some extensions of these techniques include the probabilistic MHT (PMHT)[84] that uses a soft-gating procedure and Monte Carlo (MC)-JPDA[85], which uses a sample based on the JPDA algorithm.

An alternative solution to the multiple targets tracking problem is to view the set of

observations collectively, and to try to estimate the set of target states directly, where the correct report-to-track association is considered unobservable [86]. The disadvantage of this approach is that the continuity of the individual target tracks are not kept. One such method uses Finite Set Statistics for multiple target tracking [87], with an approach analogous to the recursion used in Bayesian filtering by constructing multiple targets posterior distributions. The time required for calculating joint multi-target likelihoods grows exponentially with the number of targets so is therefore not very practical for sequential target estimation as this may need to be undertaken in real time. A practical alternative to Bayesian multiple target tracking was proposed [23] for propagating the first-order statistical moment, or probability hypothesis density (PHD), instead of the multiple target posterior.

An overview of this technique is given in the next section and the mathematical framework is described in Chapter 3.

The mathematical foundation of the multiple target filtering methods used in this thesis are based on the theory of random finite set(RFS), which was first studied by Matheron [88] in the 1970s. Mahler constructed finite set statistics (FISST)[23][86] [89] from the mathematical theory of point processes [90] and RFS theory in the mid-1990s as a way of extending classical single-sensor, single-target statistics to a multi-sensor, multi-target statistics of finite-set variates. The multi-target states and observations are represented as Random Finite Sets (RFSs) from which a theoretically optimal Bayesian multi-sensor multi-target filter can be derived [87]. The multi-target Bayes filter is not tractable for real-time implementations due to the combinatorial complexity of the multiple target likelihoods and so the optimal filter must be approximated. A recursive approach was proposed to propagate the first-order statistical moment, or expectation, of the multi-target posterior distribution based on the Stein-Winter probability hypothesis density [91]. This was called the PHD filter [23]. The predictive density is approximated by a Poisson point process to track potentially many targets, including birth, death and spawning of targets automatically.

Although the foundation was established in the form of FISST, its relationship to conventional probability was not entirely clear. Vo, Singh and Doucet established the relationship between FISST and conventional probability [92], which led to the development of a SMC multi-target filter. In addition, a SMC implementation of the

PHD filter is proposed in the form of a multi-target particle filter which operates on sets of observations to provide a multi-modal intensity function from which the multiple target states are determined.

The advantage of the particle PHD filter is that it can track a variable number of targets, and estimates both the number of targets and their locations. It avoids the need for data association techniques as part of the multiple target frameworks, since the identities of the individual targets are not required. In addition to estimating the number of targets and their states at each point in time, it is also important in tracking scenarios to know the trajectories of the targets and to be able to distinguish among different targets.

The Gaussian mixture probability hypothesis density (GM-PHD) filter was derived recently to provide a closed-form solution to the PHD filter [93]. It was shown that, under linear Gaussian assumptions, the posterior intensity at any point in time is a Gaussian mixture. The means and covariances of the Gaussians are determined from the Kalman filtering equations and the weights are calculated according to the PHD filter update equation. The multiple target states of the GM-PHD mixture are determined from the Gaussian components with the highest weights. It can be shown that Gaussians within the mixture track the evolution of individual target states which ensures the continuity of target identity.

5.2. Data association method

Nearest neighbor method

The basic solution approach for the MTT problem is the nearest neighbor (NN) method [18]. This method allows only the measurement closest in statistical distance to the predicted track to be used to update the target state estimate. Nearest neighbor method, it easy to implement, less computation, but is only applicable to the environment of high signal-to-noise ratio and low target density. In high clutter density environment, the associated effect of NN method is often unsatisfactory. The global nearest neighbor (GNN) algorithm, is also known as the 2-D assignment algorithm [94] to a certain extent improved the tracking effect. In the GNN method, the closest measurement in statistical distance to the predicted track is used to update the target

state estimate.

GNN method considers all measurement to clutter, existing track and new track association, and selects the best overall hypothesis. At each time step, an association matrix containing measurement-to-source likelihoods is formed, and the assignment problem is then solved as a convex optimization problem. While being global, GNN represents a hard decision for each measurement and only one data association hypothesis is thus considered. In some more complex scenarios, making a hard decision may be insufficient.

PDA and JPDA

The probabilistic data association (PDA) [19] method is based on computing the posterior probability of each candidate measurement found in a validation gate, and assumes that only one real target is present and all other measurements are Poisson distributed clutter. When calculating the equivalent echo, PDA method considers the associated gate of each target as a whole, but does not consider the impact of adjacent target echo, so it is only applicable to the single or isolated multiple targets environment, and has no track initiation and extinction mechanism.

The joint probabilistic data association (JPDA) [20] method combines all of the potential candidates for association to a track in a single statistically most probable update, takes account of the statistical distribution of the track errors and clutter, and assumes that more than one of the candidates is a target, and the rest are false alarms. However, JPDA can only handle a fixed number of targets and its performance suffers when targets are closely spaced.

Multiple hypothesis tracking

Multiple hypothesis tracking (MHT)[21][22] is a data association method that considers association of sequences of measurements and evaluates the probabilities of all hypothesis. Quickly, the number of possible hypotheses grows very large, and therefore methods to reduce the number of hypotheses have been suggested. These include clustering to reduce combinatorial complexity, pruning of low probability hypotheses and merging of similar hypotheses.

MHT exhaustively searches for all possible measurements to tracks associations over a number of time intervals so that measurements available at subsequent time intervals

can be used to resolve uncertainties in associational formed at present. At each time interval, it maintains a number of hypotheses that consider all possible ways of associating past measurements to targets. Since a measurement could be clutter, be generated by an existing target or be generated by a new born target, a hypothesis may discard a measurements as a clutter, and associate it with an existing track or use it to initialize a new track while ensuring a measurement is only assigned to one track at maximum. As the new set of measurements is available, a new set of hypotheses is formed for each existing hypothesis. The hypothesis with the highest posterior likelihood is returned.

While the MHT method attempts to provide all possible hypotheses over a certain number of most recent frames and chooses the most likely one, its practicality and feasibility are hampered since it requires an enumeration of an exponentially increasing number of feasible joint association hypotheses to evaluate probabilities. Due to these combinatorial problems, the data association problem makes up the bulk of the computational load in multiple targets tracking methods.

5.3. Random finite set approach to multiple target tracking

The RFS approach to multi-target tracking provides a Bayesian framework for recursive update of the multi-target posterior density with the noisy measurement set received at the sensor. The PHD filter is a computationally efficient filter based on the RFS framework and can be used to jointly estimate the number of targets and their states from the noisy measurement sets available up to the current time interval.

The main idea behind the RFS approach to multi-target tracking is to first treat the collection of targets as a set-valued state called multi-target state and the collection of measurements as a set-valued observation, called multi-observation and then to characterize uncertainties present in a multi-target tracking problem by modeling these set-valued entities as random finite sets.

Rest of this section presents an overview of the RFS approach to multi-target tracking.

5.3.1. Random finite set

In essence, a RFS X is simply a finite set-valued random variable whose cardinality $|X|$ as well as the values of the individual elements in X are also random. The randomness of $|X|$ is described by a discrete probability distribution and an appropriate density characterizes the joint distribution of the elements of X for each $|X|$. In the context of multi-target tracking, the cardinality of $|X|$ could represent the number of targets or the number of measurements, and the values of the elements of X could represent the individual target states or the individual measurements received at the sensor.

Let N_k be the number of targets present with states, $x_{k,1}, \cdots, x_{k,N_k}$ each taking values in a state space $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ at time k, and M_k the number of observations received, $z_{k,1}, \cdots, z_{k,M_k}$ each taking values in an observation space $\mathcal{Z} \subseteq \mathbb{R}^{n_z}$ at time k. Let

$$X_k = \{x_{k,1}, \cdots, x_{k,N_k}\} \in \mathcal{F}(\mathcal{X}) \tag{5.1}$$

$$Z_k = \{z_{k,1}, \cdots, z_{k,M_k}\} \in \mathcal{F}(\mathcal{Z}) \tag{5.2}$$

denote the set of targets and observations received at time k, where $\mathcal{F}(\mathcal{X})$ and $\mathcal{F}(\mathcal{Z})$ denote the respective collections of all finite subsets of $\mathcal{X}$ and $\mathcal{Z}$. By modeling the multi-target state and multi-target observation as RFSs, the multi-target filtering problem can be posed as a Bayesian filtering problem with state space $\mathcal{F}(\mathcal{X})$ and observation space $\mathcal{F}(\mathcal{Z})$. Given a realization X_{k-1} of the multi-target RFS at time $k-1$, the behavior of each target with state $x_{k-1} \in X_{k-1}$ is modeled by the RFS $S_{k|k-1}(x_{k-1})$ that can either take on state $\{x_k\}$ with probability $p_{S,k}(x_{k-1})$ or $\emptyset$ with probability $1 - p_{S,k}(x_{k-1})$. The evolution of x_k from x_{k-1} is modeled by $f_{k|k-1}(\cdot\,|x_{k-1})$. A new target may appear in the horizon either due to spontaneous target birth independently of existing targets or by spawning from a target at time $k-1$. Hence at time k, the multi-target state modeled by the RFS X_k is given by

$$X_k = \left(\bigcup_{x \in X_{k-1}} S_{k|k-1}(x)\right) \cup \left(\bigcup_{x \in X_{k-1}} B_{k|k-1}(x)\right) \cup \Gamma_k, \tag{5.3}$$

where $B_{k|k-1}(x)$ denotes the RFS of targets spawned at time k from the target with

state x at time $k-1$ and Γ_k denotes the RFS of spontaneous birth at time k. It is assumed that RFSs $S_{k|k-1}(\cdot)$, $B_{k|k-1}(\cdot)$ and Γ_k are independent of each other.

Similarly the RFS observation model incorporates the measurement likelihood, target detection uncertainty at the sensor and clutter. Assuming the set-valued observation at time k is modeled by the RFS Z_k, it is obtained as

$$Z_k = \left(\bigcup_{x \in X_k} \Theta_k(x)\right) \cup K_k \tag{5.4}$$

where $\Theta_k(x)$ denotes the RFS of measurements generated by the single-target state x at time k, and K_k denotes the RFS of clutter measurements or false alarms at time k. For each $x_k \in X_k$, $\Theta_k(x_k)$ either contributes measurement $\{z_k\}$ with a probability of detection $p_{D,k}(x_k)$ or $\emptyset$ with a probability of $1 - p_{D,k}(x_k)$. The generation of z_k from x_k is modeled by $g_k(\cdot|x_k)$. It is assumed that the RFSs $\Theta_k(X_k)$ and K_k are independent of each other.

The uncertainties in the evolution of X_k over time and the generation of Z_k can be respectively characterized by the multi-target transition density, $f_{k|k-1}(X_k|X_{k-1})$ and the multi-target measurement likelihood, $g_k(Z_k|X_k)$. The multi-target transition density incorporates all aspects of the multi-target motion such as the random number of targets, individual target dynamics, target birth, target spawning, target death and target interactions. Similarly, the multi-target measurement likelihood incorporates all sensor characteristics such as the measurement likelihood, the target detection probability and clutter models. Given that the multi-target posterior density of the multi-target state X_{k-1} given $Z_{1:k-1}$, $p_{k-1}(\cdot|Z_{1:k-1})$ is known at time $k-1$, the multi-target posterior density $p_k(\cdot|Z_{1:k})$ at time k is given by multi-target Bayes filter.

Traditionally, it is difficult to compute the multi-target transition density and measurement likelihood as a Radon-Nikodym derivatives of the appropriate probability measures. FISST provides a set of mathematical tools that can be used to construct the multi-target transition density and measurement likelihood from the underlying physical models of sensors, individual target dynamics, target birth and target death.

The multi-target Bayes filter

Given the multi-target posterior density $p_{k-1}(\cdot|Z_{1:k-1})$ at time $k-1$, the multi-target density predicted to time k is given as

$$p_{k-1}(X_k|Z_{1:k-1}) = \int f_{k|k-1}(X_k|X_{k-1})\, p_{k-1}(X_{k-1}|Z_{1:k-1})\mu_s(dX_{k-1}) \tag{5.5}$$

where μ_s is an appropriate reference measure on $\mathcal{F}(\mathcal{X})$. Detailed description on the multi-target Bayes recursion is presented in literature [95].

The updated multi-target posterior density (or the multi-target filtered density) $p_k(\cdot\,|Z_{1:k})$ is obtained from the predicted multi-target density using the measurement Z_k set available at time k as

$$p_k(X_k|Z_{1:k}) = \frac{g_k(Z_k|X_k)p_{k|k-1}(X_k|Z_{1:k-1})}{\int g_k(Z_k|X)p_{k|k-1}(X|Z_{1:k-1})\mu_s(dX)} \tag{5.6}$$

While the full Bayes filters have been used in practice with success for a small number of targets, it is computationally intractable as the number of targets increases. Analogous to the Kalman filter that provides a computationally efficient approximation to the theoretically optimal single-target Bayes filter, an approximation to the multi-target Bayes filter, called the PHD filter has been proposed by Mahler in literatures [87][23]. The basic idea is to approximate the multi-target posterior density by its first order statistical moments and to propagate it with the PHD filter.

5.3.2. Probability hypothesis density filter

The optimal multi-target Bayesian recursions are computationally intractable. The FISST theory provides a powerful tool that allows the extension of the Bayesian inference to multi-target tracking cases directly. The probability hypothesis density (PHD) filter is a suboptimal alternative to the multi-target Bayes filter. Instead of propagating the multi-target posterior density in time, the PHD filter propagates the posterior intensity of the RFS of targets, a first-order statistical moment of the posterior multi-target state, and does not require any data association computations.

For a RFS X on $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ with a probability distribution $\mathcal{P}$, its first order moment, is a non-negative function v on $\mathcal{X}$, called the intensity or the PHD function, with the property that for any closed subset $S \subseteq \mathcal{X}$

$$\int_S v(x)dx = \int |X \cap S| \mathcal{P}(dX) \tag{5.7}$$

where $|X|$ denotes the cardinality of X. Given the intensity function v, its integral over any region S gives an estimate for the number of elements in X that are present in S. The local maxima of the intensity function v are points in $\mathcal{X}$ with the highest local concentration of expected number of elements, and hence can be used to generate estimates for the elements of X.

PHD recursion

The PHD filter is a computationally cheaper alternative to propagating the multi-target posterior density recursively in time and propagates the posterior intensity function of the multi-target RFS as follows: Given the posterior intensity v_{k-1} at time $k-1$, the intensity function $\mathrm{v_{k|k-1}}$ to time k is predicted as

$$v_{k|k-1}(x) = \int \left[p_{S,k}(\xi) f_{k|k-1}(x|\xi) + \beta_{k|k-1}(x|\xi)\right] v_{k-1}(\xi) d\xi + \gamma_k(x) \tag{5.8}$$

and the posterior intensity $\mathrm{v_k}$ at time k is updated as

$$v_k(x) = \left[1 - p_{D,k}(x)\right] v_{k|k-1}(x) + \sum_{z \in Z_k} \frac{p_{D,k}(x) g_k(z|x) v_{k|k-1}(x)}{\kappa_k(z) + \int p_{D,k}(\xi) g_k(z|\xi) v_{k|k-1}(\xi) d\xi} \tag{5.9}$$

where $\kappa_k(\cdot)$ is the intensity of the clutter RFS and equals $\lambda_c c_k(z)$; Z_k is the multi-target measurement available at time k; $\gamma_k(\cdot)$ denotes the intensity of spontaneous target birth; $\beta_{k|k-1}(\cdot|\xi)$ denotes the intensity of the target RFS spawned by a target of previous state ξ at time k; and $c_k(\cdot)$ denotes the clutter density that is often assumed to be uniform density in the literature.

For the recursion given in formula (5.8) and (5.9), the following assumptions hold:

1). Each target evolves and generates measurements independently of one another;
2). The birth RFS and the surviving RFSs are independent of each other;
3). The clutter RFS is Poisson and independent of the measurement RFSs;
4). The predicted multi-target RFS is Poisson.

First three assumptions are common in most multi-target applications. The fourth assumption is a simplification needed to derive the PHD update. A RFS X is a Poisson with the mean $N = \int v(x) dx$ and given a cardinality, elements of X are i.i.d. according to v/N. Thus a Poisson RFS is completely characterized by its intensity.

5.3.3. Implementations of PHD filter

Though the PHD recursion consists of equations that are considerable simpler than those of the multi-target Bayes filter, it still requires solving multi-dimensional integrals that does not have closed-form solutions in general.

The PHD filter is a computationally efficient approximation of the RFS Bayes multi-target filter, which propagates the first moment or the intensity function of the multi-target RFS. Nevertheless, the PHD recursion requires solving multi-dimensional integrals that does not have closed-form solutions in general. SMC methods provide a way of solving such integrals and a generalized SMC implementation of PHD filter, called the SMC-PHD filter has been proposed in literatures [95] [96].

However, the existing SMC implementations of the PHD filter only provide the state estimates of individual targets. It keeps no record of the target identities and avoids the problem of associating the state estimates obtained at each time interval to individual targets. Recently, some attempts have been made to address the task of associating individual state estimates to their respective targets for the SMC-PHD filter [97][98].

Sequential Monte Carlo implementation

For general multi-target models that include nonlinear and non-Gaussian target dynamical models, the SMC approximation of the PHD filter has been proposed as a general multi-target tracking algorithm. At each time step the posterior intensity function is approximated by a weighted set of particles, from which the state estimates of individual targets are generated via standard clustering techniques. The expected number of the targets is given by the sum of the particle weights.

Given an initial intensity function v_0 at time step $k = 0$ and the sequence of measurement sets $Z_{1:k}$ up to time step k, the posterior intensity function at time step $k > 0$ is given as follows.

Initialization Step: At time step $k = 0$, the SMC-PHD filter is initialized with the particle representation of the initial intensity function v_0, i.e., $\left\{x_0^{(i)}, w_0^{(i)}\right\}_{i=1}^{L_0}$, such that

$$\hat{v}_0(x) = \sum_{i=1}^{L_0} w_0^{(i)} \delta(x - x_0^{(i)}) \tag{5.10}$$

where $\delta(x)$ is the Dirac delta function, L_0 is the number of particles at $k = 0$,

$w_0^{(i)} = N_0/L_0$ and N_0 denotes the number of targets at $k = 0$.

Prediction Step: For time step $k > 0$, given that the intensity function v_{k-1} at time step $k-1$ is approximated by the set of particles and their weights, $\left\{x_{k-1}^{(i)}, w_{k-1}^{(i)}\right\}_{i=1}^{L_{k-1}}$ as

$$\hat{v}_{k-1}(x) = \sum_{i=1}^{L_{k-1}} w_{k-1}^{(i)} \delta(x - x_{k-1}^{(i)}) \tag{5.11}$$

The predicted intensity function $\hat{v}_{k|k-1}$ at time step k is given by

$$\hat{v}_{k|k-1}(x) = \sum_{i=1}^{L_{k-1}+J_k} \tilde{w}_{k|k-1}^{(i)} \delta(x - \tilde{x}_k^{(i)}) \tag{5.12}$$

where

$$\tilde{x}_k^{(i)} \sim \begin{cases} q_k\left(\cdot \,|x_{k-1}^{(i)}, Z_k\right), & i = 1, \cdots, L_{k-1} \\ p_k(\cdot \,|Z_k), & i = L_{k-1}+1, \cdots, L_{k-1}+J_k \end{cases} \tag{5.13}$$

and

$$\tilde{w}_{k|k-1}^{(i)} = \begin{cases} \dfrac{\phi_{k|k-1}\left(\tilde{x}_k^{(i)}, x_{k-1}^{(i)}\right)}{q_k\left(\tilde{x}_k^{(i)}|x_{k-1}^{(i)}, Z_k\right)} w_{k-1}^{(i)}, & i = 1, \cdots, L_{k-1} \\ \dfrac{1}{J_k} \dfrac{\gamma_k\left(\tilde{x}_k^{(i)}\right)}{p_k\left(\tilde{x}_k^{(i)}|Z_k\right)}, & i = L_{k-1}+1, \cdots, L_{k-1}+J_k \end{cases} \tag{5.14}$$

In the particle representation of $\hat{v}_{k|k-1}$, the L_{k-1} particles are predicted forward from time step $k-1$ by the kernel $\phi_{k|k-1}$ and the additional J_k particles are drawn to detect the new born targets. The number of particles drawn at each time step can be function of time step k to accommodate the time-varying number of the new born targets so long as the average number of the new born particles per target is maintained, i.e., $J_k = \rho \int \gamma_k(x)dx$ and ρ denotes the number of particles per target.

Update Step: Given that the particle representation of the predicted intensity function is available at time step k, i.e., $\left\{\tilde{x}_k^{(i)}, \tilde{w}_{k|k-1}^{(i)}\right\}_{i=1}^{L_{k-1}+J_k}$, the updated intensity function $\hat{v}_k$ is given by

$$\hat{v}_k(x) = \sum_{i=1}^{L_{k-1}+J_k} \tilde{w}_k^{(i)} \delta(x - \tilde{x}_k^{(i)}) \tag{5.15}$$

For $i = 1, \cdots, L_{k-1} + J_k$, the particle weight $\widetilde{w}_k^{(i)}$ is updated as

$$\widetilde{w}_k^{(i)} = \left[1 - p_{D,k}\left(\tilde{x}_k^{(i)}\right) + \sum_{z \in Z_k} \frac{p_{D,k}\left(\tilde{x}_k^{(i)}\right) g_k\left(z|\tilde{x}_k^{(i)}\right)}{\kappa_k(z) + C_k(z)}\right] \widetilde{w}_{k|k-1}^{(i)} \tag{5.16}$$

where $C_k(z) = \sum_{j=1}^{L_{k-1}+J_k} p_{D,k}\left(\tilde{x}_k^{(j)}\right) g_k\left(z|\tilde{x}_k^{(j)}\right) \widetilde{w}_{k|k-1}^{(j)}$. The update step of the SMC-PHD filter uses the measurement set $\mathrm{Z_k}$ to update the particle weights at each time step.

Resampling Step: Given a particle representation of the updated intensity function $\hat{v}_k$, i.e., $\left\{\tilde{x}_k^{(i)}, \widetilde{w}_k^{(i)}\right\}_{i=1}^{L_{k-1}+J_k}$, the new set of particles and their weights, i.e., $\left\{x_k^{(i)}, w_k^{(i)}/\widetilde{N}_k\right\}_{i=1}^{L_k}$ are obtained by resampling from $\left\{\tilde{x}_k^{(i)}, \widetilde{w}_k^{(i)}/\widetilde{N}_k\right\}_{i=1}^{L_{k-1}+J_k}$, where $\widetilde{N}_k = \sum_{i=1}^{L_{k-1}+J_k} \widetilde{w}_k^{(i)}$ denotes the estimate of the target number at time step k. The particle weights are scaled by $\widetilde{N}_k$ so that the sum of the particle weights in the resampled particle set is the same as before.

The resampling step is needed in the filter to avoid the problem of degeneracy as a number of the particle weights would otherwise become negligible after a few iterations. It should be noted that the resampling step given above maintains the number of particles per target constant so that the number of particle does not increase with time, i.e., $L_k = \rho \widetilde{N}_k$. Otherwise, we have $L_k = L_{k-1} + J_k$ as the J_k number of particles are added at each time step.

Once the SMC-PHD filter is initialized at time step $k = 0$, the prediction, update and resampling steps are repeated at the subsequent time steps as the measurement set Z_k becomes available.

Multi-target state estimation

From the particle representation of the posterior intensity $\left\{x_k^{(i)}, w_{k|k-1}^{(i)}\right\}_{i=1}^{L_k}$ at each time step, the state estimates of the individual targets are generated by extracting peaks of $\hat{v}_k$ via clustering.

This has been commonly achieved via the Expectation Maximization (EM) method in literature [99] and the k-means algorithm in literature [95].

Gaussian mixture implementation

The PHD recursion does not admit closed-form solutions in general. However, for a limited case of multi-target tracking problems, a closed-form solution exists and is given by the Gaussian mixture probability hypothesis density (GM-PHD) filter [93][100]. Under linear Gaussian assumptions on the target motion and the observation model, and Gaussian assumption on the target birth, the posterior intensity function is approximated by a sum of Gaussians whose means, covariances and weights are analytically propagated in time.

The GM-PHD recursion forms the basis of a general multi-target tracking algorithm, the GM-PHD filter, which has been shown to track an unknown and time-varying number of targets.

The GM-PHD recursion propagates the intensity function that is approximated with a Gaussian mixture by analytically propagating the weights, means and covariances of the Gaussian mixture terms. The updated intensity function is also a Gaussian mixture.

Linear multi-target models

The linear Gaussian multiple target model includes a number of assumptions on the birth, death and detection of targets as well as the linear Gaussian assumptions on the target dynamical model. The assumptions are summarized below.

Each target follows a linear Gaussian dynamical model and the sensor has a linear Gaussian measurement model, i.e.,

$$f_{k|k-1}(x|\zeta) = \mathcal{N}(x; F_{k-1}\zeta, Q_{k-1}), \tag{5.17}$$

$$g_k(z|x) = \mathcal{N}(z; H_k x, R_k), \tag{5.18}$$

where $\mathcal{N}(\cdot; m, P)$ denotes a Gaussian density with mean m and covariance P, F_{k-1} is the state transition matrix, Q_{k-1} is the process noise covariance, H_k is the observation matrix, and R_k is the observation noise covariance.

The survival and detection probabilities are both state independent, i.e.,

$$p_{S,k}(\mathrm{x}) = p_{S,k}, \tag{5.19}$$

$$p_{D,k}(x) = p_{D,k}. \tag{5.20}$$

The intensities of the birth and spawn RFSs are Gaussian mixtures of the form

$$\gamma_k(x) = \sum_{i=1}^{J_{\gamma,k}} w_{\gamma,k}^{(i)} \mathcal{N}\left(x; m_{\gamma,k}^{(i)}, P_{\gamma,k}^{(i)}\right) \tag{5.21}$$

$$\beta_{k|k-1}(x|\zeta) = \sum_{j=1}^{J_{\beta,k}} w_{\beta,k}^{(j)} \mathcal{N}\left(x; F_{\beta,k-1}^{(j)}\zeta + d_{\beta,k-1}^{(j)}, Q_{\beta,k-1}^{(j)}\right) \tag{5.22}$$

where $J_{\gamma,k}$, $w_{\gamma,k}^{(i)}$, $m_{\gamma,k}^{(i)}$, $P_{\gamma,k}^{(i)}$, $i = 1, \cdots, J_{\gamma,k}$, are given model parameters that determine the shape of the birth intensity; similarly, $J_{\beta,k}$, $w_{\beta,k}^{(j)}$, $F_{\beta,k-1}^{(j)}$, $d_{\beta,k-1}^{(j)}$, and $Q_{\beta,k-1}^{(j)}$, $j = 1, \cdots, J_{\beta,k}$, determine the shape of the spawning intensity of a target with previous state ζ.

The GM-PHD recursion consists of the following prediction and update steps.

Prediction Step: Given that the posterior intensity v_{k-1} at time step $k-1$ is a Gaussian mixture of the form

$$v_{k-1}(x) = \sum_{i=1}^{J_{k-1}} w_{k-1}^{(i)} \mathcal{N}\left(x; m_{k-1}^{(i)}, P_{k-1}^{(i)}\right) \tag{5.23}$$

Then, the predicted intensity to time step k is also a Gaussian mixture and is given by

$$v_{k|k-1}(x) = v_{\mathrm{S,k|k-1}}(x) + v_{\mathrm{\beta,k|k-1}}(x) + \gamma_k(x) \tag{5.24}$$

where $\gamma_k(x)$ is given in formula (5.21).

$$v_{S,\mathrm{k|k-1}}(x) = p_{S,k} \sum_{j=1}^{J_{k-1}} w_{k-1}^{(j)} \mathcal{N}\left(x; m_{S,k|k-1}^{(j)}, P_{S,k|k-1}^{(j)}\right) \tag{5.25}$$

$$m_{S,k|k-1}^{(j)} = F_{k-1} m_{k-1}^{(j)}, \tag{5.26}$$

$$P_{S,k|k-1}^{(j)} = Q_{k-1} + F_{k-1} P_{k-1}^{(j)} (F_{k-1})^T, \tag{5.27}$$

$$v_{\mathrm{\beta,k|k-1}}(x) = \sum_{j=1}^{J_{k-1}} \sum_{l=1}^{J_{\beta,k}} w_{k-1}^{(j)} w_{\beta,k}^{(l)} \mathcal{N}\left(x; m_{\beta,k|k-1}^{(j,l)}, P_{\beta,k|k-1}^{(j,l)}\right), \tag{5.28}$$

$$m_{\beta,k|k-1}^{(j,l)} = F_{k-1}^{(l)} m_{k-1}^{(j)} + d_{\beta,k-1}^{(l)}, \tag{5.29}$$

$$P_{\beta,k|k-1}^{(j,l)} = Q_{\beta,k-1}^{(l)} + F_{\beta,k-1}^{(l)} P_{\beta,k-1}^{(j)} \left(F_{\beta,k-1}^{(l)}\right)^T. \tag{5.30}$$

Update Step: Assuming that the predicted intensity $v_{k|k-1}$ to time step k is a Gaussian mixture of the form

$$v_{k|k-1}(x) = \sum_{i=1}^{J_{k|k-1}} w_{k|k-1}^{(i)} \mathcal{N}(x; m_{k|k-1}^{(i)}, P_{k|k-1}^{(i)}) \tag{5.31}$$

The posterior intensity v_k at time step k is also a Gaussian mixture, and is given by

$$v_k(x) = (1 - p_{D,k}) v_{k|k-1}(x) + \sum_{z \in Z_k} v_{D,k}(x; z) \tag{5.32}$$

where

$$v_{D,k}(x; z) = \sum_{j=1}^{J_{k|k-1}} w_k^{(j)}(z) \mathcal{N}\left(x; m_{k|k}^{(j)}(z), P_{k|k}^{(j)}\right), \tag{5.33}$$

$$w_k^{(j)}(z) = \frac{p_{D,k} w_{k|k-1}^{(j)} q_k^{(j)}(z)}{\kappa_k(z) + p_{D,k} \sum_{l=1}^{J_{k|k-1}} w_{k|k-1}^{(l)} q_k^{(l)}(z)}, \tag{5.34}$$

$$q_k^{(j)}(z) = \mathcal{N}\left(z; H_k m_{k|k-1}^{(j)}, R_k + H_k P_{k|k-1}^{(j)} (H_k)^T\right), \tag{5.35}$$

$$m_{k|k}^{(j)}(z) = m_{k|k-1}^{(j)} + K_k^{(j)} \left(z - H_k m_{k|k-1}^{(j)}\right), \tag{5.36}$$

$$P_{k|k}^{(j)} = \left[I - K_k^{(j)} H_k\right] P_{k|k-1}^{(j)}, \tag{5.37}$$

$$K_k^{(j)} = P_{k|k-1}^{(j)} (H_k)^T \left(H_k P_{k|k-1}^{(j)} (H_k)^T + R_k\right)^{-1}. \tag{5.38}$$

The prediction step and update step of the GM-PHD recursion forms the basis of a general multi-target tracking algorithm, called the GM-PHD filter. Given that an initial intensity function v_0 at time step $k = 0$ is a known Gaussian mixture, the posterior intensity function at time step $k > 0$ is also a Gaussian mixture from which the estimates of individual target states need to be extracted via peak extractions. The

expected number of targets $\widehat{N}_{k|k-1}$ and $\widehat{N}_k$ associated with $v_{k|k-1}$ and v_k are obtained by summing the appropriate mixture weights. The closed-form recursions for $\widehat{N}_{k|k-1}$ and $\widehat{N}_k$ are as follows:

$$\widehat{N}_{k|k-1} = \widehat{N}_{k-1}\left(p_{S,k} + \sum_{j=1}^{J_{\beta,k}} w_{\beta,k}^{(j)}\right) + \sum_{j=1}^{J_{\gamma,k}} w_{\gamma,k}^{(j)}, \tag{5.39}$$

$$\widehat{N}_k = \widehat{N}_{k|k-1}\left(1 - p_{D,k}\right) + \sum_{z\in Z_k}\sum_{j=1}^{J_{k|k-1}} w_k^{(j)}(z). \tag{5.40}$$

The estimate of the target number given above suffers from the instability in the low probability of target detection.

Given that the posterior intensity function at each time step is a mixture of weighted Gaussian terms, the means of all Gaussian terms give the local maxima of the intensity function. For the multi-target state estimation, given that a weight threshold is w_{Th}, the state estimates of individual targets are obtained by selecting means of the Gaussian terms with weights greater than w_{Th},

$$\hat{X}_k = \left\{m_k^{(i)} : w_k^{(i)} > w_{Th}\right\}. \tag{5.41}$$

As a result, the GM-PHD filter avoids the need for standard clustering techniques that are needed in the SMC-PHD filter.

5.4. Multiple target tracking based on PHD filter and feature measurement

As pointed out in the previous section, the GM-PHD filter can track an unknown and time-varying number of targets. For tracking moving objects in the video sequences, how to apply GM-PHD filter and extract the feature measurement random set is a challenge. In this section, we are motivated by the work in literature [23], and propose a method for extracting the feature measurement random set.

Let $Z_k = \{z_1, \cdots, z_k\}$ is the multi-target measurement set. The posterior intensity is defined as

$$v_{k|k-1}(x) = v_{k|k-1}(x|Z^{(k-1)}) \tag{5.42}$$

$$v_k(x) = v_k(x|Z^{(k)}) \tag{5.43}$$

$$v_{k|k-1}[h] = \int h(x)v_{k|k-1}(x)\,dx \tag{5.44}$$

Assume that $f_{k|k-1}(X|Z^{(k-1)})$ is approximately Poisson, i.e.,

$$f_{k|k-1}\left(X\middle|Z^{(k-1)}\right) \cong e^{-\mu}\mu^n s(x_1)\cdots s(x_n) \tag{5.45}$$

The generating function of posterior density, $G_{k|k-1}[h]$ at time $k-1$ has this form $G_{k|k-1}[h] = e^{\mu s[h]-\mu}$. For some $\mu \geq 0$ and probability density $s(x)$, in which case $v_{k|k-1}\left(x\middle|Z^{(k-1)}\right) = \mu \cdot s(x)$, where $s[h] = \int h(x)s(x)dx$, $h(x)$ is a real-valued function. Then, the PHD approximate Bayes corrector equation is

$$v_k(x) \cong F_k(Z_k|x)v_{k|k-1}(x) \tag{5.46}$$

where

$$F_k(Z|x) = \sum_{z\in Z_k} \frac{P_{D,k}(x)L_z(x)}{\lambda c(z)+v_{k|k-1}[P_D L_z]} + 1 - p_{D,k}(x) \tag{5.47}$$

For the detailed proof process, the reader is referred to the section VIK in literature [23].Where $L_{k,z}(x) = f_k(z|x)$ is the sensor likelihood function, hereafter abbreviated as $L_z(x)$. And, we also suppose that the sensor has an infinite field of view, such as $P_{D,k}(x) = P_D$ is a constant. We suppose no missed detections and no false alarms can occur in the update stage of PHD. Given that $P_{D,k}(x) = 1$, then, we can obtain the posterior intensity $v_k(x)$ at time k.

$$v_k(x) \cong \sum_{z\in Z_k} v_k(x|z) \tag{5.48}$$

where

$$v_k(x|z) = \frac{L_z(x)}{v_{k|k-1}[L_z]} v_{k|k-1}(x) \tag{5.49}$$

So, we can obtain the approximate posterior intensity $\hat{v}_k(x)$

$$v_k(x) \cong \hat{v}_k(x) = L_z(x)\, v_{k|k-1}(x) \tag{5.50}$$

where $L_z(x)$ is the sensor likelihood function. In visual tracking, we can adopt the feature histogram model of tracking object to evaluate this feature likelihood, and find the feature measure random set.

5.4.1. State vector and dynamic model

The state vector at time k of a single object typically consists of kinematic and region parameters. In this paper, the moving object is modeled by a rectangular bounding box defined in terms of the dynamic state

$$\mathrm{X} = \{x, \dot{x}, y, \dot{y}, h_x, h_y\}^{\mathrm{T}} \tag{5.51}$$

where x and y represent the center coordinates of the rectangular box for tracking an object, $\dot{x}$ and $\dot{y}$ represent the respective velocity components, h_x, h_y denote the height and width of half axes, respectively. And, the state dynamic is typically described by

$$\mathrm{X_k} = A\mathrm{X_{k-1}} + Bw_{k-1}, \tag{5.52}$$

where A and B are the deterministic components of the model, and w_{k-1} is a Gaussian distribution of zero mean with covariance matrix $Q = diag\left\{\sigma_x^2, \sigma_{\dot{y}}^2, \sigma_{h_x}^2, \sigma_{h_y}^2\right\}$, describing the uncertainty in the state vector.

$$A = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.53}$$

$$B = \begin{bmatrix} T^2/2 & 0 & 0 & 0 \\ T & 0 & 0 & 0 \\ 0 & T^2/2 & 0 & 0 \\ 0 & T & 0 & 0 \\ 0 & 0 & T^2/2 & 0 \\ 0 & 0 & 0 & T^2/2 \end{bmatrix} \tag{5.54}$$

5.4.2. Monte Carlo sampling

In this step, we obtain the feature measurement random set by Monte Carlo sampling. For feature measurement, we adopt the color and texture features, which are described in detail in section 3.3.2 feature likelihood models in Chapter 3.

At time k, we firstly compute the predicted intensity $v_{k|k-1}(x)$, which is a Gaussian mixture. Next, we sample particles $\{x^i, l^i, w^i\}_{i=1}^N$ from $v_{k|k-1}(x)$ by the Monte Carlo method, where w^i is the weight for particle x^i, $l^i$is the integer label of particle x^i.Then, the posterior intensity $v_k(x)$ will be approximated by $\{x^i, l^i, \varphi^i\}$, where $\varphi^i = L(x^i)w^i$. Next, the particles set will be re-sampled and clustered. To accurately express the appearance of tracking target, we adopt the color and texture feature to present the sampling particles. Lastly, by fusing the color and LBP texture measurement to obtain the feature measurement, we can obtain the measurement result set $Z_{f,k}$ at time k.

$$Z_{f,k} = \{Z_{f,1}, \cdots, Z_{f,n}\} \tag{5.55}$$

where $Z_{f,i}$ is the center state of i-th cluster, n is the number of tracking target, and $f \in \{C, T\}$, C and T denote the color and texture feature, respectively. Note that these associated cluster indices at time k for each of the cluster created at time k, we adopt the cluster labels to associate each cluster. So we sum the weights associated with the particles in each group with the same label.

5.4.3. Feature measurement fusion

In the tracking process, there are complex conditions, such as illumination changes, partial occlusion, and similar backgrounds. Use of a single feature is inadequate for meeting the tracking needs, although the computational complexity of the weighted algorithm is low. Multiple-feature fusion can better utilize information provided by the features supplementing each other and can improve the robustness of the algorithm. Thus, according to the above description, we apply an adaptive weight to fuse the color and texture measurement.

$$Z_{F,k} = \alpha \cdot Z_{C,k} + (1 - \alpha) \cdot Z_{T,k} \tag{5.56}$$

where, $Z_{C,k}$ denotes the color measurement set, $Z_{T,k}$ denotes the texture measurement, $Z_{F,k}$ denotes the fusion feature measurement set at time k, α is the proportion to the number of color particle weight larger than a threshold TH in the color and texture feature. In the experiment, we set TH=0.5. Note that the larger the value is, the more similar color feature distributions.

Algorithm flow

The flow of the proposed tracking algorithm is described as follows:

Input: Video frames $F_1, F_2, \cdots, F_n$.

Output: Target regions according to target states in each frame.

Algorithm:

(1) Select the target objects in initializing frame manually, and calculate the appearance model with color and texture feature for the first frame.

(2) Initialize the corresponding parameters of Gaussian mixture $\left\{w_{k-1}^{(i)}, m_{k-1}^{(i)}, P_{k-1}^{(i)}\right\}_{i=1}^{J_{k-1}}$, the measurement set $Z_{c,k}$ and $Z_{t,k}$ and form the posterior intensity corresponding to the formula (5.23).

(3) Input a new frame and predict the birth targets $\left\{w_{k|k-1}^{(j)}, m_{k|k-1}^{(j)}, P_{k|k-1}^{(j)}\right\}_{j=1}^{J_{\gamma,k}}$.

(4) Predict the survival targets $\left\{w_{k|k-1}^{(i)}, m_{k|k-1}^{(i)}, P_{k|k-1}^{(i)}\right\}_{i=1}^{J_{k-1}}$ according to the formulas (5.25),(5.26) and (5.27).

(5) Calculate the color and texture measurements $Z_{c,k}$, $Z_{t,k}$, and form feature measurement Z_k according to the formula (5.56).

(6) Update the Gaussian mixture components of missing targets and existing targets $\left\{w_k^{(i)}, m_k^{(i)}, P_k^{(i)}\right\}_{i=1}^{J_{k-1}+J_{\gamma,k}}$.

(7) Prune and merge the number of Gaussian components.

(8) Estimate the number of targets, extract the target state and draw the tracking rectangle in current frame.

(9) Update the reference model of color and texture feature of tracking targets and go to step 3.

5.5. Experiment results

Test conditions

The results presented in this section were obtained on a dataset of targets extracted from three different test sequences (Figure 5.2). First tracking sequence S1 was taken from an on-board camera on the highway. Second tracking sequences S2 was from a public dataset [39]. Third tacking sequences S3 was extracted from the PETS 2001 dataset [38].The parameters of the tracker were set experimentally and were different for different datasets. Two hundred particles were used per frame. The results presented were obtained from a MATLAB implementation running under Windows on a PC with a Core 3.40GHz CPU.

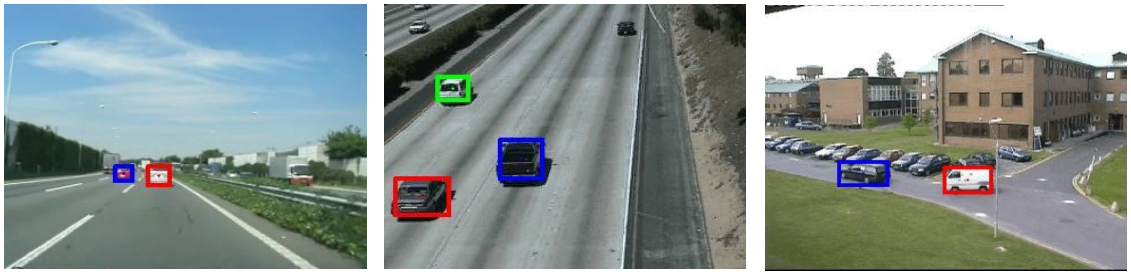


Figure 5.2 Targets of the evaluation data sets S1, S2 and S3.

Performance evaluation

The performance evaluation was based on the optimal sub-pattern assignment (OSPA) metric[101][102]. The OSPA metric is a mathematically and intuitively consistent metric, which tries to capture the estimate quality both the cardinality and multi-target states.

Let $X = \{x_1, \cdots, x_m\}$ and $Y = \{y_1, \cdots, y_n\}$ represent the real multi-target state set and the estimated one with the cardinality m and n, where $m, n \in \mathbb{N}_0 = \{0,1,2, \cdots\}$, and Π_k be the set of permutations on $\{1,2, \cdots, k\}$ for any $k \in \mathbb{N} = \{1,2, \cdots\}$. For $1 \leq p < \infty$, $c > 0$, the OSPA metric of order p with cut off at c is defined as

$$\overline{d}_p^{(c)}(X,Y) = \left[\frac{1}{n}\left(\min_{\pi \epsilon \Pi_n} \sum_{i=1}^{m} d^{(c)}(x_i, y_{\pi(i)})^p + c^p(n-m)\right)\right]^{1/p} \quad (5.57)$$

If $m \leq n$, then $\bar{d}_p^{(c)}(X,Y) = \bar{d}_p^{(c)}(Y,X)$, if $m > n$; moreover,

$$\bar{d}_{\infty}^{(c)}(X,Y) = \begin{cases} \min_{\pi \epsilon \Pi_n} \max_{1 \le i \le n} d^{(c)}\left(x_i, y_{\pi(i)}\right), & if\ m = n \\ c, & if\ m \neq n \end{cases} \tag{5.58}$$

is either case set the distance to zero if $m = n = 0$. In this experiment, we set c=70 and p=2. To capture the average performance ,we run 100 Monte Carlo(MC) trials for each filter with the same target tracks but independently generated measurements.

We implemented the comparison between our proposed method (referred to as FM-PHD) and the JPDA[20]. The JPDA is a method to estimate states of objects based on enumerating and computing probabilities of all possible associations between objects and observations. The S1 sequence undergoes large scale changes, illumination change and camera shake in a clutter environment. The tracking results are illustrated in Figure 5.3, in which four representative frames (1000, 1041, 1055, and 1087) are shown, where rows 1 and 2 correspond to JPDA and our method. The tracking results clearly reflect the performance of the different methods. Specially, for the on-board camera, the camera shake caused a great impact on the tracking targets. We can see that our proposed tracker performed well throughout the sequence.

Figure 5.3 Tracking results of S1 sequence. Rows 1 and 2 correspond to JPDA and FM-PHD, respectively (1000, 1041, 1055 and 1087).

The second sequence, S2, undergoes large scale change and camera shake in a cluttered environment. Some tracking results are shown in Figure 5.4. The frame indexes are 115,120,123 and 130. The JPDA method could not provide accurate state information for the vehicles under these conditions.

In the third experiment using the S3 sequence, the tracking targets are the two

moving vehicles undergoing translation, scale change. The tracking results are illustrated in Figure 5.5, in which four representative frames (2589, 2602, 2609, and 2629) are shown, where rows 1 and 2 correspond to JPDA and our method, respectively. We can see that our proposed tracker performed well from beginning to the end. The OSPA distance curves

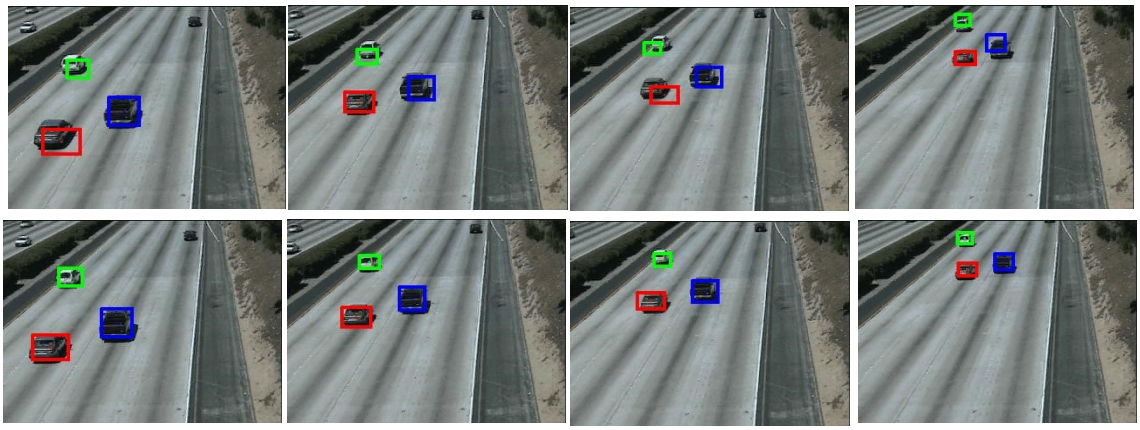


Figure 5.4 Tracking results of S2 sequence. Rows 1 and 2 correspond to JPDA and FM-PHD, respectively (115, 120, 123 and 130).

corresponding to test frames are shown in Figure 5.6. The results showed that our method is better than the JPDA tracker.

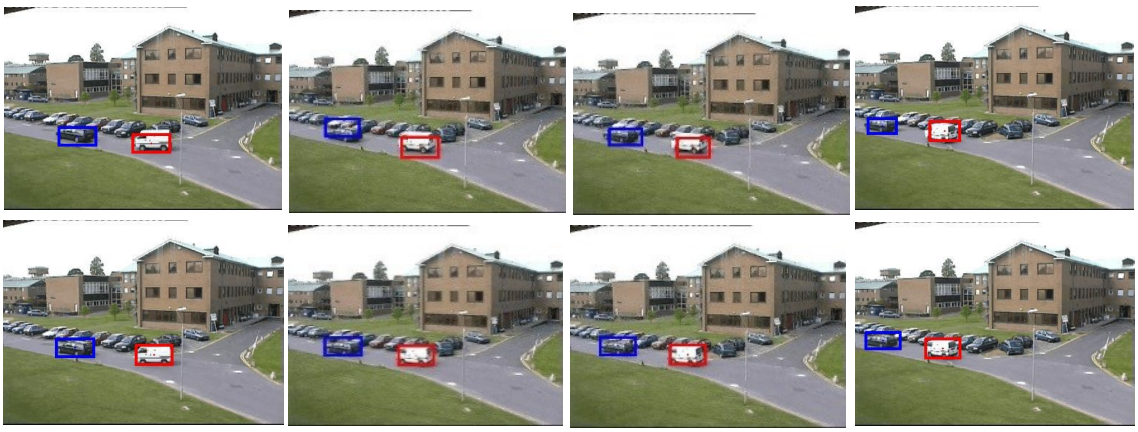


Figure 5.5 Tracking results of S3 sequence. Rows 1 and 2 correspond to JPDA and FM-PHD, respectively (2589, 2602, 2609 and 2629).

Lastly, we list the average OSPA distance of the comparing methods for three

sequences as shown in Table 5.1. From the results, it is easy to see that the proposed method is more accuracy than the JPDA method.

5.6. Summary

In this chapter, we firstly introduce the classical data association methods for MTT, and then analyze the characteristic of these ones. Secondly, we present the PHD filter based on RFS theories. And we apply the GM-PHD filter to track multiple vehicles. For

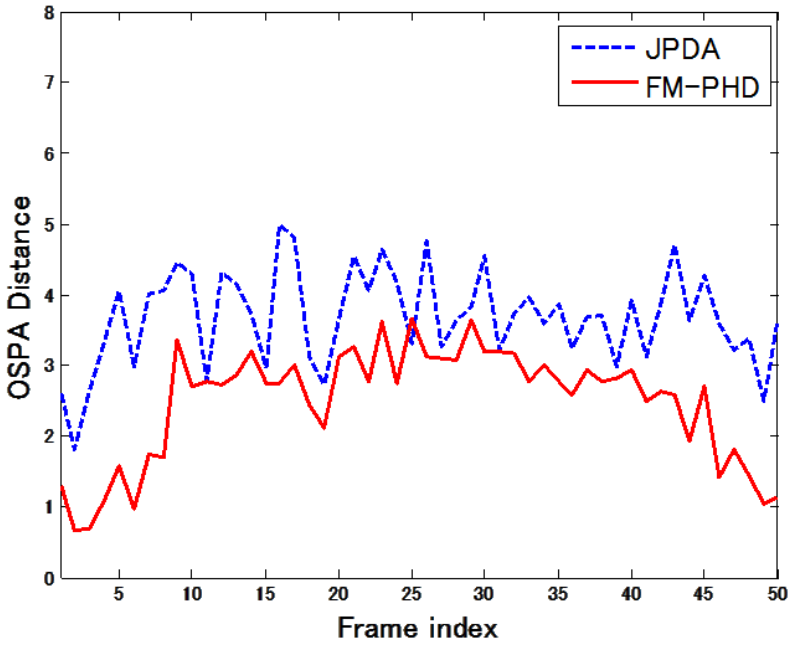


Figure 5.6 The OSPA distance of JPDA and FM-PHD for the S3 sequence.

Table 5.1 The average OSPA distance of test sequences.

Sequence	JPDA(pixel)	FM-PHD (pixel)
S1	4.04	2.50
S2	4.59	3.17
S3	3.68	2.48

visual tracking problem, it is difficult to extract the random set of visual target. So we propose a method using the likelihood function and feature measurement to approximate the PHD. For the feature measurement, we adopt an adaptive weight to fuse color measurement and texture measurement, and extract feature measurement

based on Monte Carlo technique, and implement the PHD filter using the GM method. Lastly, contrasting the JPDA method, experimental results clearly demonstrated the effectiveness of our proposed method.

Chapter 6. Conclusions and future work

In this thesis, the problem of moving target detection and tracking in the visual surveillance system has been studied. We proposed four methods for moving target detection, single target tracking based on particle filter, and multiple target tracking. We consider robustness and accuracy as the major design goals of our work. In this finial chapter, we will summarize our research and briefly describe some areas that merit future research.

6.1. Research summary

Due to the dynamic environmental conditions and interference factors such as illumination changes, shadows, stop-and-go vehicles and camera vibrations, we proposed a novel background subtraction for moving target detection. Firstly, the characteristics of each color space, such as RGB, HSV and YCbCr color space, are analyzed. For these characteristics, we propose the color statistical background model based on YCbCr color space. The contrastive analysis of mean and standard deviation of extracted background model show the validity of proposed background. In particular, this model can effectively reduce the influence of illumination changes, and dramatically adapt to environmental changes. Secondly, we propose a new multiple feature similarity fusion algorithm using the Choquet integral to class the foreground and background, and provide a new idea for high precision target detection under the complex conditions. The choice of classification features directly affects the result of moving target detection. Color feature is one of the most commonly used features, but it is difficult to accurately describe image information for the instability of natural background, and the individual classification feature is sensitive to the dynamic changes of scene. So, it will cause the inaccurate extraction of moving target. Therefore, to fuse the ULBP texture feature and color feature, a new method for moving target detection using fuzzy integral is presented. Due to the complex conditions, the background maintenance process is an important step. Thus, by analyzing of traditional blind and selective background maintenance process, we propose an adaptive background

maintenance method to adapt the complex condition. Comparing experiments show that the proposed method has good performance.

In Chapter 3, a multiple feature fusion algorithm for single target tracking based on particle filter is proposed. We firstly introduce a review of particle filter techniques, including Bayesian estimation, Monte Carlo simulation, Bayesian importance sampling, sequential importance sampling, particle degeneracy and resampling, and analysis of the advantages and disadvantages of the corresponding methods. Next, we propose an adaptive mechanism to fuse color, edge, and texture features for moving target tracking using particle filter. The proposed mechanism not only fuses multiple features to improve the representation ability of tracking target, but dynamically balances the effect of feature similarity and feature discriminability among target object, candidate and adjacent background to obtain the adaptive feature weight. We adopt the Bhattacharyya coefficient to represent the similarity, and use the variance of the log-likelihood ratio to describe the discriminability. Moreover, we compare other state-of-the-art algorithms to demonstrate the effectiveness of our proposed method.

In Chapter 4, for the tracking problem of complex conditions, such as fast moving, large scale change, rotation, and mutual occlusion, etc., we propose a robust vehicle tracking based on SURF feature in a particle filter framework. Combining the color and LBP feature, the proposed method improves the represent ability of tracking target. And, the dynamic update mechanism of target template is proposed to capture appearance changes, and the size of tracking window is also modified dynamically by balancing the weights of three feature distribution. The weights of each particle are allocated by an improved distance kernel function method. Specifically, the proposed method of adopting new feature points for target template can objectively reflect tracking target changes, and effectively overcome the disadvantage of random selection mechanism. Lastly, we use the different challenging sequences to test the efficient and robust performance of our approach.

In Chapter 5, for the multiple target tracking, we firstly introduce the classical data association methods, such as NN, PDA, JPDA and MHT, and analyze the characteristic of these methods. Secondly, we introduce the PHD filter based on random finite set theories. The PHD filter is a promising approach for multi-target tracking which propagates the PHD or the first moment of the multi-target posterior density instead of

the full multi-target posterior density. Based on this idea, we propose a multiple vehicle tracking method FM-PHD using the feature measurement to approximate the posterior density. To improve the representation ability of tracking target, we adopt an adaptive weight to fuse the color and LBP features which are extracted by Monte Carlo method. Lastly, we use Gaussian mixture to implement the tracking method, and verify the effectiveness and accuracy through the different types test sequences.

6.2. Future work

The methods we presented for intelligence visual surveillance show promising results. However, some further studies should be done for moving target detection and tracking in a complex background.

(1) Moving target occlusion problem: In complex background, if partial occlusion or mutual occlusion occurs at the long time in sequences, the moving target detection method cannot segment the multiple moving targets accurately. Therefore, how to solve the occlusion problem of moving targets is worth further study.

(2) Multiple cameras surveillance problem: Surveillance systems based on multiple cameras can not only expand the monitoring scope, but obtain more image information in different directions. Therefore, moving target detection using multiple cameras can not only eliminate easily shadow effect, but improve the performance of moving target detection in complex background. However, with the introduction of multiple cameras, it brings other problems, such as multiple event scheduling, multiple image registration and so on.

(3) Recognition and analysis of tracking results: Our methods can be improved and extended in the following ways, target recognition, target classification and activity analysis. Thus, these methods can provide more comprehensive and accurate information for the users of intelligent surveillance system.

Bibliography

[1]http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2010:207:0001:0013:EN:PDF.

[2] http://www.nilim.go.jp/japanese/its/0frame/index_b.htm.

[3] P. G. Micchalopoulos, “Vehicle detection through video image processing AUTOSCOPE system”, IEEE Transaction on Vehicular Technology, Vol. 40, No. 1, pp. 21-29, 1991.

[4] Berthold K. P. Horn and Brian G. Schunck, “Determining Optical Flow”, Artifical Intelligence, Vol.17, pp. 185-203, 1981.

[5] J. L. Barron, D. J. Fleet and S. S. Beauchemin, “Performance of Optical Flow Techniques”, Int. Journal of Computer Vision, Vol. 12, No. 1, pp. 43-77, 1994.

[6] A. Verri, S. Uras and E.D. Micheli, “Motion segmentation from optical flow”, In: Proc. of the 5th Alvey Vision Conf., pp. 209-214, 1989.

[7] Y. Mae, Y. Shirai, J. Miura, and et al, “Object tracking in cluttered background based on optical flow and edges”, Proc. of the Int. Conf. on Pattern Recognition, Vienna, Austria, pp. 196-200,1996.

[8] J. Shin, S. Kim, S. Kang, and et al, “Optical flow-based real-time object tracking using non-prior training active feature model”, ELSEVIER Real-Time Imaging, Vol.11,pp. 204-218, 2005.

[9] A. J. Lipton, H. Fujiyoshi, and R. S. Patil, “Moving target classification and tracking from real-time video”, Proc. of the 4th IEEE Workshop on Applications of Computer Vision, Princeton, NJ, pp. 8-14, October, 1998.

[10] S. Gupte, O. Masoud, R. K. Martin and N. P. Papanikolopoulos, “Detection and classification of vehicles”, IEEE Transactions on Intelligent Transportation Systems, Vol. 3, No.1, pp. 37-47, 2002.

[11] R. Cucchiara, C. Grana, M. Piccardi and A. Prati, “Statistic and Knowledge-based Moving object detection in traffic scenes”, Proc. of IEEE Intelligent Transportation Systems, pp. 27-32, 2000.

[12] N. Friedman and S. Russell, “Image segmentation in video sequences: A probabilistic approach”, Proc. of the 13th Annual Conf. on Uncertainty in Artificial

Intelligence (UAI-97), pp. 175-181, Morgan Kaufmann Publishers, Inc., (San Francisco,CA), 1997.

[13] R. E. Kalman, “A new approach to linear filtering and prediction problems”, Transactions of the ASME-Journal of Basic Engineering, Vol.82, No. SeriesD, pp. 35-45, 1960.

[14] E. A. Wan and R. V. D. Merwe, “The unscented kalman filter for nonlinear estimation”, Proc. of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Controls Symposium, pp. 153-158, 2000.

[15] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, pp. 603-619, 2002.

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

[17] P. Perez, C. Hue, J. Vermaak, and M. Ganget, “Color-based probabilistic tracking”, Proc. of the European Conf. on Computer Vision, pp. 661-675, 2002.

[18] Y. Bar-Shalom, X. Li, “Multitarget-multisensor tracking: principles and techniques”, YBS Publishing, Storrs, CT, 1995.

[19] Y. Bar-Shalom, and E. TSE, “Tracking in a cluttered environment with probabilistic data association”, Automatica, Vol. 11, pp. 451-460,1975.

[20] Y. Bar-Shalom and T. E. Fortmann, “Tracking and data association”, Academic Press, San Diego, 1988.

[21] D. Reid, “An algorithm for tracking multiple targets”, IEEE Transactions on Automatic Control, Vol. 24, No. 6, pp. 843-854,1979.

[22] S. Blackman, “Multiple hypothesis tracking for multiple target tracking”, IEEE A&E System Magazine, Vol. 19, No.1, part 2.pp. 5-18, 2004.

[23] R. Mahler, “Multitarget Bayes filtering via first-order multi-target moments”, IEEE Transactions on Aerospace and Electronic Systems, Vol. 39, No. 4, pp. 1152-1178, 2003.

[24] S. Haykin, Kalman Filtering and Neural Networks, Wiley, 2001.

[25] L. Li, W. Huang, I. Y. Gu, and Q. Tian, “Statistical modeling of complex backgrounds for foreground object detection”, IEEE Transactions on Image Processing,

Vol. 13, No. 11, pp. 1459-1472, 2004.

[26] T. Horprasert, D. Harwood, L. S. Davis, “A statistical approach for real-time robust background subtraction and shadow detection”, IEEE ICCV’99, Frame-Rate Workshop, Corfu, Greece, 1999.

[27] A. Elgammal, D. Harwood, and L. Davis, “Non-parametric model for background subtraction”, Proceeding of IEEE ICCV’99 Frame-rate workshop, 1999.

[28] C. Stauffer and W. Grimson, “Learning patterns of activity using real-time tracking”, IEEE Transaction On Pattern Analysis and Machine Intelligence, Vol. 22, pp. 747-757, 2000.

[29] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, “Background modeling and subtraction by codebook construction”, IEEE Int. Conf. on Image Processing (ICIP), Vol. 5, pp. 3061-3064, 2004.

[30] F. E. Baf, T. Bouwmans, and B. Vachon, “A fuzzy approach for background subtraction”, IEEE Int. Conf. on Image Processing, pp. 2648-2651, 2008.

[31] N. A. Mandellos, I. Keramitsoglou, and C. T. Kiranoudis, “A background subtraction algorithm for detecting and tracking vehicles”, Expert Systems with Applications, Vol. 38, pp. 1619-1631, 2011.

[32] Y. Ding, W. Li, J. Fan, H. Yang, “A fuzzy background model for moving object detection”, IEEE Int. Conf. on Computer-Aided Design and Computer Graphics, pp. 610-613, 2009.

[33] T. Ojala, M. Pietikäinen, and D. Harwood, “A Comparative study of texture measures with classification based on feature distributions”, Pattern Recognition, Vol. 29, No. 1, pp. 51-59, 1996.

[34] M. Pietikäinen, T. Ojala, and Z. Xu, “Rotation-Invariant texture classification using feature distributions”, Pattern Recognition, Vol. 33, pp. 43-52, 2000.

[35] T. Ojala, M. Pietikäinen, and T. Mäenpää, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns”, IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 24, No. 7, pp. 971-987, 2002.

[36] H. Zhang, and D. Xu “Fusing Color and Texture Features for Background Model”, Third Int. Conf. on Fuzzy Systems and Knowledge Discovery, Vol. 4223, No. 7, pp. 887–893, 2006.

[37] W. Zhang, X.Z. Fang, and X.K. Yang, “Moving vehicles segmentation based on

Bayesian framework for Gaussian motion model", Pattern Recognition Letters, Vol. 27, No. 9, pp. 956-967, 2006.

[38] PETS2001 Dataset: ftp://ftp.cs.rdg.ac.uk/pub/PETS2001/DATASET1/.

[39] ATON Project Tested Dataset: http://cvrr.ucsd.edu/aton/shadow/ .

[40] L. Li, and W. Huang, "Statistical Modeling of Complex Background for Foreground Object Detection", IEEE Transactions on Image Processing, Vol. 13, No. 11, pp. 1459-1472, 2004.

[41] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 25, No. 5, pp. 564-567, 2003.

[42] S. Spor and R. Rabenstein, "A real-time face tracker for color video", IEEE Int. Conf. on Acoustics, Speech and Signal Processing, Vol. 3, pp. 1493-1496, 2001.

[43] D. Koller, J. Weber, and J. Malik, "Towards real time visual based tracking in cluttered traffic scenes", Proc. of the Intelligent Vehicles '94 Symposium, pp. 201-206, 1994.

[44] I. Mikic, M. Trivedi, E. Hunter, and P. Cosman, "Human body model acquisition and tracking using voxel data", Int. Journal of Computer Vision, Vol. 53, No. 3, pp. 199-223, 2003.

[45] J. J. LaViola, "A comparison of unscented and extended Kalman filtering for estimating quaternion motion", Proc. of American Control Conf., Vol. 3, pp. 2435-2440, 2003.

[46] A. Doucet, N. D. Freitas, and N. Gordon, "Sequential Monte Carlo methods in practice", Springer-Verlag, New York, 2001.

[47] K. Numiaro, E. Koller-Meier, and L. V. Gool, "An adaptive color-based particle filter", Image and Vision Computing, Vol. 21, No. 1, pp. 99-110,2003.

[48] K. Numiaro, E. Koller-Meier, and L. V. Gool, "Object tracking with an adaptive color-based particle filter", Pattern Recognition 24th DAGM Symposium 2002, Lect. Notes Computer Science, Vol. 2449, pp. 353-360, 2002.

[49] Y. Rui and Y. Chen, "Better proposal distribution object tracking using unscented particle filter", IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, Vol. 2, pp. 786-793, 2001.

[50] E. Maggio and A. Cavallaro, "Multi-part target representation for color tracking",

Proc. IEEE Int. Conf. on Image Processing, Vol. 2, pp. 729-731, 2005.

[51] C. Shen, A. V. D. Hengel, and A. Dick, "Probabilistic multiple cue integration for particle filter based tracking", Proc.7 th Int. Conf. Digital Image Computing, pp. 309-408, 2003.

[52] P. Perez, J. Vermaak, and A. Blake, "Data fusion for visual tracking with particles", Proc. IEEE, Vol. 92, No. 3, pp. 495-513, 2004.

[53] P. Brasnett, L. Mihaylova, D. Bull, and N. Canagarajah, "Sequential Monte Carlo tracking by fusing multiple features in video sequences", Image and Vision Computing, Vol. 25, No. 8, pp. 1217-1227, 2007.

[54] P. Wu, L. Kong, F. Zhao, and X. Li, "Particle filter tracking based on color and SIFT features", Int. Conf. on Audio, Language and Image Processing, pp. 932-937, 2008.

[55] Q. Zhang, T. Rui, H. Fang, and J. Zhang, "Particle filter object tracking based on Harris-SIFT feature matching", Procedia Engineering, Vol. 29, pp. 924-929, 2012.

[56] T. R. Bayes, "An Essay towards solving a Problem in the Doctrine of Chances", Philosophical Transactions of the Royal Society of London, Vol. 53, pp. 370-418, 1763 Reprinted in Biometrika, Vol. 45, 1958.

[57] J. M. Bernardo and A.F.M. Smith, "Bayesian Theory", Second edition, New York: Wiley, 1998.

[58] B. Ristic, S. Arulampalam and N. Gordon, "Beyond the Kalman Filter : Particle Filter for Tracking Applications", Artech House, 2004.

[59] J. D. Hol, T. B. Schon and F. Gustafsson, "On resampling algorithms for particle filters", Proc. of Nonlinear Statistical Signal Processing Workshop (NSSPW), pp. 79-82. 2006.

[60] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift", Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition, Vol. 2, pp. 142–149, 2000.

[61] F. Aherne, N. Thacker, and P. Rockett, "The Bhattacharyya metric as an absolute similarity measure for frequency coded data", Kybernetika, Vol. 34, No. 4, pp. 363-368, 1998.

[62] T. Kailath, "The divergence and Bhattacharyya distance measures in signal selection", IEEE Transactions on Communication Technology, Vol. 15, No. 1, pp. 52–60, 1967.

[63] Y. T. Chen and C. S. Chen, “Fast human detection using a novel boosted cascading structure with meta stages”, IEEE Transactions on Image Processing, Vol. 17, No. 8, pp. 1452-1464, 2008.

[64] K. Levi and Y. Weiss, “Learning object detection from a small number of examples: The importance of good features”, Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition, Vol. 2, pp. 53–60, 2004.

[65] Z. Xu, P. Shi, and X. Xu, “Adaptive subclass discriminant analysis color space learning for visual tracking”, Proc. PCM 2008, Lect. Notes Computer Science, Vol. 5353, pp. 902-905, 2008.

[66] M. Heikkila and M. Pietikainen, “A texture-based method for modeling the background and detecting moving objects”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, No. 4, pp. 657-662, 2006.

[67] Y. Jin and F. Mokhtarian, “Data fusion for robust head tracking by particles”, Proc. 2nd Joint IEEE Int. Workshop on VS-PETS, pp. 33-40, 2005.

[68] R. T. Collins, Y. Liu, and M. Leordeanu, “Online selection of discriminative tracking features”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 27, No. 10, pp. 1631-1643, 2005.

[69] M. J. Swain and D. H. Ballard, “Color indexing”, Int. Journal of Computer Vision, Vol. 7, No. 1, pp. 11-32, 1991.

[70] http://www.elec.qmul.ac.uk/staffinfo/andrea/multi-feature.html.

[71] http://www.ces.clemson.edu/~stb/research/headtracker/seq/.

[72] D. Doermann and D. Mihalcik, “Tools and techniques for video performance evaluation”, Proc. IEEE Int. Conf. on Pattern Recognition, Barcelona, Spain, Vol. 4, pp. 167-170, 2000.

[73] H. Bay, T. Tuytelaars, and L.Van Gool, “SURF: speeded up robust features”, Computer Vision-ECCV 2006, Lecture Notes in Computer Science, Vol. 3951, pp. 404-417, 2006.

[74] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-Up Robust Features (SURF) ”, Computer Vision and Image Understanding, Vol. 110, pp. 346-359, 2008.

[75] T. Lindeberg, “Scale-space for discrete signals”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, No. 3, pp. 234-254,1990.

[76] M. Brown and D. Lowe, “Invariant features from interest point groups”, Proc. of

British Machine Vision Conf. (BMVC), pp. 253-262, 2002.

[77] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.27, No.10, pp. 1615-1630, 2005.

[78] M. A. Fishler and R. C. Boles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography”, Communications of the ACM, Vol. 24, No. 6, pp. 381-395, 1981.

[79] R. Hartley and A. Zisserman, “Multiple view geometry in computer vision, second edition”, Cambridge University Press, 2003.

[80] Q. Miao, G. Wang, C. Shi, X. Lin, and Z. Ruan, “A new framework for on-line object tracking based on SURF”, Pattern Recognition Letters, Vol.32, pp. 1564-1571, 2011.

[81] M. Grabner, H. Grabner, and H. Bischof, “Learning features for tracking”, IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1-8, 2007.

[82] S. Cheng and C. Hu, “Particle filter tracking algorithm based on multi-information fusion”, Int. Conf. on Information Engineering and Computer Science, pp. 1-4, 2009.

[83] PETS2000dataset: ftp://ftp.cs.rdg.ac.uk/pub/PETS2000/testimages/.

[84] R. L. Streit and T.E. Luginbuhl, “Probabilistic Multi-Hypothesis Tracking”, NUWC-NPT Technical Report 10 428, Naval Undersea Warfare Center, Newport, Rhode Island, 1995.

[85] D. Schulz, W. Burgard, D. Fox, and A.B. Cremers, “People tracking with a mobile robot using sample-based Joint Probabilistic Data Association Filters”, Int. Journal of Robotics Research, pp. 99-116, 2003.

[86] I.R. Goodman, R.P.S. Mahler, and H.T. Nguyen, “Mathematics of Data Fusion”, Kluwer Academic Publisher, 1997.

[87] R. P. S. Mahler, “An introduction to multisource-multitarget statistics and its applications”, Technical monograph, Lockheed Martin, 2000.

[88] G. Matheron, “Random sets and integral geometry”, J. Wiley, 1975.

[89] R. Mahler, “Global integrated data fusion”, Proc. Seventh Nat. Symposium On Sensor Fusion, 1, (Unclassified) Sandia National Laboratories, Albuquerque, ERIM Ann Arbor MI, pp. 187-199, 1994.

[90] D.J. Daley and D. Vere-Jones, “An introduction to the theory of point processes”,

Springer, 1988.

[91] R. Mahler, "A theoretical foundation for the Stein-Winter Probability hypothesis density multitarget tracking approach", Proc. 2000 MSS Nat'l Symposium on Sensor and Data Fusion, Vol. I (Unclassified), 2000.

[92] B. Vo, S. Singh, and A. Doucet, "Sequential Monte Carlo methods for multi-target filter with Random Finite Sets", IEEE Transactions on Aerospace and Electronic Systems, Vol. 41,No.4,pp. 1224-1245, 2005.

[93] B. Vo and W. K. Ma, "The Gaussian Mixture Probability Hypothesis Density Filter", IEEE Transactions on Signal Processing, Vol. 54, No. 11, pp. 4091-4104, 2006.

[94] S. Blackman," Multiple target tracking with radar applications", Artech House, MA, 1986.

[95] B. Vo, S. Singh, and A. Doucet, "Sequential Monte Carlo methods for multi-garget filtering with random finite sets", IEEE Transactions on Aerospace and Electronic Systems, Vol. 41, No. 4, pp. 1224-1245, 2005.

[96] B. Vo, S. Singh, and A. Doucet, "Sequential Monte Carlo implementation of the PHD filter for multi-target tracking", Proc. 6 th Int. Conf. on Information Fusion, Vairns, Australia, pp. 792-799, 2003.

[97] K. Panta, B. Vo, and S. Singh, "Improved probability hypothesis density filter for multi-target tracking", Proc. Third Int. Conf. on Intelligent Sensing and Information Processing, pp. 213-218, 2005.

[98] K. Panta, B. Vo, and S. Singh, "Novel data association schemes for the probability hypothesis density filter", IEEE Transactions on Aerospace and Electronic Systems, Vol. 43, No.2, pp. 556-570, 2007.

[99] H. Sidenbladh, "Multi-target particle filtering for the probability hypothesis density", In Proc. Of Fusion, pp. 1110-1117, 2003.

[100] B. Vo and W. K. Ma, "A close-form solution for the probability hypothesis density filter", Proc. of 7 th Int. Conf. on Information Fusion, Vol. 2, pp. 856-863, 2005.

[101] D. Schuhmacher, B. T. Vo and B. N.V o, "A consistent metric for performance evaluation of multi-object filters", IEEE Transactions on Signal Processing, Vol.56, No. 8, Part 1, pp. 3447-3457, 2008.

[102] D. Schuhmacher and A. Xia, "A new metric between point process distributions", Submitted, Aug. 2007, available online at http:// arxiv.org/abs/0708.2777.

[103] http://ja.wikipedia.org/wiki/HSV_cone.jpg

[104] Z. Wang and G. J. Klir, “Fuzzy measure theory”, Plenum, 1992.

Publications of Author

Published Papers and International Conferences

[1] Xiaofeng Lu, Takashi Izumi, Lin Teng, Tadahiro Horie, Lei Wang, “A Novel Background Subtraction Method for Moving Vehicle Detection,” IEEJ Transactions on Fundamentals and Materials, Vol. 132, No. 10, pp. 857-863, 2012.

[2] Xiaofeng Lu, Takashi Izumi, Lin Teng, Lei Wang, “A adaptive multiple-feature fusion for moving-object tracking using particle filter,” Int. Conf. on Instrumentation, Control, Information Technology and System Integration, SICE Annual Conf., pp. 1649-1656, 2013.

[3] Lin Teng, Takashi Izumi, Xiaofeng Lu, Fumio Wakui, “A Method of the Optimum Route Search by Fuzzy-AHP”, IEEJ Transactions on Electronics, Information and Systems, Vol. 133, No. 6, pp. 1269-1276, 2013.

[4] Lin Teng, Takashi Izumi, Xiaofeng Lu, “ A new proposal to reflect preferences into route search for individual driver,” Int. Conf. on Instrumentation, Control, Information Technology and System Integration, SICE Annual Conf., pp. 753-756, 2013.

[5] Xiaofeng Lu, Takashi Izumi, Lin Teng, Lei Wang, “Particle filter vehicle tracking based on SURF feature matching”, IEEJ Journal of Industry Applications. (Accepted)

[6] Xiaofeng Lu, Takashi Izumi, Lin Teng, Lei Wang, “Multiple visual targets tracking via probability hypothesis density filter and feature measurement”, IEEJ Transaction on Electrical and Electronic Engineering. (Submitted)

[7] Xiaofeng Lu, Takashi Izumi, Tomoaki Takahashi, Lei Wang, “Moving vehicle detection based on fuzzy background subtraction”, the 2014 IEEE Int. Conf. on Fuzzy Systems. (Submitted)

Academic Society Meetings and Symposia

[1] Xing Ye, Yuki Ono, Xiaofeng Lu, and Takashi Izumi, “Traffic flow analysis based on video image examination of the number measurement of vehicles”, Technical Meeting of Chiba sub-branch, Tokyo branch, IEE of Japan, 4-7, 2011.

[2] Tadahiro Horie, Yuki Ono, Xiaofeng Lu, and Takashi Izumi, “Extraction of preceding vehicles based on shadow in the vehicle front image—Vehicle tracking by particle filter”, IEE-Japan Industry Applications Society Conf., Y-109, 2011.

[3] Lin Teng, Xiaofeng Lu, and Takashi Izumi, “Optimal route search for driver—Introduction of AHP sensibility evaluation”, IEE-Japan Industry Applications Society Conf., 2-62, 2011.

[4] Lin Teng, Xiaofeng Lu, and Takashi Izumi, “Extraction of optimal route search for driver by AHP”, IEE-Japan Technical Meeting on Intelligent Transport Systems, ITS-11-17, pp. 5-8, 2011.

[5] Lin Teng, Xiaofeng Lu, and Takashi Izumi, “Extraction of Optimal route search for driver: Approach of preference by Fuzzy-AHP”, The 27th Symposium of Fuzzy system, Vol. 27, pp. 811-814, TG1-4, 2011.

[6] Yuki Ono, Xiaofeng Lu, and Takashi Izumi, “Extraction of preceding vehicles based on shadow in the vehicle front view—Improvement of method of area limitation by white line”, IEE-Japan Industry Applications Society Conf., Y-110, 2011.

[7] Xiaofeng Lu, Tadahiro Horie, Takashi Izumi, Lin Teng, “A Novel Background Subtraction Method for Moving Vehicle Detection”, IEE-Japan Technical Meeting on Intelligent Transport Systems, ITS-12-24, pp.47-50, 2012.

[8] Xiaofeng Lu, Takashi Izumi, Lin Teng，and Tadahiro Horie, “A Novel Background Subtraction Method for Moving Vehicle Detection”, IEE Japan Joint Technical Meeting on Light Application and Visual Science and Instrumentation and Measurement. LAV-12-008, IM-12-016, pp.35-40, 2012.

[9] Xiaofeng Lu, Takashi Izumi, “A novel background subtraction for moving vehicle detection”, Technical Meeting of Chiba sub-branch, Tokyo branch, IEE of Japan,1-1, 2012.

[10] Lin Teng, Takashi Izumi, and Xiaofeng Lu, “Study of driver optimal route search by Fuzzy-AHP”, The 11th Symposium of Intelligent Transport Systems of Japan, 2-B-03, 2012.

[11] Tadahiro Horie, Kazuki Matsubara, Xiaofeng Lu, and Takashi Izumi, “Preceding vehicle tracking by adapting a particle filter in the image processing”, IEE-Japan Industry Applications Society Conf., Y-130, 2012.

[12] Xiaofeng Lu, Takashi Izumi, Hiroto Seki, and Tomoaki Takahashi, “A

consideration of moving object tracking using particle filter", IEE-Japan Technical Meeting on Intelligent Transport Systems, ITS-13-039, TER-13-065, pp. 53-56, 2013.

[13] Tadahiro Horie, Xiaofeng Lu, and Takashi Izumi, "Detection of preceding vehicles based on shadow in the vehicle front view--Reduction of shadow false detection under the vehicles by vehicles classifier", The 2013 Annual Meeting of the Institute of Electrical Engineers of Japan, 4-208, 2013.

[14] Lin Teng, Takashi Izumi, Kazuaki Hiwatashi, and Xiaofeng Lu, "Driver preference route search by Fuzzy-AHP--Study of route search using the DRM-DB", IEE-Japan Industry Applications Society Conf., 4-16, 2013.

[15] Satoru Ishibashi, Xiaofeng Lu, Hiroto Seki, Takashi Izumi, "Detection of Preceding Vehicles in the Vehicles Front View--Optimization of Edge Superimposed Using Multi-buffer", Technical Meeting of Chiba sub-branch, Tokyo branch, IEE of Japan, 4-7, 2013.

APPENDIX A

For the proposed second method, different type public datasets were adopted to evaluate the effectiveness of our method. A first head target (H1) was from a public dataset [70], and another head target (H2) was part of an in-house dataset [71]. The challenging factors include clutter, translation, rotation, partial occlusion, and scale changes.

The H1 sequence underwent translation, rotation, and partial occlusion in the cluttered background. The final tracking result is illustrated in Figure A.1, in which four representative frames (29, 84, 103, and 130) are shown. Our algorithm performed well. The evaluation results of these algorithms are shown in Figure A.3 (a).

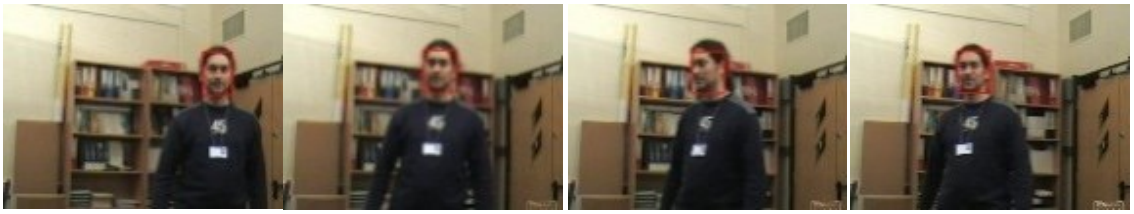


Figure A.1 Tracking results of H1 sequence.

In the second experiment, we used the H2 sequence to compare the performance of our tracker with the other trackers in handling translation, partial occlusion, and scale changes. The tracking results are illustrated in Figure A.2, in which four representative frames (400, 435, 466, and 480) are shown, where rows 1, 2, 3 and 4 correspond to Edge, FW, AMF-MS, and our tracker, respectively. We can see that our proposed tracker performed well throughout the sequence. The Edge tracker drifted from the target quickly, and the FW and AMF-MS trackers both had larger tracking errors under the partial occlusion. The evaluation results of these methods are shown in Figure A.3 (b). The evaluation curves accurately reflect the tracking performance in the case where partial occlusion occurs.

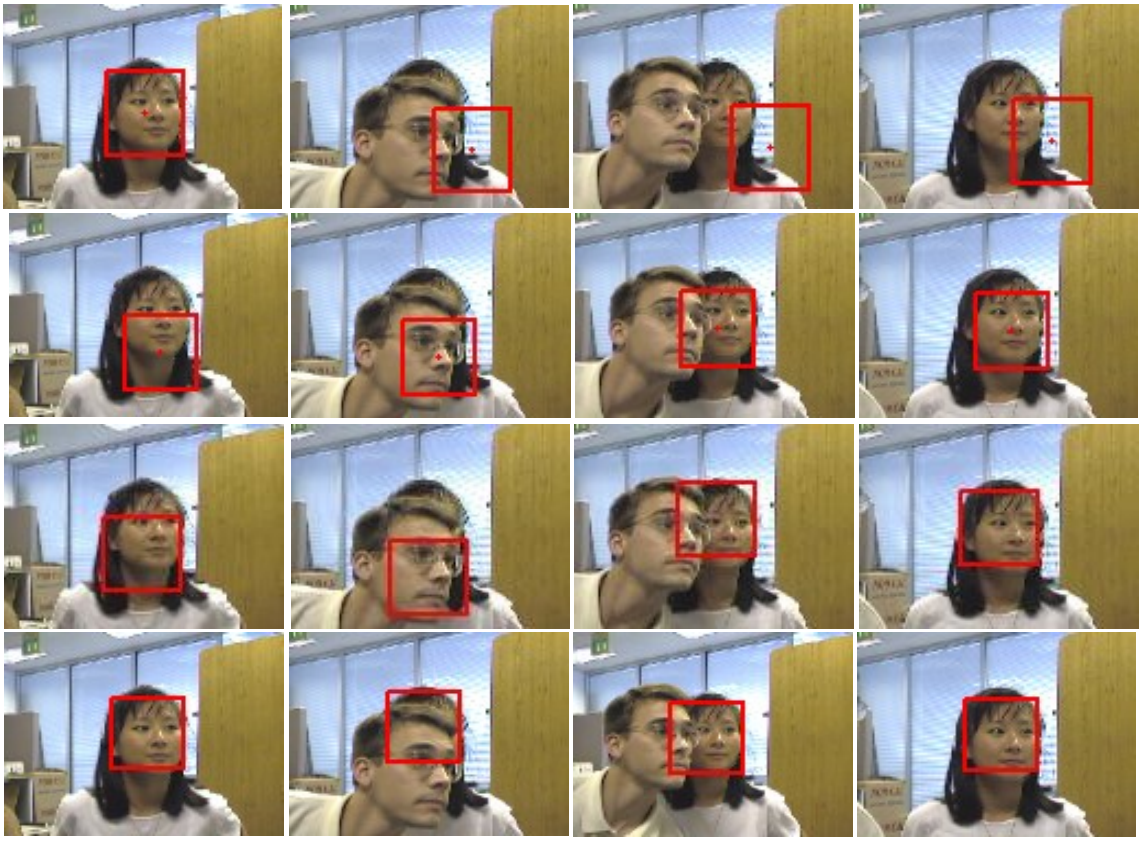
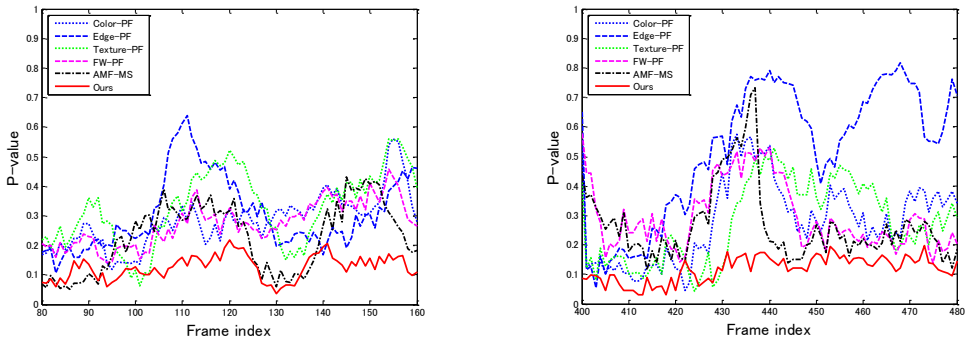


Figure A.2 Tracking results of H2 sequence. Rows 1, 2, 3 and 4 correspond to Edge, FW, AMF-MS, and our tracker, respectively (frames 400, 435, 466, and 480).



(a) (b)

Figure A.3 Evaluation results. (a) H1 (from frame 80 to frame 160); (b) H2 (from frame 400 to frame 480);

In our proposed tracker, we used a multiplication rule to balance the effects of similarity and discriminability, and we calculated the optimization weights of the three

features. The weight curves of the three individual features are plotted in Figure A.4. In the H1 sequence, starting from frame 100 when the head turned to the right, the edge weight gradually increased, and from frame 128 when the head moved out of the cluttered background of the bookshelves, the edge weight gradually decreased. The texture feature was more reliable than the color feature.

Table A.1 The average tracking errors (in pixels) of the compared methods.

	Color-PF	Edge-PF	Texture-PF	FW-PF	AMF-MS	Ours
H1	11.1	9.1	14.1	10.3	8.1	4.3
H2	8.0	17.6	8.3	8.2	7.0	3.7

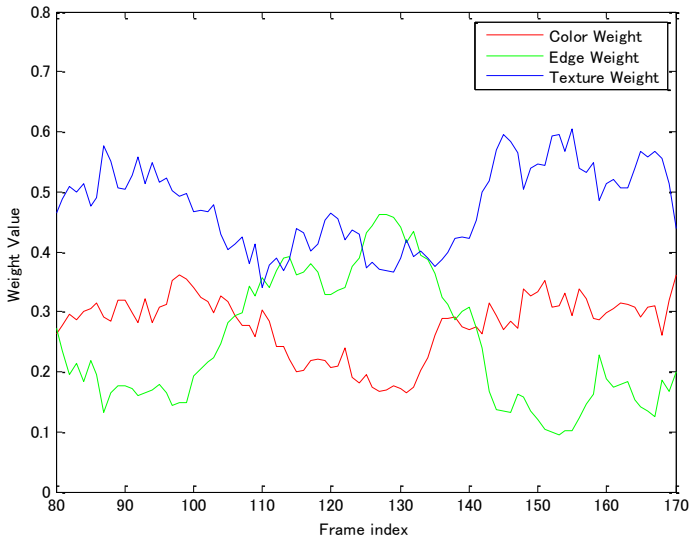


Figure A.4 The weight curves of our method for the H1 sequence.