Proceedings of the IEEE
International Conference on Automation and Logistics
Shenyang, China August 2009

Research on Traffic Moving Object Detection, Tracking and Track-Generating

Shanming Lin and Jun Tang
Computer and Information Engineering College
Hohai University
Changzhou, jiangsu Province, China
linsm@hhuc.edu.cn

Xuewu Zhang and Yanyun Lv
Computer and Information Engineering College
Hohai University
Changzhou, jiangsu Province, China
zhangxw@hhu.edu.cn

***Abstract* - This paper presents algorithms for vision-based detection, tracking and trajectory generation of vehicles in simple video sequences of traffic scenes which are registered by a stationary camera. Processing is done by three steps: improved vehicle detection method, faster tracking algorithm and new vehicle trajectory generation technique. Vehicles are viewed as rectangular blocks with certain active action. The propounded method is based on the correspondences between regions and vehicles, as the vehicles moving through the video sequence. The vehicle trajectory generation technique, used for gathering statistical traffic data for analysis in vehicle flow detection, is also described in detail. Experimental results from highway scenes demonstrate the effectiveness of the vehicle detection and tracking method.**

***Index Terms* - Object detection, Object tracking, Trajectory generating, Vehicle flow detection.**

I. Introduction

The primary task of moving vehicles detection and tracking is to detect moving target from video images. The outline of the target and background can be separately extracted by making use of spatiotemporal redundant information. Video tracking is mean to match the outline of the target in the following video frame. Clearly, the quality of video tracking has a significant influence on the effect of the next image module. With in-depth research in this field, there are more and more algorithms that are applied to moving vehicle detection and tracking. There are three conventional approaches to automated detect moving target: temporal differencing (two-frame or three-frame)[Anderson et al.,1985]; optical flow [Barron et al.,1994] and background subtraction [Wren et al.,1997]. Temporal differencing is very adaptive to dynamic environments, but generally does a poor job of extracting all relevant feature pixels. Optical flow can be used to detect independently moving objects in the presence of camera motion; however, most optical flow computation methods are very complex and are inapplicable to real-time algorithms without specialize hardware. Background subtraction supplies the most complete feature data, but is extremely sensitive to dynamic scene changes due to illumination and external events. The approach proposed here is supposed to make background subtraction more powerful to surrounding dynamism. The keystone viewpoint is to maintain a developmental statistical prototype of the background to provide a mechanism that adapts to slow changes in the environment. This single Gaussian model was used in [1]. However, pixel values often have complex distributions and more elaborate models are needed. Gaussian mixture model (GMM) was proposed for background subtraction in [2]. But the method required establishing model for each pixel, and at the same time a large number of parameters have to be updated. We present here an improved algorithm base on RGB color distribution. Not only the parameters but also the number of components in the mixture model constantly adapt for each pixel. By choosing the number of components for each pixel in an on-line procedure, the algorithm can adapt to the scene automatically and fully.

As to the tracking algorithm, earlier results were based on the famous Kalman filtering [3], which can obtain optimal solution in the case of linear dynamics and Gaussian noise. Unfortunately, very few practical visual tracking problems belong to this case. For nonlinear or non-Gaussian problems, it is impossible to estimate the probability distribution analytically and many algorithms have been proposed to approximate them. The particle filter [4], also known as sequential Monte Carlo or Condensation, is the most popular approach which recursively constructs the posterior probability distribution function of the state space using Monte Carlo integration. However, there are high computational demands in the approach, and this is the shortcoming to apply particle filtering in real-time systems. Mean shift [5], a typical and popular method, is a robust non-parametric method for climbing density gradients to discover the upmost of probability distributions. Mean shift explores the local energy landscape, using only a single tentative. This approach is computational effective, but it is accessible to converge to local top-flight. In this paper, we proposed a new tracking algorithm 'the Mean Shift Embedded Particle Filter' (MSEPF) to integrate advantages of the above two methods. The MSEPF leads to more efficient sampling by shifting samples to their neighbouring modes, conquering the degeneracy problem, and requires fewer particles to maintain excellent performance, resulting in low computational load. Then, adding the trajectory generation module, we can detect the majority of traffic flow data from the module data. The paper is structured as follows: we begin Chapter II with a description of the background of mixed Gaussian extraction principle in detail, followed by the results of experiments charts and images. Improved particle-filter method for studying and testing will be indicated in chapter III properly. In Chapter IV, the main points of the path generation module and simulation results will be presented .The conclusion for this paper will be given in Chapter V.

II. Gaussian Mixture Background extraction

If each pixel resulted from a particular exterior under particular illumination, a single Gaussian would be enough to model the pixel value while accounting for remove noise. If only lighting changed over time, a single, adaptive Gaussian per pixel would be sufficient. In practice, multiple surfaces often emerge in the view frustum of a particular pixel and the change of illumination conditions. Thus, multiple, adaptive Gaussians are necessary. We use a mixture of adaptive Gaussians to estimate this course.

We consider the values of a particular pixel over time as a "pixel process". The "pixel process" is a time series of pixel values, e.g. scalars for gray values or vectors for colour images. At any time, a particular pixel, $\{x_0, y_0\}$ is its history,

$$\{X_1, \ldots X_t\} = \{I(x_0, y_0, i), 1 \le i \le t\}. \qquad (1)$$

I was defined as the image frame-sequence. The value of each pixel represents a measurement of the radiance in the direction. With a quiet background and static illumination, that value of change would be relatively small. If we imagine that independently, Gaussian noise is incurred in the sampling process; its density can be described by a single Gaussian distribution centred at the mean pixel value. Unfortunately, the most live video frames include illumination changes, situation changes, and moving objects.

If illumination alteration occurred in a static scene, it would be necessary for the Gaussian to track those changes. If a static object was increased to the scene and was not incorporated into the background until it had been there longer than the last object, the corresponding pixels could be advised as foreground for discretionarily long periods. This could lead to accumulated errors in the foreground estimation, resulting in poor tracking behaviour. An additional aspect of variation occurs if moving objects are present in the scene. Even a relatively consistently colour moving object is generally expected to produce more variance than a "static" object. Also, in short, there must be more data to maintain the background distributions because the pixel values for different objects are often not in the same color.

For the sake of brevity, supposing each pixel value in the time t has been expressed as: $X_t = \left(x_t^r, x_t^g, x_t^b\right)$. In this paper, we record the change of pixel values in a certain time, the pixel histogram is shown below, and we can see its pixels satisfy the Gaussian distribution. (This assumes that the red, green and blue pixel values are independent and have the same variances.)

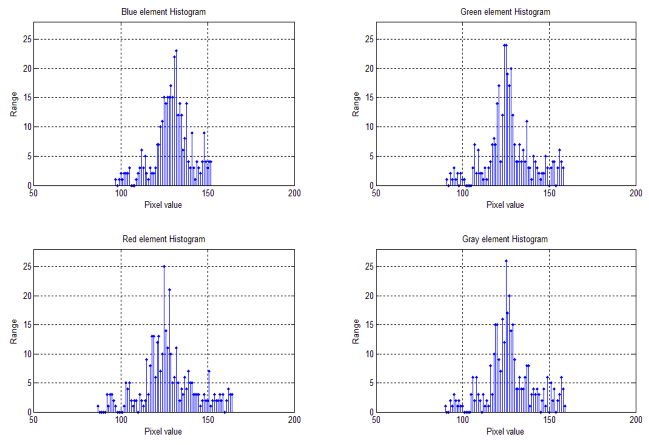


Fig.1 the pixel histogram of one point located at (150,150) in about 340 background frames (a) Blue element histogram (b) Green element histogram (c) Red element histogram (d) Gray element histogram

The recent history of each pixel $\{X_1, \ldots X_t\}$ is modelled by a mixture of K Gaussian distributions. The probability of observing the current pixel value is:

$$f(X_t) = \sum_{i=1}^{K} \tau_{i,t} * \eta\left(x, \mu_{i,t}, \sum\nolimits_{i,t}\right). \qquad (2)$$

Where K is the number of distributions, $\tau_{i,t}$ is an estimate of the weight of the i^{th} Gaussian in the mixture at time t, $\mu_{i,t}$ is the mean value of the i^{th} Gaussian in the mixture at time t, $\sum_{i,t}$ is the covariance matrix of the i^{th} Gaussian in the mixture at time t, and where $\eta\left(x, \mu_{i,t}, \sum_{i,t}\right)$ is a Gaussian probability density function:

$$\eta\left(x, \mu_{i,t}, \sum\nolimits_{i,t}\right) = \frac{1}{(2\pi)^{\frac{n}{2}} \left|\sum_{i,t}\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_t - \mu_{i,t})^T \sum_{i,t}^{-1}(x_t - \eta_{i,t})} \qquad (3)$$

K is determined by the available memory and computational power. Currently, the data from 3 to 6 are used. Also, for computational reasons, the covariance matrix is assumed to be of the following form:

$$\sum\nolimits_{i,t} = \sigma_{i,t}^2 I. \qquad (4)$$

That is to say, establish a one-dimensional Gaussian mixture model for each color. For each input pixel value, if

$$\left|I_t - \mu_{i,t-1}\right| \le D * \sigma_{i,t-1}. \qquad (5)$$

Where $\mu_{i,t-1}$, D and $\sigma_{i,t-1}$ represent mean value, parameter and standard variance respectively. If the match between I_t and the Gaussian mixture model function is good, its parameters are updated by the following formula:

$$\begin{cases} \tau_{i,t} = (1-\alpha)\tau_{i,t-1} + \alpha \\ \mu_{i,t} = (1-\rho)\mu_{i,t-1} + \rho I_t \\ \sigma_{i,t}^2 = (1-\rho)\sigma_{i,t-1}^2 + \rho(I_t - \mu_{i,t})^2 \end{cases} . \quad (6)$$

Where α is customized as learning rate, $0 \le \alpha \le 1$, ρ is the parameter for learning rate .

If none of the K distributions match the current pixel value, the least probable distribution is replaced with a distribution with the current value as its mean value, an initially high variance, and low prior weight. The prior weights of the K distributions $\tau_{i,t}$ are adjusted as follows:

$$\tau_{i,t} = (1-\alpha)\tau_{i,t-1} . \quad (7)$$

Finally, all the normalized weights were lined up, according to the Gaussian distribution from small to large order. The larger has a smaller variance with the emergence of a greater probability which reflect the distribution characteristics of pixel in the background scene. Because the probability of one pixel regarded as "background state" is usually bigger than the "state of foreground". If the absolute difference between I_t and the mean value of each background distribution is D times bigger than the standard variance of the distribution, then I_t was considered as a foreground pixel, else it is labelled as background pixels.

III. Improved Particle Filter Tracking Method

The Kalman filter provides an effective solution to the linear Gaussian filtering problem. However, where there is nonlinearity, either in the model specification or the observation process, therefore, other methods are required. Methods known popularly as "particle filters" are considered. These include the condensation algorithm and the Bayesian bootstrap or sampling importance re-sampling (SIR) filter. These filters represent the posterior distribution of the state variables by a system of particles which evolves and adapts recursively as new information becomes available.

In order to develop the details of the algorithm, let $\{X_K^i, w_K^i\}_{i=1}^{N_s}$ denote a random measure that characterizes the posterior distribution function $p(x_K \mid z_{1:K})$.

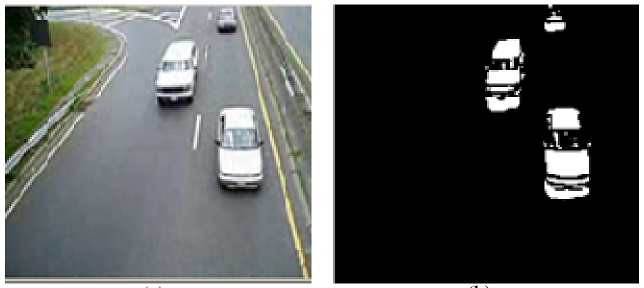

(a) (b)

Fig.2 Background extraction method (a) current image (b) foreground image

Where $\{X_K^i, i=1,\dots N_s\}$ is a set of support points with associated weights $\{w_K^i, i=1,\dots N_s\}$. The weights are normalized as: $\sum_{i=1}^{N_s} w_K^i = 1$ and $\{X_K^i, i=1,\dots N_s\}$ is the set of all states up to time K.

Suppose $p(x) \propto q(x)$ is a probability density which is difficult to draw samples but for $q(x)$ can be evaluated. In addition, $q(x)$ is an important density function that could be easily generated from sampling. According to Bayesian theory:

$$q(X_K \mid Z_{1:K}) = q(X_K \mid X_{K-1}, Z_{1:K})\, q(X_{K-1} \mid Z_{1:K-1}) . \quad (8)$$

The weight value can be written as:

$$w_K^i = w_{K-1}^i \frac{p(Z_K \mid X_K^i)\, p(X_K^i \mid X_{K-1}^i)}{q(X_K^i \mid X_{K-1}^i, Z_{1:K})} . \quad (9)$$

Weights can be normalized as:

$$\bar{w}_K^i = \frac{w_K^i}{\sum_{i=1}^{N_s} w_K^i} . \quad (10)$$

Furthermore, if $q(X_K \mid X_{K-1}, Z_{1:K}) = q(X_K \mid X_{K-1}, Z_K)$, then the importance density becomes only dependent on X_{K-1} and Z_K. In such scenarios, only X_K^i need to be stored, therefore, one can discard the path X_{K-1}^i and history of observations $Z_{1:K-1}$. The modified weight is then:

$$w_K^i = w_{K-1}^i \frac{p(Z_K \mid X_K^i)\, p(X_K^i \mid X_{K-1}^i)}{q(X_K^i \mid X_{K-1}^i, Z_K)} . \quad (11)$$

and the posterior filtered density can be approximated as:

$$p(X_K \mid Z_{1:K}) \approx \sum_{i=1}^{N_s} \bar{w}_K^i \delta(X_K - X_K^i) . \quad (12)$$

A common problem with conventional particle filters is the degeneracy phenomenon. After a few iterations, one sample will have small weight. This means most samples may have very low probability, and their contribution to the posterior estimation becomes negligible. So a large of computational effort is wasted. The brute measure to handle the problem is to use a very large number of samples. In order to overcome the degeneracy problem, we proposed a new tracking algorithm, the Mean Shift Embedded Particle Filter (MSEPF), which integrates advantages of the two methods.

The mean shift algorithm is a kernel based way for efficient object tracking. Kernel function $H(\bullet)$ is regarded as probability density function if it meets certain statistical moment restrictions which can be used for non-parametric probability density estimates. If the sample sets are obtained from density function $f(x)$ after the N times independently sample, then the density estimation function can be defined as:

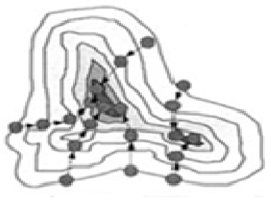
$$\hat{f}(x) = \sum_i w_i H(x - x_i) . \quad (13)$$

The weight coefficient w_i meet the restriction condition: $\sum_i w_i = 1$. The mean-shift vector can be expressed as:

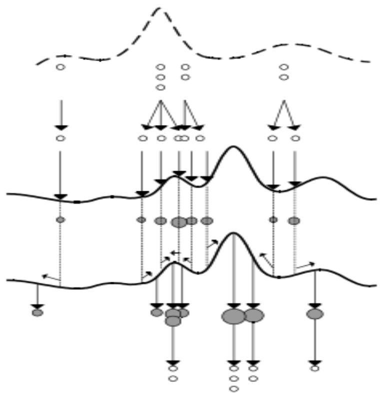
$$m(x) - x = \frac{\sum_i w_i k(x - x_i) x_i}{\sum_i w_i k(x - x_i)} - x . \quad (14)$$

Where $m(x)$ is the mean value of the sample x. In iterative process, data point moving to the mean of sample is called mean-shift algorithm. In the iterative process, sequence $\{x, m(x), m(m(x)), \ldots\}$ is named as the x trajectory. In brief, mean-shift method is continuously moving towards post-weighted sample mean, the target location is to meet the location of maxima (or local maxima).

In our paper, mean shift method is applied to each sample based on observation density. After mean shift iterations, samples are moved to have large weights, the algorithm concentrates on sample with large weights. So the degeneracy problem is efficiently overcome. At the same time, if the iteration time is set properly, the resultant samples will not contain too many repeated points, so the problem of impoverishment is also reduced. Meanwhile, many samples will converge to the same local maximum, so we can use fewer samples than the conventional particle filter. As we known, the computational load of particle filtering largely depends on the number of samples. So, the efficiency of particle filtering is improved greatly. Selective re-sampling also is adopted to prevent many samples from having small weights even after mean shift finding. A graphical representation of the MSPF is shown in Fig.3 (b).

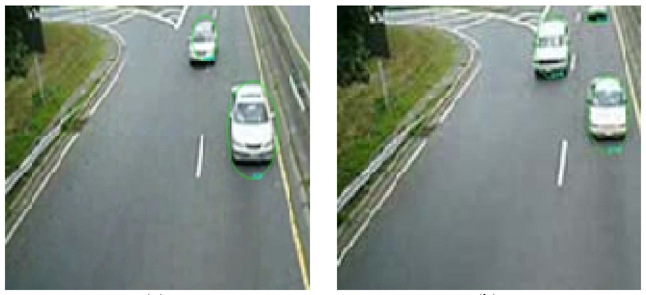


(a)



(b)

Fig.3 (a) Sketch map for mean-shift
(b) A graphical representation of MSPF method



(a) (b)

Fig.4 the results of tracking algorithm (a) 7th frame (b) 25th frame

IV. Trajectory generation and the application to traffic flux detection

With improved particle filter tracking algorithm to track the vehicle, we add a trajectory generation module.

The main steps of the algorithm are shown as follows:

1. Extracting the contour of moving vehicle from foreground image frame.
2. Judge the size of the contour, if too small, it can be removed as noise; else please go to the step 3.
3. Labeled those contours with different ID, then calculating the cancroids, width and length of those contours.
4. To predict the cancroids, measure the width and length of those contours in current frame with Kalman filter.
5. Making contrast between the predicted value and the calculated value, then do some modify to the generated trajectory.
6. Using image smoothing algorithm to deal with the trajectory, then saving ID of the contours and the trajectories to computer file.
7. If there is another contour, skip to the step 2, else end the procedure.

In this paper, the coordinate transformation between world coordinate system and computer image coordinate is necessary in video image processing. Because the computer images are two-dimensional imaging plane, pixel as the units of measurement, to the contrast, the world coordinate system can be seen as the actual three-dimensional space. As shown in Fig. 5(a).

The moving trajectory for the 5th vehicle is lay out in Fig.5(b), it records the course that the vehicle to be tracked from the beginning to the disappearing. From the figure, we can not only gain the average speed of each vehicle and instantaneous speed, but also the traffic capacity, etc. For example, firstly selecting a group of contour pixels without changing the shape of the characteristics in the video frames, then to make match with the pixels of similar characteristics in next frames. Thus, the moving distance of the vehicle can be obtained. So, the vehicle speed can be gotten from the two images frame-interval time. The particular algorithm is: to obtain the centric coordinates $(x_1, y_1), (x_2, y_2)$ of the target area in some time-interval, the moving distance D of the cent-

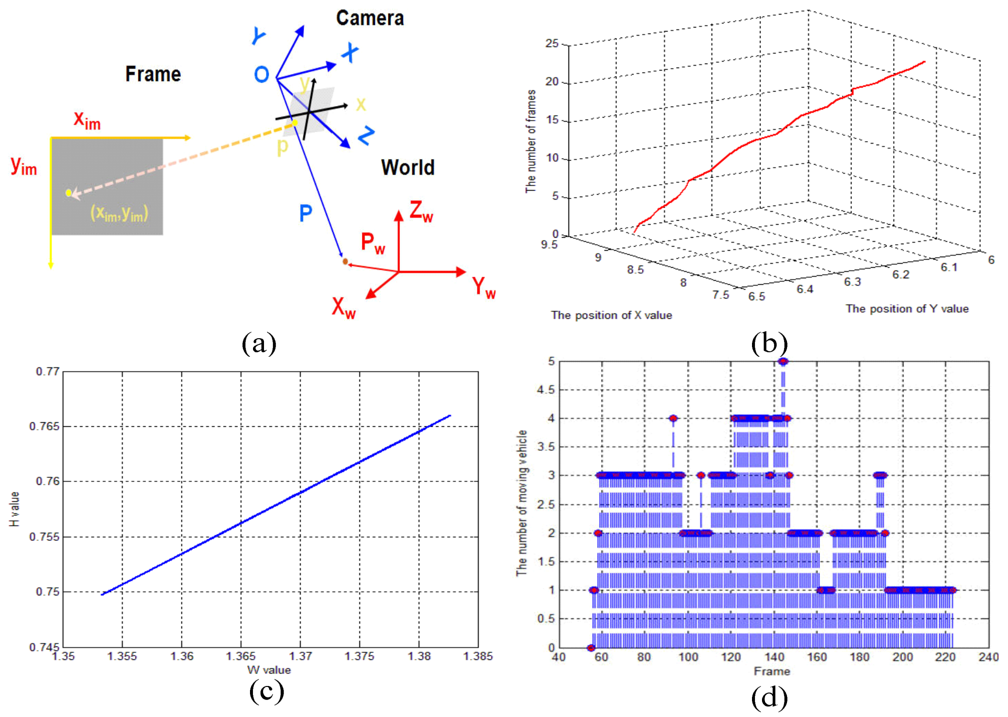


Fig.5 (a) Coordinate transformation between world and image (b) Vehicle moving trajectory (c) Length and width changes of the moving vehicle (d) Vehicle flow in frames

roid can be gotten from the difference between the above two points, then multiplied by the system initialization parameter (the coefficient of proportionality l between the pixel of computer image coordinates and actual path length of world coordinate system).The actual speed v can be gained if we use the actual path length L to divide the time interval Δt of two corresponding images.

$$D(x,y) = \{|x_1 - x_2|, |y_1 - y_2|\} . \tag{15}$$

$$L = l \bullet D . \tag{16}$$

$$v = \frac{L}{\Delta t} . \tag{17}$$

Fig.5(c) describes the process: the vehicle move from faraway to nearby in the video. Reflected in the two-dimensional coordinate system, is the course that length L and width W of objects change from small to big. However, during the process of the target moving from far to nearby in the video, L and W will not change too much. Through the transformation from computer image coordinate to world coordinate, actual vehicle's true length and width can be obtained. Based on the ratio of vehicle length and width, the type of vehicle can be identified. For example, large trucks (2.4m), medium vehicles (1.8m), small car (1.4m). Table 1 reflects the track record of multi-objective situation in some video frame. Fig.5(d) reflects the situation for the tracked vehicle in a certain frame; it indicated the num-ber of traveling vehicles in highway in a certain period of time, which can be used to detect traffic flow. In unit time, the traffic flow can be obtained according to the number of vehicles. In this way, we can get corresponding flow in different time interval. In addition, with the prior information, we can also calculate vehicle speed and the waiting time.

V. SUMMARY

In this paper, we presented a vehicle detection, tracking and trajectory generation system that is designed to operate in congested traffic. In order to make the system be less sensitive to illumination problem, we proposed a transcendental background extraction method. Meanwhile, we incorporated the mean-shift algorithm into the particle-filter and enable it to track an object even when vehicle overlapping occurs; this method has the advantages of small computational load and little response time. In addition, the trajectory generating module can provide some other information about the vehicle, such as trajectory, position and size. To analyze trajectory, we can detect the vast majority of traffic flow data, including traffic flow, vehicle speed, etc. Initial experimental results from highway scenes were presented. Future works will focus on analyzing the traffic scene under heavy load.

Table 1
THE BEGINNING AND END FRAMES FOR THE TRACKED CAR

CAR	BEGIN	END	CAR	BEGIN	END
Car01	56	93	Car07	122	146
Car02	58	97	Car08	140	161
Car03	59	107	Car09	144	173
Car04	92	147	Car10	168	193
Car05	107	137	Car11	174	191
Car06	112	145	Car12	188	233

References

[1] C. Wren, A. Azabayejani, T. Darrell, "Real-time tracking of the human body", IEEE Trans. Pattern Analysis Machine Int.vol.19, no.7, pp.780-785, 1997.

[2] C. Stauffer, W. E. L. Grimson, "Adaptive background mixture models for real-time tracking", Computer Vision and Pattern Recognition, vol. 2, pp. 246-252, 1999.

[3] Grewal, M. S., Angus, P. A, "Kalman filtering and practice", NJ, SA: Prentice Hall, Upper Saddle River, 1993.

[4] S. Maskell and N. Gordon, "A Tutorial on Particle Filters for On-line Nonlinear/Non-Gaussian Bayesian Tracking", in Proc. IEEE Workshop" Target Tracking: Algorithms and Applications, pp.5-28, 2001.

[5] D. Comaniciu and V. Ramesh, "Mean Shift and Optimal Prediction for Efficient Object Tracking", ICIP, Vancouver, pp.70-73, 2000.

[6] S.Gupte, O. Masoud, R.F.K Martin, N.P. Papanikolopoulos, "Detection and classification of vehicles", IEEE Trans. Intel. Transport Syst, vol.3, no.1, 2002.

[7] R. Rad, M. Jamzad, "Vehicle Tracking and Classification in Highways Using Region and Boundary Extraction", In: IASTED International conference on Visualization, Imaging and Image Processing, Benalmedena, Spain, Sept.8–10, 2003.

[8] Roberts, J. M, "Attentive Visual Tracking and Trajectory Estimation for Dynamic Scene Segmentation", Ph. D. thesis, University of Southampton, 1994.

[9] Dorin Comaniciu, et al, "Kernel-Based Object Tracking", IEEE Transactions on Pattern Analysis and Machine Intelligence,vol.25, no.5, May 2003.

[10]M. Isard and A. Blake, "CONDENSATION-Conditional Density Propagation for Visual Tracking", International Journal on Computer Vision, vol.129, no.1, pp.5-28, 1998.

[11]Jiang Gangyi, Yu Mei, Ye Xien, "New method of vision based vehicle tracking and traffic parameter estimation", Journal of Circuits and Systems, vol.6, no.4, pp.69-73, 2001.

[12]A. Doucet, N. Gordon, "Sequential Monte Carlo Methods in Practice", Springer-Verlag, New York, 2001.

[13]Caifeng Shan, Yucheng Wei, T ieniu Tan, "Real Time Hand Tracking by Combining Particle Filtering and Mean Shift", in: proc IEEE Sixth International Conference on Automatic Face and Gesture Recognition, pp. 669-674,2004.

[14]A. Rajagopalan and R. Chellappa, "Vehicle Detection and Tracking in Video", Proc. IEEE Int'l Conf. Image Processing, pp.351-354, 2000.

[15]A. Monnet, A. Mittal, N. Paragios and V. Ramesh, "Background Modeling and Subtraction of Dynamic Scenes", In Proceedings ICCV, pp.1305–1312,2003.