

Vehicle detection and tracking under various lighting conditions using a particle filter

Y.-M. Chan¹ S.-S. Huang² L.-C. Fu¹ P.-Y. Hsiao³ M.-F. Lo⁴

¹Department of Computer Science and Information Engineering, National Taiwan University, Taipei

²Department of Computer and Communication Engineering, National Kaohsiung First University of Science and Technology, NKFUST, Kaohsiung, Taiwan, TW 811

³Department of Electrical Engineering, National University of Kaohsiung, Kaohsiung University Road, Nan-Tzu District, Kaohsiung, Taiwan, TW 811

⁴Chung-Shan Institute of Science and Technology, Missile and Rocket Department, Rong'an 3rd St., Zhongli City, Taoyuan County, TW 325

E-mail: yimingchan@gmail.com

Abstract: The authors propose a vision-based automatic system to detect preceding vehicles on the highway under various lighting and different weather conditions. To adapt to different characteristics of vehicle appearance under various lighting conditions, four cues including underneath shadow, vertical edge, symmetry and taillight are fused for the vehicle detection. The authors achieve this goal by generating probability distribution of vehicle under particle filter framework through the processes of initial sampling, propagation, observation, cue fusion and evaluation. Unlike normal particle filter focusing on single target distribution in a state space, the authors detect multiple vehicles with a single particle filter through a high-level tracking strategy using clustering. In addition, the data-driven initial sampling technique helps the system detect new objects and prevent the multi-modal distribution from collapsing to the local maxima. Experiments demonstrate the effectiveness of the proposed system.

1 Introduction

Vision-based driver assistance systems (DAS) [1–4] have recently become popular because of the superior power of the environment description and the low price. The applications include lane recognition, night view system, obstacle avoidance etc. This study focuses on the vision-based DAS using the particle filter with a high-level tracking technique. Although the vehicle pattern is specific and regular, it is still hard to recognise them from natural scenes.

There are numerous approaches proposed for vehicle detection. Among the day time vehicle detection scenarios, there are three major categories: knowledge-based [5–7], appearance-based [2, 8–11] and template-based approaches [12, 13]. Knowledge-based approach, without the necessity of the training process, is more efficient and more general than template-based or appearance-based approaches; efficiency and generality are important in real-time applications. Thus, we choose a knowledge-based scheme.

Among the knowledge-based approaches, the intensity-based symmetry method [5–7] is often used in the car-following situation, as the rear of a vehicle is typically symmetrical. In the ARGO project, Broggi *et al.* [14] produced a symmetry map by combining the grey-level information and the horizontal-edge symmetry attributes. Then, the position of the vehicle's bottom is found by fitting a template to the edge map. This method is limited

to, however, the strict car-following situations or the symmetrical object. Besides, it may cause false alarms by the symmetrical objects, for example, fences.

In the low-light condition, different cues are used. Chern and Hou [15] proposed a monocular vision system on the highway at night. Wang *et al.* [16] proposed a vehicle detection system based on the taillight detection. The detection of taillights is through blob template of taillight in the night scene. However, the usage of single cue is not robust enough for vehicle detection under various lighting conditions. Accordingly, there are some approaches using multiple cues. Huang *et al.* [17] employed vertical edge, underneath shadow and symmetry as the prior knowledge of the vehicle. Tao and Debrunner [18] used sequential Monte Carlo to fuse line and colour features for the vehicle. Martin and Bernt [19] argued that applying multiple cue integration with democratic integration can improve robustness of visual tracking. The effectiveness of the democratic integration with CONDENSATION to achieve the goal of multi-hypothesis tracking is shown in [20].

Particle filter, known as CONDENSATION [20], is widely used in the areas of visual tracking [20–28]. It can adequately model the multi-modal and non-Gaussian distribution. It is not only easier to track multiple objects but it is also simpler to integrate multiple vision cues. We propose a way to integrate these features intelligently and effectively.

To achieve multiple target tracking, we need to track a multi-modal density distribution. There are three major

classes of approaches dealing with multi-target tracking in the literatures [21, 23]: (i) one particle filter for each target [23], (ii) single particle filter for all targets modelled in the state space [21] and (iii) single particle filter without correspondence modelling. One particle filter for each target is an intuitive approach, but often converges to a local maximum when two similar objects intersect. Modelling every target in the state space explicitly can help distinguish occluded targets. However, two issues make it improper for real-world cases; the number of targets is assumed to be constant and the state space grows exponentially with respect to the target number. Using one particle filter for all targets without correspondence modelling is an efficient solution; the number of particles and the size of the state space are less than those of the other two. All we need to do for this scheme is to solve data association of the targets in the multi-modal distribution. To extend such a particle filter approach to its multi-target tracking version, we cluster particles into several groups by the Basic Sequential Algorithm Scheme (BSAS) [29], an efficient clustering algorithm generating effective results. After clustering, we use a Kalman filter [30] with constant velocity model to accomplish the high-level tracking. The goal of the multi-target tracking is achieved without heavy computational burden.

The hereby proposed vehicle detection and tracking architecture is shown in Fig. 1. The detection and tracking algorithms are highly coupled using a particle filter framework and a high-level tracking module. First, we enhance the initial sampling for earlier convergence and use multiple cues to improve the robustness of vehicle detection. The data-driven initial sampling draws particles from high-likelihood area instead of randomly from the whole state space; it is important for fast convergence when a new target appears. The system can redistribute particles into multi-modal distribution; otherwise, multi-modal distribution may collapse into local maxima. Second, in order to apply this vehicle tracking algorithm under various lighting conditions, we use multiple cues for each candidate. These cues are integrated by the cue fusion procedure to obtain the final likelihood of the candidate. In

order to extend the particle filter into to accommodate multiple targets, the aforementioned BSAS is adopted to cluster the particles. After an iteration of the particle filter algorithm, the most probable candidate will propagate to the next frame through a high-level tracking technique.

The rest of the paper is organised as follows. Section 2 shows the first two processes, initial sampling and propagation, of our system. In Section 3, both observation and evaluation will be described. Section 4 shows the detection and tracking mechanisms of our system. Experiments are shown in Section 5. Finally, we conclude the paper in Section 6.

2 Particle filter and hypothesis generation

2.1 Introduction to particle filters

Particle filter, a non-parametric Bayesian filter, is generally used to generate the density of the state space through iterations using Bayes' rule under Markov assumptions. It can deal with general distributions or multi-modal distributions by relaxing the Gaussian assumptions. A general Bayesian filter maximises the posterior term using Bayes' rule as shown in the following equation

$$p(x_t|Z_t) = \eta p(z_t|x_t) \int p(x_t|x_{t-1})p(x_{t-1}|Z_{t-1})dx_{t-1} \quad (1)$$

The posterior term, $p(x_t|Z_t)$, is the conditional probability of state vector x_t given the measurement history $Z_t = Z_{t-1} \cup z_t$; the measurement at the time instance t is denoted as z_t and the history of the measurement from the beginning to the time instance $t - 1$ is denoted as $Z_{t-1} = \{z, \dots, z_{t-1}\}$. η is the normalisation constant. $p(z_t|x_t)$ is also known as the likelihood probability of the measurement z_t given the state vector x_t . $p(x_t|x_{t-1})$ is the state transition probability without knowing current measurement. Equation (1) can be taken into iterative steps for deriving the current state. In the particle filter framework, the particles 'evolve' to the best approximation distribution using Bayes' rule. Each particle

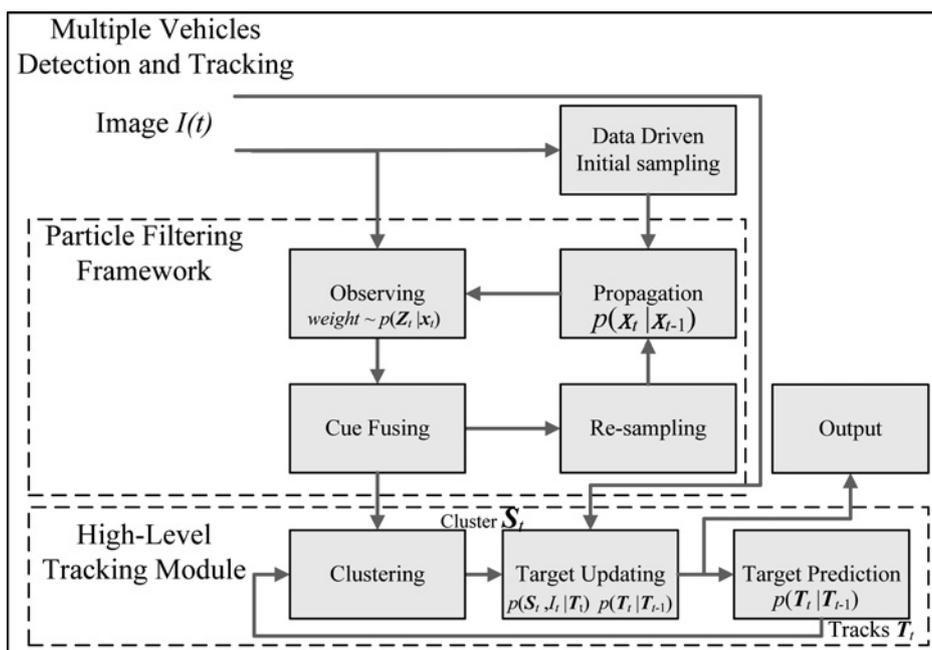


Fig. 1 Architecture of the vehicle detection and tracking procedure

is encoded with a state vector and its weight to represent a hypothesis with confidence. The re-sampling of the particle filter, generation of particles, is done by drawing particles according to their associated weights at the previous time instance; the higher the weight is, the higher the probability of ‘survival’ the particle is with.

If the number of particles is close to infinity, the final estimation will converge to the true posterior density. In general, the computational complexity of the particle filtering is linearly proportional to n , the number of particles. Owing to efficiency, we prefer not to use too many particles. In our experiments, the number is less than 2000.

Following the manner of a Bayesian filter, particle filter framework ‘observes’ and ‘propagates’ particles according to the probabilities $p(z_t|x_t)$ and $p(x_t|x_{t-1})$, respectively. The final state distribution is ‘estimated’ by the final particle state vectors and weights.

2.2 Data-driven initial sampling

These ‘particles’ in particle filters are all possible candidates of a vehicle. A vehicle is represented by a vector described as follows

$$\begin{aligned} \mathbf{x}_t^k &= (u_t, v_t, w_t, h_t, \Delta u_t, \Delta v_t, \Delta w_t)^t \\ (u_t, v_t): & (U, V) \text{ coordinates of the top left corner} \\ (w_t, h_t): & (\text{width, height}) \text{ of the sample} \\ (\Delta u_t, \Delta v_t, \Delta w_t) &= (u_t - u_{t-1}, v_t - v_{t-1}, w_t - w_{t-1}) \end{aligned} \quad (2)$$

where \mathbf{x}_t^k is the state vector of the k th vehicle candidate at time step t . The schematic description of these notations is shown in Fig. 2a. Without loss of generality, Δh_t is assumed to be proportional to Δw_t in the on-road vehicle cases and is hence omitted in our state vector. The measurement vector at time step t is defined as $z_t = (z_t^{\text{im}}, z_t^{\text{un}}, z_t^{\text{ve}}, z_t^{\text{tl}})^t$ which includes image z_t^{im} , vertical edge map z_t^{ve} , underneath shadow map z_t^{un} and taillight map z_t^{tl} . We use the Sobel edge detector with orientation constraint to generate vertical edge map, that is, only vertical edge pixels are kept. The underneath shadow map is generated by horizontal orientation constraints and an intensity threshold, that is, only dark horizontal edges are kept. We generate taillight map by thresholds in the ‘R–B image’ described by Wang *et al.* [16], where headlights and street lamp with white colour are restrained. Then, the set of particles \mathbf{X}_{t-1} at time $t - 1$ can be generated by assigning each particle a weight corresponding to the likelihood probability as shown below

$$\mathbf{X}_{t-1} = \{ \langle \mathbf{x}_{t-1}^1, w_{t-1}^1 \rangle, \langle \mathbf{x}_{t-1}^2, w_{t-1}^2 \rangle, \dots, \langle \mathbf{x}_{t-1}^n, w_{t-1}^n \rangle \} \quad (3)$$

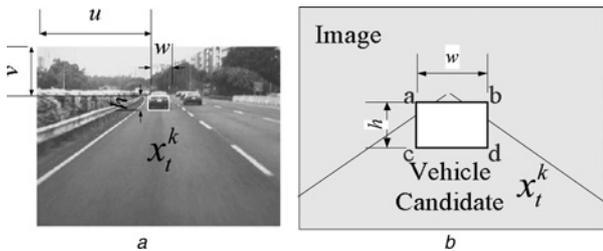


Fig. 2 Schematic description of notations

a Vehicle candidate vector example

b Vehicle candidate for evaluating the likelihood. The rectangle abcd denotes a vehicle candidate

where $\langle \mathbf{x}_{t-1}^k, w_{t-1}^k \rangle$ is the k th particle at time $t - 1$, with state vector \mathbf{x}_{t-1}^k , the associated weight w_{t-1}^k , and the total number of particles equals to n . Without loss of generality, the particle weights are arranged in descending order, that is, $\forall k, j \in N, 1 \leq k < j \leq n \Rightarrow w_{t-1}^k \geq w_{t-1}^j$.

If the particles are well distributed around vehicles, the particles will converge to the solutions soon. The ‘initial guess’ is important for a system to converge more quickly. Similar to the work proposed by Isard and Blake [31], we use the data-driven technique to provide a better guess of initial sampling by drawing particles according to the currently observed image. The areas in the image with the underneath shadow cue and the vertical cue are initially considered as particles for further inference in particle filtering framework. At each round, 10% of the particles are generated from data-driven initial sampling, and 90% of the particles are derived from the previous time step. The 10% new particles help detect new targets and make the system converge faster.

2.3 Propagation

In the propagation stage, the particles will propagate to the new locations as a new ‘guess’. This stage consists of two steps, namely, deterministic drift and randomised diffusion. In the deterministic drift step, we use a constant velocity model for each particle, since the relative velocity of vehicles during a short time interval can be regarded as a constant. Thus, we add the velocity component to the centre coordinate of the particle set to predict the state of particle at next time step. The state vector after deterministic drift, namely, \mathbf{x}'_t , can be expressed as

$$\begin{aligned} \mathbf{x}'_t &= \left(u_{t-1} + \Delta u_{t-1}, v_{t-1} + \Delta v_{t-1}, w_t + \Delta w_{t-1}, \right. \\ &\quad \left. h_t + \Delta w_{t-1} \frac{h_{t-1}}{w_{t-1}}, \Delta u_{t-1}, \Delta v_{t-1}, \Delta w_{t-1} \right)^t \end{aligned} \quad (4)$$

Note that particles with identical state vector will move to the same location. Next, we randomly diffuse, by moving slightly every element of the vehicle candidate. This is important since we cannot predict the true dynamic model of the target, and random move can give the particle a chance to shift to the correct state.

$$\tilde{\mathbf{x}}_t = \mathbf{x}'_t + N(0, \Sigma) \quad (5)$$

where $\tilde{\mathbf{x}}_t$ in (5) is the state vector after random diffusion, and $N(0, \Sigma)$ is a normal distribution noise with zero mean and covariance matrix Σ . After this step, we predict the possible locations of the vehicles at the current time instant by drifting and diffusing the estimated particles at the previous time.

3 Hypothesis verification

According to observations, there are many characteristics of a vehicle. We refer to these characteristics as ‘cues’ for measuring every candidate. We assign a weight to every candidate by evaluating its corresponding likelihood probability. The more similar the candidate is to a vehicle pattern, the higher the likelihood score of the corresponding candidate is.

3.1 Observations

Each particle is associated with a likelihood probability according to the measurement vectors as shown in Fig. 3. The likelihood of each measurement can be evaluated by the currently observed image with respect to the four cues, namely, vertical edge, underneath shadow, symmetry and taillight. We will explain how each cue is evaluated as follows. The notations are defined in a schematic form as shown in Fig. 2b. The rectangle $abcd$ in Fig. 2b denotes a vehicle candidate.

3.1.1 Vertical edge cue: A vehicle candidate will contain edge pixels on the vertical boundaries if the bounding box fits the vehicle properly [17]. We define the likelihood of a vertical edge cue as the number of total vertical edge pixels divided by vertical boundaries. The exponential term of the likelihood probability can be expressed as follows

$$g^v(z_t|x_t) = \frac{\text{Vertical edge pixels on } \overline{ac} + \overline{bd}}{\overline{ac} + \overline{bd}} \quad (6)$$

The function $g^v(z_t|x_t)$ will return value 1 for ideal case, that is, $\overline{ac} + \overline{bd}$ in Fig. 2b are fully overlapped with vertical edge pixels. The likelihood probability in (7) is proportional to the exponential of the function $g^v(z_t|x_t)$, where the η^v is a normalising term.

$$p^{(v)}(z_t|x_t) = \frac{1}{\eta^v} \exp(g^v(z_t|x_t))$$

$$\Rightarrow p^{(v)}(z_t|x_t) \propto \exp(g^v(z_t|x_t)) \quad (7)$$

3.1.2 Underneath shadow cue: The shadow under the vehicle is a strong feature for a vehicle [17]. We use horizontal edge detection with low intensity to locate the underneath pixels. The ratio between the number of underneath shadow pixels is computed and the bottom boundary of the vehicle candidate, whose formal definition is expressed in (8). The underneath shadow cue will return

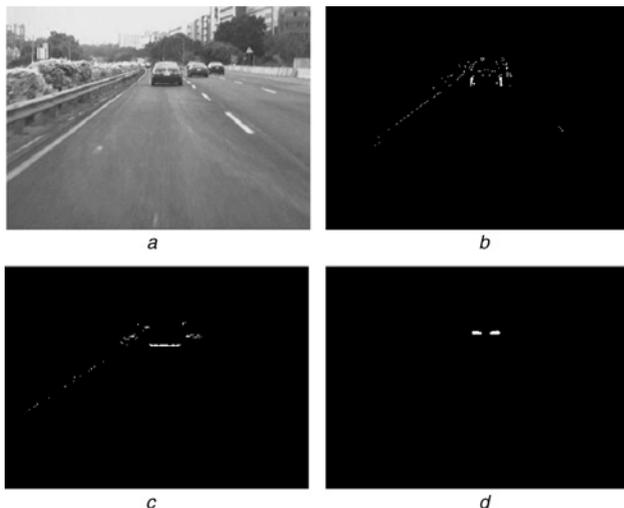


Fig. 3 Measurement vectors

- a Original image z_t^{im}
- b Vertical edge map z_t^{ve}
- c Underneath map z_t^{un}
- d Taillight map z_t^{tl}

0 if there is no underneath pixel overlapped with the edge \overline{cd} in Fig. 2b. Equation (9) shows, the likelihood probability of the underneath cue is defined in proportion to the exponential of the function $g^u(z_t^{un}|x_t)$.

$$g^u(z_t^{un}|x_t) = \frac{\text{Underneath pixels on } \overline{cd}}{\overline{cd}} \quad (8)$$

$$p(z_t^{un}|x_t) = \frac{1}{\eta^u} \exp(g^u(z_t^{un}|x_t))$$

$$\Rightarrow p(z_t^{un}|x_t) \propto \exp(g^u(z_t^{un}|x_t)) \quad (9)$$

3.1.3 Taillight cue: In the night scene, taillights are the most significant feature for a vehicle [15, 16]. The taillight will appear as a blob in the night scene image. We assume that there are two taillights in the vehicle candidate bounding box. We search two taillight spots in the bounding box as [16]; bright blobs in the R–B image are extracted as candidates. We calculate the distance between two farthest taillights if there are more than two possible ones. The taillight cue uses the taillight distance in the bounding box divided by the width of the box to represent the likelihood of the vehicle candidate, denoted as $g^{tl}(z_t^{tl}|x_t)$ which is shown in the following equation

$$g^{tl}(z_t^{tl}|x_t) = \frac{\text{Taillight distance}}{w} \quad (10)$$

It will return a high value if the two taillight spots are close to the vertical edges of the bounding box as shown in Fig. 2b. The exponential of the taillight cue provides a proportional term of the likelihood probability of such cue

$$p(z_t^{tl}|x_t) = \frac{1}{\eta^{tl}} \exp(g^{tl}(z_t^{tl}|x_t))$$

$$\Rightarrow p(z_t^{tl}|x_t) \propto \exp(g^{tl}(z_t^{tl}|x_t)) \quad (11)$$

where η_{tl} is the normalising term.

3.1.4 Symmetry cue: Horizontal symmetry in the rear part is a property of a vehicle [14]. We define a symmetry cue function within the bounding box as the ratio between the total symmetry pixels and the bounding box width. The formula of this symmetry cue and the associated likelihood probability can be expressed as follows

$$g^s(z_t^{im}|x_t) = \frac{\sum_{i=1}^{w/2} \sum_{j=1}^h k(i, j)}{(wh/2)} \quad (12)$$

$$k(i, j) = \begin{cases} 1, & \text{if } \left| \frac{I(i, j) - I(w - i, j)}{I(i, j)} \right| < \theta_{\text{sym}} \\ 0, & \text{else} \end{cases} \quad (13)$$

$$p(z_t^{im}|x_t) = \frac{1}{\eta^s} \exp(g^s(z_t^{im}|x_t))$$

$$\Rightarrow p(z_t^{im}|x_t) \propto \exp(g^s(z_t^{im}|x_t)) \quad (14)$$

where the notation $I(i, j)$ here means the intensity of the image at the coordinate (i, j) with respect to the top left corner point of the bounding box, h is the bounding box height, w is the bounding box width, θ_{sym} is the threshold for detecting a symmetry pair and η^s is the normalising term. Note that the function $k(i, j)$ will return value 1 if the two intensities of

the corresponding pixel pair are similar enough. The symmetry cue function defined in (12) refers to the ratio between the total intensity-wise similar corresponding pixel pairs and the half-width of the bounding box.

3.2 Cue fusion

We will use only one integrated likelihood density for one vehicle candidate, although we may obtain multi-modal observations. It should be conceivable that cue fusion helps to build a robust DAS under various lighting conditions. For example, the underneath shadow cue is good at day time whereas the taillight cue is good at night time. To integrate four likelihood functions derived from four cues aforementioned, we can evaluate the weighted likelihood $p(z_t^i|x_t^i)$ as multiplications of likelihood of every cue and a constant η based on a simplified assumption that each cue is conditionally independent of the other cue.

$$p(z_t^i|x_t^i) = \eta p(z_t^{vc}|x_t^i) p(z_t^{un}|x_t^i) p(z_t^{tl}|x_t^i) p(z_t^{im}|x_t^i) \quad (15)$$

By rewriting (15) into the exponential term, we obtain

$$p(z_t^i|x_t^i) \propto \exp(W_t^v g^v(z_t^{vc}|x_t^i) + W_t^u g^u(z_t^{un}|x_t^i) + W_t^{tl} g^{tl}(z_t^{tl}|x_t^i) + W_t^s g^s(z_t^{im}|x_t^i)) \quad (16)$$

where W_t^i is the fusion weight of the cue i at time step t . Equivalently speaking, the fused likelihood probability is equal to the exponential term of the weighted sum of every cue. The higher the confidence of a cue is, the higher the fusion weight of the cue assigned. For example, we use 0.3, 0.3, 0.1 and 0.3 for vertical edge, underneath shadow, taillight and symmetry cue in daytime scenes.

4 Vehicle detection and tracking

After cue fusion stage, we have examined the particles in the current time step and have got the likelihood of each particle. Each target produces several particles in the state space. To evaluate the final belief of the state, we perform the data associations with particles. Then, we track each target with one Kalman filter. The details are shown as follows.

4.1 Evaluation of the particle filter

Isard and Blake [20] chose weighted sum of particles to represent the final belief for single target tracking. It is not adequate for our system, since the target distribution is a multi-modal distribution which corresponds to multiple objects in our case. The modes or local maxima are the most possible solutions at the current time. As we only draw finite number of particles in the state space, we can hardly find the maximum likelihood particles as the best belief.

Under such consideration, we approximate the belief state by computing the weighted mean of particles belonging to one mode. Thus, we cluster particles first, and then compute the weighted mean for the final belief. In our case, the number of cluster is unknown so that the traditional supervised clustering techniques like k -means are not applicable. Thus, an unsupervised clustering technique named BSAS [29] is adopted.

The procedure of the BSAS is described as follows. Generate the first cluster at the beginning. For all other

particles, check the distance between the existing clusters and the particles. If the shortest distance between the input particle and a cluster is greater than C_d and the number of total number of clusters is less than N_{max} , construct a new cluster with this particle. Otherwise, find the closest cluster and put this particle in. Given X_t , a set of particles at time t , we can form the clusters $S_t = \{s^1, s^2, \dots, s^{N_t}\}$, where s^i denotes the i th cluster, and N_t the number of clusters at time t . The flow chart of a BSAS is shown in Fig. 4, where C_d is the threshold of cluster radius and N_{max} is the maximum number of clusters.

One of the issues of the BSAS algorithm is that the sequence of input will significantly affect the clustering results. The first input particle always forms a cluster, and then the other particles are merged with the closest cluster among clusters already formed. Therefore, we need to cluster meaningful particles first. This gives us the idea to improve the solution of the BSAS, that is, to feed the solutions of the previous time step first. At the beginning, the input sequence is randomly chosen to form clusters. At later time steps, instead of random input sequence, we feed the previous cluster first to generate meaningful clusters. As the distance between these particles is large (particles with distance within C_d will be clustered) and they are meaningful (solutions from the previous time step), the results of the clustering will be more stable.

4.2 High-level tracking

The particle filter will generate a set of possible vehicle candidates. The performance of the particle filter will depend on the number of the particles and the quality of the discriminability of feature cues. The usage of more particles and the better cue function may help improve the performance. This will result in, however, a decrease of the efficiency.

Instead of keeping combination of all hypotheses, we maintain the vehicle target candidates in a high-level framework. Without putting lots of particles, we update each target (track) x_i in the high-level framework. Since the number of targets is usually small (<20), we can update the target through a more complicated classifier without bringing heavy computational burden.

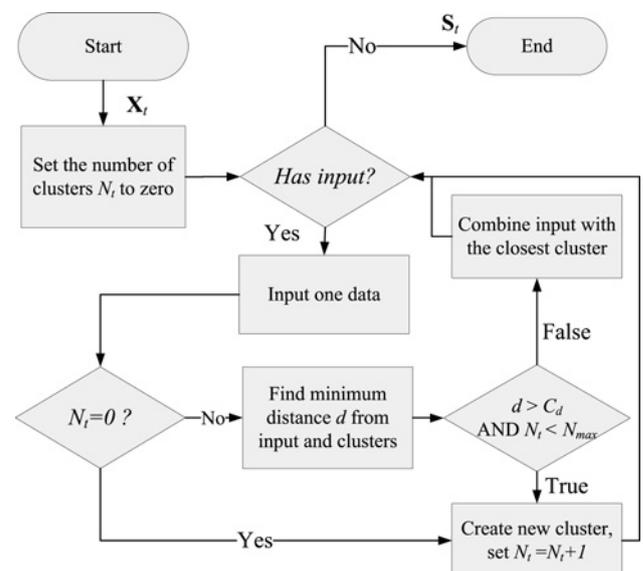


Fig. 4 Flow chart of the BSAS

Given clusters S_t and image I_t at time t , tracks $T_{t-1} = \{x_1, x_2, \dots, x_k\}_{t-1}$ at the time instance $t - 1$, the high-level tracker can be modelled as maximising a posterior probability $p(T_t|S_t, I_t)$.

$$\max p(T_t|S_t, I_t) = \max(p(S_t, I_t|T_t)p(T_t|T_{t-1})) \quad (17)$$

where $p(S_t, I_t|T_t)$ stands for the likelihood probability of cluster set S_t and image I_t given tracks T_t , and $p(T_t|T_{t-1})$ is the transition probability given tracks at the time instance $t - 1$.

In the high-level tracking module, we maintain each track (a cluster) with a Kalman filter. We perform data association between targets and the current generated particle clusters. The initial likelihood is assigned by a Gaussian function with input equal to the distance between targets and the particle clusters. The greater the distance is, the less the likelihood becomes. Then, the likelihood of each candidate is updated through a Haar-like feature [32] classifier from AdaBoost training. If the learnt classifier return 'false', the target will be assigned a small likelihood. To model the transition probability $p(T_t|T_{t-1})$, we choose a constant velocity model.

5 Experiments

In the experiments, a CCD camera is mounted behind the driving mirror to consecutively grab the images of the road scene at the driving speed between 30 and 100 km/h. The grabbed images (320×240) are delivered to the mobile computer equipped with Intel® Pentium® M 1.4 GHz processor and 512 MB RAM through IEEE 1394 interface.

The observation stage in the proposed particle filtering contains four different vision cues, namely, vertical edge cue, taillight cue, underneath shadow cue and symmetry cue. The measurement vectors are shown in Fig. 5. Some examples of special lighting conditions are shown in Fig. 6. The over exposure, due to high contrast, makes the vertical edges blur and hardly recognisable; the high-level tracking



Fig. 6 Special lighting conditions

- a Inside a tunnel
- b Over exposure

strategy gives us a chance to track vehicles even when the features of the vehicle are obscure.

To show the performance of the data-driven mechanism, we evaluate the results with and without data-driven approach. The results are shown in Fig. 7. We use 1800 particles in these results. Without data-driven mechanism, only two vehicles can be detected in Fig. 7b. In Fig. 8, the particles are projected to the bird's-eye-view. Fig. 8a shows the case with the data-driven mechanism, which helps particles to converge to the meaningful area; the variance of the data-driven clusters is less than that without data-driven mechanism. Without data-driven mechanism, particles scatter in the state space as shown in Fig. 8b. The analysis



Fig. 7 Comparison of data-driven results

- a Data-driven results
- b Without data-driven results

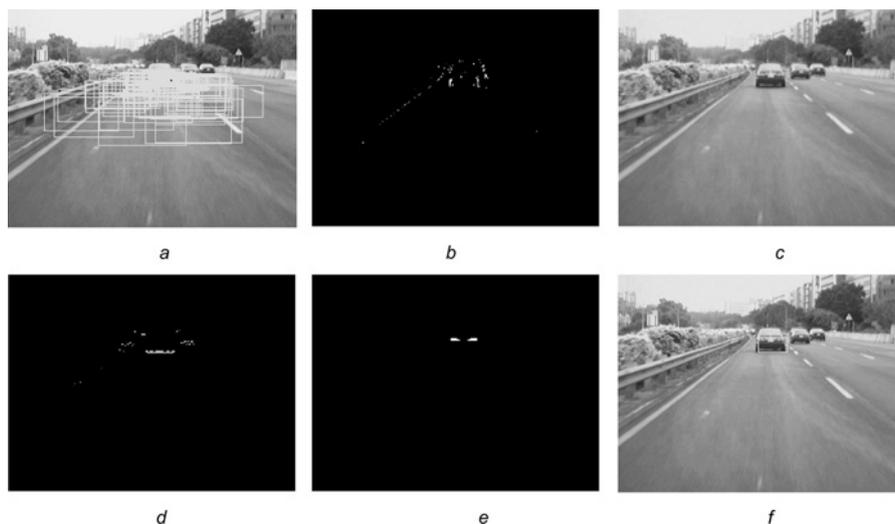


Fig. 5 Measurement vectors and the detection result

- a Initial sampling
- b Vertical edge map
- c Original image
- d Underneath map
- e Taillight map
- f Detection result

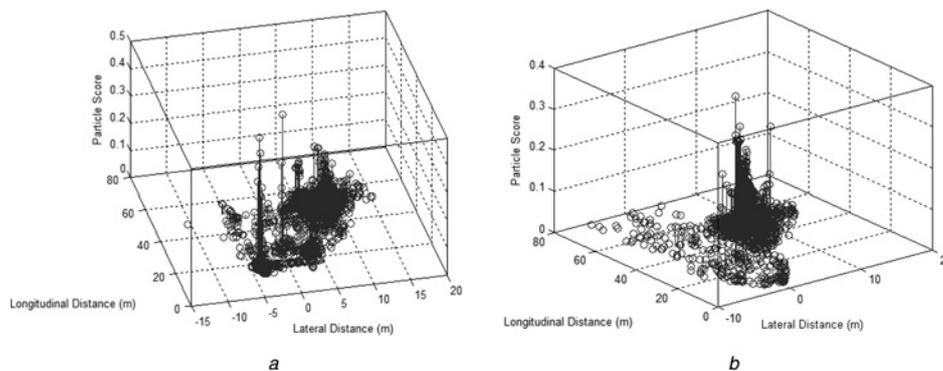


Fig. 8 Comparison of data-driven results

a Data-driven results

b Without data driven

Vertical axis shows the longitudinal distance and the lateral axis shows the lateral distance of the target from the host vehicle

Table 1 Data-driven analysis

	High-score clusters	Cluster variance mean
with data driven	17	0.2834
without data driven	16	0.3280

of the clusters is shown in Table 1. Without data-driven mechanism, particles are attracted by a local maximum; particles are ‘trapped’ into a huge cluster in the right-hand side. On the contrary, more particles are found with high-score targets when the data-driven approach is employed.

For detection rate analysis, we use more than 28 000 images with more than 5000 vehicles within 50 m for several experiments. The testing videos from highway were classified into different scenarios. ‘Normal’ condition stands for bright condition with clear scenes. ‘Through tunnel’ stands for a video sequence with several tunnels. ‘Under overpass’ and ‘Sunset, forward shadow’ represent different shadow appearance on the ground. ‘Sprinkle’ and ‘Sunny Rain’ stand for slightly rainy conditions. The details of the detection result are shown in Table 2. The ‘Hit’ means the number of correct detection of vehicles and the ‘Miss’ represents the number of false negatives, that is, the number of missed vehicles. The detection rate of the system is up to 99% at clear scenes, even when the shadow appears on the ground or the host vehicle changes lane. The performance will have slightly deterioration in the case of entering into a tunnel since the blooming effect blurs the vertical edge and the underneath shadow of the preceding vehicles. In the special scenes of sunny rain while facing sun, the quality of the images is even worse. The rain drops on the windshield

Table 2 Vehicle detection rate of this system

	Hit	Miss	Detection rate, %	Precision, %
normal	473	4	99.37	98.96
through tunnel	685	148	82.35	98.85
left turn, changing lane	765	29	96.47	80.63
under overpass	247	47	84.35	99.26
sunset, forward shadow	2138	101	95.53	90.22
sprinkle	460	51	90.00	95.04
sunny rain, facing sun	378	23	94.51	98.44
total	5146	403	92.84	92.42

refract the light and distorted the preceding vehicles. The sunshine, when facing the sun, lengthens the shadow of vehicles and blurs the vertical edges of preceding vehicle, which causes the degradation of the performance in the last scene. The high-level tracking in our system gives us a chance to track the preceding vehicle with short period of ‘missing.’ The overall system performance is 92% detection rate with 92% precision. In the near future, the performance will be improved if better vehicle cues for low-light conditions are sought.

To compare our system with others, we construct AdaBoost classifiers using Haar-like features [32]. The AdaBoost classifier is very successful in the face detection domain. Also, the AdaBoost Classifier is publicly available in the OpenCV library. Here, about 1500 positive samples, each of which contains only a vehicle, are chosen from the MIT LabelMe [33] database for the training phase. At the same time, 3770 negative samples are selected from our videos. To reduce the false alarm of the classifier, the negative samples are selected both from the false detection results and manually selected road scenes. Totally, 273 weak classifiers are selected to construct a 17-stage cascaded AdaBoost classifier. The details of the detection results are shown in Table 3. Due to the lighting condition variation, the detection result is poor in case of inside a tunnel, since the classifiers missed all vehicles in the tunnel. The detection rate is 59.42%, against our 82.35%. In the cases of ‘Under overpass’ and ‘Sprinkle’, vehicle types absent from the training samples are usually missed in the test scenario. The detection rates are 79.11 and 79.84% against our 84.35 and 90%. The drawback of learning classifiers is hard to collect good training samples. Under different lighting conditions, we cannot train a single classifier with high-detection rate and high precision using training samples with different lighting conditions; mixture of different samples will let the classifier compromise between the day and night thresholds. That means scenes which appear in fewer training samples will be sacrificed. On the contrary, our system performance is better than the AdaBoost classifiers under various conditions.

Table 3 Performance of the AdaBoost classifiers

	Hit	Miss	Detection rate, %	Precision, %
through tunnel	495	338	59.42	82.55
under overpass	232	62	79.11	78.47
sprinkle	408	103	79.84	84.98

Table 4 Vehicle detection and tracking processing time

Average, ms	Σ , ms	Minimum	Maximum
47.03	0.344	46.41	47.48

The time for processing vehicle detection and tracking of each frame is less than 48 ms. Since the processing time is closely related to the number of particles, the standard deviation of the processing time is small when the particle number is determined. The details of the processing time with the number of particles = 100 are shown in Table 4.

6 Conclusions

A statistical framework called particle filter is adopted to integrate the four cues including vertical edge, underneath shadow, symmetry and the taillights. By fusing multiple cues for vehicle detection, our system can achieve high detection rate under various lighting conditions. Besides the particle filtering framework, a high-level tracking strategy is proposed to deal with the image with blurring effect for a short period. With BSAS clustering technique and initial data-driven sampling, multiple vehicles can be tracked with only one set of particles. We maintain the high-level solutions by assigning each solution a Kalman filter with constant velocity motion model, and then propagate the solution to the BSAS input. Multiple objects can be tracked at the same time in an efficient way. Finally, seven videos are taken to validate the effectiveness of our system. The detection rate is 92.84% even under some difficult environment and various lighting conditions. Compared with traditional AdaBoost classifier, our system is more reliable under various lighting conditions. The frame rate of our proposed system is roughly 20 frames per second. In some cases, however, the detection results are unsatisfactorily owing to poor quality of the image. In the future, we will introduce more cues, learned through online or off-line mechanisms, to further improve our system performance.

7 Acknowledgments

This work is partly supported by the Chung-Shan Institute of Science and Technology, Taiwan, R.O.C. We also gratefully acknowledge the support by the National Science Council, Taiwan, ROC, grant NSC99-2221-E-390-039-MY3, and by the 'Aim for the Top University Plan' from the National Taiwan Normal University and the Ministry of Education.

8 References

- Sun, Z., Bebis, G., Miller, R.: 'On-road vehicle detection: a review', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2006, **28**, pp. 694–711
- Sun, Z., Bebis, G., Miller, R.: 'Monocular Precrash vehicle detection: features and classifiers', *IEEE Trans. Image Process.*, 2006, **15**, pp. 2019–2034
- Khammari, A., Nashashibi, F., Abramson, Y., Laurgeau, C.: 'Vehicle detection combining gradient analysis and AdaBoost classification'. Proc. IEEE Conf. on Intelligent Transportation Systems, 2005, pp. 66–71
- Cheng, H., Zheng, N., Sun, C.: 'Boosted Gabor features applied to vehicle detection'. Proc. Int. Conf. on Pattern Recognition, Hong Kong, 2006, vol. 1, pp. 662–666
- Liu, W., Wen, X., Duan, B., Yuan, H.A.Y.H., Wang, N.A.W.N.: 'Rear vehicle detection and tracking for lane change assist'. Proc. IEEE Intelligent Vehicles Symp., 2007, pp. 252–257
- Wen, X., Yuan, H., Yang, C., Song, C., Duan, B., Zhao, H.: 'Improved Haar wavelet feature extraction approaches for vehicle detection'. Proc. IEEE Conf. on Intelligent Transportation Systems, Washington, USA, 2007, pp. 1050–1053
- Kim, S., Oh, S.-Y., Kang, J., *et al.*: 'Front and rear vehicle detection and tracking in the day and night times using vision and sonar sensor fusion'. Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 2005, pp. 2173–2178
- Bai, H., Wu, J., Liu, C.: 'Motion and Haar-like features based vehicle detection'. Proc. Int. Multi-Media Modelling Conf., 2006, pp. 356–359
- Wang, C.C.R., Lien, J.J.J.: 'Automatic vehicle detection using local features a statistical approach', *IEEE Trans. Intell. Transp. Syst.*, 2008, **9**, pp. 83–96
- Sivaraman, S., Trivedi, M.M.: 'A general active-learning framework for on-road vehicle recognition and tracking', *IEEE Trans. Intell. Transp. Syst.*, 2010, **11**, pp. 267–276
- Tsai, Y.-M., Huang, K.-Y., Tsai, C.-C., Chen, L.-G.: 'Learning-based vehicle detection using up-scaling schemes and predictive frame pipeline structures'. Int. Conf. on Pattern Recognition, 2010, pp. 3101–3104
- Handmann, U., Kalinke, T., Tzomakas, C., Werner, M., Seelen, W.v.: 'An image processing system for driver assistance', *Image Vis. Comput.*, 2000, **18**, pp. 367–376
- Bensrhair, A., Bertozzi, M., Broggi, A., Miche, P., Mousset, S., Toulminet, G.: 'A cooperative approach to vision-based vehicle detection'. Proc. IEEE Intelligent Transportation Systems Conf., 2001, pp. 207–212
- Broggi, A., Bertozzi, M., Fascioli, A., Bianco, C.G.L., Piazzzi, A.: 'Visual perception of obstacles and vehicles for platooning', *IEEE Trans. Intell. Transp. Syst.*, 2000, **1**, (3), pp. 164–176
- Chern, M.-Y., Hou, P.-C.: 'The lane recognition and vehicle detection at night for a camera-assisted car on highway'. IEEE Int. Conf. on Robotics and Automation, 2003, vol. 2, pp. 2110–2115
- Wang, C.-C., Huang, S.-S., Fu, L.-C., Hsiao, P.-Y.: 'Driver assistance system for lane detection and vehicle recognition with night vision'. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 2005, pp. 3314–3319
- Huang, S.-S., Chen, C.-J., Hsiao, P.-Y., Fu, L.-C.: 'On-board vision system for lane recognition and front-vehicle detection to enhance driver's awareness'. IEEE Int. Conf. on Robotics and Automation, 2004, vol. 3, pp. 2456–2461
- Tao, X., Debrunner, C.: 'Stochastic car tracking with line- and colour-based features', *IEEE Trans. Intell. Transp. Syst.*, 2004, **5**, pp. 324–328
- Martin, S., Bernt, S.: 'Towards robust multi-cue integration for visual tracking', *Mach. Vis. Appl.*, 2003, **14**, pp. 50–58
- Isard, M., Blake, A.: 'CONDENSATION—conditional density propagation for visual tracking', *Int. J. Comput. Vis.*, 1998, **29**, pp. 5–28
- Chang, C., Ansari, R., Khokhar, A.: 'Multiple object tracking with kernel particle filter'. Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, 2005, vol. 1, pp. 566–573
- Zhou, S.K., Chellappa, R., Moghaddam, B.: 'Visual tracking and recognition using appearance-adaptive models in particle filters', *IEEE Trans. Image Process.*, 2004, **13**, pp. 1491–1506
- Hue, C., Le Cadre, J.P., Perez, P.: 'Sequential Monte Carlo methods for multiple target tracking and data fusion', *IEEE Trans. Signal Process.* [see also *IEEE Trans. Acoustics, Speech, and Signal Process.*], 2002, **50**, pp. 309–325
- Wei, D., Piater, J.: 'Tracking by cluster analysis of feature points using a mixture particle filter'. Proc. IEEE Conf. on Advanced Video and Signal Based Surveillance, 2005, pp. 165–170
- Yang, C., Duraiswami, R., Davis, L.: 'Fast multiple object tracking via a hierarchical particle filter'. Proc. IEEE Int. Conf. on Computer Vision, 2005, vol. 1, pp. 212–219
- Yang, T., Aitken, V.: 'Uniform clustered particle filtering for robot localization'. Proc. 2005 American Control Conf., 2005, vol. 7, pp. 4607–4612
- Fu, C.-M., Huang, C.-L., Chen, Y.-S.: 'Vision-based preceding vehicle detection and tracking'. Proc. Int. Conf. on Pattern Recognition, 2006, vol. 2, pp. 1070–1073
- Hue, C., Le Cadre, J.P., Perez, P.: 'Tracking multiple objects with particle filtering', *IEEE Trans. Aerosp. Electron. Syst.*, 2002, **38**, pp. 791–812
- Theodoridis, S., Koutroubas, K.: 'Pattern recognition' (Academic Press, 2003), p. 433
- Kalman, R.: 'A new approach to linear filtering and prediction', *J. Basic Eng.*, 1960, **82**, series D, pp. 35–45
- Isard, M., Blake, A.: 'CONDENSATION: unifying low-level and high-level tracking in a stochastic framework'. Proc. European Conf. on Computer Vision, 1998, vol. 1, pp. 893–908
- Viola, P., Jones, M.: 'Rapid object detection using a boosted cascade of simple features'. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, 2001, pp. 511–518
- LabelMe, MIT, Computer Science and Artificial Intelligence Laboratory, <http://labelme.csail.mit.edu/>